COP8™ MICROCONTROLLER DATABOOK

1996/1997 Edition

COP8 Family 1

COP8 OTP Products 2

COP8 32K OTP Products 3

COP8 Basic Family Products 4

COP8 Feature Family Products 5

MICROWIRE/PLUS™ Peripherals 6

COP8 Applications 7

Appendices/Physical Dimensions 8

TRADEMARKS

Following is the most current list of National Semiconductor Corporation's trademarks and registered trademarks.

ABiC™
Abuseable™
AirShare™
Anadig™
APPS™
ARi1™
ASPECT™
AT/LANTIC™
Auto-Chem Deflasher™
BCP™
BI-FET™
BI-FET II™
BI-LINE™
BIPLAN™
BLC™
BLX™
BMAC™
Boomer®
Brite-Lite™
BSI™
BSI-2™
CDD™
CDL™
CGS™
CIM™
CIMBUS™
CLASIC™
COMBO®
COMBO I™
COMBO II™
CompactRISC™
CompactSPEECH™
COPS™ microcontrollers
COP8™
CRD™
CROSSVOLT™
CSNI™
CTI™
CYCLONE™
DA4™
DENSPAK™
DIB™
DISCERN™
DISTILL™
DNR®
DPVM™
E2CMOS™
ELSTAR™
Embedded System Processor™
EP™
E-Z-LINK™
FACT™
FACT Quiet Series™
FAIRCAD™
Fairtech™
FAST®
FastLock™
FASTr™
GENIX™
GNX™
GTO™
HEX 3000™
HiSeC™
HPC™
HyBal™
I3L®
ICM™
Integral ISE™
Intelisplay™
Inter-LERIC™
Inter-RIC™
ISE™
ISE/06™
ISE/08™
ISE/16™
ISE32™
ISOPLANAR™
ISOPLANAR-Z™
LERIC™
LMCMOS™
M2CMOS™
Macrobus™
Macrocomponent™
MACSI™
MAPL™
MAXI-ROM®
Microbus™ data bus
MICRO-DAC™
μPot™
μtalker™
Microtalker™
MICROWIRE™
MICROWIRE/PLUS™
MOLE™
MPA™
MST™
Naked-8™
National®
National Semiconductor®
National Semiconductor Corp.®
NAX 800™
NeuFuz™
Nitride Plus™
Nitride Plus Oxide™
NML™
NOBUS™
NS486™
NSC800™
NSCISE™
NSX-16™
NS-XC-16™
NTERCOM™
NURAM™
OPAL™
Overture™
OXISS™
P2CMOS™
Perfect Watch™
PLAN™
PLANAR™
PLAYER™
PLAYER+™
PLLatinum™
Plus-2™
Polycraft™
POP™
Power + Control™
POWERplanar™
QS™
QUAD3000™
Quiet Series™
QUIKLOOK™
RAT™
RIC™
RICKIT™
RTX16™
SCENIC™
SCX™
SERIES/800™
Series 32000®
SIMPLE SWITCHER®
SNI™
SNIC™
SofChek™
SONIC™
SpeechPro™
SPiKe™
SPIRE™
Staggered Refresh™
STAR™
Starlink™
STARPLEX™
ST-NIC™
SuperAT™
Super-Block™
SuperChip™
SuperScript™
Switchers Made Simple®
SYS32™
TapePak®
TDS™
TeleGate™
The National Anthem®
TinyPak™
TLC™
Trapezoidal™
TRI-CODE™
TRI-POLY™
TRI-SAFE™
TRI-STATE®
TROPIC™
Tropic Pele'™
Tropic Reef™
TURBOTRANSCEIVER™
TWISTER™
VIP™
VR32™
WATCHDOG™
XMOS™
XPU™
Z STAR™
883B/RETS™
883S/RETS™

Dolby® is a registered trademark of Dolby Labs.
I2C® is a registered trademark of Philips.
IBM®, PC®, PC-AT® and PC-XT® are registered trademarks of International Business Machines Corporation.
iceMASTER™ is a trademark of MetaLink Corporation.

LIFE SUPPORT POLICY

NATIONAL'S PRODUCTS ARE NOT AUTHORIZED FOR USE AS CRITICAL COMPONENTS IN LIFE SUPPORT DEVICES OR SYSTEMS WITHOUT THE EXPRESS WRITTEN APPROVAL OF THE PRESIDENT OF NATIONAL SEMICONDUCTOR CORPORATION. As used herein:

1. Life support devices or systems are devices or systems which, (a) are intended for surgical implant into the body, or (b) support or sustain life, and whose failure to perform, when properly used in accordance with instructions for use provided in the labeling, can be reasonably expected to result in a significant injury to the user.
2. A critical component is any component of a life support device or system whose failure to perform can be reasonably expected to cause the failure of the life support device or system, or to affect its safety or effectiveness.

National Semiconductor Corporation 2900 Semiconductor Drive, P.O. Box 58090, Santa Clara, California 95052-8090 1-800-272-9959
TWX (910) 339-9240

National Semiconductor

Product Status Definitions

Definition of Terms

Data Sheet Identification	Product Status	Definition
Advance Information	Formative or In Design	This data sheet contains the design specifications for product development. Specifications may change in any manner without notice.
Preliminary	First Production	This data sheet contains preliminary data, and supplementary data will be published at a later date. National Semiconductor Corporation reserves the right to make changes at any time without notice in order to improve design and supply the best possible product.
No Identification Noted	Full Production	This data sheet contains final specifications. National Semiconductor Corporation reserves the right to make changes at any time without notice in order to improve design and supply the best possible product.
Obsolete	Not In Production	This data sheet contains specifications on a product that has been discontinued by National Semiconductor Corporation. The data sheet is printed for reference information only.

National Semiconductor Corporation reserves the right to make changes without further notice to any products herein to improve reliability, function or design. National does not assume any liability arising out of the application or use of any product or circuit described herein; neither does it convey any license under its patent rights, nor the rights of others.

Table of Contents

Alphanumeric Index viii

Section 1 COP8 Family

COP8 Family 1-3

Section 2 COP8 OTP Products

COP8SAA7/COP8SAB7/COP8SAC7 8-Bit One-Time Programmable (OTP) Microcontroller 2-3

COP8780C/COP8781C/COP8782C 8-Bit One-Time Programmable (OTP) Microcontroller 2-58

COP87L20CJ/COP87L22CJ 8-Bit One-Time Programmable (OTP) Microcontrollers with Multi-Input Wake-Up and Brown Out Detector 2-79

COP87L40CJ/COP87L42CJ 8-Bit One-Time Programmable (OTP) Microcontrollers with Multi-Input Wake-Up and Brown Out Detector 2-105

COP87L84BC 8-Bit One-Time Programmable (OTP) Microcontroller with CAN Interface 2-132

COP87L88EB/COP87L89EB 8-Bit One-Time Programmable (OTP) Microcontroller with CAN Interface, A/D and UART 2-183

COP87L88CL/COP87L84CL 8-Bit One-Time Programmable (OTP) Microcontroller 2-252

COP87L88CF/COP87L84CF 8-Bit One-Time Programmable (OTP) Microcontroller with A/D Converter 2-282

COP8ACC7 8-Bit One-Time Programmable (OTP) Microcontroller with High Resolution A/D Conversion 2-314

COP87L88EK/COP87L84EK 8-Bit One-Time Programmable (OTP) Microcontroller with Analog Function Block 2-349

COP87L88EG/COP87L84EG 8-Bit One-Time Programmable (OTP) Microcontroller with UART and Three Multi-Function Timers 2-381

COP87L88FH/COP87L84FH 8-Bit One-Time Programmable (OTP) Microcontroller with UART and Three Multi-Function Timers and Multiply/Divide Block 2-418

COP87L88GG 8-Bit One-Time Programmable (OTP) Microcontroller with UART and Three Multi-Function Timers 2-458

COP87L88GD 8-Bit One-Time Programmable (OTP) Microcontroller with A/D Converter 2-497

COP87L88KG 8-Bit One-Time Programmable (OTP) Microcontroller with UART and Three Multi-Function Timers 2-499

Section 3 COP8 32K OTP Products

COP87L40RJ/COP87L42RJ 8-Bit One-Time Programmable (OTP) Microcontroller with 32 Kbytes of Program Memory 3-3

COP87L88RK/COP87L84RK 8-Bit One-Time Programmable (OTP) Microcontroller with Analog Function Block and 32 Kbytes of Program Memory 3-29

COP87L88RG 8-Bit One-Time Programmable (OTP) Microcontroller with 32 Kbytes of Program Memory 3-61

COP87L84RG 8-Bit One-Time Programmable (OTP) Microcontroller with 32 Kbytes of Program Memory 3-98

COP87L88RD 8-Bit One-Time Programmable (OTP) Microcontroller with A/D Converter and 32 Kbytes of Program Memory 3-135

COP87L88RW 8-Bit One-Time Programmable (OTP) Microcontroller with Pulse Train Generators and Capture Modules 3-137

Section 4 COP8 Basic Family Products

COP912C/COP912CH 8-Bit Microcontroller 4-3

COP620C/COP622C/COP640C/COP642C/COP820C/COP822C/COP840C/COP842C/COP920C/COP922C/COP940C/COP942C 8-Bit Microcontroller 4-24

Table of Contents (Continued)

Section 4 COP8 Basic Family Products (Continued)
COP820CJ/COP822CJ/COP823CJ 8-Bit Microcontroller with Multi-Input Wake-Up and Brown Out Detector 4-47
COP840CJ/COP842CJ/COP940CJ/COP942CJ 8-Bit Microcontrollers with Multi-Input Wake-Up and Brown Out Detector 4-75
COP680C/COP681C/COP682C/COP880C/COP881C/COP882C/COP980C/COP981C/COP982C Microcontrollers 4-106
Section 5 COP8 Feature Family Products
COP884BC/COP684BC 8-Bit Microcontrollers with CAN Interface 5-3
COP888EB/COP889EB/COP688EB/COP689EB 8-Bit Microcontroller with CAN Interface, A/D, and UART 5-59
COP688CL/COP684CL/COP888CL/COP884CL/COP988CL/COP984CL 8-Bit Microcontroller 5-130
COP888CF/COP884CF/COP988CF/COP984CF 8-Bit CMOS Microcontroller with A/D Converter 5-167
COP8ACC5 8-Bit Microcontroller with High Resolution A/D Conversion 5-202
COP688EK/COP684EK/COP888EK/COP884EK/COP988EK/COP984EK 8-Bit Microcontroller with Analog Function Block 5-240
COP688GD/COP888GD/COP988GD 8-Bit Microcontroller with A/D Converter 5-279
COP888CG/COP884CG 8-Bit Microcontroller with UART and Three Multi-Function Timers 5-281
COP688CS/COP684CS/COP888CS/COP884CS/COP988CS/COP984CS 8-Bit Microcontroller with UART and One Multi-Function Timer 5-319
COP688EG/COP684EG/COP888EG/COP884EG/COP988EG/COP984EG 8-Bit Microcontroller with UART and Three Multi-Function Timers 5-362
COP688FH/COP684FH/COP888FH/COP884FH/COP988FH/COP984FH 8-Bit Microcontroller with UART, Three Multi-Function Timers and Multiply/Divide Block .. 5-406
COP688GG/COP888GG 8-Bit Microcontroller with UART, and Three Multi-Function Timers 5-449
COP688HG/COP888HG 8-Bit Microcontroller with UART, and Three Multi-Function Timers 5-492
COP688KG/COP888KG 8-Bit Microcontroller with UART, and Three Multi-Function Timers 5-535
COP888GW 8-Bit Microcontroller with Pulse Train Generators and Capture Modules .. 5-576
Section 6 MICROWIRE/PLUS Peripherals
MICROWIRE/PLUS: Serial Interface 6-3
ADC0811 8-Bit Serial I/O A/D Converter with 11-Channel Multiplexer 6-7
ADC08031/ADC08032/ADC08034/ADC08038 8-Bit High Speed Serial I/O A/D Converters with Multiplexer Options, Voltage Reference and Track/Hold Functions . 6-8
ADC0852/ADC0854 Multiplexed Comparator with 8-Bit Reference Divider 6-9
ADC1038 10-Bit Serial I/O A/D Converters with Analog Multiplexer and Track/Hold Function 6-10
ADC12038 Self-Calibrating 12-Bit Plus Sign Serial I/O A/D Converters with MUX and Sample/Hold 6-11
COP472-3 Liquid Crystal Display Controller 6-12
MM5450/MM5451 LED Display Drivers 6-18
MM5483 Liquid Crystal Display Driver 6-19
MM5484 16-Segment LED Display Drivers 6-20
MM5486 LED Display Driver 6-21
MM58241 High Voltage Display Driver 6-22
MM58341 High Voltage Display Driver 6-23

Table of Contents (Continued)

Section 6 MICROWIRE/PLUS Peripherals (Continued)

MM58342 High Voltage Display Driver 6-24
NM93C13/NM93C14 256-/1024-Bit Serial EEPROM 6-25
NM93C06/NM93C46/NM93C56/NM93C66 256-/1024-/2048-/4096-Bit Serial EEPROM (MICROWIRE Bus Interface) 6-26
NM93C46A 1024-Bit Serial Interface, Standard Voltage CMOS EEPROM 6-27
NM93C56A 2048-Bit Serial Interface, Standard Voltage CMOS EEPROM (MICROWIRE Bus Interface) 6-28
NM93C66A 4096-Bit Serial Interface, Standard Voltage CMOS EEPROM (MICROWIRE Bus Interface) 6-29
NM93C86A 16,384-Bit Serial Interface CMOS EEPROM (MICROWIRE Synchronous Bus) 6-30
NM93CS06/NM93CS46/NM93CS56/NM93CS66 (MICROWIRE Bus Interface) 256-/1024-/2048-/4096-Bit Serial EEPROM with Data Protect and Sequential Read 6-31
LMC1982 Digitally-Controlled Stereo Tone and Volume Circuit with Two Selectable Stereo Inputs 6-32
LMC1983 Digitally-Controlled Stereo Tone and Volume Circuit with Three Selectable Stereo Inputs 6-33
LMC1992 Digitally-Controlled Stereo Tone and Volume Circuit with Four-Channel Input-Selector 6-34
LMC835 Digitally-Controlled Graphic Equalizer 6-35
LM1971 μPot Digitally Controlled 62 dB Audio Attenuator with Mute 6-36
LM1972 μPot 2-Channel 78 dB Audio Attenuator with Mute 6-37
LM1973 μPot 3-Channel 76 dB Audio Attenuator with Mute 6-38

Section 7 COP8 Applications

AB-15 Protecting Data in Serial EEPROMs 7-3
AB-22 Automatic Low Cost Thermostat 7-5
AN-521 Dual Tone Multiple Frequency (DTMF) 7-7
AN-579 MICROWIRE/PLUS Serial Interface for COP800 Family 7-16
AN-596 COP800 MathPak 7-28
AN-607 Pulse Width Modulation A/D Conversion Techniques with COP800 Family Microcontrollers 7-64
AN-662 COP800 Based Automated Security/Monitoring System 7-71
AN-663 Sound Effects for the COP800 Family 7-79
AN-666 DTMF Generation with a 3.58 MHz Crystal 7-102
AN-673 2-Way Multiplexed LCD Drive and Low Cost A/D Converter Using V/F Techniques with COP8 Microcontrollers 7-130
AN-681 PC MOUSE Implementation Using COP800 7-149
AN-714 Using COP800 Devices to Control DC Stepper Motors 7-174
AN-734 MF2 Compatible Keyboard with COP8 Microcontrollers 7-184
AN-739 RS-232C Interface with COP800 7-204
AN-755 NM95C12 Flexibility in Industrial Control Applications 7-216
AN-758 Using National's MICROWIRE EEPROM 7-227
AN-794 Using an EEPROM-I²C Interface NM24C02/03/04/05/08/09/16/17 7-238
AN-823 Timekeeping Using a COP800 Microcontroller 7-247
AN-824 8-Channel 8-Bit PWM Controller 7-249
AN-841 Software for Interfacing the COP800 Family Microcontrollers to National's MICROWIRE EEPROMs 7-252
AN-871 Selling National's Write Protected "CS" Series EEPROMs to High Volume OEMs 7-258

Table of Contents (Continued)

Section 7 COP8 Applications (Continued)
AN-936 How to Use the NM93C86A Serial EEPROM as a PC/Laptop Detachable Printer File Memory Card (DPFMC) 7-262
AN-952 Low Cost A/D Conversion Using COP800 7-269
AN-953 LCD Triplex Drive with COP820CJ 7-278
AN-982 COP888GW Features and Applications 7-302
AN-983 Simple, Cost Effective A/D Conversion Using COP888EK 7-320
AN-1042 COP8 Instruction Set Performance Evaluation 7-324
AN-1043 Comparison of COP878x to the Enhanced COP8SAx7 Family—Hardware/Software Considerations 7-366
AN-1044 Comparison of COP82xCJ to the Enhanced COP8SAx7 Family—Hardware/Software Considerations 7-371
AN-1048 Replacing Dedicated Protocol Controllers with Code Efficient and Configurable Microcontrollers—Low Speed CAN Network Applications 7-376
AN-1049 Using CAN Networking for Cost Effective DC Motor Control in Vehicle Body Electronics 7-381
AN-1050 Understanding and Eliminating EMI in Microcontroller Applications 7-385

Section 8 Appendices/Physical Dimensions
Surface Mount 8-3
PLCC Packaging 8-23
Physical Dimensions 8-27
Bookshelf
Distributors
Worldwide Sales Offices

Alpha-Numeric Index

AB-15 Protecting Data in Serial EEPROMs 7-3
AB-22 Automatic Low Cost Thermostat 7-5
ADC0811 8-Bit Serial I/O A/D Converter with 11-Channel Multiplexer 6-7
ADC0852 Multiplexed Comparator with 8-Bit Reference Divider 6-9
ADC0854 Multiplexed Comparator with 8-Bit Reference Divider 6-9
ADC08031 8-Bit High Speed Serial I/O A/D Converter with Multiplexer Options, Voltage Reference and Track/Hold Functions 6-8
ADC08032 8-Bit High Speed Serial I/O A/D Converter with Multiplexer Options, Voltage Reference and Track/Hold Functions 6-8
ADC08034 8-Bit High Speed Serial I/O A/D Converter with Multiplexer Options, Voltage Reference and Track/Hold Functions 6-8
ADC08038 8-Bit High Speed Serial I/O A/D Converter with Multiplexer Options, Voltage Reference and Track/Hold Functions 6-8
ADC1038 10-Bit Serial I/O A/D Converters with Analog Multiplexer and Track/Hold Function 6-10
ADC12038 Self-Calibrating 12-Bit Plus Sign Serial I/O A/D Converters with MUX and Sample/Hold 6-11
AN-521 Dual Tone Multiple Frequency (DTMF) 7-7
AN-579 MICROWIRE/PLUS Serial Interface for COP800 Family 7-16
AN-596 COP800 MathPak 7-28
AN-607 Pulse Width Modulation A/D Conversion Techniques with COP800 Family Microcontrollers 7-64
AN-662 COP800 Based Automated Security/Monitoring System 7-71
AN-663 Sound Effects for the COP800 Family 7-79
AN-666 DTMF Generation with a 3.58 MHz Crystal 7-102
AN-673 2-Way Multiplexed LCD Drive and Low Cost A/D Converter Using V/F Techniques with COP8 Microcontrollers 7-130
AN-681 PC MOUSE Implementation Using COP800 7-149
AN-714 Using COP800 Devices to Control DC Stepper Motors 7-174
AN-734 MF2 Compatible Keyboard with COP8 Microcontrollers 7-184
AN-739 RS-232C Interface with COP800 7-204
AN-755 NM95C12 Flexibility in Industrial Control Applications 7-216
AN-758 Using National's MICROWIRE EEPROM 7-227
AN-794 Using an EEPROM-I^{2}C Interface NM24C02/03/04/05/08/09/16/17 7-238
AN-823 Timekeeping Using a COP800 Microcontroller 7-247
AN-824 8-Channel 8-Bit PWM Controller 7-249
AN-841 Software for Interfacing the COP800 Family Microcontrollers to National's MICROWIRE EEPROMs 7-252
AN-871 Selling National's Write Protected "CS" Series EEPROMs to High Volume OEMs 7-258
AN-936 How to Use the NM93C86A Serial EEPROM as a PC/Laptop Detachable Printer File Memory Card (DPFMC) 7-262
AN-952 Low Cost A/D Conversion Using COP800 7-269
AN-953 LCD Triplex Drive with COP820CJ 7-278
AN-982 COP888GW Features and Applications 7-302
AN-983 Simple, Cost Effective A/D Conversion Using COP888EK 7-320
AN-1042 COP8 Instruction Set Performance Evaluation 7-324
AN-1043 Comparison of COP878x to the Enhanced COP8SAx7 Family—Hardware/Software Considerations 7-366
AN-1044 Comparison of COP82xCJ to the Enhanced COP8SAx7 Family—Hardware/Software Considerations 7-371
AN-1048 Replacing Dedicated Protocol Controllers with Code Efficient and Configurable Microcontrollers—Low Speed CAN Network Applications 7-376

Alpha-Numeric Index (Continued)

AN-1049 Using CAN Networking for Cost Effective DC Motor Control in Vehicle Body Electronics 7-381
AN-1050 Understanding and Eliminating EMI in Microcontroller Applications 7-385
COP8 Family 1-3
COP8ACC5 8-Bit Microcontroller with High Resolution A/D Conversion 5-202
COP8ACC7 8-Bit One-Time Programmable (OTP) Microcontroller with High Resolution A/D Conversion 2-314
COP8SAA7 8-Bit One-Time Programmable (OTP) Microcontroller 2-3
COP8SAB7 8-Bit One-Time Programmable (OTP) Microcontroller 2-3
COP8SAC7 8-Bit One-Time Programmable (OTP) Microcontroller 2-3
COP87L20CJ 8-Bit One-Time Programmable (OTP) Microcontroller with Multi-Input Wake-Up and Brown Out Detector 2-79
COP87L22CJ 8-Bit One-Time Programmable (OTP) Microcontroller with Multi-Input Wake-Up and Brown Out Detector 2-79
COP87L40CJ 8-Bit One-Time Programmable (OTP) Microcontroller with Multi-Input Wake-Up and Brown Out Detector 2-105
COP87L40RJ 8-Bit One-Time Programmable (OTP) Microcontroller with 32 Kbytes of Program Memory 3-3
COP87L42CJ 8-Bit One-Time Programmable (OTP) Microcontroller with Multi-Input Wake-Up and Brown Out Detector 2-105
COP87L42RJ 8-Bit One-Time Programmable (OTP) Microcontroller with 32 Kbytes of Program Memory 3-3
COP87L84BC 8-Bit One-Time Programmable (OTP) Microcontroller with CAN Interface 2-132
COP87L84CF 8-Bit One-Time Programmable (OTP) Microcontroller with A/D Converter 2-282
COP87L84CL 8-Bit One-Time Programmable (OTP) Microcontroller 2-252
COP87L84EG 8-Bit One-Time Programmable (OTP) Microcontroller with UART and Three Multi-Function Timers 2-381
COP87L84EK 8-Bit One-Time Programmable (OTP) Microcontroller with Analog Function Block 2-349
COP87L84FH 8-Bit One-Time Programmable (OTP) Microcontroller with UART and Three Multi-Function Timers and Multiply/Divide Block 2-418
COP87L84RG 8-Bit One-Time Programmable (OTP) Microcontroller with 32 Kbytes of Program Memory 3-98
COP87L84RK 8-Bit One-Time Programmable (OTP) Microcontroller with Analog Function Block and 32 Kbytes of Program Memory 3-29
COP87L88CF 8-Bit One-Time Programmable (OTP) Microcontroller with A/D Converter 2-282
COP87L88CL 8-Bit One-Time Programmable (OTP) Microcontroller 2-252
COP87L88EB 8-Bit One-Time Programmable (OTP) Microcontroller with CAN Interface, A/D and UART 2-183
COP87L88EG 8-Bit One-Time Programmable (OTP) Microcontroller with UART and Three Multi-Function Timers 2-381
COP87L88EK 8-Bit One-Time Programmable (OTP) Microcontroller with Analog Function Block 2-349
COP87L88FH 8-Bit One-Time Programmable (OTP) Microcontroller with UART and Three Multi-Function Timers and Multiply/Divide Block 2-418
COP87L88GD 8-Bit One-Time Programmable (OTP) Microcontroller with A/D Converter 2-497
COP87L88GG 8-Bit One-Time Programmable (OTP) Microcontroller with UART and Three Multi-Function Timers 2-458
COP87L88KG 8-Bit One-Time Programmable (OTP) Microcontroller with UART and Three Multi-Function Timers 2-499

Alpha-Numeric Index (Continued)

COP87L88RD 8-Bit One-Time Programmable (OTP) Microcontroller with A/D Converter and 32 Kbytes of Program Memory ... 3-135
COP87L88RG 8-Bit One-Time Programmable (OTP) Microcontroller with 32 Kbytes of Program Memory ... 3-61
COP87L88RK 8-Bit One-Time Programmable (OTP) Microcontroller with Analog Function Block and 32 Kbytes of Program Memory ... 3-29
COP87L88RW 8-Bit One-Time Programmable (OTP) Microcontroller with Pulse Train Generators and Capture Modules ... 3-137
COP87L89EB 8-Bit One-Time Programmable (OTP) Microcontroller with CAN Interface, A/D and UART ... 2-183
COP472-3 Liquid Crystal Display Controller ... 6-12
COP620C 8-Bit Microcontroller ... 4-24
COP622C 8-Bit Microcontroller ... 4-24
COP640C 8-Bit Microcontroller ... 4-24
COP642C 8-Bit Microcontroller ... 4-24
COP680C Microcontroller ... 4-106
COP681C Microcontroller ... 4-106
COP682C Microcontroller ... 4-106
COP684BC 8-Bit Microcontroller with CAN Interface ... 5-3
COP684CL 8-Bit Microcontroller ... 5-130
COP684CS 8-Bit Microcontroller with UART and One Multi-Function Timer ... 5-319
COP684EG 8-Bit Microcontroller with UART and Three Multi-Function Timers ... 5-362
COP684EK 8-Bit Microcontroller with Analog Function Block ... 5-240
COP684FH 8-Bit Microcontroller with UART, Three Multi-Function Timers and Multiply/Divide Block ... 5-406
COP688CL 8-Bit Microcontroller ... 5-130
COP688CS 8-Bit Microcontroller with UART and One Multi-Function Timer ... 5-319
COP688EB 8-Bit Microcontroller with CAN Interface, A/D, and UART ... 5-59
COP688EG 8-Bit Microcontroller with UART and Three Multi-Function Timers ... 5-362
COP688EK 8-Bit Microcontroller with Analog Function Block ... 5-240
COP688FH 8-Bit Microcontroller with UART, Three Multi-Function Timers and Multiply/Divide Block ... 5-406
COP688GD 8-Bit Microcontroller with A/D Converter ... 5-279
COP688GG 8-Bit Microcontroller with UART, and Three Multi-Function Timers ... 5-449
COP688HG 8-Bit Microcontroller with UART, and Three Multi-Function Timers ... 5-492
COP688KG 8-Bit Microcontroller with UART, and Three Multi-Function Timers ... 5-535
COP689EB 8-Bit Microcontroller with CAN Interface, A/D, and UART ... 5-59
COP820C 8-Bit Microcontroller ... 4-24
COP820CJ 8-Bit Microcontroller with Multi-Input Wake-Up and Brown Out Detector ... 4-47
COP822C 8-Bit Microcontroller ... 4-24
COP822CJ 8-Bit Microcontroller with Multi-Input Wake-Up and Brown Out Detector ... 4-47
COP823CJ 8-Bit Microcontroller with Multi-Input Wake-Up and Brown Out Detector ... 4-47
COP840C 8-Bit Microcontroller ... 4-24
COP840CJ 8-Bit Microcontroller with Multi-Input Wake-Up and Brown Out Detector ... 4-75
COP842C 8-Bit Microcontroller ... 4-24
COP842CJ 8-Bit Microcontroller with Multi-Input Wake-Up and Brown Out Detector ... 4-75
COP880C Microcontroller ... 4-106
COP881C Microcontroller ... 4-106
COP882C Microcontroller ... 4-106
COP884BC 8-Bit Microcontroller with CAN Interface ... 5-3
COP884CF 8-Bit CMOS Microcontroller with A/D Converter ... 5-167

Alpha-Numeric Index (Continued)

COP884CG 8-Bit Microcontroller with UART and Three Multi-Function Timers 5-281
COP884CL 8-Bit Microcontroller 5-130
COP884CS 8-Bit Microcontroller with UART and One Multi-Function Timer 5-319
COP884EG 8-Bit Microcontroller with UART and Three Multi-Function Timers 5-362
COP884EK 8-Bit Microcontroller with Analog Function Block 5-240
COP884FH 8-Bit Microcontroller with UART, Three Multi-Function Timers and Multiply/Divide Block 5-406
COP888CF 8-Bit CMOS Microcontroller with A/D Converter 5-167
COP888CG 8-Bit Microcontroller with UART and Three Multi-Function Timers 5-281
COP888CL 8-Bit Microcontroller 5-130
COP888CS 8-Bit Microcontroller with UART and One Multi-Function Timer 5-319
COP888EB 8-Bit Microcontroller with CAN Interface, A/D, and UART 5-59
COP888EG 8-Bit Microcontroller with UART and Three Multi-Function Timers 5-362
COP888EK 8-Bit Microcontroller with Analog Function Block 5-240
COP888FH 8-Bit Microcontroller with UART, Three Multi-Function Timers and Multiply/Divide Block 5-406
COP888GD 8-Bit Microcontroller with A/D Converter 5-279
COP888GG 8-Bit Microcontroller with UART, and Three Multi-Function Timers 5-449
COP888GW 8-Bit Microcontroller with Pulse Train Generators and Capture Modules 5-576
COP888HG 8-Bit Microcontroller with UART, and Three Multi-Function Timers 5-492
COP888KG 8-Bit Microcontroller with UART, and Three Multi-Function Timers 5-535
COP889EB 8-Bit Microcontroller with CAN Interface, A/D, and UART 5-59
COP912C 8-Bit Microcontroller 4-3
COP912CH 8-Bit Microcontroller 4-3
COP920C 8-Bit Microcontroller 4-24
COP922C 8-Bit Microcontroller 4-24
COP940C 8-Bit Microcontroller 4-24
COP940CJ 8-Bit Microcontroller with Multi-Input Wake-Up and Brown Out Detector 4-75
COP942C 8-Bit Microcontroller 4-24
COP942CJ 8-Bit Microcontroller with Multi-Input Wake-Up and Brown Out Detector 4-75
COP980C Microcontroller 4-106
COP981C Microcontroller 4-106
COP982C Microcontroller 4-106
COP984CF 8-Bit CMOS Microcontroller with A/D Converter 5-167
COP984CL 8-Bit Microcontroller 5-130
COP984CS 8-Bit Microcontroller with UART and One Multi-Function Timer 5-319
COP984EG 8-Bit Microcontroller with UART and Three Multi-Function Timers 5-362
COP984EK 8-Bit Microcontroller with Analog Function Block 5-240
COP984FH 8-Bit Microcontroller with UART, Three Multi-Function Timers and Multiply/Divide Block 5-406
COP988CF 8-Bit CMOS Microcontroller with A/D Converter 5-167
COP988CL 8-Bit Microcontroller 5-130
COP988CS 8-Bit Microcontroller with UART and One Multi-Function Timer 5-319
COP988EG 8-Bit Microcontroller with UART and Three Multi-Function Timers 5-362
COP988EK 8-Bit Microcontroller with Analog Function Block 5-240
COP988FH 8-Bit Microcontroller with UART, Three Multi-Function Timers and Multiply/Divide Block 5-406
COP988GD 8-Bit Microcontroller with A/D Converter 5-279
COP8780C 8-Bit One-Time Programmable (OTP) Microcontroller 2-58
COP8781C 8-Bit One-Time Programmable (OTP) Microcontroller 2-58
COP8782C 8-Bit One-Time Programmable (OTP) Microcontroller 2-58

Alpha-Numeric Index (Continued)

LM1971 μPot Digitally Controlled 62 dB Audio Attenuator with Mute 6-36
LM1972 μPot 2-Channel 78 dB Audio Attenuator with Mute 6-37
LM1973 μPot 3-Channel 76 dB Audio Attenuator with Mute 6-38
LMC835 Digitally-Controlled Graphic Equalizer 6-35
LMC1982 Digitally-Controlled Stereo Tone and Volume Circuit with Two Selectable Stereo Inputs 6-32
LMC1983 Digitally-Controlled Stereo Tone and Volume Circuit with Three Selectable Stereo Inputs 6-33
LMC1992 Digitally-Controlled Stereo Tone and Volume Circuit with Four-Channel Input-Selector 6-34
MICROWIRE/PLUS: Serial Interface 6-3
MM5450 LED Display Driver 6-18
MM5451 LED Display Driver 6-18
MM5483 Liquid Crystal Display Driver 6-19
MM5484 16-Segment LED Display Drivers 6-20
MM5486 LED Display Driver 6-21
MM58241 High Voltage Display Driver 6-22
MM58341 High Voltage Display Driver 6-23
MM58342 High Voltage Display Driver 6-24
NM93C06 256-Bit Serial EEPROM (MICROWIRE Bus Interface) 6-26
NM93C13 256-Bit Serial EEPROM 6-25
NM93C14 1024-Bit Serial EEPROM 6-25
NM93C46 1024-Bit Serial EEPROM (MICROWIRE Bus Interface) 6-26
NM93C46A 1024-Bit Serial Interface, Standard Voltage CMOS EEPROM 6-27
NM93C56 2048-Bit Serial EEPROM (MICROWIRE Bus Interface) 6-26
NM93C56A 2048-Bit Serial Interface, Standard Voltage CMOS EEPROM (MICROWIRE Bus Interface) 6-28
NM93C66 4096-Bit Serial EEPROM (MICROWIRE Bus Interface) 6-26
NM93C66A 4096-Bit Serial Interface, Standard Voltage CMOS EEPROM (MICROWIRE Bus Interface) 6-29
NM93C86A 16,384-Bit Serial Interface CMOS EEPROM (MICROWIRE Synchronous Bus) 6-30
NM93CS06 (MICROWIRE Bus Interface) 256-Bit Serial EEPROM with Data Protect and Sequential Read 6-31
NM93CS46 (MICROWIRE Bus Interface) 1024-Bit Serial EEPROM with Data Protect and Sequential Read 6-31
NM93CS56 (MICROWIRE Bus Interface) 2048-Bit Serial EEPROM with Data Protect and Sequential Read 6-31
NM93CS66 (MICROWIRE Bus Interface) 4096-Bit Serial EEPROM with Data Protect and Sequential Read 6-31

Section 1

COP8™ Family

Section 1 Contents

COP8 Family 1-3

National Semiconductor

The 8-Bit COP8™ Family: Optimized for Value

CPU/Instruction Set Features

- 1 μs instruction cycle time
- Up to 14 multi-sourced vectored interrupt servicing each interrupt source with independent vector
- Versatile and easy to use instruction set
- 8-bit stack pointer (SP)—(Stack in RAM)
- Two 8-bit register indirect memory pointers (B,X)
- High code efficiency with majority of instructions being single byte/single cycle (77%)
- True bit manipulation
- BCD arithmatic instructions

Peripheral Features

- On-chip ROM from 768 bytes to 24k bytes
- On-chip OTP EPROM up to 32k bytes
- On-chip RAM from 64 bytes to 1k byte
- Up to three 16-bit multi-function timers, with two 16-bit registers supporting
  - — Processor independent PWM mode
  - — External event counter mode
  - — Input capture mode
- Idle timer
- Multi-Input Wakeup (up to 8 wakeup pins) with optional interrupts (8)
- Full duplex UART
- CAN interface
- A/D (8-bit, 8 channel)
- Analog function block
- Analog comparators
- WATCHDOG™ and clock monitor logic
- MICROWIRE/PLUS™ serial I/O
- Brown out detection
- Power-On-Reset
- Multiply/Divide function

I/O Features

- Memory mapped I/O
- Software selectable I/O
  - — TRI-STATE® outputs
  - — Push-Pull outputs
  - — Weak Pull-Up input
  - — High impedance input
- Schmitt trigger inputs
- High current outputs
- Pin efficient (i.e., 40 pins in 44-pin package are devoted to useful I/O)
- Packages: 16-pin to 68-pin packages

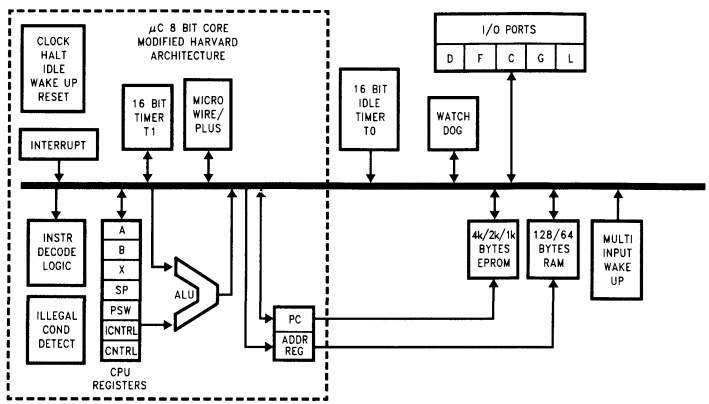
Fully Static CMOS

- M²CMOS™ fabrication
- Low current drain (typically < 1 μA)
- Two power saving modes: HALT and IDLE
- Single supply operation: 2.3V to 6.0V
- Temperature ranges: 0°C to +70°C, −40°C to +85°C, −55°C to +125°C

Development Support

- Emulation and OTP devices
- Real time emulation and full program debug offered by MetaLink Development System

An Example of COP888 Block Diagram (COP8SAx7)



TL/XX/0073–3

Embedded Control: Practical Solutions to Real Problems

Microcontrollers have played an important role in the semiconductor industry for quite some time. Unlike microprocessors, which typically address a range of more computation intensive, general purpose applications, microcontrollers are based on a central processing unit, data memory and input/output circuitry that are designed primarily for specific, single function applications.

During the 1970s, microcontrollers were initially used in simple applications such as calculators and digital watches. But the combination of decreasing costs and increasing integration and performance has created many new application opportunities over the years. Even as the bulk of application growth occurs in the 8-bit arena, the same issues that system designers were concerned with in the 4-bit world continue in force today. These include cost/performance tradeoffs, low power and low voltage capabilities, time to market, space/pin efficiency, and ease of design.

- **Cost/Performance.** A price difference of just a few pennies can be the gating factor in today's 8-bit design decisions. Manufacturers must offer a wide range of cost/performance options in order to meet customer demands.
- **Low Power and Low Voltage.** The increasing range of mobile and/or battery-powered applications is placing a premium on low-power, low-voltage, CMOS and BiCMOS embedded control solutions.
- **Time to Market.** The microcontroller's architecture, functionality, and feature set have a major influence on product design cycles in today's competitive market, with its shrinking windows of opportunity.
- **Space/Pin Efficiency.** Real estate and board configuration considerations demand maximum space and I/O pin efficiency, particularly given today's high integration and small product form factors.
- **Ease of Design.** A familiar and easy to use application design environment—including complete development tool support—is one of the driving factors affecting today's 8-bit microcontroller design decisions.

All of these issues must be considered when searching for the appropriate 8-bit microcontroller to meet specific application needs.

National Semiconductor has been a prominent player in the worldwide microcontroller market, and its COP8 family of products spans today's range of applications, providing customers with a wealth of options at every price/performance point in the 8-bit microcontroller market.

Designers can select from a variety of building blocks centered around a common memory-mapped core and modified Harvard architecture. These building blocks include ROM, RAM, user programmable memory, UART, comparator, A/D, and I/O functions.

The COP8 family incorporates 1 μs instruction cycle times, watchdog and clock monitors, multi-input wake up circuitry and National's MICROWIRE/PLUS interface. In addition, National's COP8 microcontrollers are available in a wide variety of temperature range configurations from $-55°C$ through $+125°C$—optimizing them for rugged industrial and military applications.

COP8 Benefits

The COP8 family provides designers with a number of features that result in substantial benefits. These include a code-efficient instruction set, low power/voltage features, efficient I/O, a flexible and configurable design methodology, robust design tools, and electromagnetic interference (EMI) control.

The COP8 family's compact, efficient and easy-to-program instruction set enables designers to reduce time to market for their products. Thanks to the instruction set, efficient ROM utilization lowers costs while providing the opportunity to integrate additional functionality on-chip. Low voltage operation, low current drain, multi-input wakeup and several power saving modes reduce power consumption for today's increasing range of handheld, battery-driven applications. And an array of user-friendly development tools—including hardware from MetaLink, and state-of-the-industry assemblers, and C compilers help design engineers save valuable development time.

National's Configurable Controller Methodology (CCM) for the COP8 family creates "whole products" that are bug-free, fully tested and characterized, and supported by a range of documentation and hardware/software tools. National developed CCM because the majority of customer requests for new products have typically called for reconfigurations of existing proven blocks—such as RAM, ROM, timers, comparators, UARTs, and I/O.

EMI Reduction Technology

COP8 products incorporate circuitry that guards against electromagnetic interference—an increasing problem in todays microcontroller board designs. National's patented EMI reduction technology offers low EMI clock circuitry, EMI-optimized pinouts, gradual turn-on outputs (GTO), and on-chip choke device to help customers circumvent many of the EMI issues influencing embedded control designs.

Core Architecture

All COP8 devices use a modified Harvard architecture, which means that program memory and data memory are accessed separately using independent address/data buses. This type of architecture offers the advantage of faster operation because the next instruction can be fetched from program memory while the current data memory transfer operation is carried out. The COP8 architecture is a "modified" Harvard version because data tables can be accessed from program memory by using a special instruction, Load Accumulator Indirect (LAID).

The core CPU has an 8-bit accumulator (A), a 16-bit program counter (PC), two 8-bit data pointers (B and X), and 8-bit stack pointer (SP), and 8-bit processor status word (PSW), an 8-bit control register (CNTRL), and a bank of general-purpose 8-bit registers. All RAM, I/O ports, and registers (except for the accumulator and program counter) are mapped into the data memory address space.

The COP8 device communicates with other devices through several configurable I/O ports or through the MICROWIRE/PLUS serial I/O interface. The I/O ports are designated by letter names such as Port C, Port D, Port G, Port I, and Port L. The number of ports and port pins vary with the device type and package type.

All COP8 devices have at least one 16-bit, general-purpose timer that can be programmed to operate in any of three modes: Pulse Width Modulation (PWM), external event counter, or input capture mode. Many COP8 devices have two or more of these timers and/or special-purpose timers such as the IDLE mode timer.

Peripheral Blocks

Several different on-chip peripheral devices are available in different COP8 devices, with multiple peripherals available in some versions of COP8 devices. Some of the peripheral blocks available are:

- Comparator
- Analog-to-Digital Converter
- Universal Asynchronous Receiver/Transmitter (UART)
- Controller Area Network (CAN) Interface
- Hardware Multiply/Divide

Typically, the inputs and outputs of the peripherals are "alternate functions" of the programmable I/O ports of the COP8 device. In other words, the port pins can be programmed to operate as general-purpose inputs and outputs or as special-purpose inputs and outputs for the supported peripheral device.

Instruction Set

The COP8 offers a powerful and efficient instruction set. Most instructions are one byte long and take one instruction cycle to execute, resulting in compact, efficient programs. Several single-byte instructions are available that carry out multiple operations. For example, the single-byte DRSZ instruction decrements a specified register and skips the next instruction if the result is zero.

The instruction set offers a variety of addressing modes. For reading or writing data, the device offers the following addressing modes: direct, register B or X indirect, register B or X indirect with post-incrementing/decrementing, immediate, immediate short, and indirect from program memory. For transfer of program control, the device offers the following addressing modes: jump relative, jump absolute, jump absolute long, and jump indirect.

The COP8 allows any individual bit in data memory address space to be set, reset, and tested, including bits in the memory-mapped I/O ports and registers.

COP8 Families

The COP8 line of 8-bit microcontrollers is divided into two families called the "Basic Family" and "Feature Family." The Basic Family members are for lower-end, lower-cost applications that require less memory and simpler peripheral devices; whereas the Feature Family members are for applications that require more memory and more-advanced peripheral devices. However, both families share the same basic architecture and basic instruction set.

Basic Family members have from 768 bytes to 4k bytes of ROM and 64 to 128 bytes of RAM; and one 16-bit timer. A HALT mode is available to shut down the device during periods of inactivity. Devices typically have 20 or 28 pins. Simple peripheral devices such as a comparator are offered in the family.

Feature Family members have from 2k to 24k bytes of ROM and 128 to 1088 bytes of RAM; and at least two 16-bit timers. The Feature Family instruction set offers nine additional instructions to support vectored interrupts, pushing/popping the stack, and additional types of logic operations. In addition to the HALT mode, another power-down mode is available called the IDLE mode, which allows certain time-monitoring sections to operate while the rest of the device is shut down. All Feature Family members offer Multi-Input Wakeup, which provides separate inputs for edge-triggered maskable interrupts or for exiting from the HALT or IDLE mode. Devices typically have 28, 40, or 44 pins. Advanced peripheral devices such as an A/D Converter, UART, and/or CAN Interface are offered in the family.

One-Time Programmable (OTP) Devices

All COP8 devices are available in One-Time Programmable (OTP) form. OTP devices field-programmable using standard PROM programming equipment. They are useful not only for prototypes and limited-production runs, but also for normal production as well, because the additional cost over mask-programmed devices is reasonably small in many cases. For the shortest time to market and for the ability to quickly make revisions or correct bugs, OTP COP8 devices offer an attractive choice for final production.

OTP COP8 devices offer low-voltage operation and program memory security. Security is achieved by programming a bit in the device that makes the program memory unreadable from outside the chip.

Some COP8 devices are available in OTP form with an expanded memory of up to 32k bytes of program memory at a reasonable additional cost. These devices are useful for software-intensive applications when time-to-market is critical, because the software can be developed without concern for memory size constraints. Manufacturing costs can be reduced at a later time by optimizing the software to use less memory.

COP8SAx7 OTP Devices

A recent addition to the COP8 Feature Family are the COP8SAx7 OTP devices: the COP8SAA7, COP8SAB7, and COP8SAC7. These devices offer an unusual combination of OTP memory, low price, and a rich set of features:

- Low cost 8-bit OTP microcontroller
- OTP program space with read/write protection
- Quiet Design (low radiated emissions)
- Multi-Input Wakeup pins with optional interrupts (4 to 8 pins)
- 8 bytes of user storage space in EPROM
- User selectable clock options
  - —Crystal/Resonator oscillator
  - —External oscillator
  - —Internal R/C oscillator
- Internal Power-On Reset—user selectable
- WATCHDOG and Clock Monitor Logic—user selectable
- Up to 12 high current outputs

COP8SAx7 OTP Devices (Continued)

Device	EPROM	RAM	Package and I/O	
			Package Types	Number of I/O
COP8SAC7	4k	128	20 DIP/SO	16
			28 DIP/SO	24
			40 DIP	36
			44 PLCC/PQFP	40
COP8SAB7	2k	128	20 DIP/SO	16
			28 DIP/SO	24
COP8SAA7	1k	64	16 DIP/SO	12
			20 DIP/SO	16
			28 DIP/SO	24

COP8 Solutions

There are many reasons that National's COP8 families offer the best solutions to design challenges in the worldwide 8-bit microcontroller market: a code-efficient instruction set, low-power operation, I/O pin efficiency, and a "whole product" philosophy that includes superior development tools, documentation, and support. As that market continues to expand, National will continue its microcontroller technology research and development efforts to bring to the market the most advanced, cost-effective solutions.

COP8 Features/Benefits Analysis

	Key Features	Benefits
Instruction Set	• Efficient Instruction Set (77% Single Byte/Single Cycle) • Easy To Program • Compact Instruction Set • Multi Function Instructions • Ten Addressing Modes	• Efficient ROM Utilization (compact code) • Low Cost Microcontroller (small ROM size) • Fast Time To Market
Low Power	• Low Voltage Operation • Lower Current Drain • Multi-Input Wakeup • Power Savings Modes (HALT/IDLE)	• Lower Power Consumption for Hand Held Battery Driven Applications
Efficient I/O	• Software Programmable I/O • Efficient Pin Utilization • Breadth of Available Packages • Package Types Including Variety of Low Pin Count Devices • High Current Outputs • Schmitt Trigger Inputs	• Multiple Use of I/O Pins • Economical Use of External Components (lower system cost) • Cleaner Hardware Design • Choice of Optimum Package Type (price/outline/pinout)
Flexible/Powerful On-Board Features	• Smart 16-Bit Timers (processor independent PWM) • A/D • Comparators • Analog Function Block (low cost A/D) • UART • Multi-Input Wakeup • Multi-Source Hardware Interrupts • MICROWIRE/PLUS Serial Interface • Application Specific Features (CAN, Motor Control Timers, etc.)	• Timers Allow Less Software/Process Overhead for Frequency • Measurement (capture) and PWM • Cleaner Hardware (eliminating the need for external components) • Overall Cost Reduction
Safety/Software-Runaway Protection	• WATCHDOG • Software Interrupt • Clock Monitor • Brown Out Detection	• No Need for External Protection Circuitry • Brown Out Detection Allows the Use of Low Cost Power Supply
Development Tools	Hardware: • New, User Friendly, Development Tool Hardware from MetaLink • Low Cost Version of the Development Tool (Debug Module) • Various Third Party Programmers for Programming OTPs Software: • New, User Friendly Assembler, a C Compiler and a "Fuzzy" Logic Design Environment	• Saves Engineering Development Time—Fast Time to Market

COP8 Features/Applications Matrix

Market Segment		Applications	Applications Features/Functions	Microcontroller Features Required	Appropriate COP8 Devices
Consumer	Children Toys and Games	Basketball/Baseball Games Children Electronic Toys Darts Throws Juke Box Pinball Laser Gun	Battery Driven Replacing Discrete with Low Cost Driving Piezo/Speaker/LEDs Directly Very Cost Sensitive	Very Low Price Low Power Consumption Wide Voltage Range High Current Outputs Small Packages	COP912C COP920C/COP922C
	Electronic Audio Items	Audio Greeting Cards Electronic Musical Equipment	Battery Driven Tone Generation Low Power	Wide Voltage Range Low Power Consumption Efficient Table Lookup Flexible Timer	COP912C COP820C/840C/880C
	Electronic Appliances/ Tools	Small Appliances: Irons Coffee Makers Digital Scales Microwave Ovens Cookers Food Processors Blenders	Low Cost Power Supply Temp Measurement Safety Features Noise Immunity Driving LEDs/Relays/Heating Elements	Brown Out Detection On-Board Comparator High Current Outputs Watchdog/Software Interrupt Schmtt Trigger Inputs 16-Bit PWM Timer	COP820/840 COP820CJ/840CJ Family
		Household Appliances: Oven Control Dishwasher Washing Machine/Dryer Vacuum Cleaner Electronic Heater Electronic Home Control (Doorbell, Light Dimmer, Climate) Sewing Machine	Rely on Hard-Wire Relay Circuits, Timers, Counters, Mechanical Sequence Controllers Temp Control Noise Immunity Safety Features Timing Control Main Driven	Brown Out Detection On-Board Comparator On-Board A/D Watchdog/Soft Interrupt Schmitt Trigger Inputs Flexible Timers PWM Outputs High Current Outputs Safety Features	COP820CJ/840CJ (on-board comparator) COP888CF (on-board A/D)
	Portable/ Handheld/ Battery Powered	Scales Multimeters (portable) Electronic Key Laptop/Notebook Keyboard Mouse Garage Door Opener TV/Electronic Remote Control Portable PRP or Retail Pos Device Jogging Monitor Smart Cards	Battery Driven Minimal Power Consumption Low Voltage Sensing Measurement Standby Mode Flexible Package Offerings Small Physical Size	Low Voltage Operation Low Power Consumption Wide Voltage Range Power Saving Modes Multi-Input Wakeup On-Board Comparator Small Packages	COP820CJ/840CJ COP840/COP880 COP888CL (Keyboards)
Personal Communications		Cordless Phone (base/handset) Phone Dialer Answering Machine Feature Phone PBX Card CB Radios/Digital Tuners Cable Converter	Low Power Timing Serial Interfaces Low Voltage Tone Dialing Battery Saving Functions Small Physical Size	Low Current Drain Low Voltage Operation Standby Mode UART Serial Synchronous Interface 16-Bit Timers Schmitt Trigger Inputs LED Direct Drive Sufficient I/O in Small Packages	Cordless Phone: COP840/COP880 Feature Phone PBX Card: COP888CG/COP888EG Others: Generic COP8 Devices

COP8 Features/Applications Matrix (Continued)

Market Segment		Applications	Applications Features/Functions	Microcontroller Features Required	Appropriate COP8 Devices
Medical	Monitors	Thermometer Pressure Monitors Various Portable Monitors	Battery Driven Sensing/Measurement Data Transmission Low Power Low Voltage	On-Board Comparator (low cost A/D) 16-Bit Timer Low Power Consumption Low Voltage Operation	COP820CJ/840CJ (on-board comparator) COP840/COP880 COP888CL
	Medical Equipment	Bed-Side Pump/Timers Ultrasonic Imaging System Analyzers (chemical, data) Electronic Microscopes	Monitoring Data Data Transmission Timing	Serial Interface A/D 16-Bit Timers	COP888CS COP888CF COP888CG/COP888EG
Industrial	Motion Control	Motor Control Power Tools	Motor Speed Control Noisy Environment Timing Control	Flexible PWM Timers Schmitt Trigger Inputs High Current Outputs	COP820/COP840 COP888CL
	Security/ Monitoring System	Security Systems Burglar Alarms Remote Data Monitoring Systems Emergency Control Systems Security Switches	Data Transmission Monitoring (scan inputs from sensors) Keypad Scan Timing Diagnostic Data Monitoring Drive Alarm Sounders Interface to Phone System Standby Mode	UART Flexible 16-Bit PWM Timers Flexible I/O Single Slop A/D Capability Power Saving Modes (HALT, Multi-Input wakeup) Serial Synchronous Interface	Basic Systems: COP840/COP880, COP888CL (Multi-Input wakeup) More Involved Systems: COP888CS/COP888CG/COP888FH COP888EK (muxed analog inputs, constant current source)
	Misc.	Switch Controls (elevator, traffic, power switches) Sensing Control Systems/Displays Pressure Control (scales) Metering (utility, monetary, industrial) Lawn Sprinkler/Lawn Mowers Taxi Meter Coin Controls Industrial Timers Temperature Meters Gas Pump Gas/Smoke Detectors	Timing/Counting Sensing Measurement	Generic Microcontroller	Generic COP8 Microcontroller: COP820/COP840/COP880
Automotive		Radio/Tape Deck Controls Window/Seat/Mirror/Door/ Controls Heat/Climate/Controls Headlight/Antenna Power Steering Anti Theft Slave Controllers	Timing Motion Control Display Control Soft Runaway/Trap Recovery (safety considerations) EMI/Noise Immunity Serial Interfaces Standby Modes Wide Temp Range	Flexible PWM Timers Power Saving Modes Multi-Input Wakeup WATCHDOG Software Trap UART CAN Interface Special Features for Dashboard Control (counters, capture modules, MUL/DIV) Reduced EMI Wide Temp Range	Radio/Climate Control: COP888EK/888CG/888EG/888GG/888HG Seat/Motional Control, Slave Controller: COP884BC Dashboard Control: COP888GW Mirror Control, etc.: COP8 Basic Family Climate Control: COP888CF

COP8 8-Bit Microcontroller Selection Guide

	Mask ROM				Memory		I/O		Packaging					Power			Timers			
In '96 Data Book	Commercial[1] 0°C + 70°C	Industrial −40°C + 85°C	Military −40°C + 125°C	Instruction Cycle	OTP EPROM	RAM	Total I/O[3]	High Sink Current (10 mA–15 mA)	Pins	DIP	SO	PLCC	PQFP†	HALT	IDLE	Brown-Out	Normal[4]	High Speed[5]	PWM Outputs[6]	Idle Timer
	Prefix Below			µs	Bytes			Pins	Qty.	Dice Sales Available										
	COP8SAx7 OTP Family																			
	COP	COP	COP																	
New	8SAA716N9	8SAA716N8	8SAA716N7	1	1k	64	12	4	16	✓				✓	✓		1		1	✓
New	8SAA716M9	8SAA716M8	8SAA716M7	1	1k	64	12	4	16		✓			✓	✓		1		1	✓
New	8SAA720N9	8SAA720N8	8SAA720N7	1	1k	64	16	4	20	✓				✓	✓		1		1	✓
New	8SAA720M9	8SAA720M8	8SAA720M7	1	1k	64	16	4	20		✓			✓	✓		1		1	✓
New	8SAA728N9	8SAA728N8	8SAA728N7	1	1k	64	24	4	28	✓				✓	✓		1		1	✓
New	8SAA728M9	8SAA728M8	8SAA728M7	1	1k	64	24	4	28		✓			✓	✓		1		1	✓
New	8SAB720N9	8SAB720N8	8SAB720N7	1	2k	128	16	4	20	✓				✓	✓		1		1	✓
New	8SAB720M9	8SAB720M8	8SAB720M7	1	2k	128	16	4	20		✓			✓	✓		1		1	✓
New	8SAB728N9	8SAB728N8	8SAB728N7	1	2k	128	24	4	28	✓				✓	✓		1		1	✓
New	8SAB728M9	8SAB728M8	8SAB728M7	1	2k	128	24	4	28		✓			✓	✓		1		1	✓
New	8SAC720N9	8SAC720N8	8SAC720N7	1	4k	128	16	4	20	✓				✓	✓		1		1	✓
New	8SAC720M9	8SAC720M8	8SAC720M7	1	4k	128	16	4	20		✓			✓	✓		1		1	✓
New	8SAC728N9	8SAC728N8	8SAC728N7	1	4k	128	24	8	28	✓				✓	✓		1		1	✓
New	8SAC728M9	8SAC728M8	8SAC728M7	1	4k	128	24	8	28		✓			✓	✓		1		1	✓
New	8SAC740N9	8SAC740N8	8SAC740N7	1	4k	128	36	12	40	✓				✓	✓		1		1	✓
New	8SAC744V9	8SAC744V8	8SAC744V7	1	4k	128	40	12	44			✓		✓	✓		1		1	✓
New	8SAC7VEJ9	8SAC7VEJ8	8SAC7VEJ7	1	4k	128	40	12	44				✓	✓	✓		1		1	✓

COP8 8-Bit Microcontroller Selection Guide (Continued)

Features											Window
Interrupts[7]	Comparators	A/D Converter	UART	MICROWIRE/PLUS	WATCHDOG	Clock Monitor	Multi-Input Wake-Up	Reduced EMI[8]	Additional Features		Windowed DIP[9]
COP8SAx7 OTP Family											
											COP
8				✓	✓	✓	✓	✓	EPROM Security, 8 bytes of user storage space in EPROM, User selectable clock options, on-chip R/C oscillator, Internal Power-On-Reset.		
8				✓	✓	✓	✓	✓			
8				✓	✓	✓	✓	✓			
8				✓	✓	✓	✓	✓			
8				✓	✓	✓	✓	✓			
8				✓	✓	✓	✓	✓			
8				✓	✓	✓	✓	✓			
8				✓	✓	✓	✓	✓			
8				✓	✓	✓	✓	✓			
8				✓	✓	✓	✓	✓			
8				✓	✓	✓	✓	✓			8SAC720Q9
8				✓	✓	✓	✓	✓			
8				✓	✓	✓	✓	✓			8SAC728Q9
8				✓	✓	✓	✓	✓			
8				✓	✓	✓	✓	✓			8SAC740Q9
8				✓	✓	✓	✓	✓			8SAC744Q9
8				✓	✓	✓	✓	✓			

COP8 8-Bit Microcontroller Selection Guide (Continued)

In '96 Data Book	Commercial[1] 0°C + 70°C	Industrial −40°C + 85°C	Military −55°C + 125°C	EPU[10]	Debug Module[10]	Cable Adapter for Debug Module	Probe Card for iceMASTER
Development Tools							
Device Prefix Below							
COP8SAx7 OTP Family							
	COP	**COP**	**COP**				
New	8SAA716N9	8SAA716N8	8SAA716N7	COP8SA-EPU	COP8SA-DM	DM-COP8/16D	COP8SA-IM16N
New	8SAA716M9	8SAA716M8	8SAA716M7	COP8SA-EPU	COP8SA-DM	DM-COP8/16D*	COP8SA-IM16N**
New	8SAA720N9	8SAA720N8	8SAA720N7	COP8SA-EPU	COP8SA-DM	DM-COP8/20D	COP8SA-IM20N
New	8SAA720M9	8SAA720M8	8SAA720M7	COP8SA-EPU	COP8SA-DM	DM-COP8/20D*	COP8SA-IM20N**
New	8SAA728N9	8SAA728N8	8SAA728N7	COP8SA-EPU	COP8SA-DM	DM-COP8/28D	COP8SA-IM28N
New	8SAA728M9	8SAA728M8	8SAA728M7	COP8SA-EPU	COP8SA-DM	DM-COP8/28D*	COP8SA-IM28N**
New	8SAB720N9	8SAB720N8	8SAB720N7	COP8SA-EPU	COP8SA-DM	DM-COP8/20D	COP8SA-IM20N
New	8SAB720M9	8SAB720M8	8SAB720M7	COP8SA-EPU	COP8SA-DM	DM-COP8/20D*	COP8SA-IM20N**
New	8SAB728N9	8SAB728N8	8SAB728N7	COP8SA-EPU	COP8SA-DM	DM-COP8/28D	COP8SA-IM28N
New	8SAB728M9	8SAB728M8	8SAB728M7	COP8SA-EPU	COP8SA-DM	DM-COP8/28D*	COP8SA-IM28N**
New	8SAC720N9	8SAC720N8	8SAC720N7	COP8SA-EPU	COP8SA-DM	DM-COP8/20D	COP8SA-IM20N
New	8SAC720M9	8SAC720M8	8SAC720M7	COP8SA-EPU	COP8SA-DM	DM-COP8/20D*	COP8SA-IM20N**
New	8SAC728N9	8SAC728N8	8SAC728N7	COP8SA-EPU	COP8SA-DM	DM-COP8/28D	COP8SA-IM28N
New	8SAC728M9	8SAC728M8	8SAC728M7	COP8SA-EPU	COP8SA-DM	DM-COP8/28D*	COP8SA-IM28N**
New	8SAC740N9	8SAC740N8	8SAC740N7	COP8SA-EPU	COP8SA-DM	DM-COP8/40D	COP8SA-IM40N
New	8SAC744V9	8SAC744V8	8SAC744V7	COP8SA-EPU	COP8SA-DM	DM-COP8/44P	COP8SA-IM44V
New	8SAC7VEJ9	8SAC7VEJ8	8SAC7VEJ7	COP8SA-EPU	COP8SA-DM	DM-COP8/44P*	COP8SA-IM44V

* Requires surface mount adapter kit (see Debug Module Surface Mount Adapter Selection Table)

** Requires surface mount adapter kit (see iceMASTER Probe Card Surface Mount Adapter Selection Table)

COP8 Family Selection Guide (Continued)

Debug Module Surface Mount Adapter Selection Table

MHW-COP8/44P-Q	44 PLCC to 44 PQFP
DM-COP8/28D-SO	28 DIP to 28 SO
DM-COP8/20D-SO	20 DIP to 20 SO
DM-COP8/16D-SO	16 DIP to 16 SO
Optional Programming Adapter	
COP8-PGMA-44Q	44 PQFP

iceMASTER Probe Card Surface Mount Adapter Selection Table

MHW-COP8/44P-Q	44 PLCC to 44 PQFP
MHW-SOIC-28	28 DIP to 28 SO
MHW-SOIC-20	20 DIP to 20 SO
MHW-SOIC-16	16 DIP to 16 SO

Notes: *(Consult specific datasheets for exact information.)*

Note 1: Two versions available. "C" (shown) 2.3V to 4.0V operation; "CH" for 4.0V to 6.0V operation.

Note 2: Price is approximate. Varies with package, temperature range and volume.

Note 3: Bidirectional I/O pins can be bit configured by software as:
- input Hi-Z (TRI-STATE) or input with weak pull-up (internal)
- output push-pull "0" (low) or push-pull "1"(high)

Note 4: 16-bit, clocked at t_C. Can be s/w configured as: 1) PWM, 2) Input capture, 3) External event counter input.

Note 5: 8 or 16 bits, clocked at CKI (100 ns max).

Note 6: Pulsed width modulation 8/16 bits, 1 μs or 100 ns (max) resolution.

Note 7: One (dedicated) external interrupt on basic family devices, 9 external interrupts (1 dedicated, 8 configurable) on feature family devices.

Note 8: Patented EMI reducing circuitry on-chip.

Note 9: While OTP's/windowed devices are functionally identical to masked ROM devices, electrical specifications may differ.

Note 10: Debug Module and EPU have the capability to program all applicable OTP's and windowed devices (not recommended for volume programming).

Note 11: OTP's with up to 32k EPROM are available.

† Contact sales office for availability.

Packaging Options

Package Type	Code	Package Type	Code
Plastic Dual-In-Line Package (DIP)	N	Thin Quad Flat Pack	VEJ
Plastic Leaded Chip Carrier (PLCC)	V	28 Small Outline Footprint—Windowed	MHEA
Small Outline Packages—Wide Body	WM	Lead Chip Carrier	EL
Small Outline Packages—Wide Body	M	Ceramic Windowed DIP	MHD
Singulated Dice	MDC	Ceramic Windowed DIP	J
Dice in Wafer Form	DWF		

COP8 Family Selection Guide

Note: All device numbers are prefixed by indicators at top of part number column. Notes 1 to 11 are on pages 1–17.

	Mask ROM				Memory		I/O		Packaging					Power			Timers			
In '96 Data Book	Commercial[1] 0°C + 70°C	Industrial −40°C + 85°C	Military −55°C + 125°C	Instruction Cycle	ROM	RAM	Total I/O[3]	High Sink Current (10 mA–15 mA)	Pins	DIP	SO	PLCC	PQFP†	HALT	IDLE	Brown-Out	Normal[4]	High Speed[5]	PWM Outputs[6]	Idle Timer
	Prefix Below			µs	Bytes			Pins	Qty.	Dice Sales Available										
Basic Family																				
	COP	COP	COP																	
✓		823CJ		1	1k	64	11	4	16		✓			✓		✓	1	1	2	
✓		822CJ		1	1k	64	15	4	20	✓	✓			✓		✓	1	1	2	
✓		820CJ		1	1k	64	23	4	28	✓	✓			✓		✓	1	1	2	
New	942CJ	842CJ		1	2k	128	15	4	20	✓	✓			✓		✓	1	1	2	
New	940CJ	840CJ		1	2k	128	23	4	28	✓	✓			✓		✓	1	1	2	
✓	912C			2	768	64	15	0	20	✓	✓			✓			1		1	
✓	922C	822C	622C	1	1k	64	15	0	20	✓	✓			✓			1		1	
✓	920C	820C	620C	1	1k	64	23	4	28	✓	✓			✓			1		1	
✓	942C	842C	642C	1	2k	128	15	0	20	✓	✓			✓			1		1	
✓	940C	840C	640C	1	2k	128	23	4	28	✓	✓			✓			1		1	
New	982C	882C	682C	1	4k	128	15	0	20	✓	✓			✓			1		1	
✓	981C	881C	681C	1	4k	128	23	4	28	✓	✓			✓			1		1	
✓	980C	880C	680C	1	4k	128	35	8	40/44	✓		✓		✓			1		1	
Feature Family																				
Dev		888EB	688EB	1	8k	192	31/54	8	44/68			✓		✓	✓		2		2	
✓		884BC	684BC	1	2k	64	18	4	28		✓			✓	✓		1	1	2	✓
✓	984CF	884CF		1	4k	128	23	4	28	✓	✓			✓	✓		2		2	✓
✓	988CF	888CF		1	4k	128	33/37	8	40/44	✓		✓		✓	✓		2		2	✓
Dev	988GD	888GD	688GD	1	16k	256	35/39		40/44	✓		✓		✓	✓					✓
Dev		8ACC5		1	4k	128	16/24	4	20/28	✓	✓			✓	✓	1	1	1	1	
✓		884CG		1	4k	192	23	4	28	✓	✓			✓	✓		3		3	✓
✓		888CG		1	4k	192	35/39	8	40/44	✓		✓		✓	✓		3		3	✓
✓	984EG	884EG	684EG	1	8k	256	23	4	28	✓	✓			✓	✓		3		3	✓
✓	988EG	888EG	688EG	1	8k	256	35/39	8	40/44	✓		✓	✓	✓	✓		3		3	✓
New		888GG		1	16k	512	35/39	8	40/44	✓		✓		✓	✓		3		3	✓
Dev		888HG		1	20k	512	35/39	8	40/44	✓		✓		✓	✓		3		3	✓
Dev		888KG		1	24k	1k	35/39	8	40/44	✓		✓		✓	✓		3		3	✓
Dev		884FH		1	12k	512	23	8	28	✓	✓			✓	✓		3		3	✓
Dev		888FH		1	12k	512	35/39	8	40/44	✓		✓		✓	✓		3		3	✓
✓	984EK	884EK		1	8k	256	23	4	28	✓	✓			✓	✓		3		3	✓
✓	988EK	888EK		1	8k	256	35/39	8	40/44	✓		✓		✓	✓		3		3	✓
✓	984CL	884CL	684CL	1	4k	128	23	4	28	✓	✓			✓	✓		2		2	✓
✓	988CL	888CL	688CL	1	4k	128	33/39	8	40/44	✓		✓		✓	✓		2		2	✓
✓	984CS	884CS	684CS	1	4k	192	23	4	28	✓	✓			✓	✓		1		1	✓
✓	988CS	888CS	688CS	1	4k	192	35/39	8	40/44	✓		✓		✓	✓		1		1	✓
✓		888GW		1	16k	512	56	8	68			✓		✓	✓		2	2	6	✓

COP8 Family Selection Guide (Continued)

Features										OTP	Window
Interrupts[7]	Comparators	A/D Converter	UART	MICROWIRE/PLUS	WATCHDOG	Clock Monitor	Multi-Input Wake-Up	Reduced EMI[8]	Additional Features	One Time Programmable 9 and 11	Windowed DIP[9]
Basic Family											
										COP	COP
3	1				✓		✓				
3	1			✓	✓		✓			87L22CJ	
3	1			✓	✓		✓			87L20CJ	
3	1			✓	✓		✓	✓		87L42CJ	
3	1			✓	✓		✓	✓		87L40CJ	
3				✓					912 - Lowest Price	8782C	8782CJ
3				✓						8782C	8782CJ
3				✓						8781C	8781CJ
3				✓						8782C	8782CJ
3				✓						8781C	8781CJ
3				✓						8782C	8782CJ
3				✓						8781C	8781CJ
3				✓						8780C	8780CJ
Feature Family											
14		✓	1	✓	✓	✓	✓	✓	Can Interface, 8 Channel (8-bit) A/D SAR	87L88EB	
12	2			✓			✓	✓	Can Interface, Power On-Chip Reset	87L84BC	
10		✓		✓	✓	✓	✓		8 Channel (8-bit) A/D	87L84CF†	
10		✓		✓	✓	✓	✓		Successive Approximation	87L88CF†	
12		✓		✓	✓	✓	✓	✓		87L88GD	
9	1	✓		✓	✓	✓	✓	✓	Analog Function Block for Single Slope A/D Function, High Speed Capture Timer	COP8ACC7	
14	2		1	✓	✓	✓	✓	✓		87L84EG	
14	2		1	✓	✓	✓	✓	✓		87L88EG	
14	2		1	✓	✓	✓	✓			87L84EG	
14	2		1	✓	✓	✓	✓			87L88EG	
14	2		1	✓	✓	✓	✓	✓		87L88GG	
14	2		1	✓	✓	✓	✓	✓		87L88GG	
14	2		1	✓	✓	✓	✓	✓		87L88KG	
14	2		1	✓	✓	✓	✓	✓	Hardware Multiply/Divide		
14	2		1	✓	✓	✓	✓	✓	Hardware Multiply/Divide	87L88FH†	
12	1	✓		✓	✓	✓	✓	✓	6 Channel (16-bit) A/D	87L84EK	
12	1	✓		✓	✓	✓	✓	✓	Single Slope (See AN-833)	87L88EK	
10				✓	✓	✓	✓			87L84CL	
10				✓	✓	✓	✓			87L88CL	
12	1		1	✓	✓	✓	✓			87L84EG	
12	1		1	✓	✓	✓	✓			87L88EG	
14			1	✓			✓	✓	Hardware Multiply/Divide	87L88RW	

COP8 Family Selection Guide (Continued)

	Mask ROM			Development Tools			
In '96 Data Book	Commercial[1] 0°C +70°C	Industrial −40°C +85°C	Military −55°C +125°C	EPU[10]	Debug Module[10]	DIP Probe Card for iceMASTER™ 400	PLCC Probe Card for iceMASTER400
	Prefix Below			Device Prefix Below			
Basic Family							
✓	COP	COP	COP	EPU-COP	COP8-DM	MHW	MHW
✓		823CJ			840CJ		
✓		822CJ			840CJ	820CJ20DWPC	
✓		820CJ			840CJ	820CJ28DWPC	
New	942CJ	842CJ			840CJ	840CJ20DWPC	
New	940CJ	840CJ			840CJ	840CJ28DWPC	
✓	912C				880C	880C20DWPC	
✓	922C	822C	622C		880C	880C20DWPC	
✓	920C	820C	620C		880C	880C28DWPC	
✓	942C	842C	642C		880C	880C20DWPC	
✓	940C	840C	640C		880C	880C28DWPC	
New	982C	882C	682C		880C	880C20DWPC	
✓	981C	881C	681C		880C	880C28DWPC	
✓	980C	880C	680C	8780-X	880C	880C40DWPC	880C44PWPC
Feature Family							
Dev		888EB	688EB		888EB		888EB68PWPC
✓		884BC	684BC		884BC†	884BC28D5PC	
✓	984CF	884CF			888CF	884CF28DWPC	
✓	988CF	888CF			888CF	888CF40DWPC	888CF44PWPC
✓		884CG			888GG	884CG28DWPC	
✓		888CG		888GG-X	888GG	888CG40DWPC	888CG44PWPC
✓	984EG	884EG	684EG		888GG	884EG28DWPC	
✓	988EG	888EG	688EG	888GG-X	888GG	888EG40DWPC	888EG44PWPC
New		888GG		888GG-X	888GG†	888GG40DWPC	888GG44PWPC
Dev		888HG			888HG†	888GG40DWPC	888HG44PWPC
Dev		888KG			888KG†	888GG40DWPC	888KG44PWPC
Dev		884FH			888FH	888FH28DWPC	
Dev		888FH			888FH	888FH40DWPC	888FH44PWPC
✓	984EK	884EK			888EK	884EK28DWPC	
✓	988EK	888EK			888EK	884EK40DWPC	888EK44PWPC
✓	984CL	884CL	684CL		888GG	884CL28DWPC	
✓	988CL	888CL	688CL	888GG-X	888GG	888CL40DWPC	888CL44PWPC
✓	984CS	884CS	684CS		888GG	884GG28DWPC	
✓	988CS	888CS	688CS		888GG	888GG40DWPC	888EG44PWPC
✓		888GW			888GW		888RW68PWPC

COP8 Family Selection Guide (Continued)

Notes: *(Consult specific datasheets for exact information.)*

Note 1: Two versions available. "C" (shown) 2.3V to 4.0V operation; "CH" for 4.0V to 6.0V operation.

Note 2: Price is approximate. Varies with package, temperature range and volume.

Note 3: Bidirectional I/O pins can be bit configured by software as:
- input Hi-Z (TRI-STATE) or input with weak pull-up (internal)
- output push-pull "0" (low) or push-pull "1"(high)

Note 4: 16-bit, clocked at t_C. Can be s/w configured as: 1) PWM, 2) Input capture, 3) External event counter input.

Note 5: 8 or 16 bits, clocked at CKI (100 ns max).

Note 6: Pulsed width modulation 8/16 bits, 1 μs or 100 ns (max) resolution.

Note 7: One (dedicated) external interrupt on basic family devices, 9 external interrupts (1 dedicated, 8 configurable) on feature family devices.

Note 8: Patented EMI reducing circuitry on-chip.

Note 9: While OTP's/windowed devices are functionally identical to masked ROM devices, electrical specifications may differ.

Note 10: Debug Module and EPU have the capability to program all applicable OTP's and windowed devices (not recommended for volume programming).

Note 11: OTP's with up to 32k EPROM are available.

† Contact sales office for availability.

Packaging Options

Package Type	Code	Package Type	Code
Plastic Dual-In-Line Package (DIP)	N	Thin Quad Flat Pack	VEJ
Plastic Leaded Chip Carrier (PLCC)	V	28 Small Outline Footprint—Windowed	MHEA
Small Outline Packages—Wide Body	WM	Lead Chip Carrier	EL
Small Outline Packages—Wide Body	M	Ceramic Windowed DIP	MHD
Singulated Dice	MDC	Ceramic Windowed DIP	J
Dice in Wafer Form	DWF		

Section 2

COP8™ OTP Products

Section 2 Contents

COP8SAA7/COP8SAB7/COP8SAC7 8-Bit One-Time Programmable (OTP) Microcontroller . . 2-3
COP8780C/COP8781C/COP8782C 8-Bit One-Time Programmable (OTP) Microcontroller . . . 2-58
COP87L20CJ/COP87L22CJ 8-Bit One-Time Programmable (OTP) Microcontrollers with Multi-Input Wake-Up and Brown Out Detector 2-79
COP87L40CJ/COP87L42CJ 8-Bit One-Time Programmable (OTP) Microcontrollers with Multi-Input Wake-Up and Brown Out Detector 2-105
COP87L84BC 8-Bit One-Time Programmable (OTP) Microcontroller with CAN Interface 2-132
COP87L88EB/COP87L89EB 8-Bit One-Time Programmable (OTP) Microcontroller with CAN Interface, A/D and UART 2-183
COP87L88CL/COP87L84CL 8-Bit One-Time Programmable (OTP) Microcontroller 2-252
COP87L88CF/COP87L84CF 8-Bit One-Time Programmable (OTP) Microcontroller with A/D Converter 2-282
COP8ACC7 8-Bit One-Time Programmable (OTP) Microcontroller with High Resolution A/D Conversion 2-314
COP87L88EK/COP87L84EK 8-Bit One-Time Programmable (OTP) Microcontroller with Analog Function Block 2-349
COP87L88EG/COP87L84EG 8-Bit One-Time Programmable (OTP) Microcontroller with UART and Three Multi-Function Timers 2-381
COP87L88FH/COP87L84FH 8-Bit One-Time Programmable (OTP) Microcontroller with UART and Three Multi-Function Timers and Multiply/Divide Block 2-418
COP87L88GG 8-Bit One-Time Programmable (OTP) Microcontroller with UART and Three Multi-Function Timers 2-458
COP87L88GD 8-Bit One-Time Programmable (OTP) Microcontroller with A/D Converter 2-497
COP87L88KG 8-Bit One-Time Programmable (OTP) Microcontroller with UART and Three Multi-Function Timers 2-499

National Semiconductor

PRELIMINARY

COP8SAA7/COP8SAB7/COP8SAC7
8-Bit One-Time Programmable (OTP) Microcontroller

1.0 General Description

1.1 INTRODUCTION

The COPSAx7 OTP microcontrollers are members of the COP8™ feature family using an 8-bit single chip core architecture. These devices are fabricated in National Semiconductor's high-density EPROM process, and offered on a variety of packages, temperature ranges and voltage ranges to satisfy a wide variety of applications.

Key features include an 8-bit memory mapped architecture, a 16-bit timer/counter with two associated 16-bit registers supporting three modes (Processor Independent PWM generation, External Event counter, and Input Capture capabilities), two power saving HALT/IDLE modes with a multi-sourced wakeup/interrupt capability, on-chip R/C oscillator, high current outputs, user selectable options such as WATCHDOG™, Oscillator configuration, and power-on-reset.

1.2 KEY FEATURES

- Low cost 8-bit OTP microcontroller
- OTP program space with read/write protection
- Quiet Design (low radiated emissions)
- Multi-Input Wakeup pins with optional interrupts (4 to 8 pins)
- 8 bytes of user storage space in EPROM
- User selectable clock options
  - — Crystal/Resonator options
  - — Crystal/Resonator option with on-chip bias resistor
  - — External oscillator
  - — Internal R/C oscillator
- Internal Power-On Reset—user selectable
- WATCHDOG and Clock Monitor Logic—user selectable
- Up to 12 high current outputs

Device	EPROM	RAM	Package and I/O	
			Package Types	Number of I/O
COP8SAC7	4k	128	20 DIP/SO	16
			28 DIP/SO	24
			40 DIP	36
			44 PLCC/PQFP	40
COP8SAB7	2k	128	20 DIP/SO	16
			28 DIP/SO	24
COP8SAA7	1k	64	16 DIP/SO	12
			20 DIP/SO	16
			28 DIP/SO	24

1.2.1 CPU Features

- Versatile easy to use instruction set
- 1 μs instruction cycle time
- Eight multi-source vectored interrupts servicing
  - — External interrupt
  - — Idle Timer T0
  - — One Timer (with 2 interrupts)
  - — MICROWIRE/PLUS™ Serial Interface
  - — Multi-Input Wake Up
  - — Software Trap
  - — Default VIS (default interrupt)
- 8-bit Stack Pointer SP (stack in RAM)
- Two 8-bit Register Indirect Data Memory Pointers
- True bit manipulation
- Memory mapped I/O
- BCD arithmetic instructions

1.2.2 Peripheral Features

- Multi-Input Wakeup Logic
- One 16-bit timer with two 16-bit registers supporting:
  - — Processor Independent PWM mode
  - — External Event counter mode
  - — Input Capture mode
- Idle Timer
- MICROWIRE/PLUS Serial Interface (SPI Compatible)

1.2.3 I/O Features

- Software selectable I/O options
  - — TRI-STATE® Output
  - — Push-Pull Output
  - — Weak Pull Up Input
  - — High Impedance Input
- Schmitt trigger inputs on ports G and L
- UP to 12 high current outputs
- Pin efficient (i.e., 40 pins in 44-pin package are devoted to useful I/O)

1.2.4 Fully Static CMOS Design

- Low current drain (typically < 4 μA)
- Single supply operation: 2.7V to 5.5V
- Two power saving modes: HALT and IDLE

1.2.5 Temperature Ranges

0°C to +70°C, −40°C to +85°C, and −40°C to +125°C

1.2.6 Development Support

- Windowed packages for DIP and PLCC
- Real time emulation and full program debug offered by MetaLink Development System

Table of Contents

1.0 GENERAL DESCRIPTION

1.1 Introduction

1.2 Key Features

1.2.1 CPU Features

1.2.2 Peripheral Features

1.2.3 I/O Features

1.2.4 Fully Static CMOS Design

1.2.5 Temperature Ranges

1.2.6 Development Support

1.3 Block Diagram

1.4 EMI Reduction

1.5 Architecture

1.6 Instruction Set

1.6.1 Key Instruction Set Features

1.6.1 Single Byte/Single Cycle Code Execution

1.6.2 Many Single-Byte, Multifunction Instructions

1.6.3 Bit-Level Control

1.7 Packaging/Pin Efficiency

2.0 CONNECTION DIAGRAMS

2.1 Ordering Information

3.0 ELECTRICAL CHARACTERISTICS

4.0 PIN DESCRIPTIONS

5.0 FUNCTIONAL DESCRIPTION

5.1 CPU Registers

5.2 Program Memory

5.3 Data Memory

5.4 ECON (EPROM Configuration) Register

5.5 User Storage Space in EPROM

5.6 OTP Security

5.7 Reset

5.7.1 External Reset

5.7.2 On-Chip Power-On Reset

5.8 Oscillator Circuits

5.8.1 Crystal Oscillator

5.8.2 External Oscillator

5.8.3 R/C Oscillator

5.9 Control Registers

6.0 TIMERS

6.1 Timer T0 (IDLE Timer)

6.2 Timer T1

6.2.1 Mode 1. Processor Independent PWM Mode

6.2.2 Mode 2. External Event Counter Mode

6.2.3 Mode 3. Input Capture Mode

6.3 Timer Control Flags

7.0 POWER SAVING FEATURES

7.1 HALT Mode

7.2 IDLE Mode

7.3 Multi-Input Wakeup

8.0 INTERRUPTS

8.1 Introduction

8.2 Maskable Interrupts

8.3 VIS Instruction

8.3.1 VIS Execution

8.4 Non-Maskable Interrupt

8.4.1 Pending Flag

8.4.2 Software Trap

8.5 Port L Interrupts

8.6 Interrupt Summary

9.0 WATCHDOG/CLOCK MONITOR

9.1 Clock Monitor

9.2 WATCHDOG/Clock Monitor Operation

9.3 WATCHDOG and Clock Monitor Summary

9.4 Detection of Illegal Conditions

10.0 MICROWIRE/PLUS

10.1 MICROWIRE/PLUS Operation

10.1.1 MICROWIRE/PLUS Master Mode Operation

10.1.2 MICROWIRE/PLUS Slave Mode Operation

10.1.3 Alternate SK Phase Operation and SK Idle Polarity

11.0 MEMORY MAP

12.0 INSTRUCTION SET

12.1 Introduction

12.2 Instruction Features

12.3 Addressing Modes

12.3.1 Operand Addressing Modes

12.3.2 Transfer-of-Control Addressing Modes

12.4 Instruction Types

12.4.1 Arithmetic Instructions

12.4.2 Transfer-of-Control Instructions

12.4.3 Load and Exchange Instructions

12.4.4 Logical Instructions

12.4.5 Accumulator Bit Manipulation Instructions

12.4.6 Stack Control Instructions

12.4.7 Memory Bit Manipulation Instructions

12.4.8 Conditional Instructions

12.4.9 No-Operation Instruction

12.5 Register and Symbol Definition

12.6 Instruction Set Summary

Table of Contents (Continued)

12.7 Instruction Execution Time
12.5 Opcode Table

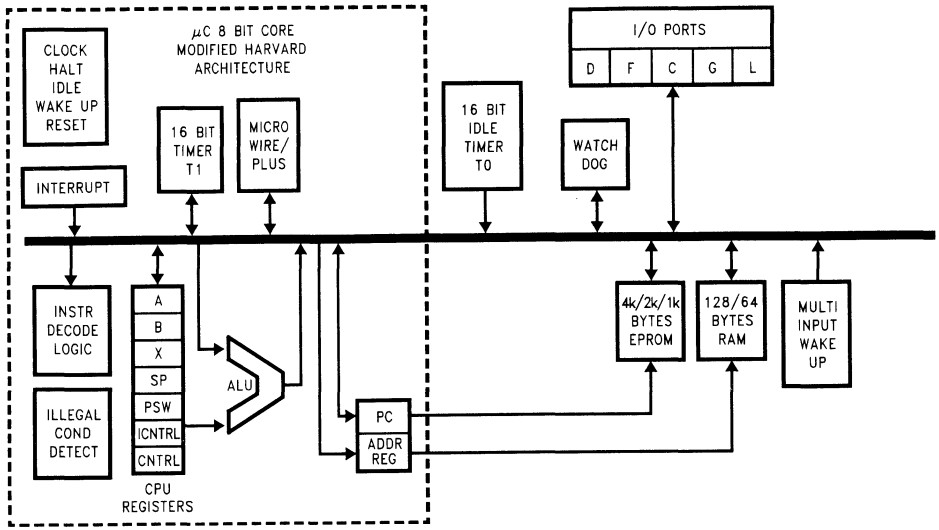
13.0 DEVELOPMENT SUPPORT

13.1 Summary
13.2 IceMASTER™ (IM) In-Circuit Emulation
13.3 IceMASTER Debug Module (DM)
13.4 IceMASTER Evaluation Programming Unit (EPU)
13.5 COP8 Assembler/Linker Software Development Tool Kit
13.6 COP8 C Compiler
13.7 Industry Wide OTP/EPROM Programming Support
13.8 Available Literature
13.9 Dial-A-Helper Service
13.10 Customer Response Center

14.0 PHYSICAL DIMENSIONS

1.0 General Description (Continued)

1.3 BLOCK DIAGRAM



TL/DD/12838–1

FIGURE 1. COP8SAx7 Block Diagram

1.0 General Description (Continued)

1.4 EMI REDUCTION

The COPSAx7 family of devices incorporates circuitry that guards against electromagnetic interference—an increasing problem in today's microcontroller board designs. National's patented EMI reduction technology offers low EMI clock circuitry, gradual turn-on output drivers (GTOs) and internal I_{CC} smoothing filters, to help circumvent many of the EMI issues influencing embedded control designs. National has achieved 15 dB–20 dB reduction in EMI transmissions when designs have incorporated its patented EMI reducing circuitry.

1.5 ARCHITECTURE

The COPSAx7 family is based on a modified Harvard architecture, which allows data tables to be accessed directly from program memory. This is very important with modern microcontroller-based applications, since program memory is usually ROM or EPROM, while data memory is usually RAM. Consequently data tables usually need to be contained in ROM or EPROM, so they are not lost when the microcontroller is powered down. In a modified Harvard architecture, instruction fetch and memory data transfers can be overlapped with a two stage pipeline, which allows the next instruction to be fetched from program memory while the current instruction is being executed using data memory. This is not possible with a Von Neumann single-address bus architecture.

The COPSAx7 family supports a software stack scheme that allows the user to incorporate many subroutine calls. This capability is important when using High Level Languages. With a hardware stack, the user is limited to a small fixed number of stack levels.

1.6 INSTRUCTION SET

In today's 8-bit microcontroller application arena cost/performance, flexibility and time to market are several of the key issues that system designers face in attempting to build well-engineered products that compete in the marketplace. Many of these issues can be addressed through the manner in which a microcontroller's instruction set handles processing tasks. And that's why COP8 family offers a unique and code-efficient instruction set—one that provides the flexibility, functionality, reduced costs and faster time to market that today's microcontroller based products require.

Code efficiency is important because it enables designers to pack more on-chip functionality into less program memory space (ROM/OTP). Selecting a microcontroller with less program memory size translates into lower system costs, and the added security of knowing that more code can be packed into the available program memory space.

1.6.1 Key Instruction Set Features

The COPSAx7 family incorporates a unique combination of instruction set features, which provide designers with optimum code efficiency and program memory utilization.

Single Byte/Single Cycle Code Execution

The efficiency is due to the fact that the majority of instructions are of the single byte variety, resulting in minimum program space. Because compact code does not occupy a substantial amount of program memory space, designers can integrate additional features and functionality into the microcontroller program memory space. Also, the majority instructions executed by the device are single cycle, resulting in minimum program execution time. In fact, 77% of the instructions are single byte single cycle, providing greater code and I/O efficiency, and faster code execution.

1.6.2 Many Single-Byte, Multifunction Instructions

The COPSAx7 instruction set utilizes many single-byte, multifunction instructions. This enables a single instruction to accomplish multiple functions, such as DRSZ, DCOR, JID, and LOAD/EXCHANGE instructions with post-incrementing and post-decrementing, to name just a few examples. In many cases, the instruction set can simultaneously execute as many as three functions with the same single-byte instruction.

JID: (Jump Indirect); Single byte instruction; decodes external events and jumps to corresponding service routines (analogous to "DO CASE" statements in higher level languages).

LAID: (Load Accumulator-Indirect); Single byte look up table instruction provides efficient data path from the program memory to the CPU. This instruction can be used for table lookup and to read the entire program memory for checksum calculations.

RETSK: (Return Skip); Single byte instruction allows return from subroutine and skips next instruction. Decision to branch can be made in the subroutine itself, saving code.

AUTOINC/DEC: (Auto-Increment/Auto-Decrement); These instructions use the two memory pointers B and X to efficiently process a block of data (analogous to "FOR NEXT" in higher level languages).

1.6.3 Bit-Level Control

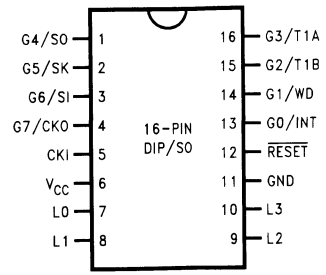
Bit-level control over many of the microcontroller's I/O ports provides a flexible means to ease layout concerns and save board space. All members of the COP8 family provide the ability to set, reset and test any individual bit in the data memory address space, including memory-mapped I/O ports and associated registers. Three memory-mapped pointers handle register indirect addressing and software stack pointer functions. The memory data pointers allow the option of post-incrementing or post-decrementing with the data movement instructions (LOAD/EXCHANGE). And 15 memory-maped registers allow designers to optimize the precise implementation of certain specific instructions.

1.7 PACKAGING/PIN EFFICIENCY

Real estate and board configuration considerations demand maximum space and pin efficiency, particularly given today's high integration and small product form factors. Microcontroller users try to avoid using large packages to get the I/O needed. Large packages take valuable board space and increases device cost, two trade-offs that microcontroller designs can ill afford.

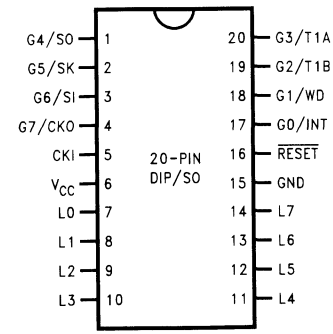
The COP8 family offers a wide range of packages and do not waste pins: up to 90.9% (or 40 pins in the 44-pin package) are devoted to useful I/O.

2.0 Connection Diagrams



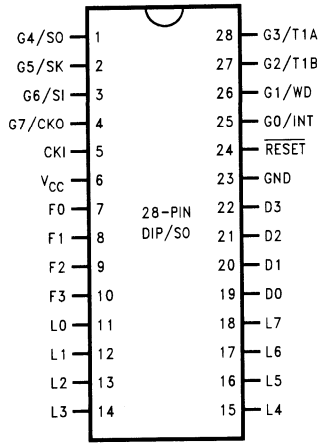
TL/DD/12838-2

Top View



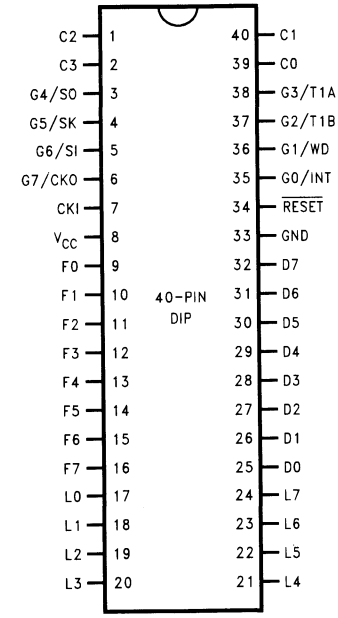
TL/DD/12838-3

Top View



TL/DD/12838-4

Top View

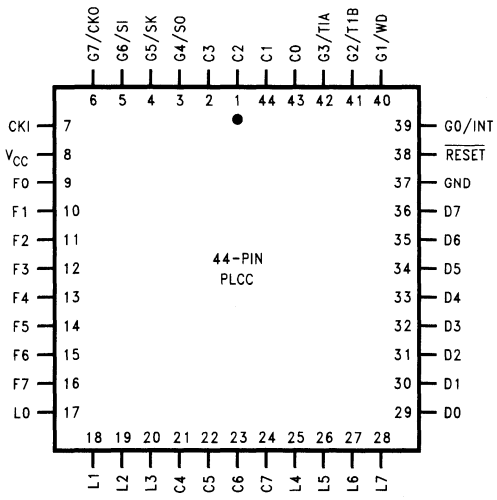


TL/DD/12838-5

Top View

FIGURE 2. Connection Diagrams

2.0 Connection Diagrams (Continued)



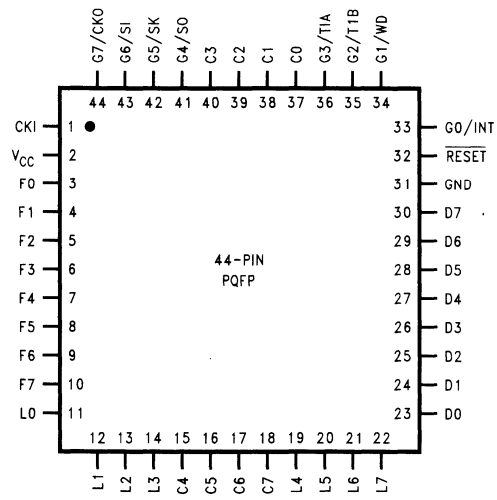


FIGURE 2. Connection Diagrams (Continued)

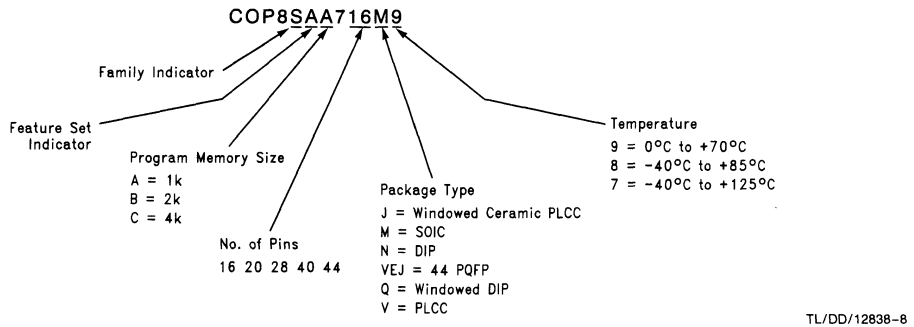


FIGURE 3. Part Numbering Scheme

2.0 Connection Diagrams (Continued)

2.1 ORDERING INFORMATION

	1k EPROM		2k EPROM		4k EPROM		4k EPROM	
Temperature	Order Number	Package	Order Number	Package	Order Number	Package	Windowed Device Order Number	Package
	COP8SAA716M9	16M						
	COP8SAA720M9	20M	COP8SAB720M9	20M	COP8SAC720M9	20M		
	COP8SAA728M9	28M	COP8SAB728M9	28M	COP8SAC728M9	28M		
	COP8SAA716N9	16N						
0°C to +70°C	COP8SAA720N9	20N	COP8SAB720N9	20N	COP8SAC720N9	20N	COP8SAC720Q9	20Q
	COP8SAA728N9	28N	COP8SAB728N9	28N	COP8SAC728N9	28N	COP8SAC728Q9	28Q
					COP8SAC740N9	40N	COP8SAC740Q9	40Q
					COP8SAC744V9	44V	COP8SAC744J9	44J
					COP8SAC7VEJ9	44PQFP		
	COP8SAA716M8	16M						
	COP8SAA720M8	20M	COP8SAB720M8	20M	COP8SAC720M8	20M		
	COP8SAA728M8	28M	COP8SAB728M8	28M	COP8SAC728M8	28M		
	COP8SAA716N8	16N						
−40°C to +85°C	COP8SAA720N8	20N	COP8SAB720N8	20N	COP8SAC720N8	20N		
	COP8SAA728N8	28N	COP8SAB728N8	28N	COP8SAC728N8	28N		
					COP8SAC740N8	40N		
					COP8SAC744V8	44V		
					COP8SAC7VEJ8	44PQFP		
					COP8SAC720M7	20M		
					COP8SAC728M7	28M		
					COP8SAC720N7	20N		
−40°C to +125°C					COP8SAC728N7	28N		
					COP8SAC740N7	40N		
					COP8SAC744V7	44V		
					COP8SAC7VEJ7	44PQFP		

3.0 Electrical Characteristics

Absolute Maximum Ratings

If Military/Aerospace specified devices are required, please contact the National Semiconductor Sales Office/Distributors for availability and specifications.

Supply Voltage (V_{CC})	7V
Voltage at Any Pin	$-0.6V$ to $V_{CC} + 0.6V$
ESD Protection Level	2 kV (Human Body Model)
Total Current into V_{CC} Pin (Source)	80 mA
Total Current out of GND Pin (Sink)	100 mA
Storage Temperature Range	$-65°C$ to $+140°C$

Note: *Absolute maximum ratings indicate limits beyond which damage to the device may occur. DC and AC electrical specifications are not ensured when operating the device at absolute maximum ratings.*

DC Electrical Characteristics $0°C \le T_A \le +70°C$ unless otherwise specified.

Parameter	Conditions	Min	Typ	Max	Units
Operating Voltage		2.7		5.5	V
Power Supply Rise Time (On-Chip Power-On Reset Selected)		10 ns		50 ms	
Power Supply Ripple (Note 1)	Peak-to-Peak			0.1 V_{CC}	V
Supply Current (Note 2)					
CKI = 10 MHz	$V_{CC} = 5.5V$, $t_C = 1\ \mu s$			6	mA
CKI = 4 MHz	$V_{CC} = 4.5V$, $t_C = 2.5\ \mu s$			2.1	mA
HALT Current (Note 3)—WATCHDOG Disabled	$V_{CC} = 5.5V$, CKI = 0 MHz		<4	8	μA
IDLE Current (Note 2)					
CKI = 10 MHz	$V_{CC} = 5.5V$, $t_C = 1\ \mu s$			1.5	mA
CKI = 4 MHz	$V_{CC} = 4.5V$, $t_C = 2.5\ \mu s$			0.8	mA
Input Levels (V_{IH}, V_{IL})					
$\overline{RESET}$					
Logic High		0.8 V_{CC}			V
Logic Low				0.2 V_{CC}	V
CKI, All Other Inputs					
Logic High		0.7 V_{CC}			V
Logic Low				0.2 V_{CC}	V
Value of the Internal Bias Resistor for the Crystal/Resonator Oscillator		0.5	1.0	2.0	MΩ
CKI Resistance to V_{CC} or GND when R/C Oscillator is Selected	$V_{CC} = 5.5V$	5	8	11	kΩ
Hi-Z Input Leakage (same as TRI-STATE output)	$V_{CC} = 5.5V$	-2		$+2$	μA
Input Pullup Current	$V_{CC} = 5.5V$, $V_{IN} = 0V$	-40		-250	μA
G and L Port Input Hysteresis		0.25 V_{CC}			V

3.0 Electrical Characteristics (Continued)

DC Electrical Characteristics $0°C \leq T_A \leq +70°C$ unless otherwise specified. (Continued)

Parameter	Conditions	Min	Typ	Max	Units
Output Current Levels					
D Outputs					
Source	$V_{CC} = 4.5V, V_{OH} = 3.3V$	−0.4			mA
	$V_{CC} = 2.7V, V_{OH} = 1.8V$	−0.2			mA
Sink	$V_{CC} = 4.5V, V_{OL} = 1.0V$	10			mA
	$V_{CC} = 2.7V, V_{OL} = 0.4V$	2			mA
L Port					
Source (Weak Pull-Up)	$V_{CC} = 4.5V, V_{OH} = 2.7V$	−10		−110	μA
	$V_{CC} = 2.7V, V_{OH} = 1.8V$	−2.5		−33	μA
Source (Push-Pull Mode)	$V_{CC} = 4.5V, V_{OH} = 3.3V$	−0.4			mA
	$V_{CC} = 2.7V, V_{OH} = 1.8V$	−0.2			mA
Sink (L0–L3, Push-Pull Mode)	$V_{CC} = 4.5V, V_{OL} = 1.0V$	10			mA
	$V_{CC} = 2.7V, V_{OL} = 0.4V$	2			mA
Sink (L4–L7, Push-Pull Mode)	$V_{CC} = 4.5V, V_{OL} = 0.4V$	1.6			mA
	$V_{CC} = 2.7V, V_{OL} = 0.4V$	0.7			mA
All Others					
Source (Weak Pull-Up Mode)	$V_{CC} = 4.5V, V_{OH} = 2.7V$	−10		−110	μA
	$V_{CC} = 2.7V, V_{OH} = 1.8V$	−2.5		−33	μA
Source (Push-Pull Mode)	$V_{CC} = 4.5V, V_{OH} = 3.3V$	−0.4			mA
	$V_{CC} = 2.7V, V_{OH} = 1.8V$	−0.2			mA
Sink (Push-Pull Mode)	$V_{CC} = 4.5V, V_{OL} = 0.4V$	1.6			mA
	$V_{CC} = 2.7V, V_{OL} = 0.4V$	0.7			mA
Allowable Sink Current per Pin (Note 6)					
D Outputs and L0 to L3				15	mA
All Others				3	mA
Maximum Input Current without Latchup (Note 4)				±200	mA
RAM Retention Voltage, Vr		2.0			V
V_{CC} Rise Time from a $V_{CC} \geq 2.0V$		1.2			μs
Input Capacitance	(Note 6)			7	pF
Load Capacitance on D2	(Note 6)			1000	pF

3.0 Electrical Characteristics (Continued)

AC Electrical Characteristics $0°C \le T_A \le +70°C$ unless otherwise specified.

Parameter	Conditions	Min	Typ	Max	Units
Instruction Cycle Time (t_C)					
Crystal/Resonator, External	$4.5V \le V_{CC} \le 5.5V$	1.0		DC	μs
	$2.7V \le V_{CC} < 4.5V$	2.0		DC	μs
Internal R/C Oscillator	$4.5V \le V_{CC} \le 5.5V$		2.0		μs
	$2.7V \le V_{CC} < 4.5V$		TBD		μs
R/C Oscillator Frequency Variation	$4.5V \le V_{CC} \le 5.5V$			±35	%
(Note 6)	$2.7V \le V_{CC} < 4.5V$			TBD	%
External CKI Clock Duty Cycle (Note 6)	fr = Max	45		55	%
Rise Time (Note 6)	fr = 10 MHz Ext Clock			12	ns
Fall Time (Note 6)	fr = 10 MHz Ext Clock			8	ns
Inputs					
t_{SETUP}	$4.5V \le V_{CC} \le 5.5V$	200			ns
	$2.7V \le V_{CC} < 4.5V$	500			ns
t_{HOLD}	$4.5V \le V_{CC} \le 5.5V$	60			ns
	$2.7V \le V_{CC} < 4.5V$	150			ns
Output Propagation Delay (Note 5)	$R_L = 2.2k, C_L = 100$ pF				
t_{PD1}, t_{PD0}					
SO, SK	$4.5V \le V_{CC} \le 5.5V$			0.7	μs
	$2.7V \le V_{CC} < 4.5V$			1.75	μs
All Others	$4.5V \le V_{CC} \le 5.5V$			1.0	μs
	$2.7V \le V_{CC} < 4.5V$			2.5	μs
MICROWIRE Setup Time (t_{UWS}) (Note 5)		20			ns
MICROWIRE Hold Time (t_{UWH}) (Note 5)		56			ns
MICROWIRE Output Propagation Delay (t_{UPD})				220	ns
MICROWIRE Maximum Shift Clock					
Master Mode				500	kHz
Slave Mode				1	MHz
Input Pulse Width (Note 6)					
Interrupt Input High Time		1			t_C
Interrupt Input Low Time		1			t_C
Timer 1 Input High Time		1			t_C
Timer 1 Input Low Time		1			t_C
Reset Pulse Width		1			μs

t_C = Instruction cycle time (Clock input frequency divided by 10).

Note 1: Maximum rate of voltage change must be < 0.5 V/ms.

Note 2: Supply and IDLE currents are measured with CKI driven with a square wave Oscillator, CKO driven 180° out of phase with CKI, inputs connected to V_{CC} and outputs driven low but not connected to a load.

Note 3: The HALT mode will stop CKI from oscillating in the R/C and the Crystal configurations. In the R/C configuration, CKI is forced high internally. In the crystal or external configuration, CKI is TRI-STATE. Measurement of I_{DD} HALT is done with device neither sourcing nor sinking current; with L. F, C, G0, and G2–G5 programmed as low outputs and not driving a load; all outputs programmed low and not driving a load; all inputs tied to V_{CC}; WATCHDOG and clock monitor disabled. Parameter refers to HALT mode entered via setting bit 7 of the G Port data register.

Note 4: Pins G6 and $\overline{\text{RESET}}$ are designed with a high voltage input network. These pins allow input voltages > V_{CC} and the pins will have sink current to V_{CC} when biased at voltages > V_{CC} (the pins do not have source current when biased at a voltage below V_{CC}). The effective resistance to V_{CC} is 750Ω (typical). These two pins will not latch up. The voltage at the pins must be limited to < 14V. **WARNING: Voltages in excess of 14V will cause damage to the pins. This warning excludes ESD transients.**

Note 5: The output propagation delay is referenced to the end of the instruction cycle where the output change occurs.

Note 6: Parameter characterized but not tested.

3.0 Electrical Characteristics (Continued)

Absolute Maximum Ratings

If Military/Aerospace specified devices are required, please contact the National Semiconductor Sales Office/Distributors for availability and specifications.

Supply Voltage (V_{CC})	7V
Voltage at Any Pin	$-0.6V$ to $V_{CC} + 0.6V$
ESD Protection Level	2 kV (Human Body Model)
Total Current into V_{CC} Pin (Source)	80 mA
Total Current out of GND Pin (Sink)	100 mA
Storage Temperature Range	$-65°C$ to $+140°C$

Note: *Absolute maximum ratings indicate limits beyond which damage to the device may occur. DC and AC electrical specifications are not ensured when operating the device at absolute maximum ratings.*

DC Electrical Characteristics $-40°C \le T_A \le +85°C$ unless otherwise specified.

Parameter	Conditions	Min	Typ	Max	Units
Operating Voltage		2.7		5.5	V
Power Supply Rise Time (On-Chip Power-On Reset Selected)		10 ns		50 ms	
Power Supply Ripple (Note 1)	Peak-to-Peak			0.1 V_{CC}	V
Supply Current (Note 2) CKI = 10 MHz	V_{CC} = 5.5V, t_C = 1 μs			6.0	mA
HALT Current (Note 3)—WATCHDOG Disabled	V_{CC} = 5.5V, CKI = 0 MHz		<4	10.0	μA
IDLE Current (Note 2) CKI = 10 MHz	V_{CC} = 5.5V, t_C = 1 μs			1.5	mA
Input Levels (V_{IH}, V_{IL})					
$\overline{\text{RESET}}$					
Logic High		0.8 V_{CC}			V
Logic Low				0.2 V_{CC}	V
CKI, All Other Inputs					
Logic High		0.7 V_{CC}			V
Logic Low				0.2 V_{CC}	V
Value of the Internal Bias Resistor for the Crystal/Resonator Oscillator		0.5	1.0	2.0	MΩ
CKI Resistance to V_{CC} or GND when R/C Oscillator is Selected	V_{CC} = 5.5V	5	8	11	kΩ
Hi-Z Input Leakage (same as TRI-STATE output)	V_{CC} = 5.5V	−2		+2	μA
Input Pullup Current	V_{CC} = 5.5V, V_{IN} = 0V	−40		−250	μA
G and L Port Input Hysteresis		0.25 V_{CC}			V

3.0 Electrical Characteristics (Continued)

DC Electrical Characteristics $-40°C \le T_A \le +85°C$ unless otherwise specified. (Continued)

Parameter	Conditions	Min	Typ	Max	Units
Output Current Levels					
D Outputs					
Source	$V_{CC} = 4.5V, V_{OH} = 3.3V$	−0.4			mA
	$V_{CC} = 2.7V, V_{OH} = 1.8V$	−0.2			mA
Sink	$V_{CC} = 4.5V, V_{OL} = 1.0V$	10			mA
	$V_{CC} = 2.7V, V_{OL} = 0.4V$	2			mA
L Port					
Source (Weak Pull-Up)	$V_{CC} = 4.5V, V_{OH} = 2.7V$	−10.0		−110	μA
	$V_{CC} = 2.7V, V_{OH} = 1.8V$	−2.5		−33	μA
Source (Push-Pull Mode)	$V_{CC} = 4.5V, V_{OH} = 3.3V$	−0.4			mA
	$V_{CC} = 2.7V, V_{OH} = 1.8V$	−0.2			mA
Sink (L0–L3, Push-Pull Mode)	$V_{CC} = 4.5V, V_{OL} = 1.0V$	10.0			mA
	$V_{CC} = 2.7V, V_{OL} = 0.4V$	2			mA
Sink (L4–L7, Push-Pull Mode)	$V_{CC} = 4.5V, V_{OL} = 0.4V$	1.6			mA
	$V_{CC} = 2.7V, V_{OL} = 0.4V$	0.7			mA
All Others					
Source (Weak Pull-Up Mode)	$V_{CC} = 4.5V, V_{OH} = 2.7V$	−10.0		−110	μA
	$V_{CC} = 2.7V, V_{OH} = 1.8V$	−2.5		−33	μA
Source (Push-Pull Mode)	$V_{CC} = 4.5V, V_{OH} = 3.3V$	−0.4			mA
	$V_{CC} = 2.7V, V_{OH} = 1.8V$	−0.2			mA
Sink (Push-Pull Mode)	$V_{CC} = 4.5V, V_{OL} = 0.4V$	1.6			mA
	$V_{CC} = 2.7V, V_{OL} = 0.4V$	0.7			mA
Allowable Sink Current per Pin (Note 6)					
D Outputs and L0 to L3				15	mA
All Others				3	mA
Maximum Input Current without Latchup (Note 4)				±200	mA
RAM Retention Voltage, Vr		2.0			V
V_{CC} Rise Time from a $V_{CC} \ge 2.0V$		1.2			μs
Input Capacitance	(Note 6)			7	pF
Load Capacitance on D2	(Note 6)			1000	pF

AC Electrical Characteristics $-40°C \le T_A \le +85°C$ unless otherwise specified.

Parameter	Conditions	Min	Typ	Max	Units
Instruction Cycle Time (t_C)					
Crystal/Resonator, External	$4.5V \le V_{CC} \le 5.5V$	1.0		DC	μs
	$2.7V \le V_{CC} < 4.5V$	2.0		DC	μs
Internal R/C Oscillator	$4.5V \le V_{CC} \le 5.5V$		2.0		μs
	$2.7V \le V_{CC} < 4.5V$		TBD		μs
R/C Oscillator Frequency Variation	$4.5V \le V_{CC} \le 5.5V$			±35	%
(Note 6)	$2.7V \le V_{CC} < 4.5V$			TBD	%
External CKI Clock Duty Cycle (Note 6)	fr = Max	45		55	%
Rise Time (Note 6)	fr = 10 MHz Ext Clock			12	ns
Fall Time (Note 6)	fr = 10 MHz Ext Clock			8	ns

3.0 Electrical Characteristics (Continued)

AC Electrical Characteristics $-40°C \leq T_A \leq +85°C$ unless otherwise specified. (Continued)

Parameter	Conditions	Min	Typ	Max	Units
Inputs					
t_{SETUP}	$4.5V \leq V_{CC} \leq 5.5V$	200			ns
	$2.7V \leq V_{CC} < 4.5V$	500			ns
t_{HOLD}	$4.5V \leq V_{CC} \leq 5.5V$	60			ns
	$2.7V \leq V_{CC} < 4.5V$	150			ns
Output Propagation Delay (Note 5)	$R_L = 2.2k$, $C_L = 100$ pF				
t_{PD1}, t_{PD0}					
SO, SK	$4.5V \leq V_{CC} \leq 5.5V$			0.7	μs
	$2.7V \leq V_{CC} < 4.5V$			1.75	μs
All Others	$4.5V \leq V_{CC} \leq 5.5V$			1.0	μs
	$2.7V \leq V_{CC} < 4.5V$			2.5	μs
MICROWIRE Setup Time (t_{UWS}) (Note 5)		20			ns
MICROWIRE Hold Time (t_{UWH}) (Note 5)		56			ns
MICROWIRE Output Propagation Delay (t_{UPD})				220	ns
MICROWIRE Maximum Shift Clock					
Master Mode				500	kHz
Slave Mode				1	MHz
Input Pulse Width (Note 6)					
Interrupt Input High Time		1			t_C
Interrupt Input Low Time		1			t_C
Timer 1 Input High Time		1			t_C
Timer 1 Input Low Time		1			t_C
Reset Pulse Width		1			μs

t_C = Instruction cycle time (Clock input frequency divided by 10).

Note 1: Maximum rate of voltage change must be < 0.5 V/ms.

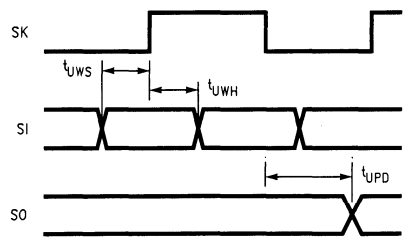
Note 2: Supply and IDLE currents are measured with CKI driven with a square wave Oscillator, CKO driven 180° out of phase with CKI, inputs connected to V_{CC} and outputs driven low but not connected to a load.

Note 3: The HALT mode will stop CKI from oscillating in the R/C and the Crystal configurations. In the R/C configuration, CKI is forced high internally. In the crystal or external configuration, CKI is TRI-STATE. Measurement of I_{DD} HALT is done with device neither sourcing nor sinking current; with L, F, C, G0, and G2–G5 programmed as low outputs and not driving a load; all outputs programmed low and not driving a load; all inputs tied to V_{CC}; clock monitor disabled. Parameter refers to HALT mode entered via setting bit 7 of the G Port data register.

Note 4: Pins G6 and $\overline{\text{RESET}}$ are designed with a high voltage input network. These pins allow input voltages > V_{CC} and the pins will have sink current to V_{CC} when biased at voltages > V_{CC} (the pins do not have source current when biased at a voltage below V_{CC}). The effective resistance to V_{CC} is 750Ω (typical). These two pins will not latch up. The voltage at the pins must be limited to < 14V. **WARNING: Voltages in excess of 14V will cause damage to the pins. This warning excludes ESD transients.**

Note 5: The output propagation delay is referenced to the end of the instruction cycle where the output change occurs.

Note 6: Parameter characterized but not tested.



TL/DD/12838–9

FIGURE 4. MICROWIRE/PLUS Timing

3.0 Electrical Characteristics (Continued)

Absolute Maximum Ratings

If Military/Aerospace specified devices are required, please contact the National Semiconductor Sales Office/Distributors for availability and specifications.

Supply Voltage (V_{CC})	7V
Voltage at Any Pin	−0.6V to V_{CC} +0.6V
ESD Protection Level	2 kV (Human Body Model)
Total Current into V_{CC} Pin (Source)	80 mA
Total Current out of GND Pin (Sink)	100 mA
Storage Temperature Range	−65°C to +140°C

Note: *Absolute maximum ratings indicate limits beyond which damage to the device may occur. DC and AC electrical specifications are not ensured when operating the device at absolute maximum ratings.*

DC Electrical Characteristics $-40°C \le T_A \le +125°C$ unless otherwise specified.

Parameter	Conditions	Min	Typ	Max	Units
Operating Voltage		4.5		5.5	V
Power Supply Rise Time (On-Chip Power-On Reset Selected)		10 ns		50 ms	
Power Supply Ripple (Note 1)	Peak-to-Peak			0.1 V_{CC}	V
Supply Current (Note 2) CKI = 10 MHz	V_{CC} = 5.5V, t_C = 1 μs			6.0	mA
HALT Current (Note 3)—WATCHDOG Disabled	V_{CC} = 5.5V, CKI = 0 MHz		<10	30	μA
IDLE Current (Note 2) CKI = 10 MHz	V_{CC} = 5.5V, t_C = 1 μs			1.5	mA
Input Levels (V_{IH}, V_{IL})					
$\overline{\text{RESET}}$					
Logic High		0.8 V_{CC}			V
Logic Low				0.2 V_{CC}	V
CKI, All Other Inputs					
Logic High		0.7 V_{CC}			V
Logic Low				0.2 V_{CC}	V
Value of the Internal Bias Resistor for the Crystal/Resonator Oscillator		0.5	1.0	2.0	MΩ
CKI Resistance to V_{CC} or GND when R/C Oscillator is Selected	V_{CC} = 5.5V	5	8	11	kΩ
Hi-Z Input Leakage	V_{CC} = 5.5V	−5		+5	μA
Input Pullup Current	V_{CC} = 5.5V, V_{IN} = 0V	−35		−400	μA
G and L Port Input Hysteresis		0.25 V_{CC}			V
Output Current Levels					
D Outputs					
Source	V_{CC} = 4.5V, V_{OH} = 3.3V	−0.4			mA
Sink	V_{CC} = 4.5V, V_{OL} = 1.0V	9			mA
L Port					
Source (Weak Pull-Up)	V_{CC} = 4.5V, V_{OH} = 2.7V	−9		−140	μA
Source (Push-Pull Mode)	V_{CC} = 4.5V, V_{OH} = 3.3V	−0.4			mA
Sink (L0–L3, Push-Pull Mode)	V_{CC} = 4.5V, V_{OL} = 1.0V	9			mA
Sink (L4–L7, Push-Pull Mode)	V_{CC} = 4.5V, V_{OL} = 0.4V	1.4			mA
All Others					
Source (Weak Pull-Up Mode)	V_{CC} = 4.5V, V_{OH} = 2.7V	−9		−140	μA
Source (Push-Pull Mode)	V_{CC} = 4.5V, V_{OH} = 3.3V	−0.4			mA
Sink (Push-Pull Mode)	V_{CC} = 4.5V, V_{OL} = 0.4V	1.4			mA

3.0 Electrical Characteristics (Continued)

DC Electrical Characteristics $-40°C \le T_A \le +125°C$ unless otherwise specified. (Continued)

Parameter	Conditions	Min	Typ	Max	Units
TRI-STATE Leakage	$V_{CC} = 5.5V$	−5		+5	μA
Allowable Sink Current per Pin (Note 6)					
D Outputs and L0 to L3				15	mA
All Others				3	mA
Maximum Input Current without Latchup (Note 4)	Room Temp			±200	mA
RAM Retention Voltage, Vr		2.0			V
V_{CC} Rise Time from a $V_{CC} \ge 2.0V$		1.2			μs
Input Capacitance	(Note 6)			7	pF
Load Capacitance on D2	(Note 6)			1000	pF

AC Electrical Characteristics $-40°C \le T_A \le +125°C$ unless otherwise specified.

Parameter	Conditions	Min	Typ	Max	Units
Instruction Cycle Time (t_C)					
Crystal/Resonator, External	$4.5V \le V_{CC} \le 5.5V$	1.0		DC	μs
Internal R/C Oscillator	$4.5V \le V_{CC} \le 5.5V$		2.0	DC	μs
R/C Oscillator Frequency Variation (Note 6)	$4.5V \le V_{CC} \le 5.5V$			TBD	%
External CKI Clock Duty Cycle (Note 6)	fr = Max	45		55	%
Rise Time (Note 6)	fr = 10 MHz Ext Clock			12	ns
Fall Time (Note 6)	fr = 10 MHz Ext Clock			8	ns
Inputs					
t_{SETUP}	$4.5V \le V_{CC} \le 5.5V$	200			ns
t_{HOLD}	$4.5V \le V_{CC} \le 5.5V$	60			ns
Output Propagation Delay (Note 5)	$R_L = 2.2k, C_L = 100$ pF				
t_{PD1}, t_{PD0}					
SO, SK	$4.5V \le V_{CC} \le 5.5V$			0.7	μs
All Others	$4.5V \le V_{CC} \le 5.5V$			1.0	μs
MICROWIRE Setup Time (t_{UWS}) (Note 5)		20			ns
MICROWIRE Hold Time (t_{UWH}) (Note 5)		56			ns
MICROWIRE Output Propagation Delay (t_{UPD})				220	ns
MICROWIRE Maximum Shift Clock					
Master Mode				500	kHz
Slave Mode				1	MHz
Input Pulse Width (Note 6)					
Interrupt Input High Time		1			t_C
Interrupt Input Low Time		1			t_C
Timer 1, 2, 3 Input High Time		1			t_C
Timer 1, 2, 3 Input Low Time		1			t_C
Reset Pulse Width		1			μs

4.0 Pin Descriptions

COPSAx7 I/O structure minimizes external component requirements. Software-switchable I/O enables designers to reconfigure the microcontroller's I/O functions with a single instruction. Each individual I/O pin can be independently configured as an output pin low, an output high, an input with high impedance or an input with a weak pull-up device. A typical example is the use of I/O pins as the keyboard matrix input lines. The input lines can be programmed with internal weak pull-ups so that the input lines read logic high when the keys are all up. With a key closure, the corresponding input line will read a logic zero since the weak pull-up can easily be overdriven. When the key is released, the internal weak pullup will pull the input line back to logic high. This flexibility eliminates the need for external pull-up resistors. The High current options are available for driving LEDs, motors and speakers. This flexibility helps to ensure a cleaner design, with less external components and lower costs. Below is the general description of all available pins.

V_{CC} and GND are the power supply pins. All V_{CC} and GND pins must be connected.

CKI is the clock input. This can come from the Internal R/C oscillator, external, or a crystal oscillator (in conjunction with CKO). See Oscillator Description section.

$\overline{\text{RESET}}$ is the master reset input. See Reset description section.

The device contains four bidirectional 8-bit I/O ports (C, G, L and F), where each individual bit may be independently configured as an input (Schmitt trigger inputs on ports L and G), output or TRI-STATE under program control. Three data memory address locations are allocated for each of these I/O ports. Each I/O port has two associated 8-bit memory mapped registers, the CONFIGURATION register and the output DATA register. A memory mapped address is also reserved for the input pins of each I/O port. (See the memory map for the various addresses associated with the I/O ports.) *Figure 5* shows the I/O port configurations. The DATA and CONFIGURATION registers allow for each port bit to be individually configured under software control as shown below:

CONFIGURATION Register	DATA Register	Port Set-Up
0	0	Hi-Z Input (TRI-STATE Output)
0	1	Input with Weak Pull-Up
1	0	Push-Pull Zero Output
1	1	Push-Pull One Output

Port L is an 8-bit I/O port. All L-pins have Schmitt triggers on the inputs.

Port L supports the Multi-Input Wake Up feature on all eight pins. The 16-pin device does not have a full complement of Port L pins. The unavailable pins are not terminated. A read operation these unterminated pins are not terminated. A read operation these unterminated pins will return unpredictable values. To minimize current drain, the unavailable pins must be programmed as outputs.

Port G is an 8-bit port. Pin G0, G2–G5 are bi-directional I/O ports. Pin G6 is always a general purpose Hi-Z input. All pins have Schmitt Triggers on their inputs. **Pin G1 serves as the dedicated WDOUT WATCHDOG output with weak pull-up if WATCHDOG feature is selected by the ECON register. The pin is a general purpose I/O if WATCHDOG feature is not selected.** If WATCHDOG feature is selected, bit 1 of the Port G configuration and data register does not have any effect on Pin G1 setup. Pin G7 is either input or output depending on the oscillator option selected. With the crystal oscillator option selected, G7 serves as the dedicated output pin for the CKO clock output. With the internal R/C or the external oscillator option selected, G7 serves as a general purpose Hi-Z input pin and is also used to bring the device out of HALT mode with a low to high transition on G7. There are two registers associated with Port G, a data register and a configuration register. Using these registers, each of the 5 I/O pins (G0, G2–G5) can be individually configured under software control.

Since G6 is an input only pin and G7 is the dedicated CKO clock output pin (crystal clock option) or general purpose input (R/C or external clock option), the associated bits in the data and configuration registers for G6 and G7 are used for special purpose functions as outlined below. Reading the G6 and G7 data bits will return zeroes.

The device will be placed in the HALT mode by writing a "1" to bit 7 of the Port G Data Register. Similarly the device will be placed in the IDLE mode by writing a "1" to bit 6 of the Port G Data Register.

Writing a "1" to bit 6 of the Port G Configuration Register enables the MICROWIRE/PLUS to operate with the alternate phase of the SK clock. The G7 configuration bit, if set high, enables the clock start up delay after HALT when the R/C clock configuration is used.

	Config. Reg.	Data Reg.
G7	CLKDLY	HALT
G6	Alternate SK	IDLE

Port G has the following alternate features:

G0 INTR (External Interrupt Input)

G2 T1B (Timer T1 Capture Input)

G3 T1A (Timer T1 I/O)

G4 SO (MICROWIRE Serial Data Output)

G5 SK (MICROWIRE Serial Clock)

G6 SI (MICROWIRE Serial Data Input)

Port G has the following dedicated functions:

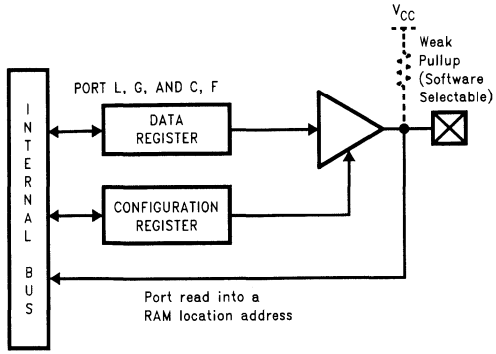
G1 WDOUT WATCHDOG and/or Clock Monitor if WATCHDOG enabled, otherwise it is a general purpose I/O

G7 CKO Oscillator dedicated output or general purpose input

Port C is an 8-bit I/O port. The 40-pin device does not have a full complement of Port C pins. The unavailable pins are not terminated. A read operation on these unterminated pins will return unpredictable values. Only the COP8SAC7 device contains Port C. The 20/28 pin devices do not offer Port C. On these devices, the associated Port C Data and Configuration registers should not be used.

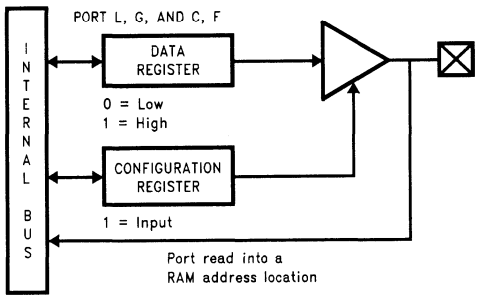
4.0 Pin Descriptions (Continued)

Port F is an 8-bit I/O port. The 28-pin device does not have a full complement of Port F pins. The unavailable pins are not terminated. A read operation on these unterminated pins will return unpredictable values.



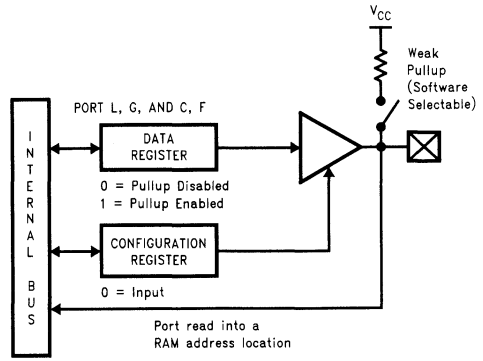
TL/DD/12838–10

FIGURE 5. I/O Port Configurations



TL/DD/12838–12

FIGURE 6. I/O Port Configurations—Output Mode



TL/DD/12838–11

FIGURE 7. I/O Port Configurations—Input Mode

Port D is an 8-bit output port that is preset high when $\overline{\text{RESET}}$ goes low. The user can tie two or more D port outputs (except D2) together in order to get a higher drive.

Note: Care must be exercised with the D2 pin operation. At RESET, the external loads on this pin must ensure that the output voltages stay above 0.7 V_{CC} to prevent the chip from entering special modes. Also keep the external loading on D2 to less than 1000 pF.

5.0 Functional Description

The architecture of the device is a modified Harvard architecture. With the Harvard architecture, the program memory EPROM is separated from the data store memory (RAM). Both EPROM and RAM have their own separate addressing space with separate address buses. The architecture, though based on the Harvard architecture, permits transfer of data from EPROM to RAM.

5.1 CPU REGISTERS

The CPU can do an 8-bit addition, subtraction, logical or shift operation in one instruction (t_C) cycle time.

There are six CPU registers:

A is the 8-bit Accumulator Register

PC is the 15-bit Program Counter Register

PU is the upper 7 bits of the program counter (PC)

PL is the lower 8 bits of the program counter (PC)

B is an 8-bit RAM address pointer, which can be optionally post auto incremented or decremented.

X is an 8-bit alternate RAM address pointer, which can be optionally post auto incremented or decremented.

SP is the 8-bit stack pointer, which points to the subroutine/interrupt stack (in RAM). With reset the SP is initialized to RAM address 02F Hex (devices with 64 bytes of RAM), or initialized to RAM address 06F Hex (devices with 128 bytes of RAM).

All the CPU registers are memory mapped with the exception of the Accumulator (A) and the Program Counter (PC).

5.2 PROGRAM MEMORY

The program memory consists of 1024, 2048, or 4096 bytes of EPROM. Table I shows the program memory sizes for the different devices. These bytes may hold program instructions or constant data (data tables for the LAID instruction, jump vectors for the JID instruction, and interrupt vectors for the VIS instruction). The program memory is addressed by the 15-bit program counter (PC). All interrupts in the device vector to program memory location 0FF Hex. The contents of the program memory read 00 Hex in the erased state.

5.3 DATA MEMORY

The data memory address space includes the on-chip RAM and data registers, the I/O registers (Configuration, Data and Pin), the control registers, the MICROWIRE/PLUS SIO shift register, and the various registers, and counters associated with the timers (with the exception of the IDLE timer). Data memory is addressed directly by the instruction or indirectly by the B, X and SP pointers.

The data memory consists of 64 or 128 bytes of RAM. Table I shows the data memory sizes for the different devices. Fifteen bytes of RAM are mapped as "registers" at addresses 0F0 to 0FE Hex. These registers can be loaded immediately, and also decremented and tested with the DRSZ (decrement register and skip if zero) instruction. The memory pointer registers X, SP and B are memory mapped into this space at address locations 0FC to 0FE Hex respectively, with the other registers (except 0FF) being available for general usage. Address location 0FF is reserved for future RAM expansion. If compatibility with future devices (with more RAM) is not desired, this location can be used as a general purpose RAM location.

The instruction set permits any bit in memory to be set, reset or tested. All I/O and registers (except A and PC) are memory mapped; therefore, I/O bits and register bits can be directly and individually set, reset and tested. The accumulator (A) bits can also be directly and individually tested.

RAM contents are undefined upon power-up.

TABLE I. Program/Data Memory Sizes

Device	Program Memory (Bytes)	Data Memory (Bytes)	User Storage (Bytes)
COP8SAA7	1024	64	8
COP8SAB7	2048	128	8
COP8SAC7	4096	128	8

5.4 ECON (EPROM CONFIGURATION) REGISTER

The ECON register is used to configure the user selectable clock, security, RAM size, power-on reset, WATCHDOG, and HALT options. The register can be programmed and read only in EPROM programming mode. Therefore, the register should be programmed at the same time as the program memory. The contents of the ECON register shipped from the factory read 00 Hex (windowed device) or 80 Hex (OTP device).

The format of the ECON register is as follows:

Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
X	POR	SECURITY	CKI 2	CKI 1	WATCH DOG	Reserved	HALT

Bit 7 = x This is for factory test. The polarity is "Don't Care."

Bit 6 = 1 Power-on reset enabled.

= 0 Power-on reset disabled.

Bit 5 = 1 Security enabled. EPROM read and write are not allowed.

= 0 Security disabled. EPROM read and write are allowed.

Bits 4, 3

= 0, 0 External CKI option selected. G7 is available as a HALT restart and/or general purpose input. CKI is clock input.

= 0, 1 R/C oscillator option selected. G7 is available as a HALT restart and/or general purpose input. CKI clock input. Internal R/C components are supplied for maximum R/C frequency.

= 1, 0 Crystal oscillator with on-chip crystal bias resistor disabled. G7 (CKO) is the clock generator output to crystal/resonator.

= 1, 1 Crystal oscillator with on-chip crystal bias resistor enabled. G7 (CKO) is the clock generator output to crystal/resonator.

Bit 2 = 1 WATCHDOG feature disabled. G1 is a general purpose I/O.

= 0 WATCHDOG feature enabled. G1 pin is WATCHDOG output with weak pullup.

Bit 1 = Reserved.

Bit 0 = 1 HALT mode disabled.

= 0 HALT mode enabled.

5.0 Functional Description (Continued)

5.5 USER STORAGE SPACE IN EPROM

There are 8 bytes of user storage space in the EPROM that are not read protected by the security bit in the ECON register. When the security bit in the ECON register is set, data in User Storage Space is write-enabled, and read-enabled. This allows the user to read and write this information while still protecting the main EPROM from tampering.

The 8 bytes of user storage space are outside the normal address range of the device, and cannot be accessed by software. This allows for the storage of non-secure information. Typical uses of this are serial numbers, data codes, copyright informations, software version, or lot numbers.

To place information into this area, the user can place the following in the assembly file:

1. Place data in the 8 bytes of user storage space
```
Data is 1 2 3 4 5 6 7 8
.USER=0x01 0x02 0x03 m0x04 0x05 0x06 0x07
0x08
```
2. Place assembly date in 8 bytes of user storage space
Date is in format DD MM YY 00 HH MM SS (all information is in Hex)
```
.USER=ASS__DATE
```
3. Place programming date in 8 bytes of user storage space
Date is in format DD MM YY 00 HH MM SS (all information is in Hex)
```
.USER=PRG__DATE
```
4. To place data in both memory space (for example, 4F9–4FF) to be read out using the LAID instruction, and the user storage space
```
Data is 1 2 3 4 5 6 7 8
. =0x4F9
.BYTE 0x01 0x02 0x03 m0x04 0x05 0x06 0x07 0x08
.USER=0x01 0x02 0x03 m0x04 0x05 0x06 0x07
0x08
```

Note: Not all programmers support the PRG__Date option. Other serialization schemes are currently being worked on with device programming manufacturers. Please contact your device programmer supplier or National for more information.

5.6 OTP SECURITY

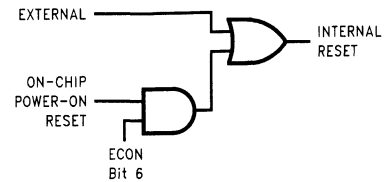
The device has a security feature, when enabled, that prevents external reading of the OTP program memory. The security bit in the ECON register determines, whether security is enabled or disabled. If the security feature is disabled, the contents of the internal EPROM may be read.

If the security feature is enabled, then any attempt to externally read the contents of the EPROM will result in the value FF Hex being read from all program locations. In addition, with the security feature enabled, the write operation to the EPROM program memory and ECON register is inhibited. The ECON register is readable regardless of the state of the security bit.

If security is being used, it is recommended that all other bits in the ECON register be programmed first. Then the security bit can be programmed.

5.7 RESET

The device is initialized when the $\overline{\text{RESET}}$ pin is pulled low or the On-chip Power-On Reset is enabled.



TL/DD/12838–13

FIGURE 8. Reset Logic

The following occurs upon initialization:

Port L: TRISTATE

Port C: TRISTATE

Port G: TRISTATE

Port F: TRISTATE

Port D: HIGH

PC: CLEARED to 0000

PSW, CNTRL and ICNTRL registers: CLEARED

SIOR: UNAFFECTED after RESET with power already applied

RANDOM after RESET at power-on

T1CNTRL: CLEARED

Accumulator, Timer 1:

RANDOM after RESET with crystal clock option (power already applied)

UNAFFECTED after RESET with R/C clock option (power already applied)

RANDOM after RESET at power-on

WKEN, WKEDG: CLEARED

WKPND: RANDOM

SP (Stack Pointer):

Initialized to RAM address 02F Hex (devices with 64 bytes of RAM), or initialized to RAM address 06F Hex (devices with 128 bytes of RAM).

B and X Pointers:

UNAFFECTED after RESET with power already applied

RANDOM after RESET at power-on

RAM:

UNAFFECTED after RESET with power already applied

RANDOM after RESET at power-on

WATCHDOG (if enabled):

The device comes out of reset with both the WATCHDOG logic and the Clock Monitor detector armed, with the WATCHDOG service window bits set and the Clock Monitor bit set. The WATCHDOG and Clock Monitor circuits are inhibited during reset. The WATCHDOG service window bits being initialized high default to the maximum WATCHDOG service window of 64k t_C clock cycles. The Clock Monitor bit being initialized high will cause a Clock Monitor error following reset if the clock has not reached the minimum specified frequency at the termination of reset. A Clock Monitor error will cause an active low error output on pin G1. This error output will continue until 16 t_C–32 t_C clock cycles following the clock frequency reaching the minimum specified value, at which time the G1 output will go high.

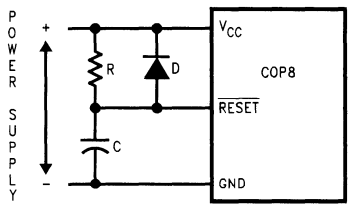
5.0 Functional Description (Continued)

5.7.1 External Reset

The $\overline{\text{RESET}}$ input when pulled low initializes the device. The $\overline{\text{RESET}}$ pin must be held low for a minimum of one instruction cycle to guarantee a valid reset. During Power-Up initialization, the user must ensure that the $\overline{\text{RESET}}$ pin is held low until the device is within the specified V_{CC} voltage. An R/C circuit on the $\overline{\text{RESET}}$ pin with a delay 5 times (5x) greater than the power supply rise time or 15 μs whichever is greater, is recommended. Reset should also be wide enough to ensure crystal start-up upon Power-Up.

$\overline{\text{RESET}}$ may also be used to cause an exit from the HALT mode.

A recommended reset circuit for this deviced is shown in *Figure 9*.



TL/DD/12838–14

RC > 5x power supply rise time or 15 μs, whichever is greater.

FIGURE 9. Reset Circuit Using External Reset

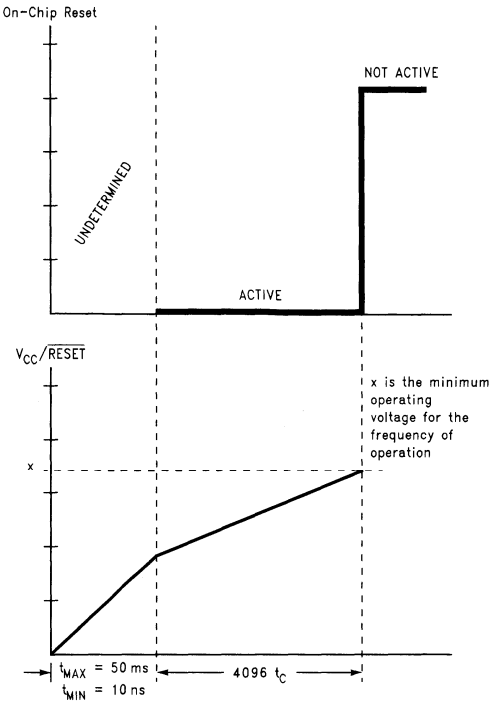
5.7.2 On-Chip Power-On Reset

The on-chip reset circuit is selected by a bit in the ECON register. When enabled, the device generates an internal reset as V_{CC} rises to a voltage level above 2.0V. The on-chip reset circuitry is able to detect both fast and slow rise times on V_{CC} (V_{CC} rise time between 10 ns and 50 ms).

Under no circumstances should the $\overline{\text{RESET}}$ pin be allowed to float. If the on-chip Power-On Reset feature is being used, $\overline{\text{RESET}}$ pin should be connected directly to V_{CC}. The output of the power-on reset detector will always preset the Idle timer to 0FFF(4096 t_C). At this time, the internal reset will be generated.

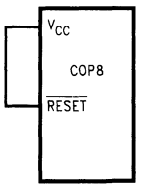
If the Power-On Reset feature is enabled, the internal reset will not be turned off until the Idle timer underflows. The internal reset will perform the same functions as external reset. The user is responsible for ensuring that V_{CC} is at the minimum level for the operating frequency within the 4096 t_C. After the underflow, the logic is designed such that no additional internal resets occur as long as V_{CC} remains above 2.0V.

The contents of data registers and RAM are unknown following the on-chip reset.



TL/DD/12838–15

FIGURE 10. Reset Timing (Power-On Reset Enabled) with V_{CC} Tied to $\overline{\text{RESET}}$



TL/DD/12838–16

FIGURE 11. Reset Circuit Using Power-On Reset

5.0 Functional Description (Continued)

5.8 OSCILLATOR CIRCUITS

There are four clock oscillator options available: Crystal Oscillator with or without on-chip bias resistor, R/C Oscillator with on-chip resistor and capacitor, and External Oscillator. The oscillator feature is selected by programming the ECON register, which is summarized in Table II.

TABLE II. Oscillator Option

ECON4	ECON3	Oscillator Option
0	0	External Oscillator
1	0	Crystal Oscillator without Bias Resistor
0	1	R/C Oscillator
1	1	Crystal Oscillator with Bias Resistor

5.8.1 Crystal Oscillator

The crystal Oscillator mode can be selected by programming ECON Bit 4 to 1. CKI is the clock input while G7/CKO is the clock generator output to the crystal. An on-chip bias resistor connected between CKI and CKO can be enabled by programming ECON Bit 3 to 1 with the crystal oscillator option selection. The value of the resistor is in the range of 0.5M to 2M (typically 1.0M). Table III shows the component values required for various standard crystal values. Resistor R2 is only used when the on-chip bias resistor is disabled. *Figure 12* shows the crystal oscillator connection diagram.

TABLE III. Crystal Oscillator Configuration, $T_A = 25°C$, $V_{CC} = 5V$

R1 (kΩ)	R2 (MΩ)	C1 (pF)	C2 (pF)	CKI Freq. (MHz)
0	1	30	30	15
0	1	32	32	10
0	1	45	30–36	4
5.6	1	100	100–156	0.455

5.8.2 External Oscillator

The External Oscillator mode can be selected by programming ECON Bit 3 to 0 and ECON Bit 4 to 0. CKI can be driven by an external clock signal provided it meets the specified duty cycle, rise and fall times, and input levels. G7/CKO is available as a general purpose input G7 and/or Halt control. *Figure 13* shows the external oscillator connection diagram.

5.8.3 R/C Oscillator

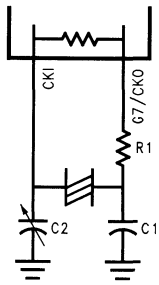
The R/C Oscillator mode can be selected by programming ECON Bit 3 to 1 and ECON Bit 4 to 0. In R/C oscillation mode, CKI is left floating, while G7/CKO is available as a general purpose input G7 and/or HALT control. The R/C controlled oscillator has on-chip resistor and capacitor for maximum R/C oscillator frequency operation. The maximum frequency is 5 MHz ± 35% for V_{CC} between 4.5V to 5.5V and temperature range of −40°C to +85°C. For max frequency operation, the CKI pin should be left floating. For lower frequencies, an external capacitor should be connected between CKI and GND. Table IV shows the oscillator frequency as a function of external capacitance on the CKI pin. *Figure 14* shows the R/C oscillator configuration.

TABLE IV. R/C Oscillator Configuration, −40°C to +85°C, V_{CC} = 4.5V to 5.5V, OSC Freq. Variation of ± 35%

External Capacitor (pF)	R/C OSC Freq (MHz)	Instr. Cycle (μs)
0	5	2.0
9	4	2.5
52	2	5.0
150	1	10
TBD	32 kHz	312.5

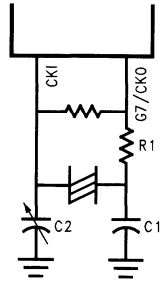
5.0 Functional Description (Continued)

With On-Chip Bias Resistor



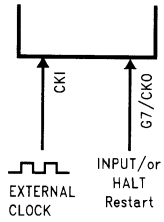
TL/DD/12838–17

Without On-Chip Bias Resistor



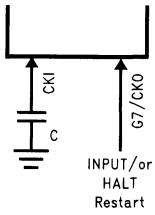
TL/DD/12838–18

FIGURE 12. Crystal Oscillator



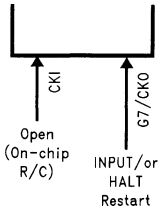
TL/DD/12838–19

FIGURE 13. External Oscillator



TL/DD/12838–20

For operation at lower than maximum R/C oscillator frequency.



TL/DD/12838–21

For operation at maximum R/C oscillator frequency.

FIGURE 14. R/C Oscillator

2

5.0 Functional Description (Continued)

5.9 CONTROL REGISTERS

CNTRL Register (Address X'00EE)

The Timer1 (T1) and MICROWIRE/PLUS control register contains the following bits:

SL1 & SL0	Select the MICROWIRE/PLUS clock divide by (00 = 2, 01 = 4, 1x = 8)
IEDG	External interrupt edge polarity select (0 = Rising edge, 1 = Falling edge)
MSEL	Selects G5 and G4 as MICROWIRE/PLUS signals SK and SO respectively
T1C0	Timer T1 Start/Stop control in timer modes 1 and 2 T1 Underflow Interrupt Pending Flag in timer mode 3
T1C1	Timer T1 mode control bit
T1C2	Timer T1 mode control bit
T1C3	Timer T1 mode control bit

T1C3	T1C2	T1C1	T1C0	MSEL	IEDG	SL1	SL0

Bit 7 Bit 0

PSW Register (Address X'00EF)

The PSW register contains the following select bits:

GIE	Global interrupt enable (enables interrupts)
EXEN	Enable external interrupt
BUSY	MICROWIRE/PLUS busy shifting flag
EXPND	External interrupt pending
T1ENA	Timer T1 Interrupt Enable for Timer Underflow or T1A Input capture edge
T1PNDA	Timer T1 Interrupt Pending Flag (Autoreload RA in mode 1, T1 Underflow in Mode 2, T1A capture edge in mode 3)
C	Carry Flag
HC	Half Carry Flag

HC	C	T1PNDA	T1ENA	EXPND	BUSY	EXEN	GIE

Bit 7 Bit 0

The Half-Carry flag is also affected by all the instructions that affect the Carry flag. The SC (Set Carry) and R/C (Reset Carry) instructions will respectively set or clear both the carry flags. In addition to the SC and R/C instructions, ADC, SUBC, RRC and RLC instructions affect the Carry and Half Carry flags.

ICNTRL Register (Address X'00E8)

The ICNTRL register contains the following bits:

T1ENB	Timer T1 Interrupt Enable for T1B Input capture edge
T1PNDB	Timer T1 Interrupt Pending Flag for T1B capture edge
μWEN	Enable MICROWIRE/PLUS interrupt
μWPND	MICROWIRE/PLUS interrupt pending
T0EN	Timer T0 Interrupt Enable (Bit 12 toggle)
T0PND	Timer T0 Interrupt pending
LPEN	L Port Interrupt Enable (Multi-Input Wakeup/Interrupt)

Bit 7 could be used as a general purpose status flag

Unused	LPEN	T0PND	T0EN	WPND	WEN	T1PNDB	T1ENB

Bit 7 Bit 0

6.0 Timers

The device contains a very versatile set of timers (T0, T1). Timer T1 and associated autoreload/capture registers power up containing random data.

6.1 TIMER T0 (IDLE TIMER)

The device supports applications that require maintaining real time and low power with the IDLE mode. This IDLE mode support is furnished by the IDLE timer T0. The Timer T0 runs continuously at the fixed rate of the instruction cycle clock, t_C. The user cannot read or write to the IDLE Timer T0, which is a count down timer.

The Timer T0 supports the following functions:

- Exit out of the Idle Mode (See Idle Mode description)
- WATCHDOG logic (See WATCHDOG description)
- Start up delay out of the HALT mode
- Timing the width of the internal power-on-reset

The IDLE Timer T0 can generate an interrupt when the twelfth bit toggles. This toggle is latched into the T0PND pending flag, and will occur every 4.096 ms at the maximum clock frequency (t_C = 1 μs). A control flag T0EN allows the interrupt from the twelfth bit of Timer T0 to be enabled or disabled. Setting T0EN will enable the interrupt, while resetting it will disable the interrupt.

6.2 TIMER T1

One of the main functions of a microcontroller is to provide timing and counting capability for real-time control tasks. The COP888 family offers a very versatile 16-bit timer/counter structure, and two supporting 16-bit autoreload/capture

6.0 Timers (Continued)

registers (R1A and R1B), optimized to reduce software burdens in real-time control applications. The timer block has two pins associated with it, T1A and T1B. Pin T1A supports I/O required by the timer block, while pin T1B is an input to the timer block.

The timer block has three operating modes: Processor Independent PWM mode, External Event Counter mode, and Input Capture mode.

The control bits T1C3, T1C2, and T1C1 allow selection of the different modes of operation.

6.2.1 Mode 1. Processor Independent PWM Mode

One of the timer's operating modes is the Processor Independent PWM mode. In this mode, the timer generates a "Processor Independent" PWM signal because once the timer is setup, no more action is required from the CPU which translates to less software overhead and greater throughput. The user software services the timer block only when the PWM parameters require updating. This capability is provided by the fact that the timer has two separate 16-bit reload registers. One of the reload registers contains the "ON" timer while the other holds the "OFF" time. By contrast, a microcontroller that has only a single reload register requires an additional software to update the reload value (alternate between the on-time/off-time).

The timer can generate the PWM output with the width and duty cycle controlled by the values stored in the reload registers. The reload registers control the countdown values and the reload values are automatically written into the timer when it counts down through 0, generating interrupt on each reload. Under software control and with minimal overhead, the PMW outputs are useful in controlling motors, triacs, the intensity of displays, and in providing inputs for data acquisition and sine wave generators.

In this mode, the timer T1 counts down at a fixed rate of t_C. Upon every underflow the timer is alternately reloaded with the contents of supporting registers, R1A and R1B. The very first underflow of the timer causes the timer to reload from the register R1A. Subsequent underflows cause the timer to be reloaded from the registers alternately beginning with the register R1B.

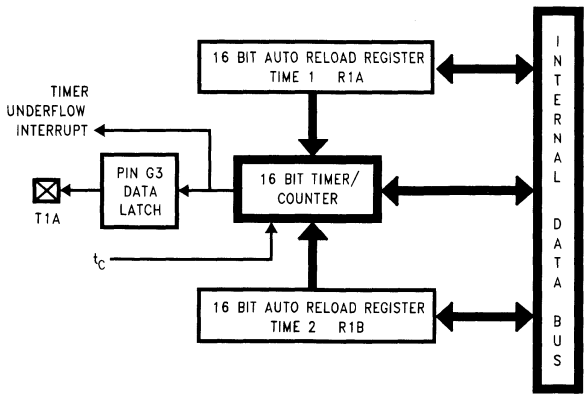
The T1 Timer control bits, T1C3, T1C2 and T1C1 set up the timer for PWM mode operation.

Figure 15 shows a block diagram of the timer in PWM mode.

The underflows can be programmed to toggle the T1A output pin. The underflows can also be programmed to generate interrupts.

Underflows from the timer are alternately latched into two pending flags, T1PNDA and T1PNDB. The user must reset these pending flags under software control. Two control enable flags, T1ENA and T1ENB, allow the interrupts from the timer underflow to be enabled or disabled. Setting the timer enable flag T1ENA will cause an interrupt when a timer underflow causes the R1A register to be reloaded into the timer. Setting the timer enable flag T1ENB will cause an interrupt when a timer underflow causes the R1B register to be reloaded into the timer. Resetting the timer enable flags will disable the associated interrupts.

Either or both of the timer underflow interrupts may be enabled. This gives the user the flexibility of interrupting once per PWM period on either the rising or falling edge of the PWM output. Alternatively, the user may choose to interrupt on both edges of the PWM output.



TL/DD/12838-22

FIGURE 15. Timer in PWM Mode

6.0 Timers (Continued)

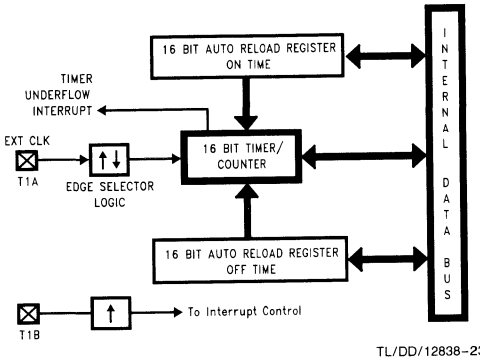
6.2.2 Mode 2. External Event Counter Mode

This mode is quite similar to the processor independent PWM mode described above. The main difference is that the timer, T1, is clocked by the input signal from the T1A pin. The T1 timer control bits, T1C3, T1C2 and T1C1 allow the timer to be clocked either on a positive or negative edge from the T1A pin. Underflows from the timer are latched into the T1PNDA pending flag. Setting the T1ENA control flag will cause an interrupt when the timer underflows.

In this mode the input pin T1B can be used as an independent positive edge sensitive interrupt input if the T1ENB control flag is set. The occurrence of a positive edge on the T1B input pin is latched into the T1PNDB flag.

Figure 16 shows a block diagram of the timer in External Event Counter mode.

Note: The PWM output is not available in this mode since the T1A pin is being used as the counter input clock.



TL/DD/12838-23

FIGURE 16. Timer in External Event Counter Mode

6.2.3 Mode 3. Input Capture Mode

The device can precisely measure external frequencies or time external events by placing the timer block, T1, in the input capture mode. In this mode, the reload registers serve as independent capture registers, capturing the contents of the timer when an external event occurs (transition on the timer input pin). The capture registers can be read while maintaining count, a feature that lets the user measure elapsed time and time between events. By saving the timer value when the external event occurs, the time of the external event is recorded. Most microcontrollers have a latency time because they cannot determine the timer value when the external event occurs. The capture register eliminates the latency time, thereby allowing the applications program to retrieve the timer value stored in the capture register.

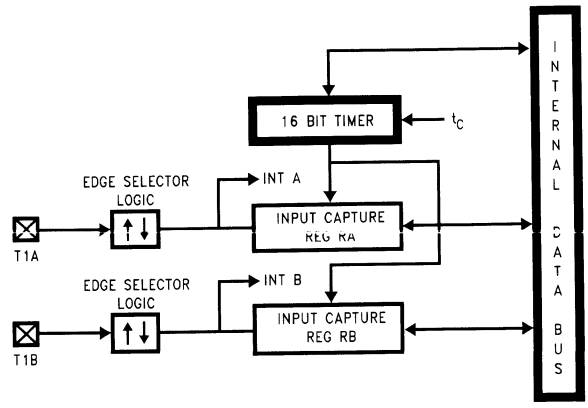
In this mode, the timer T1 is constantly running at the fixed t_C rate. The two registers, R1A and R1B, act as capture registers. Each register acts in conjunction with a pin. The register R1A acts in conjunction with the T1A pin and the register R1B acts in conjunction with the T1B pin.

The timer value gets copied over into the register when a trigger event occurs on its corresponding pin. Control bits, T1C3, T1C2 and T1C1, allow the trigger events to be specified either as a positive or a negative edge. The trigger condition for each input pin can be specified independently.

The trigger conditions can also be programmed to generate interrupts. The occurrence of the specified trigger condition on the T1A and T1B pins will be respectively latched into the pending flags, T1PNDA and T1PNDB. The control flag T1ENA allows the interrupt on T1A to be either enabled or disabled. Setting the T1ENA flag enables interrupts to be generated when the selected trigger condition occurs on the T1A pin. Similarly, the flag T1ENB controls the interrupts from the T1B pin.

Underflows from the timer can also be programmed to generate interrupts. Underflows are latched into the timer T1C0 pending flag (the T1C0 control bit serves as the timer underflow interrupt pending flag in the Input Capture mode). Consequently, the T1C0 control bit should be reset when entering the Input Capture mode. The timer underflow interrupt is enabled with the T1ENA control flag. When a T1A interrupt occurs in the Input Capture mode, the user must check both the T1PNDA and T1C0 pending flags in order to determine whether a T1A input capture or a timer underflow (or both) caused the interrupt.

Figure 17 shows a block diagram of the timer in Input Capture mode.



TL/DD/12838-24

FIGURE 17. Timer in Input Capture Mode

6.0 Timers (Continued)

6.3 TIMER CONTROL FLAGS

The control bits and their functions are summarized below.

T1C0 Timer Start/Stop control in Modes 1 and 2 (Processor Independent PWM and External Event Counter), where 1 = Start, 0 = Stop
Timer Underflow Interrupt Pending Flag in Mode 3 (Input Capture)

T1PNDA Timer Interrupt Pending Flag
T1PNDB Timer Interrupt Pending Flag

T1ENA Timer Interrupt Enable Flag
T1ENB Timer Interrupt Enable Flag
1 = Timer Interrupt Enabled
0 = Timer Interrupt Disabled

T1C3 Timer mode control
T1C2 Timer mode control
T1C1 Timer mode control

The timer mode control bits (T1C3, T1C2 and T1C1) are detailed below:

T1C3	T1C2	T1C1	Timer Mode	Interrupt A Source	Interrupt B Source	Timer Counts On
0	0	0	MODE 2 (External Event Counter)	Timer Underflow	Pos. T1B Edge	T1A Pos. Edge
0	0	1	MODE 2 (External Event Counter)	Timer Underflow	Pos. T1B Edge	T1A Neg. Edge
1	0	1	MODE 1 (PWM) T1A Toggle	Autoreload RA	Autoreload RB	t_C
1	0	0	MODE 1 (PWM) No T1A Toggle	Autoreload RA	Autoreload RB	t_C
0	1	0	MODE 3 (Capture) Captures: T1A Pos. Edge T1B Pos. Edge	Pos. T1A Edge or Timer Underflow	Pos. T1B Edge	t_C
1	1	0	MODE 3 (Capture) Captures: T1A Pos. Edge T1B Neg. Edge	Pos. T1A Edge or Timer Underflow	Neg. T1B Edge	t_C
0	1	1	MODE 3 (Capture) Captures: T1A Neg. Edge T1B Pos. Edge	Neg. T1A Edge or Timer Underflow	Pos. T1B Edge	t_C
1	1	1	MODE 3 (Capture) Captures: T1A Neg. Edge T1B Neg. Edge	Neg. T1A Edge or Timer Underflow	Neg. T1B Edge	t_C

7.0 Power Save Modes

Today, the proliferation of battery-operated based applications has placed new demands on designers to drive power consumption down. Battery-operated systems are not the only type of applications demanding low power. The power budget constraints are also imposed on those consumer/industrial applications where well regulated and expensive power supply costs cannot be tolerated. Such applications rely on low cost and low power supply voltage derived directly from the "mains" by using voltage rectifier and passive components. Low power is demanded even in automotive applications, due to increased vehicle electronics content. This is required to ease the burden from the car battery. Low power 8-bit microcontrollers supply the smarts to control battery-operated, consumer/industrial, and automotive applications.

The COPSAx7 devices offers system designers a variety of low-power consumption features that enable them to meet the demanding requirements of today's increasing range of low-power applications. These features include low voltage operation, low current drain, and power saving features such as HALT, IDLE, and Multi-Input wakeup (MIWU).

The devices offers the user two power save modes of operation: HALT and IDLE. In the HALT mode, all microcontroller activities are stopped. In the IDLE mode, the on-board oscillator circuitry and timer T0 are active but all other microcontroller activities are stopped. In either mode, all on-board RAM, registers, I/O states, and timers (with the exception of T0) are unaltered.

Clock Monitor if enabled can be active in both modes.

7.1 HALT MODE

The device can be placed in the HALT mode by writing a "1" to the HALT flag (G7 data bit). All microcontroller activities, including the clock and timers, are stopped. The WATCHDOG logic on the device is disabled during the HALT mode. However, the clock monitor circuitry, if enabled, remains active and will cause the WATCHDOG output pin (WDOUT) to go low. If the HALT mode is used and the user does not want to activate the WDOUT pin, the Clock Monitor should be disabled after the device comes out of reset (resetting the Clock Monitor control bit with the first write to the WDSVR register). In the HALT mode, the power requirements of the device are minimal and the applied voltage (V_{CC}) may be decreased to V_r (V_r = 2.0V) without altering the state of the machine.

The device supports three different ways of exiting the HALT mode. The first method of exiting the HALT mode is with the Multi-Input Wakeup feature on Port L. The second method is with a low to high transition on the CKO (G7) pin. This method precludes the use of the crystal clock configuration (since CKO becomes a dedicated output), and so may only be used with an R/C clock configuration. The third method of exiting the HALT mode is by pulling the $\overline{\text{RESET}}$ pin low.

Since a crystal or ceramic resonator may be selected as the oscillator, the Wakeup signal is not allowed to start the chip running immediately since crystal oscillators and ceramic resonators have a delayed start up time to reach full amplitude and frequency stability. The IDLE timer is used to generate a fixed delay to ensure that the oscillator has indeed stabilized before allowing instruction execution. In this case, upon detecting a valid Wakeup signal, only the oscillator circuitry is enabled. The IDLE timer is loaded with a value of 256 and is clocked with the t_C instruction cycle clock. The t_C clock is derived by dividing the oscillator clock down by a factor of 10. The Schmitt trigger following the CKI inverter on the chip ensures that the IDLE timer is clocked only when the oscillator has a sufficiently large amplitude to meet the Schmitt trigger specifications. This Schmitt trigger is not part of the oscillator closed loop. The start-up time-out from the IDLE timer enables the clock signals to be routed to the rest of the chip.

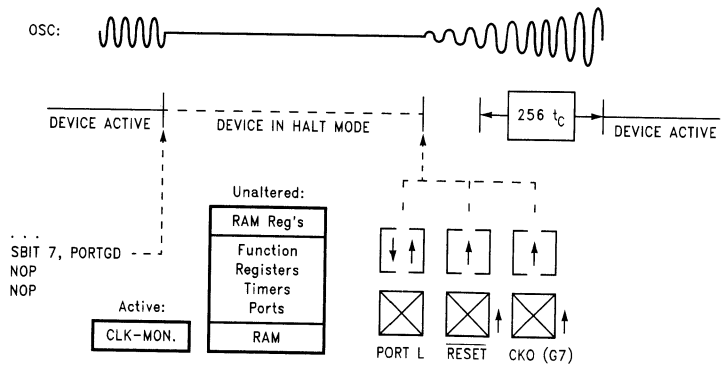
If an R/C clock option is being used, the fixed delay is introduced optionally. A control bit, CLKDLY, mapped as configuration bit G7, controls whether the delay is to be introduced or not. The delay is included if CLKDLY is set, and excluded if CLKDLY is reset. The CLKDLY bit is cleared on reset.

The device has two options associated with the HALT mode. The first option enables the HALT mode feature, while the second option disables the HALT mode selected through bit 0 of the ECON register. With the HALT mode enable option, the device will enter and exit the HALT mode as described above. With the HALT disable option, the device cannot be placed in the HALT mode (writing a "1" to the HALT flag will have no effect, the HALT flag will remain "0").

The WATCHDOG detector circuit is inhibited during the HALT mode. However, the clock monitor circuit if enabled remains active during HALT mode in order to ensure a clock monitor error if the device inadvertently enters the HALT mode as a result of a runaway program or power glitch.

If the device is placed in the HALT mode, with the R/C oscillator selected, the clock input pin (CKI) is forced to a logic high internally. With the crystal or external oscillator the CKI pin is TRI-STATE.

7.0 Power Save Modes (Continued)



TL/DD/12838-25

FIGURE 18. Wakeup from HALT

7.2 IDLE MODE

The device is placed in the IDLE mode by writing a "1" to the IDLE flag (G6 data bit). In this mode, all activities, except the associated on-board oscillator circuitry and the IDLE Timer T0, are stopped.

As with the HALT mode, the device can be returned to normal operation with a reset, or with a Multi-Input Wakeup from the L Port. Alternately, the microcontroller resumes normal operation from the IDLE mode when the twelfth bit (representing 4.096 ms at internal clock frequency of 10 MHz, t_C = 1 μs) of the IDLE Timer toggles.

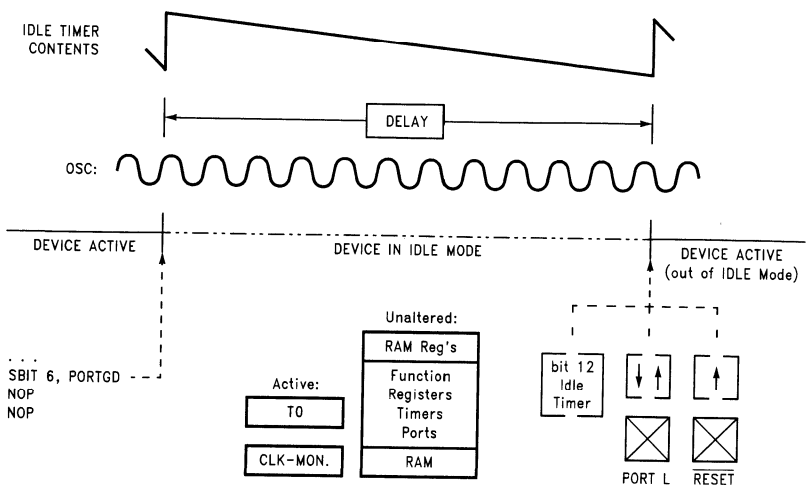
This toggle condition of the twelfth bit of the IDLE Timer T0 is latched into the T0PND pending flag.

The user has the option of being interrupted with a transition on the twelfth bit of the IDLE Timer T0. The interrupt can be enabled or disabled via the T0EN control bit. Setting the T0EN flag enables the interrupt and vice versa.

The user can enter the IDLE mode with the Timer T0 interrupt enabled. In this case, when the T0PND bit gets set, the device will first execute the Timer T0 interrupt service routine and then return to the instruction following the "Enter Idle Mode" instruction.

Alternatively, the user can enter the IDLE mode with the IDLE Timer T0 interrupt disabled. In this case, the device will resume normal operation with the instruction immediately following the "Enter IDLE Mode" instruction.

Note: It is necessary to program two NOP instructions following both the set HALT mode and set IDLE mode instructions. These NOP instructions are necessary to allow clock resynchronization following the HALT or IDLE modes.



TL/DD/12838-26

FIGURE 19. Wakeup from IDLE

7.0 Power Save Modes (Continued)

7.3 MULTI-INPUT WAKEUP

The Multi-Input Wakeup feature is used to return (wakeup) the device from either the HALT or IDLE modes. Alternately Multi-Input Wakeup/Interrupt feature may also be used to generate up to 8 edge selectable external interrupts.

Figure 20 shows the Multi-Input Wakeup logic.

The Multi-Input Wakeup feature utilizes the L Port. The user selects which particular L port bit (or combination of L Port bits) will cause the device to exit the HALT or IDLE modes. The selection is done through the register WKEN. The register WKEN is an 8-bit read/write register, which contains a control bit for every L port bit. Setting a particular WKEN bit enables a Wakeup from the associated L port pin.

The user can select whether the trigger condition on the selected L Port pin is going to be either a positive edge (low to high transition) or a negative edge (high to low transition). This selection is made via the register WKEDG, which is an 8-bit control register with a bit assigned to each L Port pin. Setting the control bit will select the trigger condition to be a negative edge on that particular L Port pin. Resetting the bit selects the trigger condition to be a positive edge. Changing an edge select entails several steps in order to avoid a Wakeup condition as a result of the edge change. First, the associated WKEN bit should be reset, followed by the edge select change in WKEDG. Next, the associated WKPND bit should be cleared, followed by the associated WKEN bit being re-enabled.

An example may serve to clarify this procedure. Suppose we wish to change the edge select from positive (low going high) to negative (high going low) for L Port bit 5, where bit 5 has previously been enabled for an input interrupt. The program would be as follows:

```
RBIT  5, WKEN    ; Disable MIWU
SBIT  5, WKEDG   ; Change edge polarity
RBIT  5, WKPND   ; Reset pending flag
SBIT  5, WKEN    ; Enable MIWU
```

If the L port bits have been used as outputs and then changed to inputs with Multi-Input Wakeup/Interrupt, a safety procedure should also be followed to avoid wakeup conditions. After the selected L port bits have been changed from output to input but before the associated WKEN bits are enabled, the associated edge select bits in WKEDG should be set or reset for the desired edge selects, followed by the associated WKPND bits being cleared.

This same procedure should be used following reset, since the L port inputs are left floating as a result of reset.

The occurrence of the selected trigger condition for Multi-Input Wakeup is latched into a pending register called WKPND. The respective bits of the WKPND register will be set on the occurrence of the selected trigger edge on the corresponding Port L pin. The user has the responsibility of clearing these pending flags. Since WKPND is a pending register for the occurrence of selected wakeup conditions, the device will not enter the HALT mode if any Wakeup bit is both enabled and pending. Consequently, the user must clear the pending flags before attempting to enter the HALT mode.

WKEN and WKEDG are all read/write registers, and are cleared at reset. WKPND register contains random value after reset.

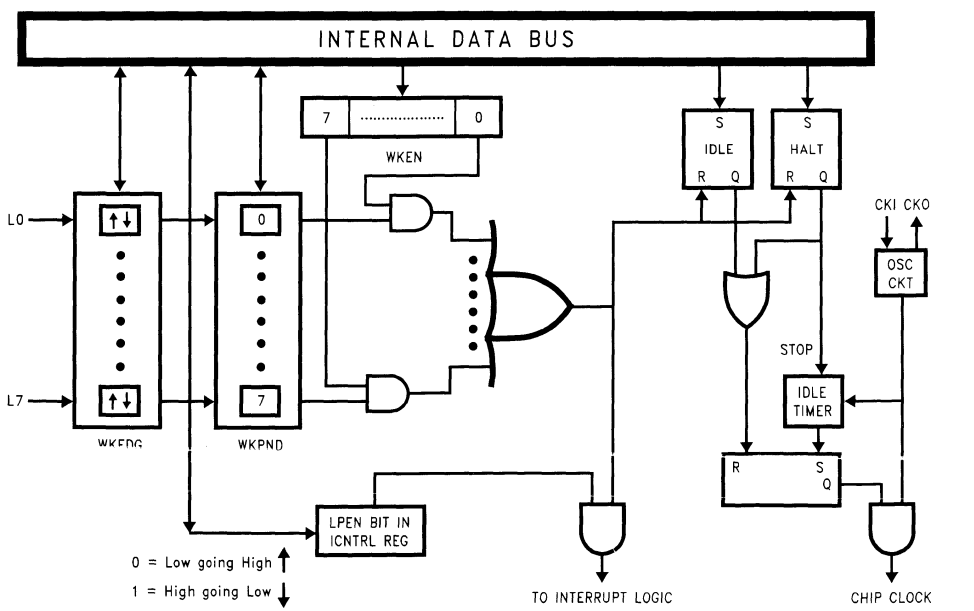


FIGURE 20. Multi-Input Wake Up Logic

8.0 Interrupts

8.1 INTRODUCTION

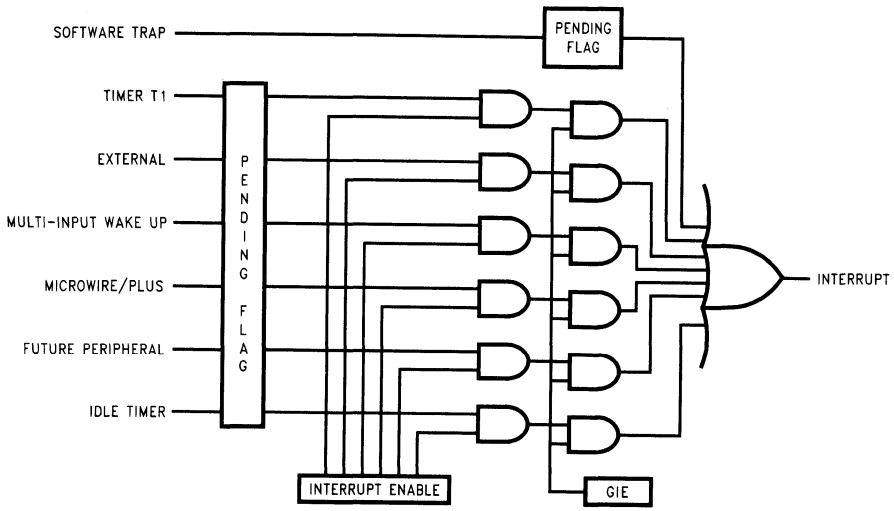
The device supports eight vectored interrupts. Interrupt sources include Timer 1, Timer T0, Port L Wakeup, Software Trap, MICROWIRE/PLUS, and External Input.

All interrupts force a branch to location 00FF Hex in program memory. The VIS instruction may be used to vector to the appropriate service routine from location 00FF Hex.

The Software trap has the highest priority while the default VIS has the lowest priority.

Each of the six maskable inputs has a fixed arbitration ranking and vector.

Figure 21 shows the Interrupt Block Diagram.



TL/DD/12838–28

FIGURE 21. Interrupt Block Diagram

8.0 Interrupts (Continued)

8.2 MASKABLE INTERRUPTS

All interrupts other than the Software Trap are maskable. Each maskable interrupt has an associated enable bit and pending flag bit. The pending bit is set to 1 when the interrupt condition occurs. The state of the interrupt enable bit, combined with the GIE bit determines whether an active pending flag actually triggers an interrupt. All of the maskable interrupt pending and enable bits are contained in mapped control registers, and thus can be controlled by the software.

A maskable interrupt condition triggers an interrupt under the following conditions:

1. The enable bit associated with that interrupt is set.
2. The GIE bit is set.
3. The device is not processing a non-maskable interrupt. (If a non-maskable interrupt is being serviced, a maskable interrupt must wait until that service routine is completed.)

An interrupt is triggered only when all of these conditions are met at the beginning of an instruction. If different maskable interrupts meet these conditions simultaneously, the highest priority interrupt will be serviced first, and the other pending interrupts must wait.

Upon Reset, all pending bits, individual enable bits, and the GIE bit are reset to zero. Thus, a maskable interrupt condition cannot trigger an interrupt until the program enables it by setting both the GIE bit and the individual enable bit. When enabling an interrupt, the user should consider whether or not a previously activated (set) pending bit should be acknowledged. If, at the time an interrupt is enabled, any previous occurrences of the interrupt should be ignored, the associated pending bit must be reset to zero prior to enabling the interrupt. Otherwise, the interrupt may be simply enabled; if the pending bit is already set, it will immediately trigger an interrupt. A maskable interrupt is active if its associated enable and pending bits are set.

An interrupt is an asychronous event which may occur before, during, or after an instruction cycle. Any interrupt which occurs during the execution of an instruction is not acknowledged until the start of the next normally executed instruction is to be skipped, the skip is performed before the pending interrupt is acknowledged.

At the start of interrupt acknowledgment, the following actions occur:

1. The GIE bit is automatically reset to zero, preventing any subsequent maskable interrupt from interrupting the current service routine. This feature prevents one maskable interrupt from interrupting another one being serviced.
2. The address of the instruction about to be executed is pushed onto the stack.
3. The program counter (PC) is loaded with 00FF Hex, causing a jump to that program memory location.

The device requires seven instruction cycles to perform the actions listed above.

If the user wishes to allow nested interrupts, the interrupts service routine may set the GIE bit to 1 by writing to the PSW register, and thus allow other maskable interrupts to interrupt the current service routine. If nested interrupts are allowed, caution must be exercised. The user must write the program in such a way as to prevent stack overflow, loss of saved context information, and other unwanted conditions.

The interrupt service routine stored at location 00FF Hex should use the VIS instruction to determine the cause of the interrupt, and jump to the interrupt handling routine corresponding to the highest priority enabled and active interrupt. Alternately, the user may choose to poll all interrupt pending and enable bits to determine the source(s) of the interrupt. If more than one interrupt is active, the user's program must decide which interrupt to service.

Within a specific interrupt service routine, the associated pending bit should be cleared. This is typically done as early as possible in the service routine in order to avoid missing the next occurrence of the same type of interrupt event. Thus, if the same event occurs a second time, even while the first occurrence is still being serviced, the second occurrence will be serviced immediately upon return from the current interrupt routine.

An interrupt service routine typically ends with an RETI instruction. This instruction sets the GIE bit back to 1, pops the address stored on the stack, and restores that address to the program counter. Program execution then proceeds with the next instruction that would have been executed had there been no interrupt. If there are any valid interrupts pending, the highest-priority interrupt is serviced immediately upon return from the previous interrupt.

8.3 VIS INSTRUCTION

The general interrupt service routine, which starts at address 00FF Hex, must be capable of handling all types of interrupts. The VIS instruction, together with an interrupt vector table, directs the device to the specific interrupt handling routine based on the cause of the interrupt.

VIS is a single-byte instruction, typically used at the very beginning of the general interrupt service routine at address 00FF Hex, or shortly after that point, just after the code used for context switching. The VIS instruction determines which enabled and pending interrupt has the highest priority, and causes an indirect jump to the address corresponding to that interrupt source. The jump addresses (vectors) for all possible interrupts sources are stored in a vector table.

The vector table may be as long as 32 bytes (maximum of 16 vectors) and resides at the top of the 256-byte block containing the VIS instruction. However, if the VIS instruction is at the very top of a 256-byte block (such as at 00FF Hex), the vector table resides at the top of the next 256-byte block. Thus, if the VIS instruction is located somewhere between 00FF and 01DF Hex (the usual case), the vector table is located between addresses 01E0 and 01FF Hex. If the VIS instruction is located between 01FF and 02DF Hex, then the vector table is located between addresses 02E0 and 02FF Hex, and so on.

Each vector is 15 bits long and points to the beginning of a specific interrupt service routine somewhere in the 32 kbyte memory space. Each vector occupies two bytes of the vector table, with the higher-order byte at the lower

8.0 Interrupts (Continued)

address. The vectors are arranged in order of interrupt priority. The vector of the maskable interrupt with the lowest rank is located to 0yE0 (higher-order byte) and 0yE1 (lower-order byte). The next priority interrupt is located at 0yE2 and 0yE3, and so forth in increasing rank. The Software Trap has the highest rank and its vector is always located at 0yFE and 0yFF. The number of interrupts which can become active defines the size of the table.

Table V shows the types of interrupts, the interrupt arbitration ranking, and the locations of the corresponding vectors in the vector table.

The vector table should be filled by the user with the memory locations of the specific interrupt service routines. For example, if the Software Trap routine is located at 0310 Hex, then the vector location 0yFE and -0yFF should contain the data 03 and 10 Hex, respectively. When a Software Trap interrupt occurs and the VIS instruction is executed, the program jumps to the address specified in the vector table.

The interrupt sources in the vector table are listed in order of rank, from highest to lowest priority. If two or more enabled and pending interrupts are detected at the same time, the one with the highest priority is serviced first. Upon return from the interrupt service routine, the next highest-level pending interrupt is serviced.

If the VIS instruction is executed, but no interrupts are enabled and pending, the lowest-priority interrupt vector is used, and a jump is made to the corresponding address in the vector table. This is an unusual occurrence, and probably the result of an error. It can result from a change in the enable bits or pending flags prior to using the VIS instruction, or from inadvertent execution of the VIS command outside of the context of an interrupt. It is a good idea to make this vector point to the Software Trap interrupt service routine or some other error handling routine. A normal RETI instruction should not be used in any such routine because the stack might not contain a valid return address.

To ensure reliable operation, the user should always use the VIS instruction to determine the source of an interrupt. Although it is possible to poll the pending bits to detect the source of an interrupt, this practice is not recommended. The use of polling allows the standard arbitration ranking to be altered, but the reliability of the interrupt system is compromised. The polling routine must individually test the enable and pending bits of each maskable interrupt. If a Software Trap interrupt should occur, it will be serviced last, even though it should have the highest priority. Under certain conditions, a Software Trap could be triggered but not serviced, resulting in an inadvertent "locking out" of all maskable interrupts by the Software Trap pending flag. Problems such as this can be avoided by using VIS instruction.

TABLE V. Interrupt Vector Table

Arbitration Ranking	Source	Description	Vector* Address (Hi-Low Byte)
(1) Highest	Software	INTR Instruction	0yFE–0yFF
(2)	Reserved	Future	0yFC–0yFD
(3)	External	G0	0yFA–0yFB
(4)	Timer T0	Underflow	0yF8–0yF9
(5)	Timer T1	T1A/Underflow	0yF6–0yF7
(6)	Timer T1	T1B	0yF4–0yF5
(7)	MICROWIRE/PLUS	BUSY Low	0yF2–0yF3
(8)	Reserved	Future	0yF0–0yF1
(9)	Reserved	Future	0yEE–0yEF
(10)	Reserved	Future	0yEC–0yED
(11)	Reserved	Future	0yEA–0yEB
(12)	Reserved	Future	0yE8–0yE9
(13)	Reserved	Future	0yE6–0yE7
(14)	Reserved	Future	0yE4–0yE5
(15)	Port L/Wakeup	Port L Edge	0yE2–0yE3
(16) Lowest	Default	VIS Instruction Execution without any interrupts	0yE0–0yE1

*y is a variable which represents the VIS block. VIS and the vector table must be located in the same 256-byte block except if VIS is located at the last address of a block. In this case, the table must be in the next block.

8.0 Interrupts (Continued)

8.3.1 VIS Execution

When the VIS instruction is executed it activates the arbitration logic. The arbitration logic generates an even number between E0 and FE (E0, E2, E4, E6 etc...) depending on which active interrupt has the highest arbitration ranking at the time of the 1st cycle of VIS is executed. For example, if the software trap interrupt is active, FE is generated. If the external interrupt is active and the software trap interrupt is not, then FA is generated and so forth. If the only active interrupt is software trap, than E0 is generated. This number replaces the lower byte of the PC. The upper byte of the PC remains unchanged. The new PC is therefore pointing to the vector of the active interrupt with the highest arbitration ranking. This vector is read from program memory and placed into the PC which is now pointed to the 1st instruction of the service routine of the active interrupt with the highest arbitration ranking.

Figure 22 illustrates the different steps performed by the VIS instruction. *Figure 23* shows a flowchart for the VIS instruction.

The non-maskable interrupt pending flag is cleared by the RPND (Reset Non-Maskable Pending Bit) instruction (under certain conditions) and upon RESET.

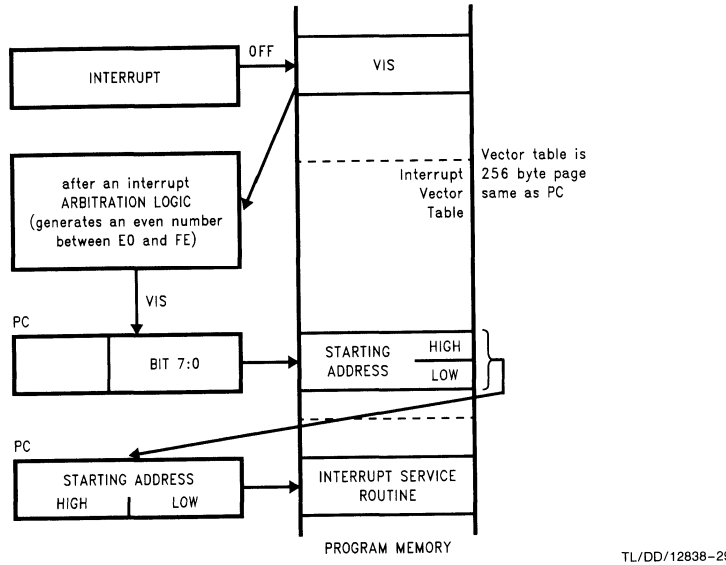
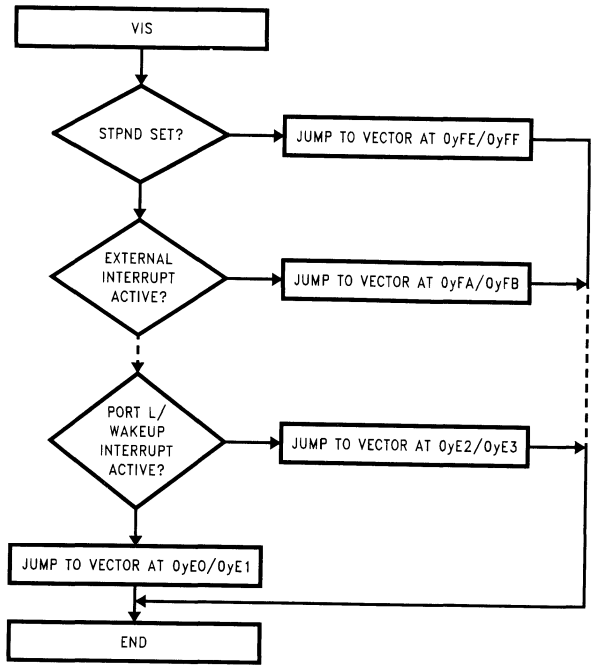


FIGURE 22. VIS Operation

8.0 Interrupts (Continued)



TL/DD/12838–30

FIGURE 23. VIS Flowchart

8.0 Interrupts (Continued)

Programming Example: External Interrupt

```
            PSW             =00EF
            CNTRL           =00EE
            RBIT            0,PORTGC
            RBIT            0,PORTGD        ; GO pin configured Hi-Z
            SBIT            IEDG, CNTRL     ; Ext interrupt polarity; falling edge
            SBIT            GIE, PSW        ; Set the GIE bit
            SBIT            EXEN, PSW       ; Enable the external interrupt
WAIT:       JP              WAIT            ; Wait for external interrupt
            .
            .
            .
            .=0FF                           ; The interrupt causes a
            VIS                             ; branch to address 0FF
                                            ; The VIS causes a branch to
                                            ; interrupt vector table
            .
            .
            .
            .=01FA                          ; Vector table (within 256 byte
            .ADDRW SERVICE                  ; of VIS inst.) containing the ext
                                            ; interrupt service routine
            .
            .
            .
SERVICE:                                    ; Interrupt Service Routine
            RBIT, EXPND, PSW                ; Reset ext interrupt pend. bit
            .
            .
            .
            RET I                           ; Return, set the GIE bit
```

8.0 Interrupts (Continued)

8.4 NON-MASKABLE INTERRUPT

8.4.1 Pending Flag

There is a pending flag bit associated with the non-maskable interrupt, called STPND. This pending flag is not memory-mapped and cannot be accessed directly by the software.

The pending flag is reset to zero when a device Reset occurs. When the non-maskable interrupt occurs, the associated pending bit is set to 1. The interrupt service routine should contain an RPND instruction to reset the pending flag to zero. The RPND instruction always resets the STPND flag.

8.4.2 Software Trap

The Software Trap is a special kind of non-maskable interrupt which occurs when the INTR instruction (used to acknowledge interrupts) is fetched from program memory and placed in the instruction register. This can happen in a variety of ways, usually because of an error condition. Some examples of causes are listed below.

If the program counter incorrectly points to a memory location beyond the available program memory space, the nonexistent or unused memory location returns zeroes which is interpreted as the INTR instruction.

If the stack is popped beyond the allowed limit (address 02F or 06F Hex), a Software Trap is triggered.

A Software Trap can be triggered by a temporary hardware condition such as a brownout or power supply glitch.

The Software Trap has the highest priority of all interrupts. When a Software Trap occurs, the STPND bit is set. The GIE bit is not affected and the pending bit (not accessible by the user) is used to inhibit other interrupts and to direct the program to the ST service routine with the VIS instruction. Nothing can interrupt a Software Trap service routine except for another Software Trap. The STPND can be reset only by the RPND instruction or a chip Reset.

The Software Trap indicates an unusual or unknown error condition. Generally, returning to normal execution at the point where the Software Trap occurred cannot be done reliably. Therefore, the Software Trap service routine should reinitialize the stack pointer and perform a recovery procedure that restarts the software at some known point, similar to a device Reset, but not necessarily performing all the same functions as a device Reset. The routine must also execute the RPND instruction to reset the STPND flag. Otherwise, all other interrupts will be locked out. To the extent possible, the interrupt routine should record or indicate the context of the device so that the cause of the Software Trap can be determined.

If the user wishes to return to normal execution from the point at which the Software Trap was triggered, the user must first execute RPND, followed by RETSK rather than RETI or RET. This is because the return address stored on the stack is the address of the INTR instruction that triggered the interrupt. The program must skip that instruction in order to proceed with the next one. Otherwise, an infinite loop of Software Traps and returns will occur.

Programming a return to normal execution requires careful consideration. If the Software Trap routine is interrupted by another Software Trap, the RPND instruction in the service routine for the second Software Trap will reset the STPND flag; upon return to the first Software Trap routine, the STPND flag will have the wrong state. This will allow maskable interrupts to be acknowledged during the servicing of the first Software Trap. To avoid problems such as this, the user program should contain the Software Trap routine to perform a recovery procedure rather than a return to normal execution.

Under normal conditions, the STPND flag is reset by a RPND instruction in the Software Trap service routine. If a programming error or hardware condition (brownout, power supply glitch, etc.) sets the STPND flag without providing a way for it to be cleared, all other interrupts will be locked out. To alleviate this condition, the user can use extra RPND instructions in the main program and in the WATCHDOG service routine (if present). There is no harm in executing extra RPND instructions in these parts of the program.

8.5 PORT L INTERRUPTS

Port L provides the user with an additional eight fully selectable, edge sensitive interrupts which are all vectored into the same service subroutine.

The interrupt from Port L shares logic with the wake up circuitry. The register WKEN allows interrupts from Port L to be individually enabled or disabled. The register WKEDG specifies the trigger condition to be either a positive or a negative edge. Finally, the register WKPND latches in the pending trigger conditions.

The GIE (Global Interrupt Enable) bit enables the interrupt function.

A control flag, LPEN, functions as a global interrupt enable for Port L interrupts. Setting the LPEN flag will enable interrupts and vice versa. A separate global pending flag is not needed since the register WKPND is adequate.

Since Port L is also used for waking the device out of the HALT or IDLE modes, the user can elect to exit the HALT or IDLE modes either with or without the interrupt enabled. If he elects to disable the interrupt, then the device will restart execution from the instruction immediately following the instruction that placed the microcontroller in the HALT or IDLE modes. In the other case, the device will first execute the interrupt service routine and then revert to normal operation. (See HALT MODE for clock option wakeup information.)

8.6 INTERRUPT SUMMARY

The device uses the following types of interrupts, listed below in order of priority:

1. The Software Trap non-maskable interrupt, triggered by the INTR (00 opcode) instruction. The Software Trap is acknowledged immediately. This interrupt service routine can be interrupted only by another Software Trap. The Software Trap should end with two RPND instructions followed by a restart procedure.
2. Maskable interrupts, triggered by an on-chip peripheral block or an external device connected to the device. Under ordinary conditions, a maskable interrupt will not interrupt any other interrupt routine in progress. A maskable interrupt routine in progress can be interrupted by the non-maskable interrupt request. A maskable interrupt routine should end with an RETI instruction.

9.0 WATCHDOG/Clock Monitor

The devices contain a user selectable WATCHDOG and clock monitor. The following section is applicable only if WATCHDOG feature has been selected in the ECON register. The WATCHDOG is designed to detect the user program getting stuck in infinite loops resulting in loss of program control or "runaway" programs.

The WATCHDOG logic contains two separate service windows. While the user programmable upper window selects the WATCHDOG service time, the lower window provides protection against an infinite program loop that contains the WATCHDOG service instruction.

The COPSAx7 devices provide the added feature of a software trap that provides protection against stack overpops and addressing locations outside valid user program space.

The Clock Monitor is used to detect the absence of a clock or a very slow clock below a specified rate on the CKI pin.

The WATCHDOG consists of two independent logic blocks: WD UPPER and WD LOWER. WD UPPER establishes the upper limit on the service window and WD LOWER defines the lower limit of the service window.

Servicing the WATCHDOG consists of writing a specific value to a WATCHDOG Service Register named WDSVR which is memory mapped in the RAM. This value is composed of three fields, consisting of a 2-bit Window Select, a 5-bit Key Data field, and the 1-bit Clock Monitor Select field. Table VI shows the WDSVR register.

TABLE VI. WATCHDOG Service Register (WDSVR)

Window Select		Key Data					Clock Monitor
X	X	0	1	1	0	0	Y
7	6	5	4	3	2	1	0

The lower limit of the service window is fixed at 256 instruction cycles. Bits 7 and 6 of the WDSVR register allow the user to pick an upper limit of the service window.

Table VII shows the four possible combinations of lower and upper limits for the WATCHDOG service window. This flexibility in choosing the WATCHDOG service window prevents any undue burden on the user software.

Bits 5, 4, 3, 2 and 1 of the WDSVR register represent the 5-bit Key Data field. The key data is fixed at 01100. Bit 0 of the WDSVR Register is the Clock Monitor Select bit.

TABLE VII. WATCHDOG Service Window Select

WDSVR Bit 7	WDSVR Bit 6	Service Window (Lower-Upper Limits)
0	0	256–8k t_C Cycles
0	1	256–16k t_C Cycles
1	0	256–32k t_C Cycles
1	1	256–64k t_C Cycles

9.1 CLOCK MONITOR

The Clock Monitor aboard the device can be selected or deselected under program control. The Clock Monitor is guaranteed not to reject the clock if the instruction cycle clock ($1/t_C$) is greater or equal to 10 kHz. This equates to a clock input rate on CKI of greater or equal to 100 kHz.

9.2 WATCHDOG/CLOCK MONITOR OPERATION

The WATCHDOG is enabled by bit 2 of the ECON register. When this ECON bit is 0, the WATCHDOG is enabled and pin G1 becomes the WATCHDOG output with a weak pullup.

The WATCHDOG and Clock Monitor are disabled during reset. The device comes out of reset with the WATCHDOG armed, the WATCHDOG Window Select bits (bits 6, 7 of the WDSVR Register) set, and the Clock Monitor bit (bit 0 of the WDSVR Register) enabled. Thus, a Clock Monitor error will occur after coming out of reset, if the instruction cycle clock frequency has not reached a minimum specified value, including the case where the oscillator fails to start.

The WDSVR register can be written to only once after reset and the key data (bits 5 through 1 of the WDSVR Register) must match to be a valid write. This write to the WDSVR register involves two irrevocable choices: (i) the selection of the WATCHDOG service window (ii) enabling or disabling of the Clock Monitor. Hence, the first write to WDSVR Register involves selecting or deselecting the Clock Monitor, select the WATCHDOG service window and match the WATCHDOG key data. Subsequent writes to the WDSVR register will compare the value being written by the user to the WATCHDOG service window value and the key data (bits 7 through 1) in the WDSVR Register. Table VIII shows the sequence of events that can occur.

The user must service the WATCHDOG at least once before the upper limit of the service window expires. The WATCHDOG may not be serviced more than once in every lower limit of the service window. The user may service the WATCHDOG as many times as wished in the time period between the lower and upper limits of the service window. The first write to the WDSVR Register is also counted as a WATCHDOG service.

The WATCHDOG has an output pin associated with it. This is the WDOUT pin, on pin 1 of the port G. WDOUT is active low. The WDOUT pin has a weak pullup in the inactive state. Upon triggering the WATCHDOG, the logic will pull the WDOUT (G1) pin low for an additional 16 t_C–32 t_C cycles after the signal level on WDOUT pin goes below the lower Schmitt trigger threshold. After this delay, the device will stop forcing the WDOUT output low. The WATCHDOG service window will restart when the WDOUT pin goes high.

A WATCHDOG service while the WDOUT signal is active will be ignored. The state of the WDOUT pin is not guaranteed on reset, but if it powers up low then the WATCHDOG will time out and WDOUT will go high.

The Clock Monitor forces the G1 pin low upon detecting a clock frequency error. The Clock Monitor error will continue until the clock frequency has reached the minimum specified value, after which the G1 output will go high following 16 t_C–32 t_C clock cycles. The Clock Monitor generates a continual Clock Monitor error if the oscillator fails to start, or fails to reach the minimum specified frequency. The specification for the Clock Monitor is as follows:

$1/t_C > 10$ kHz—No clock rejection.

$1/t_C < 10$ Hz—Guaranteed clock rejection.

9.0 WATCHDOG/Clock Monitor (Continued)

TABLE VIII. WATCHDOG Service Actions

Key Data	Window Data	Clock Monitor	Action
Match	Match	Match	Valid Service: Restart Service Window
Don't Care	Mismatch	Don't Care	Error: Generate WATCHDOG Output
Mismatch	Don't Care	Don't Care	Error: Generate WATCHDOG Output
Don't Care	Don't Care	Mismatch	Error: Generate WATCHDOG Output

9.3 WATCHDOG AND CLOCK MONITOR SUMMARY

The following salient points regarding the WATCHDOG and CLOCK MONITOR should be noted:

- Both the WATCHDOG and CLOCK MONITOR detector circuits are inhibited during RESET.
- Following RESET, the WATCHDOG and CLOCK MONITOR are both enabled, with the WATCHDOG having the maximum service window selected.
- The WATCHDOG service window and CLOCK MONITOR enable/disable option can only be changed once, during the initial WATCHDOG service following RESET.
- The initial WATCHDOG service must match the key data value in the WATCHDOG Service register WDSVR in order to avoid a WATCHDOG error.
- Subsequent WATCHDOG services must match all three data fields in WDSVR in order to avoid WATCHDOG errors.
- The correct key data value cannot be read from the WATCHDOG Service register WDSVR. Any attempt to read this key data value of 01100 from WDSVR will read as key data value of all 0's.
- The WATCHDOG detector circuit is inhibited during both the HALT and IDLE modes.
- The CLOCK MONITOR detector circuit is active during both the HALT and IDLE modes. Consequently, the device inadvertently entering the HALT mode will be detected as a CLOCK MONITOR error (provided that the CLOCK MONITOR enable option has been selected by the program).
- With the single-pin R/C oscillator option selected and the CLKDLY bit reset, the WATCHDOG service window will resume following HALT mode from where it left off before entering the HALT mode.
- With the crystal oscillator option selected, or with the single-pin R/C oscillator option selected and the CLKDLY bit set, the WATCHDOG service window will be set to its selected value from WDSVR following HALT. Consequently, the WATCHDOG should not be serviced for at least 256 instruction cycles following HALT, but must be serviced within the selected window to avoid a WATCHDOG error.
- The IDLE timer T0 is not initialized with external RESET.
- The user can sync in to the IDLE counter cycle with an IDLE counter (T0) interrupt or by monitoring the T0PND flag. The T0PND flag is set whenever the twelfth bit of the IDLE counter toggles (every 4096 instruction cycles). The user is responsible for resetting the T0PND flag.
- A hardware WATCHDOG service occurs just as the device exits the IDLE mode. Consequently, the WATCHDOG should not be serviced for at least 256 instruction cycles following IDLE, but must be serviced within the selected window to avoid a WATCHDOG error.
- Following RESET, the initial WATCHDOG service (where the service window and the CLOCK MONITOR enable/disable must be selected) may be programmed anywhere within the maximum service window (65,536 instruction cycles) initialized by RESET. Note that this initial WATCHDOG service may be programmed within the initial 256 instruction cycles without causing a WATCHDOG error.

9.4 DETECTION OF ILLEGAL CONDITIONS

The device can detect various illegal conditions resulting from coding errors, transient noise, power supply voltage drops, runaway programs, etc.

Reading of undefined ROM gets zeroes. The opcode for software interrupt is 00. If the program fetches instructions from undefined ROM, this will force a software interrupt, thus signaling that an illegal condition has occurred.

The subroutine stack grows down for each call (jump to subroutine), interrupt, or PUSH, and grows up for each return or POP. The stack pointer is initialized to RAM location 06F Hex during reset. Consequently, if there are more returns than calls, the stack pointer will point to addresses 070 and 071 Hex (which are undefined RAM). Undefined RAM from addresses 070 to 07F (Segment 0), and all other segments (i.e., Segments 4 . . . etc.) is read as all 1's, which in turn will cause the program to return to address 7FFF Hex. This is an undefined ROM location and the instruction fetched (all 0's) from this location will generate a software interrupt signaling an illegal condition.

Thus, the chip can detect the following illegal conditions:

1. Executing from undefined ROM
2. Over "POP"ing the stack by having more returns than calls.

When the software interrupt occurs, the user can re-initialize the stack pointer and do a recovery procedure before restarting (this recovery program is probably similar to that following reset, but might not contain the same program initialization procedures). The recovery program should reset the software interrupt pending bit using the RPND instruction.

10.0 MICROWIRE/PLUS

MICROWIRE/PLUS is a serial SPI compatible synchronous communications interface. The MICROWIRE/PLUS capability enables the device to interface with MICROWIRE/PLUS or SPI peripherals (i.e. A/D converters, display drivers, EEPROMs etc.) and with other microcontrollers which support the MICROWIRE/PLUS or SPI interface. It consists of an 8-bit serial shift register (SIO) with serial data input (SI), serial data output (SO) and serial shift clock (SK). *Figure 24* shows a block diagram of the MICROWIRE/PLUS logic.

The shift clock can be selected from either an internal source or an external source. Operating the MICROWIRE/PLUS arrangement with the internal clock source is called the Master mode of operation. Similarly, operating the MICROWIRE/PLUS arrangement with an external shift clock is called the Slave mode of operation.

The CNTRL register is used to configure and control the MICROWIRE/PLUS mode. To use the MICROWIRE/PLUS, the MSEL bit in the CNTRL register is set to one. In the master mode, the SK clock rate is selected by the two bits, SL0 and SL1, in the CNTRL register. Table IX details the different clock rates that may be selected.

TABLE IX. MICROWIRE/PLUS Master Mode Clock Select

SL1	SL0	SK Period
0	0	$2 \times t_C$
0	1	$4 \times t_C$
1	x	$8 \times t_C$

Where t_C is the instruction cycle clock

10.1 MICROWIRE/PLUS OPERATION

Setting the BUSY bit in the PSW register causes the MICROWIRE/PLUS to start shifting the data. It gets reset when eight data bits have been shifted. The user may reset the BUSY bit by software to allow less than 8 bits to shift. If enabled, an interrupt is generated when eight data bits have been shifted. The device may enter the MICROWIRE/PLUS mode either as a Master or as a Slave. *Figure 24* shows how two microcontroller devices and several peripherals may be interconnected using the MICROWIRE/PLUS arrangements.

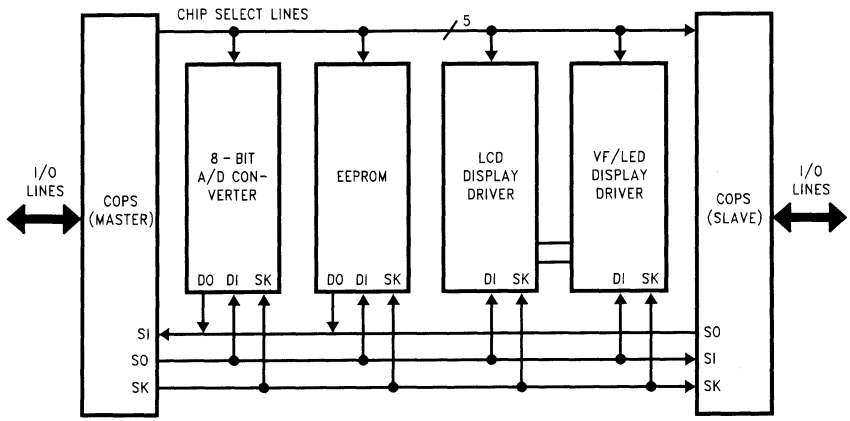
WARNING

The SIO register should only be loaded when the SK clock is in the idle phase. Loading the SIO register while the SK clock is in the active phase, will result in undefined data in the SIO register.

Setting the BUSY flag when the input SK clock is in the active phase while in the MICROWIRE/PLUS is in the slave mode may cause the current SK clock for the SIO shift register to be narrow. For safety, the BUSY flag should only be set when the input SK clock is in the idle phase.

10.1.1 MICROWIRE/PLUS Master Mode Operation

In the MICROWIRE/PLUS Master mode of operation the shift clock (SK) is generated internally. The MICROWIRE Master always initiates all data exchanges. The MSEL bit in the CNTRL register must be set to enable the SO and SK functions onto the G Port. The SO and SK pins must also be selected as outputs by setting appropriate bits in the Port G configuration register. In the slave mode, the shift clock stops after 8 clock pulses. Table X summarizes the bit settings required for Master mode of operation.



TL/DD/12838-32

FIGURE 24. MICROWIRE/PLUS Application

10.0 MICROWIRE/PLUS (Continued)

10.1.2 MICROWIRE/PLUS Slave Mode Operation

In the MICROWIRE/PLUS Slave mode of operation the SK clock is generated by an external source. Setting the MSEL bit in the CNTRL register enables the SO and SK functions onto the G Port. The SK pin must be selected as an input and the SO pin is selected as an output pin by setting and resetting the appropriate bits in the Port G configuration register. Table X summarizes the settings required to enter the Slave mode of operation.

This table assumes that the control flag MSEL is set.

TABLE X. MICROWIRE/PLUS Mode Settings

G4 (SO) Config. Bit	G5 (SK) Config. Bit	G4 Fun.	G5 Fun.	Operation
1	1	SO	Int. SK	MICROWIRE/PLUS Master
0	1	TRI-STATE	Int. SK	MICROWIRE/PLUS Master
1	0	SO	Ext. SK	MICROWIRE/PLUS Slave
0	0	TRI-STATE	Ext. SK	MICROWIRE/PLUS Slave

The user must set the BUSY flag immediately upon entering the Slave mode. This ensures that all data bits sent by the Master is shifted properly. After eight clock pulses the BUSY flag is clear, the shift clock is stopped, and the sequence may be repeated.

10.1.3 Alternate SK Phase Operation and SK Idle Polarity

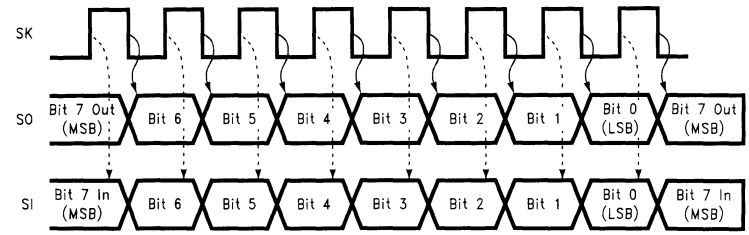
The device allows either the normal SK clock or an alternate phase SK clock to shift data in and out of the SIO register. In both the modes the SK idle polarity can be either high or low. The polarity is selected by bit 5 of Port G data register. In the normal mode data is shifted in on the rising edge of the SK clock and the data is shifted out on the falling edge of the SK clock. The SIO register is shifted on each falling edge of the SK clock. In the alternate SK phase operation, data is shifted in on the falling edge of the SK clock and shifted out on the rising edge of the SK clock. Bit 6 of Port G configuration register selects the SK edge.

A control flag, SKSEL, allows either the normal SK clock or the alternate SK clock to be selected. Resetting SKSEL causes the MICROWIRE/PLUS logic to be clocked from the normal SK signal. Setting the SKSEL flag selects the alternate SK clock. The SKSEL is mapped into the G6 configuration bit. The SKSEL flag will power up in the reset condition, selecting the normal SK signal.

TABLE XI. MICROWIRE/PLUS Shift Clock Polarity and Sample/Shift Phase

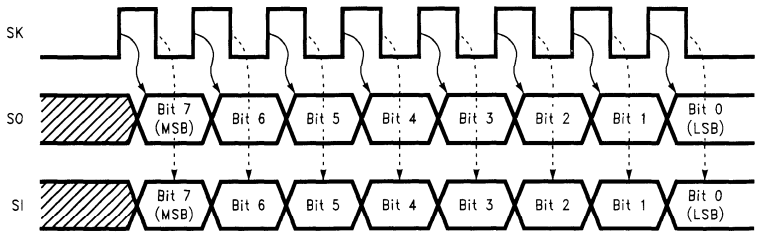
SK Phase	Port G		SO Clocked Out On:	SI Sampled On:	SK Idle Phase
	G6 (SKSEL) Config. Bit	G5 Data Bit			
Normal	0	0	SK Falling Edge	SK Rising Edge	Low
Alternate	1	0	SK Rising Edge	SK Falling Edge	Low
Alternate	0	1	SK Rising Edge	SK Falling Edge	High
Normal	1	1	SK Falling Edge	SK Rising Edge	High

10.0 MICROWIRE/PLUS (Continued)



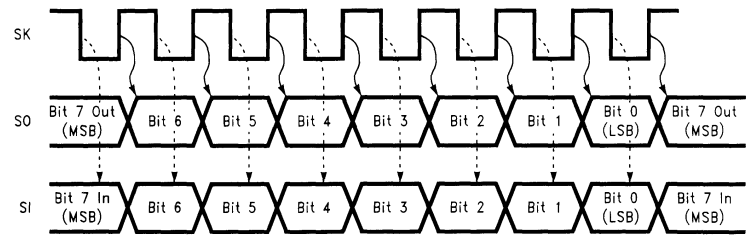
TL/DD/12838–33

FIGURE 25. MICROWIRE/PLUS SPI Mode Interface Timing, Normal SK Mode, SK Idle Phase being Low



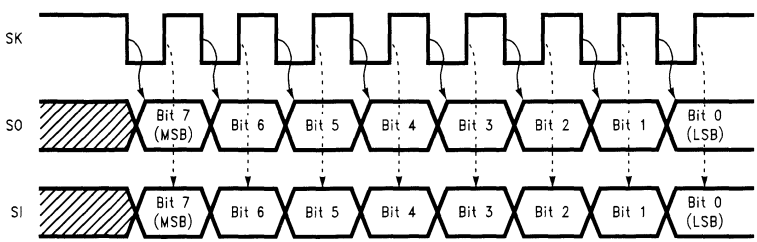
TL/DD/12838–34

FIGURE 26. MICROWIRE/PLUS SPI Mode Interface Timing, Alternate SK Mode, SK Idle Phase being Low



TL/DD/12838–35

FIGURE 27. MICROWIRE/PLUS SPI Mode Interface Timing, Alternate SK Mode, SK Idle Phase being High



TL/DD/12838–31

FIGURE 28. MICROWIRE/PLUS SPI Mode Interface Timing, Normal SK Mode, SK Idle Phase being High

11.0 Memory Map

All RAM, ports and registers (except A and PC) are mapped into data memory address space.

RAM Select	Address ADD REG	Contents
64 On-Chip RAM Bytes.	02 to 2F	On-Chip RAM (48 Bytes)
Selected by ECON Register	30 to 7F	Unused RAM (Reads as all ones)
128 On-Chip RAM Bytes	00 to 6F	On-Chip RAM (112 Bytes)
	70 to 7F	Unused RAM (Reads as all ones)
	94	Port F Data Register
	95	Port F Configuration Register
	96	Port F Input Pins (Read Only)
	97	Reserved
	C7	WATCHDOG Service Register (Reg: WDSVR)
	C8	MIWU Edge Select Register (Reg: WKEDG)
	C9	MIWU Enable Register (Reg: WKEN)
	CA	MIWU Pending Register (Reg: WKPND)
	CB to CF	Reserved
	D0	Port L Data Register
	D1	Port L Configuration Register
	D2	Port L Input Pins (Read Only)
	D3	Reserved
	D4	Port G Data Register
	D5	Port G Configuration Register
	D6	Port G Input Pins (Read Only)
	D7	Reserved
	D8	Port C Data Register
	D9	Port C Configuration Register
	DA	Port C Input Pins (Read Only)
	DB	Reserved
	DC	Port D
	DD to DF	Reserved
	E0 to E5	Reserved
	E6	Timer T1 Autoload Register T1RB Lower Byte
	E7	Timer T1 Autoload Register T1RB Upper Byte
	E8	ICNTRL Register
	E9	MICROWIRE/PLUS Shift Register
	EA	Timer T1 Lower Byte
	EB	Timer T1 Upper Byte
	EC	Timer T1 Autoload Register T1RA Lower Byte
	ED	Timer T1 Autoload Register T1RA Upper Byte
	EE	CNTRL Control Register
	EF	PSW Register
	F0 to FB	On-Chip RAM Mapped as Registers
	FC	X Register
	FD	SP Register
	FE	B Register
	FF	Segment Register

Reading any undefined memory location in the address range of 0080H–00FFH will return undefined data.

12.0 Instruction Set

12.1 INTRODUCTION

This section defines the instruction set of the COPSAx7 Family members. It contains information about the instruction set features, addressing modes and types.

12.2 INSTRUCTION FEATURES

The strength of the instruction set is based on the following features:

- Mostly single-byte opcode instructions minimize program size.
- One instruction cycle for the majority of single-byte instructions to minimize program execution time.
- Many single-byte, multiple function instructions such as DRSZ.
- Three memory mapped pointers: two for register indirect addressing, and one for the software stack.
- Sixteen memory mapped registers that allow an optimized implementation of certain instructions.
- Ability to set, reset, and test any individual bit in data memory address space, including the memory-mapped I/O ports and registers.
- Register-Indirect LOAD and EXCHANGE instructions with optional automatic post-incrementing or decrementing of the register pointer. This allows for greater efficiency (both in cycle time and program code) in loading, walking across and processing fields in data memory.
- Unique instructions to optimize program size and throughput efficiency. Some of these instructions are DRSZ, IFBNE, DCOR, RETSK, VIS and RRC.

12.3 ADDRESSING MODES

The instruction set offers a variety of methods for specifying memory addresses. Each method is called an addressing mode. These modes are classified into two categories: operand addressing modes and transfer-of-control addressing modes. Operand addressing modes are the various methods of specifying an address for accessing (reading or writing) data. Transfer-of-control addressing modes are used in conjunction with jump instructions to control the execution sequence of the software program.

12.3.1 Operand Addressing Modes

The operand of an instruction specifies what memory location is to be affected by that instruction. Several different operand addressing modes are available, allowing memory locations to be specified in a variety of ways. An instruction can specify an address directly by supplying the specific address, or indirectly by specifying a register pointer. The contents of the register (or in some cases, two registers) point to the desired memory location. In the immediate mode, the data byte to be used is contained in the instruction itself.

Each addressing mode has its own advantages and disadvantages with respect to flexibility, execution speed, and program compactness. Not all modes are available with all instructions. The Load (LD) instruction offers the largest number of addressing modes.

The available addressing modes are:

- Direct
- Register B or X Indirect
- Register B or X Indirect with Post-Incrementing/Decrementing
- Immediate
- Immediate Short
- Indirect from Program Memory

The addressing modes are described below. Each description includes an example of an assembly language instruction using the described addressing mode.

Direct. The memory address is specified directly as a byte in the instruction. In assembly language, the direct address is written as a numerical value (or a label that has been defined elsewhere in the program as a numerical value).

Example: Load Accumulator Memory Direct

LD A,05

Reg/Data Memory	Contents Before	Contents After
Accumulator	XX Hex	A6 Hex
Memory Location 0005 Hex	A6 Hex	A6 Hex

Register B or X Indirect. The memory address is specified by the contents of the B Register or X register (pointer register). In assembly language, the notation [B] or [X] specifies which register serves as the pointer.

Example: Exchange Memory with Accumulator, B Indirect

X A,[B]

Reg/Data Memory	Contents Before	Contents After
Accumulator	01 Hex	87 Hex
Memory Location 0005 Hex	87 Hex	01 Hex
B Pointer	05 Hex	05 Hex

Register B or X Indirect with Post-Incrementing/Decrementing. The relevant memory address is specified by the contents of the B Register or X register (pointer register). The pointer register is automatically incremented or decremented after execution, allowing easy manipulation of memory blocks with software loops. In assembly language, the notation [B+], [B−], [X+], or [X−] specifies which register serves as the pointer, and whether the pointer is to be incremented or decremented.

Example: Exchange Memory with Accumulator, B Indirect with Post-Increment

X A,[B+]

Reg/Data Memory	Contents Before	Contents After
Accumulator	03 Hex	62 Hex
Memory Location 0005 Hex	62 Hex	03 Hex
B Pointer	05 Hex	06 Hex

12.0 Instruction Set (Continued)

Intermediate. The data for the operation follows the instruction opcode in program memory. In assembly language, the number sign character (#) indicates an immediate operand.

Example: Load Accumulator Immediate
LD A,#05

Reg/Data Memory	Contents Before	Contents After
Accumulator	XX Hex	05 Hex

Immediate Short. This is a special case of an immediate instruction. In the "Load B immediate" instruction, the 4-bit immediate value in the instruction is loaded into the lower nibble of the B register. The upper nibble of the B register is reset to 0000 binary.

Example: Load B Register Immediate Short
LD B,#7

Reg/Data Memory	Contents Before	Contents After
B Pointer	12 Hex	07 Hex

Indirect from Program Memory. This is a special case of an indirect instruction that allows access to data tables stored in program memory. In the "Load Accumulator Indirect" (LAID) instruction, the upper and lower bytes of the Program Counter (PCU and PCL) are used temporarily as a pointer to program memory. For purposes of accessing program memory, the contents of the Accumulator and PCL are exchanged. The data pointed to by the Program Counter is loaded into the Accumulator, and simultaneously, the original contents of PCL are restored so that the program can resume normal execution.

Example: Load Accumulator Indirect
LAID

Reg/Data Memory	Contents Before	Contents After
PCU	04 Hex	04 Hex
PCL	35 Hex	36 Hex
Accumulator	1F Hex	25 Hex
Memory Location 041F Hex	25 Hex	25 Hex

12.3.2 Tranfer-of-Control Addressing Modes

Program instructions are usually executed in sequential order. However, Jump instructions can be used to change the normal execution sequence. Several transfer-of-control addressing modes are available to specify jump addresses.

A change in program flow requires a non-incremental change in the Program Counter contents. The Program Counter consists of two bytes, designated the upper byte (PCU) and lower byte (PCL). The most significant bit of PCU is not used, leaving 15 bits to address the program memory.

Different addressing modes are used to specify the new address for the Program Counter. The choice of addressing mode depends primarily on the distance of the jump. Farther jumps sometimes require more instruction bytes in order to completely specify the new Program Counter contents.

The available transfer-of-control addressing modes are:

- Jump Relative
- Jump Absolute
- Jump Absolute Long
- Jump Indirect

The transfer-of-control addressing modes are described below. Each description includes an example of a Jump instruction using a particular addressing mode, and the effect on the Program Counter bytes of executing that instruction.

Jump Relative. In this 1-byte instruction, six bits of the instruction opcode specify the distance of the jump from the current program memory location. The distance of the jump can range from −31 to +32. A JP+1 instruction is not allowed. The programmer should use a NOP instead.

Example: Jump Relative
JP 0A

Reg	Contents Before	Contents After
PCU	02 Hex	02 Hex
PCL	05 Hex	0F Hex

Jump Absolute. In this 2-byte instruction, 12 bits of the instruction opcode specify the new contents of the Program Counter. The upper three bits of the Program Counter remain unchanged, restricting the new Program Counter address to the same 4 kbyte address space as the current instruction.

(This restriction is relevant only in devices using more than one 4 kbyte program memory space.)

Example: Jump Absolute
JMP 0125

Reg	Contents Before	Contents After
PCU	0C Hex	01 Hex
PCL	77 Hex	25 Hex

Jump Absolute Long. In this 3-byte instruction, 15 bits of the instruction opcode specify the new contents of the Program Counter.

Example: Jump Absolute Long
JMP 03625

Reg/ Memory	Contents Before	Contents After
PCU	42 Hex	36 Hex
PCL	36 Hex	25 Hex

12.0 Instruction Set (Continued)

Jump Indirect. In this 1-byte instruction, the lower byte of the jump address is obtained from a table stored in program memory, with the Accumulator serving as the low order byte of a pointer into program memory. For purposes of accessing program memory, the contents of the Accumulator are written to PCL (temporarily). The data pointed to by the Program Counter (PCH/PCL) is loaded into PCL, while PCH remains unchanged.

Example: Jump Indirect
JID

Reg/ Memory	Contents Before	Contents After
PCU	01 Hex	01 Hex
PCL	C4 Hex	32 Hex
Accumulator	26 Hex	26 Hex
Memory Location 0126 Hex	32 Hex	32 Hex

The VIS instruction is a special case of the Indirect Transfer of Control addressing mode, where the double-byte vector associated with the interrupt is transferred from adjacent addresses in program memory into the Program Counter in order to jump to the associated interrupt service routine.

12.4 INSTRUCTION TYPES

The instruction set contains a wide variety of instructions. The available instructions are listed below, organized into related groups.

Some instructions test a condition and skip the next instruction if the condition is not true. Skipped instructions are executed as no-operation (NOP) instructions.

12.4.1 Arithmetic Instructions

The arithmetic instructions perform binary arithmetic such as addition and subtraction, with or without the Carry bit.

- Add (ADD)
- Add with Carry (ADC)
- Subtract (SUB)
- Subtract with Carry (SUBC)
- Increment (INC)
- Decrement (DEC)
- Decimal Correct (DCOR)
- Clear Accumulator (CLR)
- Set Carry (SC)
- Reset Carry (RC)

12.4.2 Transfer-of-Control Instructions

The transfer-of-control instructions change the usual sequential program flow by altering the contents of the Program Counter. The Jump to Subroutine instructions save the Program Counter contents on the stack before jumping; the Return instructions pop the top of the stack back into the Program Counter.

- Jump Relative (JP)
- Jump Absolute (JMP)
- Jump Absolute Long (JMPL)
- Jump Indirect (JID)
- Jump to Subroutine (JSR)
- Jump to Subroutine Long (JSRL)
- Return from Subroutine (RET)
- Return from Subroutine and Skip (RETSK)
- Return from Interrupt (RETI)
- Software Trap Interrupt (INTR)
- Vector Interrupt Select (VIS)

12.4.3 Load and Exchange Instructions

The load and exchange instructions write byte values in registers or memory. The addressing mode determines the source of the data.

- Load (LD)
- Load Accumulator Indirect (LAID)
- Exchange (X)

12.4.4 Logical Instructions

The logical instructions perform the operations AND, OR, and XOR (Exclusive OR). Other logical operations can be performed by combining these basic operations. For example, complementing is accomplished by exclusive-ORing the Accumulator with FF Hex.

- Logical AND (AND)
- Logical OR (OR)
- Exclusive OR (XOR)

12.4.5 Accumulator Bit Manipulation Instructions

The Accumulator bit manipulation instructions allow the user to shift the Accumulator bits and to swap its two nibbles.

- Rotate Right Through Carry (RRC)
- Rotate Left Through Carry (RLC)
- Swap Nibbles of Accumulator (SWAP)

12.4.6 Stack Control Instructions

- Push Data onto Stack (PUSH)
- Pop Data off of Stack (POP)

12.4.7 Memory Bit Manipulation Instructions

The memory bit manipulation instructions allow the user to set and reset individual bits in memory.

- Set Bit (SBIT)
- Reset Bit (RBIT)
- Reset Pending Bit (RPND)

12.4.8 Conditional Instructions

The conditional instruction test a condition. If the condition is true, the next instruction is executed in the normal manner; if the condition is false, the next instruction is skipped.

- If Equal (IFEQ)
- If Not Equal (IFNE)
- If Greater Than (IFGT)
- If Carry (IFC)
- If Not Carry (IFNC)
- If Bit (IFBIT)
- If B Pointer Not Equal (IFBNE)
- And Skip if Zero (ANDSZ)
- Decrement Register and Skip if Zero (DRSZ)

12.0 Instruction Set (Continued)

12.4.9 No-Operation Instruction

The no-operation instruction does nothing, except to occupy space in the program memory and time in execution.

No-Operation (NOP)

Note: The VIS is a special case of the Indirect Transfer of Control addressing mode, where the double byte vector associated with the interrupt is transferred from adjacent addresses in the program memory into the program counter (PC) in order to jump to the associated interrupt service routine.

12.5 REGISTER AND SYMBOL DEFINITION

The following abbreviations represent the nomenclature used in the instruction description and the COP8 cross-assembler.

Registers	
A	8-Bit Accumulator Register
B	8-Bit Address Register
X	8-Bit Address Register
SP	8-Bit Stack Pointer Register
PC	15-Bit Program Counter Register
PU	Upper 7 Bits of PC
PL	Lower 8 Bits of PC
C	1 Bit of PSW Register for Carry
HC	1 Bit of PSW Register for Half Carry
GIE	1 Bit of PSW Register for Global Interrupt Enable
VU	Interrupt Vector Upper Byte
VL	Interrupt Vector Lower Byte

Symbols	
[B]	Memory Indirectly Addressed by B Register
[X]	Memory Indirectly Addressed by X Register
MD	Direct Addressed Memory
Mem	Direct Addressed Memory or [B]
Meml	Direct Addressed Memory or [B] or Immediate Data
Imm	8-Bit Immediate Data
Reg	Register Memory: Addresses F0 to FF (Includes B, X and SP)
Bit	Bit Number (0 to 7)
←	Loaded with
⟷	Exchanged with

12.0 Instruction Set (Continued)

12.6 INSTRUCTION SET SUMMARY

ADD	A,Meml	ADD	A ← A + Meml
ADC	A,Meml	ADD with Carry	A ← A + Meml + C, C ← Carry, HC ← Half Carry
SUBC	A,Meml	Subtract with Carry	A ← A − $\overline{\text{Meml}}$ + C, C ← Carry, HC ← Half Carry
AND	A,Meml	Logical AND	A ← A and $\overline{\text{Meml}}$
ANDSZ	A,Imm	Logical AND Immed., Skip if Zero	Skip next if (A and Imm) = 0
OR	A,Meml	Logical OR	A ← A or Meml
XOR	A,Meml	Logical EXclusive OR	A ← A xor Meml
IFEQ	MD,Imm	IF EQual	Compare MD and Imm, Do next if MD = Imm
IFEQ	A,Meml	IF EQual	Compare A and Meml, Do next if A = Meml
IFNE	A,Meml	IF Not Equal	Compare A and Meml, Do next if A ≠ Meml
IFGT	A,Meml	IF Greater Than	Compare A and Meml, Do next if A > Meml
IFBNE	#	If B Not Equal	Do next if lower 4 bits of B ≠ Imm
DRSZ	Reg	Decrement Reg., Skip if Zero	Reg ← Reg − 1, Skip if Reg = 0
SBIT	#,Mem	Set BIT	1 to bit, Mem (bit = 0 to 7 immediate)
RBIT	#,Mem	Reset BIT	0 to bit, Mem
IFBIT	#,Mem	IF BIT	If bit #, A or Mem is true do next instruction
RPND		Reset PeNDing Flag	Reset Software Interrupt Pending Flag
X	A,Mem	EXchange A with Memory	A ⟷ Mem
X	A,[X]	EXchange A with Memory [X]	A ⟷ [X]
LD	A,Meml	LoaD A with Memory	A ← Meml
LD	A,[X]	LoaD A with Memory [X]	A ← [X]
LD	B,Imm	LoaD B with Immed.	B ← Imm
LD	Mem,Imm	LoaD Memory Immed.	Mem ← Imm
LD	Reg,Imm	LoaD Register Memory Immed.	Reg ← Imm
X	A, [B ±]	EXchange A with Memory [B]	A ⟷ [B], (B ← B ± 1)
X	A, [X ±]	EXchange A with Memory [X]	A ⟷ [X], (X ← X ± 1)
LD	A, [B±]	LoaD A with Memory [B]	A ← [B], (B ← B ± 1)
LD	A, [X±]	LoaD A with Memory [X]	A ← [X], (X ← X ± 1)
LD	[B±],Imm	LoaD Memory [B] Immed.	[B] ← Imm, (B ← B ± 1)
CLR	A	CLeaR A	A ← 0
INC	A	INCrement A	A ← A + 1
DEC	A	DECrement A	A ← A − 1
LAID		Load A InDirect from ROM	A ← ROM (PU,A)
DCOR	A	Decimal CORrect A	A ← BCD correction of A (follows ADC, SUBC)
RRC	A	Rotate A Right thru C	C → A7 → . . . → A0 → C
RLC	A	Rotate A Left thru C	C ← A7 ← . . . ← A0 ← C, HC ← A0
SWAP	A	SWAP nibbles of A	A7 . . . A4 ⟷ A3 . . . A0
SC		Set C	C ← 1, HC ← 1
RC		Reset C	C ← 0, HC ← 0
IFC		IF C	IF C is true, do next instruction
IFNC		IF Not C	If C is not true, do next instruction
POP	A	POP the stack into A	SP ← SP + 1, A ← [SP]
PUSH	A	PUSH A onto the stack	[SP] ← A, SP ← SP − 1
VIS		Vector to Interrupt Service Routine	PU ← [VU], PL ← [VL]
JMPL	Addr.	Jump absolute Long	PC ← ii (ii = 15 bits, 0 to 32k)
JMP	Addr.	Jump absolute	PC9 . . . 0 ← i (i = 12 bits)
JP	Disp.	Jump relative short	PC ← PC + r (r is −31 to +32, except 1)
JSRL	Addr.	Jump SubRoutine Long	[SP] ← PL, [SP−1] ← PU,SP−2, PC ← ii
JSR	Addr.	Jump SubRoutine	[SP] ← PL, [SP−1] ← PU,SP−2, PC9 . . . 0 ← i
JID		Jump InDirect	PL ← ROM (PU,A)
RET		RETurn from subroutine	SP + 2, PL ← [SP], PU ← [SP−1]
RETSK		RETurn and SKip	SP + 2, PL ← [SP],PU ← [SP−1], skip next instruction
RETI		RETurn from Interrupt	SP + 2, PL ← [SP],PU ← [SP−1],GIE ← 1
INTR		Generate an Interrupt	[SP] ← PL, [SP−1] ← PU, SP−2, PC ← 0FF
NOP		No OPeration	PC ← PC + 1

12.0 Instruction Set (Continued)

12.7 INSTRUCTION EXECUTION TIME

Most instructions are single byte (with immediate addressing mode instructions taking two bytes).

Most single byte instructions take one cycle time to execute.

Skipped instructions require x number of cycles to be skipped, where x equals the number of bytes in the skipped instruction opcode.

See the BYTES and CYCLES per INSTRUCTION table for details.

Bytes and Cycles per Instruction

The following table shows the number of bytes and cycles for each instruction in the format of byte/cycle.

Arithmetic and Logic Instructions

	[B]	Direct	Immed.
ADD	1/1	3/4	2/2
ADC	1/1	3/4	2/2
SUBC	1/1	3/4	2/2
AND	1/1	3/4	2/2
OR	1/1	3/4	2/2
XOR	1/1	3/4	2/2
IFEQ	1/1	3/4	2/2
IFGT	1/1	3/4	2/2
IFBNE	1/1		
DRSZ		1/3	
SBIT	1/1	3/4	
RBIT	1/1	3/4	
IFBIT	1/1	3/4	

RPND	1/1

Instructions Using A & C

CLRA	1/1
INCA	1/1
DECA	1/1
LAID	1/3
DCORA	1/1
RRCA	1/1
RLCA	1/1
SWAPA	1/1
SC	1/1
RC	1/1
IFC	1/1
IFNC	1/1
PUSHA	1/3
POPA	1/3
ANDSZ	2/2

Transfer of Control Instructions

JMPL	3/4
JMP	2/3
JP	1/3
JSRL	3/5
JSR	2/5
JID	1/3
VIS	1/5
RET	1/5
RETSK	1/5
RETI	1/5
INTR	1/7
NOP	1/1

Memory Transfer Instructions

	Register Indirect		Direct	Immed.	Register Indirect Auto Incr. & Decr.		
	[B]	[X]			[B+, B−]	[X+, X−]	
X A,*	1/1	1/3	2/3		1/2	1/3	
LD A,*	1/1	1/3	2/3	2/2	1/2	1/3	
LD B, Imm				1/1			(If B < 16)
LD B, Imm				2/2			(If B > 15)
LD Mem, Imm	2/2		3/3		2/2		
LD Reg, Imm			2/3				
IFEQ MD, Imm			3/3				

* = > Memory location addressed by B or X or directly.

12.8 OPCODE TABLE

Upper Nibble																Lower Nibble
F	E	D	C	B	A	9	8	7	6	5	4	3	2	1	0	
JP−15	JP−31	LD 0F0, #i	DRSZ 0F0	RRCA	RC	ADC A,#i	ADC A,[B]	IFBIT 0,[B]	ANDSZ A, #i	LD B,#0F	IFBNE 0	JSR x000−x0FF	JMP x000−x0FF	JP+17	INTR	0
JP−14	JP−30	LD 0F1, #i	DRSZ 0F1	*	SC	SUBC A, #i	SUB A,[B]	IFBIT 1,[B]	*	LD B,#0E	IFBNE 1	JSR x100−x1FF	JMP x100−x1FF	JP+18	JP+2	1
JP−13	JP−29	LD 0F2, #i	DRSZ 0F2	X A,[X+]	X A,[B+]	IFEQ A,#i	IFEQ A,[B]	IFBIT 2,[B]	*	LD B,#0D	IFBNE 2	JSR x200−x2FF	JMP x200−x2FF	JP+19	JP+3	2
JP−12	JP−28	LD 0F3, #i	DRSZ 0F3	X A,[X−]	X A,[B−]	IFGT A,#i	IFGT A,[B]	IFBIT 3,[B]	*	LD B,#0C	IFBNE 3	JSR x300−x3FF	JMP x300−x3FF	JP+20	JP+4	3
JP−11	JP−27	LD 0F4, #i	DRSZ 0F4	VIS	LAID	ADD A,#i	ADD A,[B]	IFBIT 4,[B]	CLRA	LD B,#0B	IFBNE 4	JSR x400−x4FF	JMP x400−x4FF	JP+21	JP+5	4
JP−10	JP−26	LD 0F5, #i	DRSZ 0F5	RPND	JID	AND A,#i	AND A,[B]	IFBIT 5,[B]	SWAPA	LD B,#0A	IFBNE 5	JSR x500−x5FF	JMP x500−x5FF	JP+22	JP+6	5
JP−9	JP−25	LD 0F6, #i	DRSZ 0F6	X A,[X]	X A,[B]	XOR A,#i	XOR A,[B]	IFBIT 6,[B]	DCORA	LD B,#09	IFBNE 6	JSR x600−x6FF	JMP x600−x6FF	JP+23	JP+7	6
JP−8	JP−24	LD 0F7, #i	DRSZ 0F7	*	*	OR A,#i	OR A,[B]	IFBIT 7,[B]	PUSHA	LD B,#08	IFBNE 7	JSR x700−x7FF	JMP x700−x7FF	JP+24	JP+8	7
JP−7	JP−23	LD 0F8, #i	DRSZ 0F8	NOP	RLCA	LD A,#i	IFC	SBIT 0,[B]	RBIT 0,[B]	LD B,#07	IFBNE 8	JSR x800−x8FF	JMP x800−x8FF	JP+25	JP+9	8
JP−6	JP−22	LD 0F9, #i	DRSZ 0F9	IFNE A,[B]	IFEQ Md,#i	IFNE A,#i	IFNC	SBIT 1,[B]	RBIT 1,[B]	LD B,#06	IFBNE 9	JSR x900−x9FF	JMP x900−x9FF	JP+26	JP+10	9
JP−5	JP−21	LD 0FA, #i	DRSZ 0FA	LD A,[X+]	LD A,[B+]	LD [B+],#i	INCA	SBIT 2,[B]	RBIT 2,[B]	LD B,#05	IFBNE 0A	JSR xA00−xAFF	JMP xA00−xAFF	JP+27	JP+11	A
JP−4	JP−20	LD 0FB, #i	DRSZ 0FB	LD A,[X−]	LD A,[B−]	LD [B−],#i	DECA	SBIT 3,[B]	RBIT 3,[B]	LD B,#04	IFBNE 0B	JSR xB00−xBFF	JMP xB00−xBFF	JP+28	JP+12	B
JP−3	JP−19	LD 0FC, #i	DRSZ 0FC	LD Md,#i	JMPL	X A,Md	POPA	SBIT 4,[B]	RBIT 4,[B]	LD B,#03	IFBNE 0C	JSR xC00−xCFF	JMP xC00−xCFF	JP+29	JP+13	C
JP−2	JP−18	LD 0FD, #i	DRSZ 0FD	DIR	JSRL	LD A,Md	RETSK	SBIT 5,[B]	RBIT 5,[B]	LD B,#02	IFBNE 0D	JSR xD00−xDFF	JMP xD00−xDFF	JP+30	JP+14	D
JP−1	JP−17	LD 0FE, #i	DRSZ 0FE	LD A,[X]	LD A,[B]	LD [B],#i	RET	SBIT 6,[B]	RBIT 6,[B]	LD B,#01	IFBNE 0E	JSR xE00−xEFF	JMP xE00−xEFF	JP+31	JP+15	E
JP−0	JP−16	LD 0FF, #i	DRSZ 0FF	*	*	LD B,#i	RETI	SBIT 7,[B]	RBIT 7,[B]	LD B,#00	IFBNE 0F	JSR xF00−xFFF	JMP xF00−xFFF	JP+32	JP+16	F

Where,

i is the immediate data

Md is a directly addressed memory location

* is an unused opcode

Note: The opcode 60 Hex is also the opcode for IFBIT #i,A

13.0 Development Support

13.1 SUMMARY

- iceMASTER: IM-COP8/400—Full feature in-circuit emulation for all COP8 products. A full set of COP8 Basic and Feature Family device and package specific probes are available.
- COP8 Debug Module: Moderate cost in-circuit emulation and development programming unit.
- COP8 Evaluation and Programming Unit: EPU-COP888GG—low cost in-circuit simulation and development programming unit.
- Assembler: COP8-DEV-IBMA. A DOS installable cross development Assembler, Linker, Librarian and Utility Software Development Took Kit.
- C Compiler: COP8C. A DOS installable cross development Software Tool Kit.
- OTP/EPROM Programmer Support: Covering needs from engineering prototype, pilot production to full production environments.
- In-factory Programming Support: Covering high volume production OTP programming.

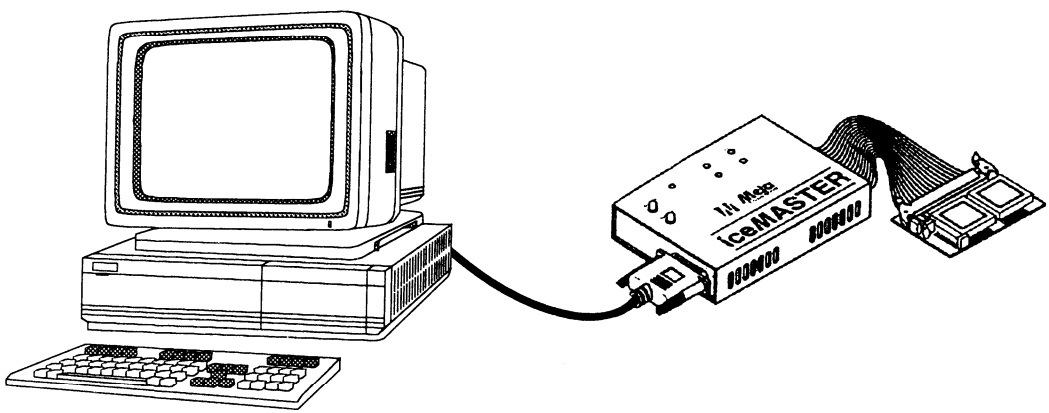
13.2 IceMASTER (IM) IN-CIRCUIT EMULATION

The iceMASTER IM-COP8/400 is a full feature, PC based, in-circuit emulation tool development and marketed by MetaLink Corporation to support the whole COP8 family of products. National and National Authorized Distributors are resale vendors for these products.

See *Figure 29* for configuration.

The iceMASTER IM-COP8/400 with its device specific COP8 Probe card provides a rich feature set for developing, testing and maintaining product:

- Real-time in-circuit emulation; full 2.7V–5.5V operation range, full DC-10 MHz clock. Chip options are programmable or jumper selectable.
- Direct connection to application board by package compatible socket or surface mount assembly.
- Full 32 kbytes of loadable programming space that overlays (replaces) the on-chip ROM or EPROM. On-chip RAM and I/O blocks are used directly or recreated on the probe as necessary.
- Full 4k frame synchronous trace memory. Address, instruction, and eight unspecified, circuit connectable trace lines. Display can be HLL source (e.g., C source), assembly or mixed.
- A full 64k hardware configurable break, trace on, trace off control, and pass count increment events.
- Tool set integrated interactive symbolic debugger—supports both assembler (COFF) and C Compiler (.COD) linked object formats.
- Real time performance profiling analysis; selectable bucket definition.
- Watch windows, content updated automatically at each execution break.
- Instruction by instruction memory/register changes displayed on source window when in single step operation.
- Single base unit and debugger software reconfigurable to support the entire COP8 family; only the probe personality needs to change. Debugger software is processor customized, and reconfigured from a master model file.
- Processor specific symbolic display of registers and bit level assignments, configured from master model file.
- Halt/Idle mode notification.
- On-Line HELP
- Includes a copy of COP8-DEV-IBMA assembler and linker SDK.



TL/DD/12838–36

FIGURE 29. COP8 iceMASTER Environment

13.0 Development Support (Continued)

IM Order Information

Base Unit	
IM-COP8/400-1	iceMASTER Base Unit, 110V Power Supply
IM-COP8/400-2	iceMASTER Base Unit, 220V Power Supply
iceMASTER Probe, COPSAx7	
COP8SA-IM44V	44 PLCC, 2.7V–5.5V
COP8SA-IM40N	40 DIP, 2.7V–5.5V
COP8SA-IM28N	28 DIP, 2.7V–5.5V
COP8SA-IM20N	20 DIP, 2.7V–5.5V
COP8SA-IM16N	16 DIP, 2.7V–5.5V
Optional Surface Mount Adapter Kits	
MHW-COP8/44P-Q	44 PLCC to 44 PQFP SM Adapter
MHW-SOIC-28	28 DIP to 28 SOIC SM Adapter
MHW-SOIC-20	20 DIP to 20 SOIC SM Adapter
MHW-SOIC-16	16 DIP to 16 SOIC SM Adapter

e.g., Target package is 44P; order: IM-COP8/400-1, COP8SA-IM44V and MHW-COP8/44V-P.

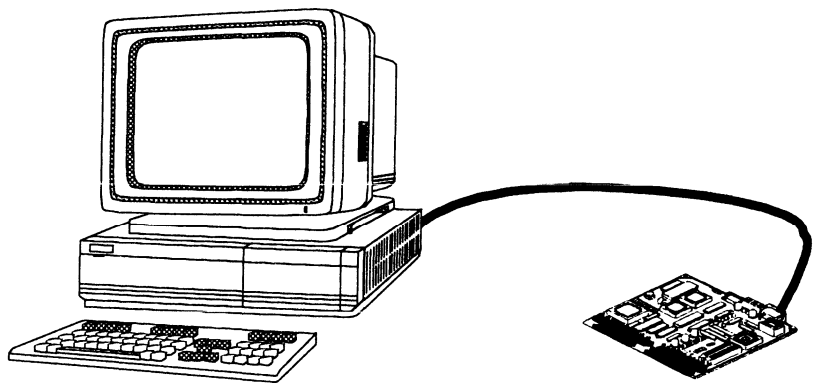
13.3 iceMASTER DEBUG MODULE (DM)

The iceMASTER Debug Module is a PC based, combination in-circuit emulation tool and COP8 based OTP/EPROM programming tool developed and marketed by MetaLink Corporation to support the whole COP8 family of products. National and National Authorized Distributors are resale vendors for these products.

See *Figure 30* for configuration.

The iceMASTER Debug Module is a moderate cost development tool. It has the capability of in-circuit emulation for a specific COP8 microcontroller and in addition serves as a programming tool for COP8 OTP and EPROM product families. Summary of features is as follows:

- Real-time in-circuit emulation; full operating voltage range operation, full DC-10 MHz clock.
- All processor I/O pins can be cabled to an application development board with package compatible cable to socket and surface mount assembly.
- Full 32 kbytes of loadable programming space that overlays (replaces) the on-chip ROM or EPROM. On-chip RAM and I/O blocks are used directly or recreated as necessary.
- 100 frames of synchronous trace memory. The display can be HLL source (C source), assembly or mixed. The most recent history prior to a break is available in the trace memory.
- Configured break points; uses INTR instruction which is modestly intrusive.
- Software—only supported features are selectable.
- Tool set integrated interactive symbolic debugger—supports both assembler (COFF) and C Compiler (.COD) SDK linked object formats.
- Instruction by instruction memory/register changes displayed when in single step operation.
- Processor specific symbolic display of registers and bit level assignments, configured from master model file.
- Halt/Idle mode notification.
- Programming menu supports full product line of programmable OTP and EPROM COP8 products. Program data is taken directly from the overlay RAM.
- Programming of 44 PQFP parts requires external programming adapters.
- Includes wallmount power supply.
- On-board V_{PP} generator from 5V input or connection to external supply supported. Requires V_{PP} level adjustment per the family programming specification (correct level is provided on an on-screen pop-down display).
- Includes a copy of COP8-DEV-IBMA assembler and linker SDK.



TL/DD/12838–37

FIGURE 30. COP8-DM Environment

13.0 Development Support (Continued)

DM Order Information

Debug Module Unit	
COP8SA-DM	
Cable Adapters, Requires One for Emulation	
DM-COP8/44P	44 PLCC
DM-COP8/40D	40 DIP
DM-COP8/28D	28 DIP
DM-COP8/20D	20 DIP
DM-COP8/16D	16 DIP
Optional Surface Mount Adapter Kits	
MHW-COP8/44P-Q	44 PLCC to 44 PQFP
DM-COP8/28D-SO	28 DIP to 28 SOIC
DM-COP8/20D-SO	20 DIP to 20 SOIC
DM-COP8/16D-SO	16 DIP to 16 SOIC
Optional Programming Adapters	
COP8-PGMA-44Q	44 PQFP

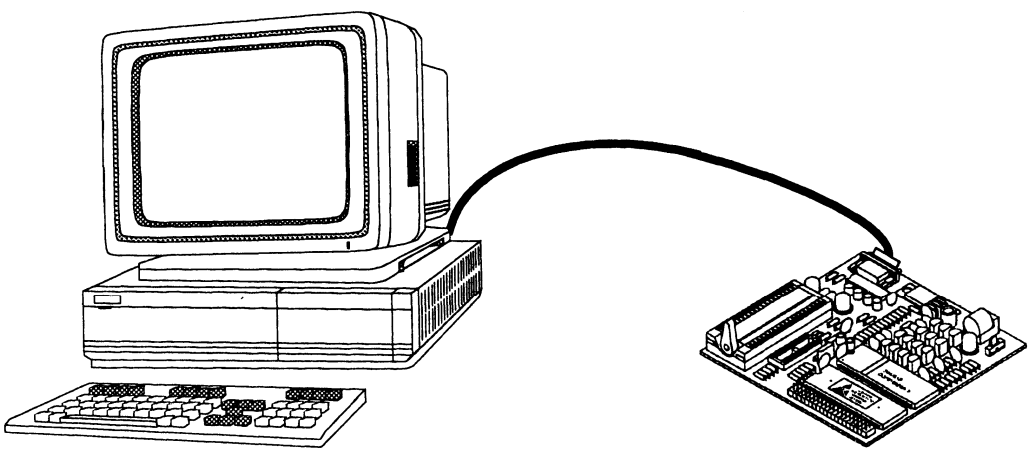
13.4 iceMASTER EVALUATION PROGRAMMING UNIT (EPU)

The iceMASTER EPU-COP888GG is a PC based, in-circuit simulation tool to support the feature family COP8 products.

See *Figure 31* for configuration.

The simulation capability is a very low cost means of evaluation the general COP8 architecture. In addition, the EPU has programming capability, with added adapters, for programming the whole COP8 product family of OTP and EPROM products. The product includes the following features:

- Non-real-time in-circuit simulation. Program overlay memory is PC resident; instructions are downloaded over RS-232 as executed. Approximate performance is 20 kHz.
- Includes a 40 pin DIP cable adapter. Other target packages are not supported. All processor I/O pins are cabled to the application development environment.
- Full 32 kbytes of loadable programmable space that overlays (replaces) the on-chip ROM or EPROM. On-chip RAM and I/O blocks are used directly or recreated as necessary.
- On-chip timer and WATCHDOG execution are not well synchronized to the instruction simulation.
- 100 frames of synchronous trace memory. The display can be HLL source (e.g., C source), assembly or mixed. The most recent history prior to a break is available in the trace memory.
- Up to eight software configured break points; uses INTR instruction which is modestly intrusive.
- Common look-feel debugger software across all MetaLink products—only supported features are selectable.
- Tool set integrated interactive symbolic debugger—supports both assembler (COFF and C Compiler (.COD) SDK linked object formats.
- Instruction by instruction memory/register changes displayed when in single step operation.
- Processor specific symbolic display of registers and bit level assignments, configured from master model file.
- Halt/Idle mode notification. Restart requires special handling.
- Programming menu supports full product line of programmable OTP and EPROM COP8 products. Only a 40 ZIF socket is available on the EPU unit. Adapters are available for other part package configurations.
- Integral wall mount power supply provides 5V and develops the required V_{PP} to program parts.
- Includes a copy of COP8-DEV-IBMA assembler, linker SDK.



TL/DD/12838–38

FIGURE 31. EPU-COP8 Tool Environment

13.0 Development Support (Continued)

Order Information

Evaluation Programming Unit	
COP8SA-EPU	Evaluation Programming Unit with debugger and programmer control software with 40-pin ZIF programming socket.
Optional Programming Adapters	
COP8-PGMA-44P-Q	44 PLCC and 44 PQFP
COP8-PGMA-28 SO	28, 20 and 16 SOIC
COP8-PGMA-878x	COP8782, COP8781, COP8780

13.5 COP8 ASSEMBLER/LINKER SOFTWARE DEVELOPMENT TOOL KIT

National Semiconductor offers a relocatable COP8 macro cross assembler, linker, librarian and utility software development tool kit. Features are summarized as follows:

- Basic and Feature Family instruction set by "device" type.
- Nested macro capability.
- Extensive set of assembler directives.
- Supported on PC/DOS platform.
- Generates National standard COFF output files.
- Integrated Linker and Librarian.
- Integrated utilities to generate ROM code file outputs.
- DUMPCOFF utility.

This product is integrated as a part of MetaLink tools as a development kit, fully supported by the MetaLink debugger. It may be ordered separately or it is bundled with the MetaLink products at no additional cost.

Assembler/Linker Order Information

Assembler SDK:	
COP8-DEV-IBMA	Assembler SDK on installable 3.5" PC®/DOS Floppy Disk Drive format. Periodic upgrades and most recent version is available on National's BBS and Internet.

13.6 COP8 C COMPILER

A C Compiler is developed and marketed by Byte Craft Limited. The COP8C compiler is a fully integrated development tool specifically designed to support the compact embedded configuration of the COP8 family of products.

Features are summarized as follows:

- ANSI C with some restrictions and extensions that optimize development for the COP8 embedded application.
- BITS data type extension. Register declaration #pragma with direct bit level definitions.

Approved List

Manufacturer	North America	Europe	Asia
BP Microsystems	(800) 225-2102 (713) 688-4600 Fax: (713) 688-0920	+49-8152-4183 +49-8856-932616	+852-234-16611 +852-2710-8121
Data I/O	(800) 426-1045 (206) 881-6444 Fax: (206) 882-1043	+44-0734-440011	Call North America
HI-LO	(510) 623-8860	Call Asia	+886-2-764-0215 Fax: +886-2-756-6403
ICE Technology	(800) 624-8949 (919) 430-7915	+44-1226-767404 Fax: 0-1226-370-434	
MetaLink	(800) 638-2423 (602) 926-0797 Fax: (602) 693-0681	+49-80915696-0 Fax: +49-80912386	+852-737-1800
Systems General	(408) 263-6667	+41-1-9450300	+886-2-917-3005 Fax: +886-2-911-1283
Needhams	(916) 924-8037 Fax: (916) 924-8065		

13.0 Development Support (Continued)

- C language support for interrupt routines.
- Expert system, rule based code generation and optimization.
- Performs consistency checks against the architectural definitions of the target COP8 device.
- Generates program memory code.
- Supports linking of compiled object or COP8 assembled object formats.
- Global optimization of linked code.
- Symbolic debug load format fully source level supported by the MetaLink debugger.

13.7 INDUSTRY WIDE OTP/EPROM PROGRAMMING SUPPORT

Programming support, in addition to the MetaLink development tools, is provided by a full range of independent approved vendors to meet the needs from the engineering laboratory to full production.

13.8 AVAILABLE LITERATURE

For more information, please see the COP8 Basic Family User's Manual, Literature Number 620895, COP8 Feature Family User's Manual, Literature Number 620897 and National's Family of 8-bit Microcontrollers COP8 Selection Guide, Literature Number 630009.

13.9 DIAL-A-HELPER SERVICE

Dial-A-Helper is a service provided by the Microcontroller Applications group. The Dial-A-Helper is an Electronic Information System that may be accessed as a Bulletin Board System (BBS) via data modem, as an FTP site on the Internet via standard FTP client application or as an FTP site on the Internet using a standard Internet browser such as Netscape or Mosaic.

The Dial-A-Helper system provides access to an automated information storage and retrieval system. The system capabilities include a MESSAGE SECTION (electronic mail, when accessed as a BBS) for communications to and from the Microcontroller Applications Group and a FILE SECTION which consists of several file areas where valuable application software and utilities could be found.

DIAL-A-HELPER BBS via a Standard Modem

Modem: CANADA/U.S.: (800) NSC-MICRO
(800) 672-6427
EUROPE: (+49) 0-814-135 13 32
Baud: 14.4k
Set-up: Length: 8-Bit
Parity: None
Stop Bit: 1
Operation: 24 Hrs., 7 Days

DIAL-A-HELPER via FTP

ftp nscmicro.nsc.com
user: anonymous
password: username@yourhost.site.domain

DIAL-A-HELPER via a WorldWide Web Browser

ftp://nscmicro.nsc.com

National Semiconductor on the WorldWide Web

See us on the WorldWide Web at: http://www.national.com

13.10 CUSTOMER RESPONSE CENTER

Complete product information and technical support is available from National's customer response centers.

CANADA/U.S.:	Tel:	(800) 272-9959
	email:	support @tevm2.nsc.com
EUROPE:	email:	europe.support@nsc.com
	Deutsch Tel:	+49 (0) 180-530 85 85
	English Tel:	+49 (0) 180-532 78 32
	Français Tel:	+49 (0) 180-532 93 58
	Italiano Tel:	+49 (0) 180-534 16 80
JAPAN:	Tel:	+81-043-299-2309
S.E. ASIA:	Beijing Tel:	(+86) 10-6856-8601
	Shanghai Tel:	(+86) 21-6415-4092
	Hong Kong Tel:	(+852) 2737-1600
	Korea Tel:	(+82) 2-3771-6909
	Malaysia Tel:	(+60-4) 644-9061
	Singapore Tel:	(+65) 255-2226
	Taiwan Tel:	+886-2-521-3288
AUSTRALIA:	Tel:	(+61) 3-9558-9999
INDIA:	Tel:	(+91) 80-559-9467

National Semiconductor

COP8780C/COP8781C/COP8782C
8-Bit One-Time Programmable (OTP) Microcontroller

General Description

The COP8780C, COP8781C and COP8782C are members of the COPS™ 8-bit microcontroller family. They are fully static microcontrollers, fabricated using double-metal, double poly silicon gate microCMOS EPROM technology. These devices are available as UV erasable or One Time Programmable (OTP). These low cost microcontrollers are complete microcomputers containing all system timing, interrupt logic, EPROM, RAM, and I/O necessary to implement dedicated control functions in a variety of applications. Features include an 8-bit memory mapped architecture, MICROWIRE/PLUS™ serial I/O, a 16-bit timer/counter with associated 16-bit autoreload/capture register, and a multi-sourced interrupt. Each I/O pin has software selectable options to adapt the device to the specific application. These devices operate over a voltage range of 4.5V to 6.0V. An efficient, regular instruction set operating at a 1 μs instruction cycle rate provides optimal throughput.

The COP8780C, COP8781C and COP8782C can be configured to EMULATE the COP880C, COP840C and COP820C microcontrollers.

Key Features

- 16-bit multi-function timer supporting
  - — PWM mode
  - — External event counter mode
  - — Input capture mode
- Crystal, RC or External Oscillator, user configurable
- 4 kbytes on-chip OTP EPROM with security feature
- 128 or 64 bytes of on-chip RAM, user configurable

I/O Features

- Memory-mapped I/O
- Software selectable I/O options (TRI-STATE®, Push-Pull, Weak Pull-Up Input, High Impedance input)
- Schmitt trigger inputs on Port G
- MICROWIRE/PLUS serial I/O
- Packages:
  - — 44 PLCC, OTP, Emulates COP880C, 36 I/O pins
  - — 40 DIP, OTP, Emulates COP880C, 36 I/O pins
  - — 28 DIP, OTP, Emulates COP820C/840C/881C, 24 I/O pins
  - — 20 DIP, OTP, Emulates COP822C/842C, 16 I/O pins
  - — 28 SO, 20 SO, OTP
  - — 44 LDCC, UV Erasable
  - — 40 CERDIP, 28 CERDIP, 20 CERDIP, UV Erasable

CPU/Instruction Set Features

- 1 μs instruction cycle time
- Three multi-source interrupts servicing
  - — External interrupt with selectable edge
  - — Timer interrupt
  - — Software interrupt
- Versatile and easy to use instruction set
- 8-bit Stack Pointer (SP)—stack in RAM
- Two 8-bit Register Indirect Data Memory Pointers (B and X)

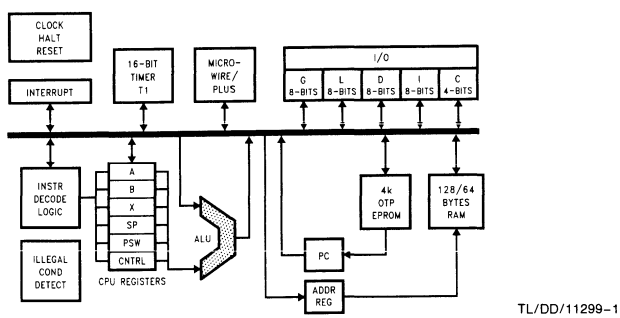
Fully Static CMOS

- Low current drain (typically < 1 μA)
- Extra-low current static HALT mode
- Single supply operation: 4.5V to 6.0V
- Temperature range: −40°C to +85°C

Development Support

- Emulation device for the COP880C, COP840C, and COP820C
- Real-time emulation and full program debug offered by MetaLink development system

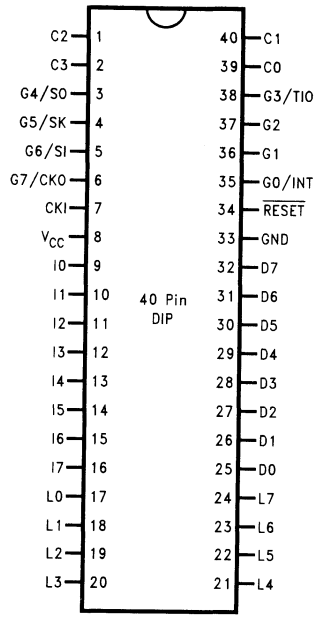
Block Diagram



TL/DD/11299–1

FIGURE 1. Block Diagram

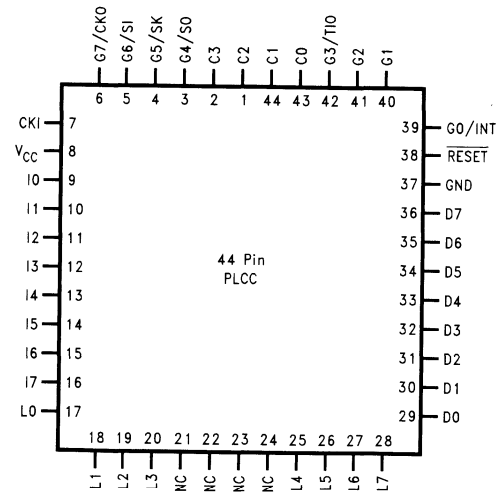
Connection Diagrams



TL/DD/11299-3

Top View

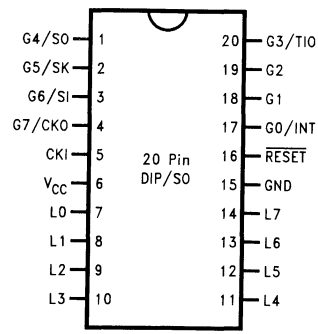
Order Number COP8780C-XXX/N or COP8780C-XXX/J
See NS Package Number J40AQ or N40A



TL/DD/11299-4

Top View

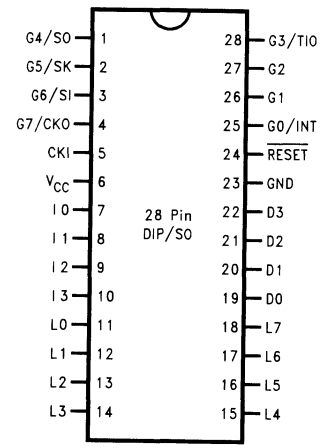
Order Number COP8780C-XXX/V or COP8780C-XXX/EL
See NS Package Number EL40C or V44A



TL/DD/11299-5

Top View

Order Number COP78782C/XXX/J, COP8782C-XXX/N or COP8782C-XXX/WM
See NS Package Number J20AQ, M20B or N20B



TL/DD/11299-6

Top View

Order Number COP8781C-XXX/J, COP8781C-XXX/N or COP8781C-XXX/WM
See NS Package Number J28AQ, M28B or N28B

FIGURE 3. Connection Diagrams

2

COP8780C/COP8781C/COP8782C

Absolute Maximum Ratings

If Military/Aerospace specified devices are required, please contact the National Semiconductor Sales Office/Distributors for availability and specifications.

Supply Voltage (V_{CC})	7V
Programming Voltage V_{PP} (RESET pin) and ME (pin G6)	13.4V
Voltage at any Pin	$-0.3V$ to $V_{CC} + 0.3V$
Total Current into V_{CC} Pin (Source)	50 mA
Total Current out of GND Pin (Sink)	60 mA
Storage Temperature Range	$-65°C$ to $+150°C$

Note: *Absolute maximum ratings indicate limits beyond which damage to the device may occur. DC and AC electrical specifications are not ensured when operating the device at absolute maximum ratings.*

DC Electrical Characteristics

COP87XXC; $-40°C \leq T_A \leq +85°C$ unless otherwise specified

Parameter	Condition	Min	Typ	Max	Units
Operating Voltage		4.5		6.0	V
Power Supply Ripple (Note 1)	Peak to Peak			0.1 V_{CC}	V
Supply Current					
CKI = 10 MHz (Note 2)	V_{CC} = 6V, t_c = 1 μs			21	mA
HALT Current (Note 3)	V_{CC} = 6V, CKI = 0 MHz			10	μA
Input Levels					
RESET, CKI					
Logic High		0.9 V_{CC}			V
Logic Low				0.1 V_{CC}	V
All Other Inputs					
Logic High		0.7 V_{CC}			V
Logic Low				0.2 V_{CC}	V
Hi-Z Input Leakage	V_{CC} = 6.0V	−2		+2	μA
Input Pullup Current	V_{CC} = 6.0V, V_{IN} = 0V	−40		−250	μA
G Port Input Hysteresis	(Note 6)		0.05 V_{CC}		V
Output Current Levels					
D Outputs					
Source	V_{CC} = 4.5V, V_{OH} = 3.8V	−0.4			mA
Sink	V_{CC} = 4.5V, V_{OL} = 1.0V	10			mA
All Others					
Source (Weak Pull-Up)	V_{CC} = 4.5V, V_{OH} = 3.2V	−10		−110	μA
Source (Push-Pull Mode)	V_{CC} = 4.5V, V_{OH} = 3.8V	−0.4			mA
Sink (Push-Pull Mode)	V_{CC} = 4.5V, V_{OL} = 0.4V	1.6			mA
TRI-STATE Leakage		−2.0		+2.0	μA
Allowable Sink/Source Current per Pin					
D Outputs (Sink)				15	mA
All Others				3	mA
Maximum Input Current (Notes 4, 6) without Latchup (Room Temp)	Room Temp			±200	mA
RAM Retention Voltage, Vr (Note 5)		2.0			V
Input Capacitance	(Note 6)			7	pF
Load Capacitance on D2	(Note 6)			1000	pF

Note 1: Rate of voltage change must be less than 0.5V/ms.

Note 2: Supply current is measured after running 2000 cycles with a square wave CKI input, CKO open, inputs at rails and outputs open.

Note 3: The HALT mode will stop CKI from oscillating in the RC and the crystal configurations. Halt test conditions: All Inputs tied to V_{CC}. L, C, and G port I/O's configured as outputs and programmed low; D outputs programmed low; the window for UV erasable packages is completely covered with an opaque cover to prevent light from falling onto the die during HALT mode test. Parameter refers to HALT mode entered via setting bit 7 of the G Port data register.

Note 4: Pins G6 and RESET are designed with a high voltage input network for factory testing. These pins allow input voltages greater than V_{CC} and the pins will have sink current to V_{CC} when biased at voltages greater than V_{CC} (the pins do not have source current when biased at a voltage below V_{CC}). The effective resistance to V_{CC} is 750Ω (typ). These two pins will not latch up. The voltage at the pins must be limited to less than 14V.

Note 5: To maintain RAM integrity, the voltage must not be dropped or raised instantaneously.

Note 6: Parameter characterized but not tested.

COP8780C/COP8781C/COP8782C

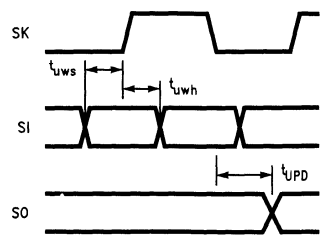
AC Electrical Characteristics $-40°C < T_A < +85°C$ unless otherwise specified

Parameter	Condition	Min	Typ	Max	Units
Instruction Cycle Time (t_c)					
Crystal/Resonator or External Clock	$V_{CC} \geq 4.5V$	1		DC	μs
R/C Oscillator Mode	$V_{CC} \geq 4.5V$	3		DC	μs
CKI Clock Duty Cycle (Note 7)	fr = Max	45		55	%
Rise Time (Note 7)	fr = 10 MHz Ext Clock			12	ns
Fall Time (Note 7)	fr = 10 MHz Ext Clock			8	ns
Inputs					
t_{SETUP}	$V_{CC} \geq 4.5V$	200			ns
t_{HOLD}	$V_{CC} \geq 4.5V$	60			ns
Output Propagation Delay	$C_L = 100$ pF, $R_L = 2.2$ kΩ				
t_{PD1}, t_{PD0}					
SO, SK	$V_{CC} \geq 4.5V$			0.7	μs
All Others	$V_{CC} \geq 4.5V$			1	μs
MICROWIRE™ Setup Time (t_{UWS})		20			ns
MICROWIRE Hold Time (t_{UWH})		56			ns
MICROWIRE Output					
Propagation Delay (t_{UPD})				220	ns
Input Pulse Width					
Interrupt Input High Time		1			t_c
Interrupt Input Low Time		1			t_c
Timer Input High Time		1			t_c
Timer Input Low Time		1			t_c
Reset Pulse Width		1.0			μs

Note 7: Parameter guaranteed by design, but not tested.

t_c = Instruction Cycle Time.

Timing Diagram



TL/DD/10802–2

FIGURE 2. MICROWIRE/PLUS Timing

Pin Descriptions

V_{CC} and GND are the power supply pins.

CKI is the clock input. This can come from an external source, a R/C generated oscillator or a crystal (in conjunction with CKO). See Oscillator description.

$\overline{\text{RESET}}$ is the master reset input. See Reset description.

PORT I is an 8-bit Hi-Z input port. The 28-pin device does not have a full complement of PORT I pins. The unavailable pins are not terminated i.e., they are floating. A read operation for these unterminated pins will return unpredictable values. The user must ensure that the software takes this into account by either masking or restricting the accesses to bit operations. The unterminated PORT I pins will draw power only when addressed.

PORT L is an 8-bit I/O port.

PORT C is a 4-bit I/O port.

Three memory locations are allocated for the L, G and C ports, one each for data register, configuration register and the input pins. Reading bits 4–7 of the C-Configuration register, data register, and input pins returns undefined data.

There are two registers associated with the L and C ports: a data register and a configuration register. Therefore, each L and C I/O bit can be individually configured under software control as shown below:

Config.	Data	Ports L and C Setup
0	0	Hi-Z Input (TRI-STATE Output)
0	1	Input with Pull-Up (Weak One Output)
1	0	Push-Pull Zero Output
1	1	Push-Pull One Output

On the 20- and 28-pin parts, it is recommended that all bits of Port C be configured as outputs to minimize current.

PORT G is an 8-bit port with 6 I/O pins (G0–G5) and 2 input pins (G6, G7). All eight G-pins have Schmitt Triggers on the inputs.

There are two registers associated with the G port: a data register and a configuration register. Therefore, each G port bit can be individually configured under software control as shown below:

Config.	Data	Port G Setup
0	0	Hi-Z Input (TRI-STATE Output)
0	1	Input with Pull-Up (Weak One Output)
1	0	Push-Pull Zero Output
1	1	Push-Pull One Output

Since G6 and G7 are input only pins, any attempt by the user to configure them as outputs by writing a one to the configuration register will be disregarded. Reading the G6 and G7 configuration bits will return zeros. The device will be placed in the HALT mode by writing a one to the G7 bit in the G-port data register.

Six pins of Port G have alternate features:

G0 INTR (an external interrupt)

G3 TIO (timer/counter input/output)

G4 SO (MICROWIRE/PLUS serial data output)

G5 SK (MICROWIRE/PLUS clock I/O)

G6 SI (MICROWIRE/PLUS serial data input)

G7 CKO crystal oscillator output (selected by programming the ECON register) or HALT Restart/general purpose input

Pins G1 and G2 currently do not have any alternate functions.

PORT D is an 8-bit output port that is preset high when $\overline{\text{RESET}}$ goes low. Care must be exercised with the D2 pin operation. At reset, the external load on this pin must ensure that the output voltage stay above 0.7 V_{CC} to prevent the chip from entering special modes. Also, keep the external loading on D2 to less than 1000 pF.

Functional Description

Figure 1 shows the block diagram of the internal architecture. Data paths are illustrated in simplified form to depict how the various logic elements communicate with each other in implementing the instruction set of the device.

ALU AND CPU REGISTERS

The ALU can do an 8-bit addition, subtraction, logical or shift operation in one cycle time.

There are five CPU registers:

A is the 8-bit Accumulator register

PU is the upper 7 bits of the program counter (PC)

PL is the lower 8 bits of the program counter (PC)

B is the 8-bit address register, can be auto incremented or decremented.

X is the 8-bit alternate address register, can be incremented or decremented.

SP is the 8-bit stack pointer, which points to the subroutine/interrupt stack in RAM. The SP must be initialized with software (usually to RAM address 06F Hex with 128 bytes of on-chip RAM selected, or to RAM address 02F Hex with 64 bytes of on-chip RAM selected). The SP is used with the subroutine call and return instructions, and with the interrupts.

B, X and SP registers are mapped into the on-chip RAM. The B and X registers are used to address the on-chip RAM. The SP register is used to address the stack in RAM during subroutine calls and returns.

PROGRAM MEMORY

The device contains 4096 bytes of UV erasable or OTP EPROM memory. This memory is mapped in the program memory address space from 0000 to 0FFF Hex. The program memory may contain either instructions or data constants, and is addressed by the 15-bit program counter (PC). The program memory can be indirectly read by the LAID (Load Accumulator Indirect) instruction for table lookup of constant data.

All locations in the EPROM program memory will contain 0FF Hex (all 1's) after the device is erased. OTP parts are shipped with all locations already erased to 0FF Hex. Unused EPROM locations should always be programmed to 00 Hex so that the software trap can be used to halt runaway program operation.

The device can be configured to inhibit external reads of the program memory. This is done by programming the security bit in the ECON (EPROM configuration) register to zero. See the ECON REGISTER section for more details.

DATA MEMORY

The data memory address space includes on-chip RAM, I/O, and registers. Data memory is addressed directly by instructions, or indirectly by means of the B, X, or SP point-

Functional Description (Continued)

ers. The device can be configured to have either 64 or 128 bytes of RAM, depending on the value of the "RAM SIZE" bit in the ECON (EPROM CONFIGURATION) register. The sixteen bytes of RAM located at data memory address 0F0–0FF are designated as "registers". These sixteen registers can be decremented and tested with the DRSZ (Decrement Register and Skip if Zero) instruction.

The three pointers X, B, and SP are memory mapped into this register address space at addresses 0FC, 0FE, and 0FD respectively. The remaining registers are available for general usage.

Any bit of data memory can be directly set, reset or tested. All of the I/O registers and control registers (except A and PC) are memory mapped. Consequently, any of the I/O bits or control register bits can be directly and individually set, reset, or tested.

Note: RAM contents are undefined upon power-up.

ECON (EPROM CONFIGURATION) REGISTER

The ECON register is used to configure the user selectable clock, security, and RAM size options. The register can be programmed and read only in EPROM programming mode. Therefore, the register should be programmed at the same time as the program memory locations 0000 through 0FFF Hex. UV erasable parts are shipped with 0FF Hex in this register while the OTP parts are shipped with 07F Hex in this register. Erasing the EPROM program memory also erases the ECON register.

The device has a security feature which, when enabled, prevents reading of the EPROM program memory. The security bit in the ECON register determines whether security is enabled or disabled. If the security option is enabled, then any attempt to externally read the contents of the EPROM will result in the value E0 Hex being read from all program memory locations. If the security option is disabled, the contents of the internal EPROM may be read. The ECON register is readable regardless of the state of the security bit.

The format of the ECON register is as follows:

TABLE I

Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
X	X	SECURITY	CKI 2	CKI 1	X	RAM SIZE	X

Bit 7 = X Don't care.

Bit 6 = X Don't care.

Bit 5 = 1 Security disabled. EPROM read and write are allowed.

= 0 Security enabled. EPROM read and write are not allowed.

Bits 4,3

= 1,1 External CKI option selected.

= 0,1 Not allowed.

= 1,0 RC oscillator option selected.

= 0,0 Crystal oscillator option selected.

Bit 2 = X Don't care.

Bit 1 = 1 Selects 128 byte RAM option. This emulates COP840 and COP880.

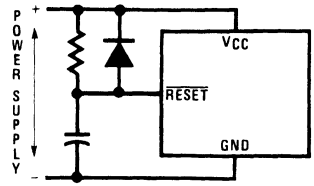
= 0 Selects 64 byte RAM option. This emulates COP820.

Bit 0 = X Don't care.

RESET

The $\overline{\text{RESET}}$ input when pulled low initializes the microcontroller. Initialization will occur whenever the $\overline{\text{RESET}}$ input is pulled low. Upon initialization, the Ports L, G and C are placed in the TRI-STATE mode and the Port D is set high. The PC, PSW and CNTRL registers are cleared. The data and configuration registers for Ports L, G and C are cleared.

The external RC network shown in *Figure 4* should be used to ensure that the $\overline{\text{RESET}}$ pin is held low until the power supply to the chip stabilizes.



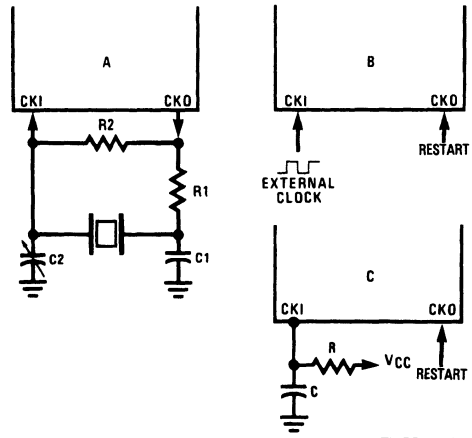
TL/DD/11299–7

RC ≥ 5X Power Supply Rise Time

FIGURE 4. Recommended Reset Circuit

OSCILLATOR CIRCUITS

Figure 5 shows the three clock oscillator configurations available for the device. The CKI 1 and CKI 2 bits in the ECON register are used to select the clock option. See the ECON REGISTER section for more details.



TL/DD/11299–8

FIGURE 5. Crystal, External and R-C Connection Diagrams

A. Crystal Oscillator

The device can be driven by a crystal clock. The crystal network is connected between the pins CKI and CKO.

Table II shows the component values required for various standard crystal frequencies.

B. External Oscillator

CKI can be driven by an external clock signal provided it meets the specified duty cycle, rise and fall times, and input levels. In External oscillator mode, G7 is available as a general purpose input and/or HALT restart control.

Functional Description (Continued)

TABLE II. Crystal Oscillator Configuration, $T_A = 25°C$

R1 (kΩ)	R2 (MΩ)	C1 (pF)	C2 (pF)	CKI Freq (MHz)	Conditions
0	1	30	30–36	10	$V_{CC} = 5V$
0	1	30	30–36	4	$V_{CC} = 5V$

TABLE III. RC Oscillator Configuration, $T_A = 25°C$

R (kΩ)	C (pF)	CKI Freq. (MHz)	Instr. Cycle (μs)	Conditions
3.3	82	2.2 to 2.7	3.7 to 4.6	$V_{CC} = 5V$
5.6	100	1.1 to 1.3	7.4 to 9.0	$V_{CC} = 5V$
6.8	100	0.9 to 1.1	8.8 to 10.8	$V_{CC} = 5V$

Note: (R/C Oscillator Configuration): $3k \leq R \leq 200k$, $50\ pF \leq C \leq 200\ pF$.

C. R/C Oscillator

CKI can be configured as a single pin RC controlled oscillator. In RC oscillator mode, G7 is available as a general purpose input and/or HALT restart control.

Table III shows the variation in the oscillator frequencies as functions of the component (R and C) values.

HALT MODE

The device supports a power saving mode of operation: HALT. The controller is placed in the HALT mode by setting the G7 data bit, alternatively the user can stop the clock input. (Stopping the clock input will draw more current than setting the G7 data bit.) In the HALT mode all internal processor activities including the clock oscillator are stopped. The fully static architecture freezes the state of the controller and retains all information until continuing. In the HALT mode, power requirements are minimal as it draws only leakage currents and output current. The applied voltage (V_{CC}) may be decreased down to Vr (minimum RAM retention voltage) without altering the state of the machine.

There are two ways to exit the HALT mode: via the $\overline{\text{RESET}}$ or by the G7 pin. A low on the $\overline{\text{RESET}}$ line reinitializes the microcontroller and starts execution from address 0000H. In external and RC oscillator modes, a low to high transition on the G7 pin also causes the microcontroller to come out of the HALT mode. Execution resumes at the address following the HALT instruction. Except for the G7 data bit, which gets reset, all RAM locations retain the values they had prior to execution of the "HALT" instruction. It is required that the first instruction following the "HALT" instruction be a "NOP" in order to synchronize the clock.

INTERRUPTS

The device has a sophisticated interrupt structure to allow easy interface to the real world. There are three possible interrupt sources, as shown below.

A maskable interrupt on external G0 input (positive or negative edge sensitive under software control)

A maskable interrupt on timer underflow or timer capture

A non-maskable software/error interrupt on opcode zero

INTERRUPT CONTROL

The GIE (global interrupt enable) bit enables the interrupt function. This is used in conjunction with ENI and ENTI to select one or both of the interrupt sources. This bit is reset when interrupt is acknowledged.

ENI and ENTI bits select external and timer interrupts respectively. Thus the user can select either or both sources to interrupt the microcontroller when GIE is enabled.

IEDG selects the external interrupt edge (0 = rising edge, 1 = falling edge). The user can get an interrupt on both rising and falling edges by toggling the state of IEDG bit after each interrupt.

IPND and TPND bits signal which interrupt is pending. After an interrupt is acknowledged, the user can check these two bits to determine which interrupt is pending. This permits the interrupts to be prioritized under software. The pending flags have to be cleared by the user. Setting the GIE bit high inside the interrupt subroutine allows nested interrupts.

The software interrupt does not reset the GIE bit. This means that the controller can be interrupted by other interrupt sources while servicing the software interrupt.

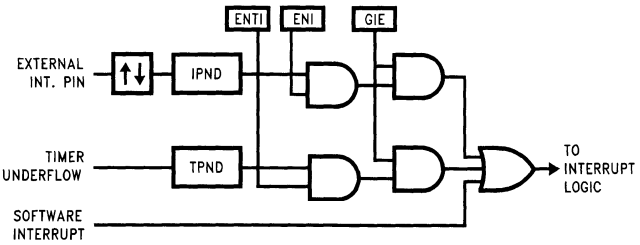
INTERRUPT PROCESSING

The interrupt, once acknowledged, pushes the program counter (PC) onto the stack and the stack pointer (SP) is decremented twice. The Global Interrupt Enable (GIE) bit is reset to disable further interrupts. The microcontroller then vectors to the address 00FFH and resumes execution from that address. This process takes 7 cycles to complete. At the end of the interrupt subroutine, any of the following three instructions return the processor back to the main program: RET, RETSK or RETI. Either one of the three instructions will pop the stack into the program counter (PC). The stack pointer is then incremented twice. The RETI instruction additionally sets the GIE bit to re-enable further interrupts.

Any of the three instructions can be used to return from a hardware interrupt subroutine. The RETSK instruction should be used when returning from a software interrupt subroutine to avoid entering an infinite loop.

Note: There is always the possiblity of an interrupt occurring during an instruction which is attempting to reset the GIE bit or any other interrupt enable bit. If this occurs when a single cycle instruction is being used to reset the interrupt enable bit, the interrupt enable bit will be reset but an interrupt may still occur. This is because interrupt processing is started at the same time as the interrupt bit is being reset. To avoid this scenario, the user should always use a two, three, or four cycle instruction to reset interrupt enable bits.

Functional Description (Continued)



TL/DD/11299-9

FIGURE 6. Interrupt Block Diagram

DETECTION OF ILLEGAL CONDITIONS

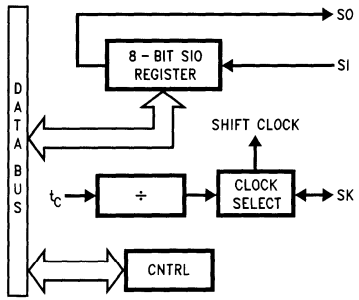
The device incorporates a hardware mechanism that allows it to detect illegal conditions which may occur from coding errors, noise and "brown out" voltage drop situations. Specifically, it detects cases of executing out of undefined EPROM area and unbalanced stack situations.

Reading an undefined EPROM location returns 00 (hexadecimal) as its contents. The opcode for a software interrupt is also "00". Thus a program accessing undefined EPROM will cause a software interrupt.

Reading an undefined RAM location returns an FF (hexadecimal). The subroutine stack on the device grows down for each subroutine call. By initializing the stack pointer to the top of RAM, the first unbalanced return instruction will cause the stack pointer to address undefined RAM. As a result the program will attempt to execute from FFFF (hexadecimal), which is an undefined EPROM location and will trigger a software interrupt.

MICROWIRE/PLUS

MICROWIRE/PLUS is a serial synchronous bidirectional communications interface. The MICROWIRE/PLUS capability enables the device to interface with any of National Semiconductor's MICROWIRE peripherals (i.e. A/D converters, display drivers, EEPROMS, etc.) and with other microcontrollers which support the MICROWIRE/PLUS interface. It consists of an 8-bit serial shift register (SIO) with serial data input (SI), serial data output (SO) and serial shift clock (SK). *Figure 7* shows the block diagram of the MICROWIRE/PLUS interface.



TL/DD/11299-10

FIGURE 7. MICROWIRE/PLUS Block Diagram

The shift clock can be selected from either an internal source or an external source. Operating the MICROWIRE/PLUS interface with the internal clock source is called the Master mode of operation. Operating the MICROWIRE/PLUS interface with an external shift clock is called the Slave mode of operation.

The CNTRL register is used to configure and control the MICROWIRE/PLUS mode. To use the MICROWIRE/PLUS, the MSEL bit in the CNTRL register is set to one. The SK clock rate is selected by the two bits, SL0 and SL1, in the CNTRL register. Table IV details the different clock rates that may be selected.

TABLE IV

SL1	SL0	SK Cycle Time
0	0	$2t_C$
0	1	$4t_C$
1	x	$8t_C$

where,

t_C is the instruction cycle time.

MICROWIRE/PLUS OPERATION

Setting the BUSY bit in the PSW register causes the MICROWIRE/PLUS arrangement to start shifting the data. It gets reset when eight data bits have been shifted. The user may reset the BUSY bit by software to allow less than 8 bits to shift. The device may enter the MICROWIRE/PLUS mode either as a Master or as a Slave. *Figure 8* shows how two device microcontrollers and several peripherals may be interconnected using the MICROWIRE/PLUS arrangement.

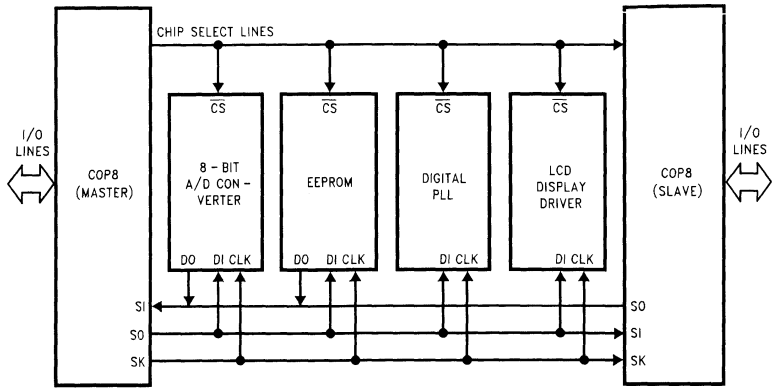
Master MICROWIRE/PLUS Operation

In the MICROWIRE/PLUS Master mode of operation the shift clock (SK) is generated internally by the device. The MICROWIRE/PLUS Master always initiates all data exchanges *(Figure 8)*. The MSEL bit in the CNTRL register must be set to enable the SO and SK functions on the G Port. The SO and SK pins must also be selected as outputs by setting appropriate bits in the Port G configuration register. Table V summarizes the bit settings required for Master mode of operation.

SLAVE MICROWIRE/PLUS OPERATION

In the MICROWIRE/PLUS Slave mode of operation the SK clock is generated by an external source. Setting the MSEL

Functional Description (Continued)



TL/DD/11299-11

FIGURE 8. MICROWIRE/PLUS Application

bit in the CNTRL register enables the SO and SK functions on the G Port. The SK pin must be selected as an input and the SO pin selected as an output pin by appropriately setting up the Port G configuration register. Table V summarizes the settings required to enter the Slave mode of operation.

The user must set the BUSY flag immediately upon entering the Slave mode. This will ensure that all data bits sent by the Master will be shifted properly. After eight clock pulses the BUSY flag will be cleared and the sequence may be repeated *(Figure 8)*.

TABLE V

G4 Config. Bit	G5 Config. Bit	G4 Fun.	G5 Fun.	G6 Fun.	Operation
1	1	SO	Int. SK	SI	MICROWIRE Master
0	1	TRI-STATE	Int. SK	SI	MICROWIRE Master
1	0	SO	Ext. SK	SI	MICROWIRE Slave
0	0	TRI-STATE	Ext. SK	SI	MICROWIRE Slave

TIMER/COUNTER

The device has a powerful 16-bit timer with an associated 16-bit register enabling it to perform extensive timer functions. The timer T1 and its register R1 are each organized as two 8-bit read/write registers. Control bits in the register CNTRL allow the timer to be started and stopped under software control. The timer-register pair can be operated in one of three possible modes. Table VI details various timer operating modes and their requisite control settings.

MODE 1. TIMER WITH AUTO-LOAD REGISTER

In this mode of operation, the timer T1 counts down at the instruction cycle rate. Upon underflow the value in the register R1 gets automatically reloaded into the timer which continues to count down. The timer underflow can be programmed to interrupt the microcontroller. A bit in the control register CNTRL enables the TIO (G3) pin to toggle upon timer underflows. This allows the generation of square-wave outputs or pulse width modulated outputs under software control *(Figure 9)*.

MODE 2. EXTERNAL COUNTER

In this mode, the timer T1 becomes a 16-bit external event counter. The counter counts down upon an edge on the TIO pin. Control bits in the register CNTRL program the counter to decrement either on a positive edge or on a negative edge. Upon underflow the contents of the register R1 are automatically copied into the counter. The underflow can also be programmed to generate an interrupt *(Figure 9)*.

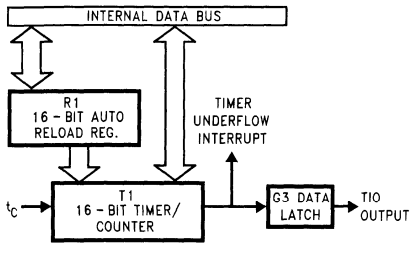
MODE 3. TIMER WITH CAPTURE REGISTER

Timer T1 can be used to precisely measure external frequencies or events in this mode of operation. The timer T1 counts down at the instruction cycle rate. Upon the occurrence of a specified edge on the TIO pin the contents of the timer T1 are copied into the register R1. Bits in the control register CNTRL allow the trigger edge to be specified either as a positive edge or as a negative edge. In this mode the user can elect to be interrupted on the specified trigger edge *(Figure 10)*.

TABLE VI. Timer Operating Modes

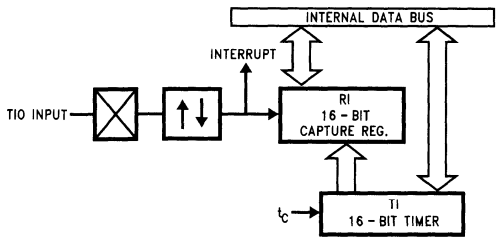
CNTRL Bits 7 6 5	Operation Mode	T Interrupt	Timer Counts On
0 0 0	External Counter w/Auto-Load Reg.	Timer Underflow	TIO Pos. Edge
0 0 1	External Counter w/Auto-Load Reg.	Timer Underflow	TIO Neg. Edge
0 1 0	Not Allowed	Not Allowed	Not Allowed
0 1 1	Not Allowed	Not Allowed	Not Allowed
1 0 0	Timer w/Auto-Load Reg.	Timer Underflow	t_C
1 0 1	Timer w/Auto-Load Reg./Toggle TIO Out	Timer Underflow	t_C
1 1 0	Timer w/Capture Register	TIO Pos. Edge	t_C
1 1 1	Timer w/Capture Register	TIO Neg. Edge	t_C

Functional Description (Continued)



TL/DD/11299-12

FIGURE 9. Timer/Counter Auto Reload Mode Block Diagram

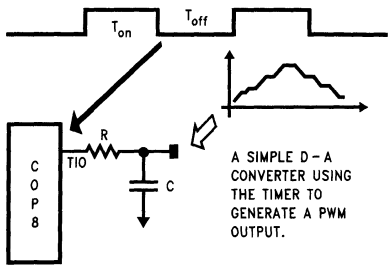


TL/DD/11299-13

FIGURE 10. Timer Capture Mode Block Diagram

TIMER PWM APPLICATION

Figure 11 shows how a minimal component D/A converter can be built out of the Timer-Register pair in the Auto-Reload mode. The timer is placed in the "Timer with auto reload" mode and the TIO pin is selected as the timer output. At the outset the TIO pin is set high, the timer T1 holds the on time and the register R1 holds the signal off time. Setting TRUN bit starts the timer which counts down at the instruction cycle rate. The underflow toggles the TIO output and copies the off time into the timer, which continues to run. By alternately loading in the on time and the off time at each successive interrupt a PWM frequency can be easily generated.



TL/DD/11299-14

FIGURE 11. Timer Application

Control Registers

CNTRL REGISTER (ADDRESS X'00EE)

The Timer and MICROWIRE/PLUS control register contains the following bits:

SL1 & SL0	Select the MICROWIRE/PLUS clock divide-by
IEDG	External interrupt edge polarity select (0 = rising edge, 1 = falling edge)
MSEL	Enable MICROWIRE/PLUS functions SO and SK
TRUN	Start/Stop the Timer/Counter (1 = run, 0 = stop)
TC3	Timer input edge polarity select (0 = rising edge, 1 = falling edge)
TC2	Selects the capture mode
TC1	Selects the timer mode

TC1	TC2	TC3	TRUN	MSEL	IEDG	S1	S0

Bit 7 Bit 0

PSW REGISTER (ADDRESS X'00EF)

The PSW register contains the following select bits:

GIE	Global interrupt enable
ENI	External interrupt enable
BUSY	MICROWIRE/PLUS busy shifting
IPND	External interrupt pending
ENTI	Timer interrupt enable
TPND	Timer interrupt pending
C	Carry Flag
HC	Half carry Flag

HC	C	TPND	ENTI	IPND	BUSY	ENI	GIE

Bit 7 Bit 0

Addressing Modes

REGISTER INDIRECT

This is the "normal" mode of addressing for the device. The operand is the memory location addressed by the B register or X register.

DIRECT

The instruction contains an 8-bit address field that directly points to the data memory location for the operand.

IMMEDIATE

The instruction contains an 8-bit immediate field as the operand.

REGISTER INDIRECT (AUTO INCREMENT AND DECREMENT)

This is a register indirect mode that automatically increments or decrements the B or X register after executing the instruction.

RELATIVE

This mode is used for the JP instruction, the instruction field is added to the program counter to get the new program location. JP has a range of −31 to +32 to allow a one byte relative jump (JP + 1 is implemented by a NOP instruction). There are no "pages" when using JP, all 15 bits of PC are used.

Memory Map

All RAM, ports and registers (except A and PC) are mapped into data memory address space.

RAM Select	Address	Contents
64 On-Chip RAM Bytes Selected by ECON reg.	00–2F 30–7F	48 On-Chip RAM Bytes Unused RAM Address Space (Reads as all 1's)
128 On-Chip RAM Bytes Selected by ECON reg.	00–6F 70–7F	112 On-chip RAM Bytes Unused RAM Address Space (Reads as all 1's)
	80 to BF	Expansion Space for On-Chip EERAM
	C0 to CF	Expansion Space for I/O and Registers
	D0 to DF D0 D1 D2 D3 D4 D5 D6 D7	On-Chip I/O and Registers Port L Data Register Port L Configuration Register Port L Input Pins (Read Only) Reserved for Port L Port G Data Register Port G Configuration Register Port G Input Pins (Read Only) Port I Input Pins (Read Only)
	D8 D9 DA DB DC DD–DF	Port C Data Register Port C Configuration Register Port C Input Pins (Read Only) Reserved for Port C Port D Data Register Reserved for Port D
	E0 to EF E0–E7 E8 E9 EA EB EC ED EE EF	On-Chip Functions and Registers Reserved for Future Parts Reserved MICROWIRE/PLUS Shift Register Timer Lower Byte Timer Upper Byte Timer Autoload Register Lower Byte Timer Autoload Register Upper Byte CNTRL Control Register PSW Register
	F0 to FF FC FD FE	On-Chip RAM Mapped as Registers X Register SP Register B Register

Reading unused memory locations below 7FH will return all ones. Reading other unused memory locations will return undefined data.

Instruction Set

REGISTER AND SYMBOL DEFINITIONS

Registers

A	8-bit Accumulator register
B	8-bit Address register
X	8-bit Address register
SP	8-bit Stack pointer register
PC	15-bit Program counter register
PU	upper 7 bits of PC
PL	lower 8 bits of PC
C	1-bit of PSW register for carry
HC	Half Carry
GIE	1-bit of PSW register for global interrupt enable

Symbols

[B]	Memory indirectly addressed by B register
[X]	Memory indirectly addressed by X register
Mem	Direct address memory or [B]
Meml	Direct address memory or [B] or Immediate data
Imm	8-bit Immediate data
Reg	Register memory: addresses F0 to FF (Includes B, X and SP)
Bit	Bit number (0 to 7)
←	Loaded with
↔	Exchanged with

Instruction Set

ADD	add	A ← A + Meml
ADC	add with carry	A ← A + Meml + C, C ← Carry HC ← Half Carry
SUBC	subtract with carry	A ← A + $\overline{\text{Meml}}$ + C, C ← Carry HC ← Half Carry
AND	Logical AND	A ← A and Meml
OR	Logical OR	A ← A or Meml
XOR	Logical Exclusive-OR	A ← A xor Meml
IFEQ	IF equal	Compare A and Meml, Do next if A = Meml
IFGT	IF greater than	Compare A and Meml, Do next if A > Meml
IFBNE	IF B not equal	Do next if lower 4 bits of B ≠ Imm
DRSZ	Decrement Reg. ,skip if zero	Reg ← Reg − 1, skip if Reg goes to 0
SBIT	Set bit	1 to bit, Mem (bit = 0 to 7 immediate)
RBIT	Reset bit	0 to bit, Mem
IFBIT	If bit	If bit, Mem is true, do next instr.
X	Exchange A with memory	A ↔ Mem
LD A	Load A with memory	A ← Meml
LD mem	Load Direct memory Immed.	Mem ← Imm
LD Reg	Load Register memory Immed.	Reg ← Imm
X	Exchange A with memory [B]	A ↔ [B] (B ← B ± 1)
X	Exchange A with memory [X]	A ↔ [X] (X ← X ± 1)
LD A	Load A with memory [B]	A ← [B] (B ← B ± 1)
LD A	Load A with memory [X]	A ← [X] (X ← X ± 1)
LD M	Load Memory Immediate	[B] ← Imm (B ← B ± 1)
CLRA	Clear A	A ← 0
INCA	Increment A	A ← A + 1
DECA	Decrement A	A ← A − 1
LAID	Load A indirect from ROM	A ← ROM(PU,A)
DCORA	DECIMAL CORRECT A	A ← BCD correction (follows ADC, SUBC)
RRCA	ROTATE A RIGHT THRU C	C → A7 → . . . → A0 → C
SWAPA	Swap nibbles of A	A7 . . . A4 ↔ A3 . . . A0
SC	Set C	C ← 1, HC ← 1
RC	Reset C	C ← 0, HC ← 0
IFC	If C	If C is true, do next instruction
IFNC	If not C	If C is not true, do next instruction
JMPL	Jump absolute long	PC ← ii (ii = 15 bits, 0 to 32k)
JMP	Jump absolute	PC11..0 ← i (i = 12 bits)
JP	Jump relative short	PC ← PC + r (r is −31 to +32, not 1)
JSRL	Jump subroutine long	[SP] ← PL,[SP-1] ← PU,SP-2,PC ← ii
JSR	Jump subroutine	[SP] ← PL,[SP-1] ← PU,SP-2,PC11.. 0 ← i
JID	Jump indirect	PL ← ROM(PU,A)
RET	Return from subroutine	SP+2,PL ← [SP],PU ← [SP-1]
RETSK	Return and Skip	SP+2,PL ← [SP],PU ← [SP-1],Skip next instruction
RETI	Return from Interrupt	SP+2,PL ← [SP],PU ← [SP-1],GIE ← 1
INTR	Generate an interrupt	[SP] ← PL,[SP−1] ← PU,SP-2,PC ← 0FF
NOP	No operation	PC ← PC + 1

Bits 7–4

F	E	D	C	B	A	9	8	7	6	5	4	3	2	1	0	Bits 3–0
JP -15	JP -31	LD 0F0,#i	DRSZ 0F0	RRCA	RC	ADC A, #i	ADC A, [B]	IFBIT 0,[B]	*	LD B, 0F	IFBNE 0	JSR 0000–00FF	JMP 0000–00FF	JP + 17	INTR	0
JP -14	JP -30	LD 0F1,#i	DRSZ 0F1	*	SC	SUBC A, #i	SUBC A,[B]	IFBIT 1,[B]	*	LD B, 0E	IFBNE 1	JSR 0100–01FF	JMP 0100–01FF	JP + 18	JP + 2	1
JP -13	JP -29	LD 0F2,#i	DRSZ 0F2	X A, [X+]	X A, [B+]	IFEQ A, #i	IFEQ A,[B]	IFBIT 2,[B]	*	LD B, 0D	IFBNE 2	JSR 0200–02FF	JMP 0200–02FF	JP + 19	JP + 3	2
JP -12	JP -28	LD 0F3,#i	DRSZ 0F3	X A, [X−]	X A, [B−]	IFGT A, #i	IFGT A,[B]	IFBIT 3,[B]	*	LD B, 0C	IFBNE 3	JSR 0300–03FF	JMP 0300–03FF	JP + 20	JP + 4	3
JP -11	JP -27	LD 0F4,#i	DRSZ 0F4	*	LAID	ADD A, #i	ADD A,[B]	IFBIT 4,[B]	CLRA	LD B, 0B	IFBNE 4	JSR 0400–04FF	JMP 0400–04FF	JP + 21	JP + 5	4
JP -10	JP -26	LD 0F5,#i	DRSZ 0F5	*	JID	AND A, #i	AND A,[B]	IFBIT 5,[B]	SWAPA	LD B, 0A	IFBNE 5	JSR 0500–05FF	JMP 0500–05FF	JP + 22	JP + 6	5
JP -9	JP -25	LD 0F6,#i	DRSZ 0F6	X A, [X]	X A, [B]	XOR A, #i	XOR A,[B]	IFBIT 6,[B]	DCORA	LD B, 9	IFBNE 6	JSR 0600–06FF	JMP 0600–06FF	JP + 23	JP + 7	6
JP -8	JP -24	LD 0F7,#i	DRSZ 0F7	*	*	OR A, #i	OR A,[B]	IFBIT 7,[B]	*	LD B, 8	IFBNE 7	JSR 0700–07FF	JMP 0700–07FF	JP + 24	JP + 8	7
JP -7	JP -23	LD 0F8,#i	DRSZ 0F8	NOP	*	LD A, #i	IFC	SBIT 0,[B]	RBIT 0,[B]	LD B, 7	IFBNE 8	JSR 0800–08FF	JMP 0800–08FF	JP + 25	JP + 9	8
JP -6	JP -22	LD 0F9,#i	DRSZ 0F9	*	*	*	IFNC	SBIT 1,[B]	RBIT 1,[B]	LD B, 6	IFBNE 9	JSR 0900–09FF	JMP 0900–09FF	JP + 26	JP + 10	9
JP -5	JP -21	LD 0FA,#i	DRSZ 0FA	LD A, [X+]	LD A, [B+]	LD [B+],#i	INCA	SBIT 2,[B]	RBIT 2,[B]	LD B, 5	IFBNE 0A	JSR 0A00–0AFF	JMP 0A00–0AFF	JP + 27	JP + 11	A
JP -4	JP -20	LD 0FB,#i	DRSZ 0FB	LD A, [X−]	LD A, [B−]	LD [B−],#i	DECA	SBIT 3,[B]	RBIT 3,[B]	LD B, 4	IFBNE 0B	JSR 0B00–0BFF	JMP 0B00–0BFF	JP + 28	JP + 12	B
JP -3	JP -19	LD 0FC,#i	DRSZ 0FC	LD Md, #i	JMPL	X A,Md	*	SBIT 4,[B]	RBIT 4,[B]	LD B, 3	IFBNE 0C	JSR 0C00–0CFF	JMP 0C00–0CFF	JP + 29	JP +13	C
JP -2	JP -18	LD 0FD,#i	DRSZ 0FD	DIR	JSRL	LD A, Md	RETSK	SBIT 5,[B]	RBIT 5,[B]	LD B, 2	IFBNE 0D	JSR 0D00–0DFF	JMP 0D00–0DFF	JP + 30	JP +14	D
JP -1	JP -17	LD 0FE,#i	DRSZ 0FE	LD A, [X]	LD A, [B]	LD [B], #i	RET	SBIT 6, [B]	RBIT 6, [B]	LD B, 1	IFBNE 0E	JSR 0E00–0EFF	JMP 0E00–0EFF	JP + 31	JP +15	E
JP -0	JP -16	LD 0FF,#1	DRSZ 0FF	*	*	*	RETI	SBIT 7,[B]	RBIT 7,[B]	LD B, 0	IFBNE 0F	JSR 0F00–0FFF	JMP 0F00–0FFF	JP + 32	JP + 16	F

where, i is the immediate data Md is a directly addressed memory location * is an unused opcode (see following table)

Instruction Execution Time

Most instructions are single byte (with immediate addressing mode instruction taking two bytes).

Most single instructions take one cycle time to execute.

Skipped instructions require x number of cycles to be skipped, where x equals the number of bytes in the skipped instruction opcode.

See the BYTES and CYCLES per INSTRUCTION table for details.

Bytes and Cycles per Instruction

The following table shows the number of bytes and cycles for each instruction in the format of byte/cycle.

Arithmetic Instructions (Bytes/Cycles)

	[B]	Direct	Immed.
ADD	1/1	3/4	2/2
ADC	1/1	3/4	2/2
SUBC	1/1	3/4	2/2
AND	1/1	3/4	2/2
OR	1/1	3/4	2/2
XOR	1/1	3/4	2/2
IFEQ	1/1	3/4	2/2
IFGT	1/1	3/4	2/2
IFBNE	1/1		
DRSZ		1/3	
SBIT	1/1	3/4	
RBIT	1/1	3/4	
IFBIT	1/1	3/4	

Memory Transfer Instructions (Bytes/Cycles)

	Register Indirect		Direct	Immed.	Register Indirect Auto Incr & Decr		
	[B]	[X]			[B+, B−]	[X+, X−]	
X A,*	1/1	1/3	2/3		1/2	1/3	
LD A,*	1/1	1/3	2/3	2/2	1/2	1/3	
LD B,Imm				1/1			(If B < 16)
LD B,Imm				2/3			(If B > 15)
LD Mem,Imm			3/3		2/2		
LD Reg,Imm				2/3			

* = > Memory location addressed by B or X or directly.

Instructions Using A & C

Instructions	Bytes/Cycles
CLRA	1/1
INCA	1/1
DECA	1/1
LAID	1/3
DCORA	1/1
RRCA	1/1
SWAPA	1/1
SC	1/1
RC	1/1
IFC	1/1
IFNC	1/1

Transfer of Control Instructions

Instructions	Bytes/Cycles
JMPL	3/4
JMP	2/3
JP	1/3
JSRL	3/5
JSR	2/5
JID	1/3
RET	1/5
RETSK	1/5
RETI	1/5
INTR	1/7
NOP	1/1

Bytes and Cycles per Instruction (Continued)

The following table shows the instructions assigned to unused opcodes. This table is for information only. The operations performed are subject to change without notice. Do not use these opcodes.

Unused Opcode	Instruction	Unused Opcode	Instruction
60	NOP	A9	NOP
61	NOP	AF	LD A, [B]
62	NOP	B1	C → HC
63	NOP	B4	NOP
67	NOP	B5	NOP
8C	RET	B7	X A, [X]
99	NOP	B9	NOP
9F	LD [B], #i	BF	LD A, [X]
A7	X A, [B]		
A8	NOP		

Development Support

SUMMARY

- iceMASTER™: IM-COP8/400—Full feature in-circuit emulation for all COP8 products. A full set of COP8 Basic and Feature Family device and package specific probes are available.
- COP8 Debug Module: Moderate cost in-circuit emulation and development programming unit.
- COP8 Evaluation and Programming Unit: EPU-COP880C—low cost In-circuit simulation and development programming unit.
- Assembler: COP8-DEV-IBMA. A DOS installable cross-development Assembler, Linker, Librarian and Utility Software Development Tool Kit.
- C Compiler: COP8C. A DOS installable cross development Software Tool Kit.
- OTP/EPROM Programmer Support: Covering needs from engineering prototype, pilot production to full production environments.

Development Support (Continued)

IceMASTER (IM) IN-CIRCUIT EMULATION

The iceMASTER IM-COP8/400 is a full feature, PC-based in-circuit emulation tool developed and marketed by MetaLink Corporation to support the whole COP8 family of products. National is a resale vendor for these products.

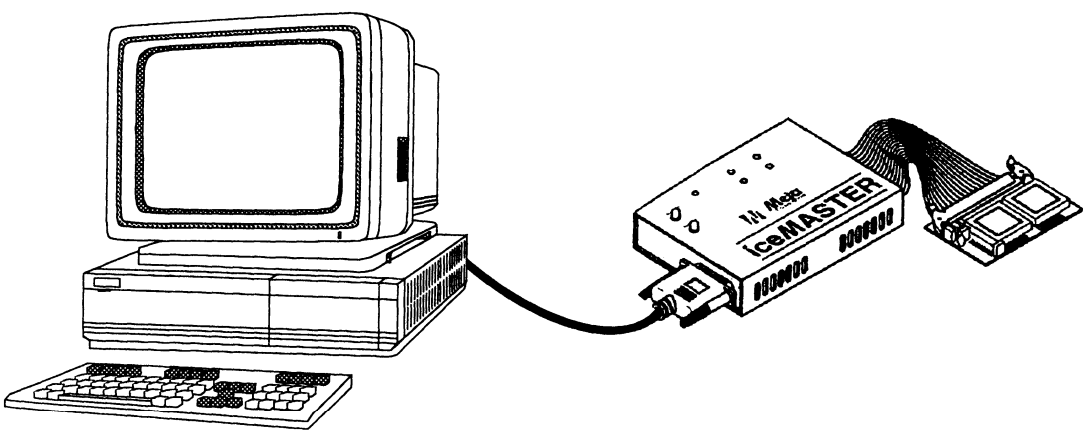
See *Figure 12* for configuration.

The iceMASTER IM-COP8/400 with its device specific COP8 Probe provides a rich feature set for developing, testing and maintaining product:

- Real-time in-circuit emulation; full 2.4-5.5VDC operation range, full DC-10 MHz clock. Chip options are programmable or jumper selectable.
- Direct connection to application board by package compatible socket or surface mount assembly.
- Full 32 kbyte of loadable programming space that overlays (replaces) the on-chip ROM or EPROM. On-chip RAM and I/O blocks are used directly or recreated on the probe as necessary.
- Full 4k frame synchronous trace memory. Address, instruction, and 8 unspecified, circuit connectable trace lines. Display can be HLL source (e.g., C source), assembly or mixed.
- A full 64k hardware configurable break, trace on, trace off control, and pass count increment events.
- Tool set integrated interactive symbolic debugger—supports both assembler (COFF) and C Compiler (.COD) linked object formats.
- Real time peformance profiling analysis; selectable bucket definition.
- Watch windows, content updated automatically at each execution break.
- Instruction by instruction memory/register changes displayed on source window when in single step operation.
- Single base unit and debugger software reconfigurable to support the entire COP8 family; only the probe personality needs to change. Debugger software is processor customized, and reconfigured from a master model file.
- Processor specific symbolic display of registers and bit level assignments, configured from master model file.
- Halt/Idle mode notification.
- On-line HELP customized to specific processor using master model file.
- Includes a copy of COP8-DEV-IBMA assembler and linker SDK.

IM Order Information

Base Unit:	
IM-COP8/400-1	iceMASTER base unit, 110V power supply
IM-COP8/400-2	iceMASTER base unit, 220V power supply
iceMASTER Probe	
MHW-880C20DWPC MHW-880C28DWPC MHW-880C40DWPC MHW-880C44PWPC	20 DIP 28 DIP 40 DIP 44 PLCC
DIP to SO Adapters	
MHW-SOIC20 MHW-SOIC28	20 SO 28 SO



TL/DD/11299-15

FIGURE 12. COP iceMASTER Environment

Development Support (Continued)

IceMASTER DEBUG MODULE (DM)

The iceMASTER Debug Module is a PC based, combination in-circuit emulation tool and COP8 based OTP/EPROM programming tool developed and marketed by MetaLink Corporation to support the whole COP8 family of products. National is a resale vendor for these products.

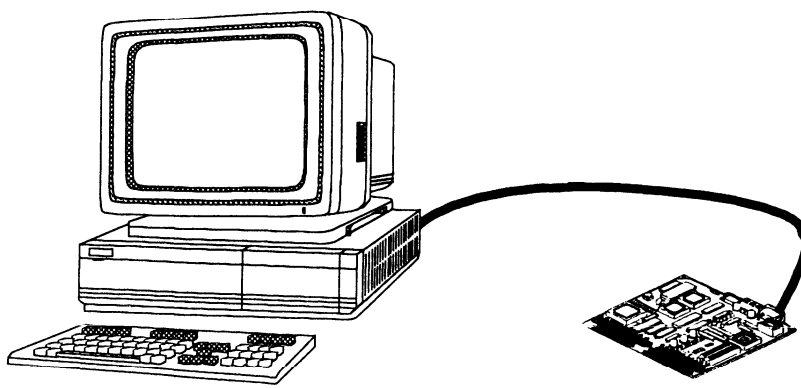
See *Figure 13* for configuration.

The iceMASTER Debug Module is a moderate cost development tool. It has the capability of in-circuit emulation for a specific COP8 microcontroller and in addition serves as a programming tool for COP8 OTP and EPROM product families. Summary of features is as follows:

- Real-time in-circuit emulation; full operating voltage range operation, full DC-10 MHz clock.
- All processor I/O pins can be cabled to an application development board with package compatible cable to socket and surface mount assembly.
- Full 32 kbyte of loadable programming space that overlays (replaces) the on-chip ROM or EPROM. On-chip RAM and I/O blocks are used directly or recreated as necessary.
- 100 frames of synchronous trace memory. The display can be HLL source (C source), assembly or mixed. The most recent history prior to a break is available in the trace memory.
- Configured break points; uses INTR instruction which is modestly intrusive.
- Software—only supported features are selectable.
- Tool set integrated interactive symbolic debugger—supports both assembler (COFF) and C Compiler (.COD) SDK linked object formats.
- Instruction by instruction memory/register changes displayed when in single step operation.
- Debugger software is processor customized, and reconfigured from a master model file.
- Processor specific symbolic display of registers and bit level assignments, configured from master model file.
- Halt/Idle mode notification.
- Programming menu supports full product line of programmable OTP and EPROM COP8 products. Program data is taken directly from the overlay RAM.
- Programming of 44 PLCC and 68 PLCC parts requires external programming adapters.
- Includes wall mount power supply.
- On-board V_{PP} generator from 5V input or connection to external supply supported. Requires V_{PP} level adjustment per the family programming specification (correct level is provided on an on-screen pop-down display).
- On-line HELP customized to specific processor using master model file.
- Includes a copy of COP8-DEV-IBMA assembler and linker SDK.

DM Order Information

Debug Module Unit	
COP8-DM/880C	
Cable Adapters	
DM-COP8/20D	20 DIP
DM-COP8/28D	28 DIP
DM-COP8/40D	40 DIP
DM-COP8/44P	44 PLCC
DIP to SO Adapters	
COP8-DM/20D-SO	20 SO
COP8-DM/28D-SO	28 SO



TL/DD/11299–16

FIGURE 13. COP8-DM Environment

Development Support (Continued)

IceMASTER EVALUATION PROGRAMMING UNIT (EPU)

The iceMASTER EPU-COP880C is a PC based, in-circuit simulation tool to support the feature family COP8 products.

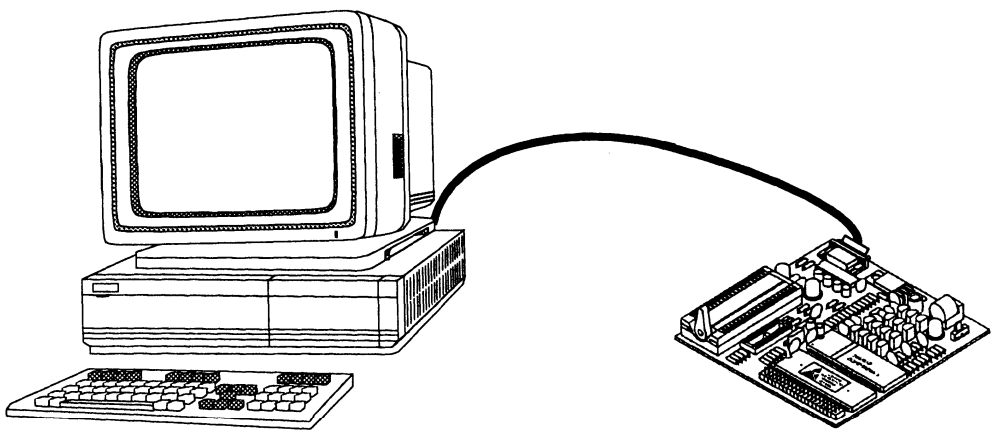
See *Figure 14* for configuration.

The simulation capability is a very low cost means of evaluating the general COP8 architecture. In addition, the EPU has programming capability, with added adapters, for programming the whole COP8 product family of OTP and EPROM products. The product includes the following features:

- Non-real-time in-circuit simulation. Program overlay memory is PC resident; instructions are downloaded over RS-232 as executed. Approximate performance is 20 kHz.
- Includes a 40 pin DIP cable adapter. Other target packages are not supported. All processor I/O pins are cabled to the application development environment.
- Full 32 kbytes of loadable programming space that overlays (replaces) the on-chip ROM or EPROM. On-chip RAM and I/O blocks are used directly or recreated as necessary.
- On-chip timer and WATCHDOG™ execution are not well synchronized to the instruction simulation.
- 100 frames of synchronous trace memory. The display can be HLL source (e.g., C source), assembly or mixed. The most recent history prior to a break is available in the trace memory.
- Up to 8 software configured break points; uses INTR instruction which is modestly intrusive.
- Common look-feel debugger software across all MetaLink products—only supported features are selectable.
- Tool set integrated interactive symbolic debugger—supports both assembler (COFF) and C Compiler (.COD) SDK linked object formats.
- Instruction by instruction memory/register changes displayed when in single step operation.
- Processor specific symbolic display of registers and bit level assignments, configured from master model file.
- Halt/Idle mode notification. Restart requires special handling.
- Programming menu supports full product line of programmable OTP and EPROM COP8 products. Only a 40 ZIF socket is available on the EPU unit. Adapters are available for other part package configurations.
- Integral wall mount power supply provides 5V and develops the required V_{PP} to program parts.
- Includes a copy of COP8-DEV-IBMA assembler, linker SDK.

EPU Order Information

Evaluation Programming Unit	
EPU-COP880C	Evaluation Programming Unit with debugger and programmer control software and 40 ZIF programming socket
General Programming Adapters	
COP8-PGMA-DS	28 and 20 DIP and SOIC adapter
COP8-PGMA-DS44P	28 and 20 DIP and SOIC plus 44 PLCC adapter



TL/DD/11299–17

FIGURE 14. EPU-COP8 Tool Environment

Development Support (Continued)

COP8 ASSEMBLER/LINKER SOFTWARE DEVELOPMENT TOOL KIT

National Semiconductor offers a relocateable COP8 macro cross assembler, linker, librarian and utility software development tool kit. Features are summarized as follows:

- Basic and Feature Family instruction set by "device" type.
- Nested macro capability.
- Extensive set of assembler directives.
- Supported on PC/DOS platform.
- Generates National standard COFF output files.
- Integrated Linker and Librarian.
- Integrated utilities to generate ROM code file outputs.
- DUMPCOFF utility.

This product is integrated as a part of MetaLink tools as a development kit, fully supported by the MetaLink debugger. It may be ordered separately or it is bundled with the MetaLink products at no additional cost.

Order Information

Assembler SDK:	
COP8-DEV-IBMA	Assembler SDK on installable 3.5" PC/DOS Floppy Disk Drive format. Periodic upgrades and most recent version is available on National's BBS and Internet.

COP8 C COMPILER

A C Compiler is developed and marketed by Byte Craft Limited. The COP8C compiler is a fully integrated development tool specifically designed to support the compact embedded configuration of the COP8 family of products.

Features are summarized as follows:

- ANSI C with some restrictions and extensions that optimize development for the COP8 embedded application.
- BITS data type extension. Register declaration #pragma with direct bit level definitions.
- C language support for interrupt routines.
- Expert system, rule based code geration and optimization.
- Performs consistency checks against the architectural definitions of the target COP8 device.
- Generates program memory code.
- Supports linking of compiled object or COP8 assembled object formats.
- Global optimization of linked code.
- Symbolic debug load format fully source level supported by the MetaLink debugger.

CROSS REFERENCE TABLE

The following cross reference table lists the COP800 devices which can be emulated with the COP87XXC single-chip, form fit and function emulators.

Single-Chip Emulator Selection Table

Device Number	Package	Description	Emulates
COP8780CV	44 PLCC	One Time Programmable (OTP)	COP880C
COP8780CEL	44 LDCC	UV Erasable	COP880C
COP8780CN	40 DIP	OTP	COP880C
COP8780CJ	40 DIP	UV Erasable	COP880C
COP8781CN	28 DIP	OTP	COP881C, COP840C, COP820C
COP8781CJ	28 DIP	UV Erasable	COP881C, COP840C, COP820C
COP8781CWM	28 SO	OTP	COP881C, COP840C, COP820C
COP8782CN	20 DIP	OTP	COP842C, COP822C
COP8782CJ	20 DIP	UV Erasable	COP842C, COP822C
COP8782CWM	20 SO	OTP	COP842C, COP822C

INDUSTRY WIDE OTP/EPROM PROGRAMMING SUPPORT

Programming support, in addition to the MetaLink development tools, is provided by a full range of independent approved vendors to meet the needs from the engineering laboratory to full production.

Development Support (Continued)

Approved List

Manufacturer	North America	Europe	Asia
BP Microsystems	(800) 225-2102 (713) 688-4600 Fax: (713) 688-0920	+49-8152-4183 +49-8856-932616	+852-234-16611 +852-2710-8121
Data I/O	(800) 426-1045 (206) 881-6444 Fax: (206) 882-1043	+44-0734-440011	Call North America
HI-LO	(510) 623-8860	Call Asia	+886-2-764-0215 Fax: +886-2-756-6403
ICE Technology	(800) 624-8949 (919) 430-7915	+44-1226-767404 Fax: 0-1226-370-434	
MetaLink	(800) 638-2423 (602) 926-0797 Fax: (602) 693-0681	+49-80 9156 96-0 Fax: +49-80 9123 86	+852-737-1800
Systems General	(408) 263-6667	+41-1-9450300	+886-2-917-3005 Fax: +886-2-911-1283
Needhams	(916) 924-8037 Fax: (916) 924-8065		

AVAILABLE LITERATURE

For more information, please see the COP8 Basic Family User's Manual, Literature Number 620895, COP8 Feature Family User's Manual, Literature Number 620897 and National's Family of 8-bit Microcontrollers COP8 Selection Guide, Literature Number 630009.

DIAL-A-HELPER SERVICE

Dial-A-Helper is a service provided by the Microcontroller Applications group. The Dial-A-Helper is an Electronic Information System that may be accessed as a Bulletin Board System (BBS) via data modem, as an FTP site on the Internet via standard FTP client application or as an FTP site on the Internet using a standard Internet browser such as Netscape or Mosaic.

The Dial-A-Helper system provides access to an automated information storage and retrieval system. The system capabilities include a MESSAGE SECTION (electronic mail, when accessed as a BBS) for communications to and from the Microcontroller Applications Group and a FILE SECTION which consists of several file areas where valuable application software and utilities could be found.

DIAL-A-HELPER BBS via a Standard Modem

Modem: CANADA/U.S.: (800) NSC-MICRO
(800) 672-6427
EUROPE: (+49) 0-8141-351332
Baud: 14.4k
Set-Up: Length: 8-Bit
Parity: None
Stop Bit: 1
Operation: 24 Hours, 7 Days

DIAL-A-HELPER via FTP

ftp nscmicro.nsc.com
user: anonymous
password: username@yourhost.site.domain

DIAL-A-HELPER via a WorldWide Web Browser

ftp://nscmicro.nsc.com

National Semiconductor on the WorldWide Web

See us on the WorldWide Web at: http://www.national.com

Development Support (Continued)

CUSTOMER RESPONSE CENTER

Complete product information and technical support is available from National's customer response centers.

CANADA/U.S.:	Tel: email:	(800) 272-9959 support @tevm2.nsc.com
EUROPE:	email:	europe.support@nsc.com
	Deutsch Tel:	+ 49 (0) 180-530 85 85
	English Tel:	+ 49 (0) 180-532 78 32
	Français Tel:	+ 49 (0) 180-532 93 58
	Italiano Tel:	+ 49 (0) 180-534 16 80
JAPAN:	Tel:	+ 81-043-299-2309
S.E. ASIA:	Beijing Tel:	(+ 86) 10-6856-8601
	Shanghai Tel:	(+ 86) 21-6415-4092
	Hong Kong Tel:	(+ 852) 2737-1600
	Korea Tel:	(+ 82) 2-3771-6909
	Malaysia Tel:	(+ 60-4) 644-9061
	Singapore Tel:	(+ 65) 255-2226
	Taiwan Tel:	+ 886-2-521-3288
AUSTRALIA:	Tel:	(+ 61) 3-9558-9999
INDIA:	Tel:	(+ 91) 80-559-9467

Programming Considerations

In addition to the application program, the ECON register needs to be programmed as well. The following tables provide examples of some ECON register values. For more detailed information refer to the ECON REGISTER section.

EPROM Security Disabled

RAM Memory	External CKI	RC Oscillator	Crystal Oscillator
64 Bytes	38	30	20
128 Bytes	3A	32	22

EPROM Security Enabled

RAM Memory	External CKI	RC Oscillator	Crystal Oscillator
64 Bytes	18	10	00
128 Bytes	1A	12	02

EPROM programmer manufacturers may not all calculate a "checksum" the same way. Before implementing an in-house verification by comparing checksums, need to ensure how each programming system utilized calculates a checksum. It is strongly recommended not to include the ECON register in the checksum for not all programmers include this byte in their calculated checksum.

ERASING THE COP8780C EPROM

The EPROM program memory is erased by exposing the transparent window on the UV erasable packages to an ultraviolet light source. Erasure begins to occur when exposed to light with wavelengths shorter than approximately 4000 Angstroms (Å). It should be noted that sunlight and certain types of fluorescent lamps have wavelengths in the 3000Å to 4000Å range.

After programming, "truly opaque" labels should be placed over the window of the device to prevent functional failure due to the generation of photo currents, erasure and excessive HALT current. The term "truly opaque" needs additional clarification when used in the context of covering quartz windows on these devices. The typical white colored stickers or labels are normally used for they are easy to write on.

These stickers are not opaque but translucent, they do let a certain percentage of UV-light through. The black write-protect labels supplied with 5.25" floppy disks work extremely well. If problems are encountered during programming (fails blank check) or during normal operation (intermittent functional or logical failures), need to determine first if an appropriate opaque label is being used to cover the quartz window at all times. Note that the device will also draw more current than normal (especially in HALT mode) when the window of the device is not covered with an opaque label.

The recommended erasure procedure for the device is exposure to short wave ultraviolet light which has a wavelength of 2537Å. The integrated dose (UV intensity × exposure time) for erasure should be a minimum of 30W-sec/cm^2.

The device should be placed within one inch of the lamp tubes during erasure. Some lamps have a filter on their tubes which should be removed before erasure. The following table shows the minimum erasure time for various light intensities.

Minimum Erasure Time

Light Intensity (Micro-Watts/cm^2)	Erasure Time* (Minutes)
15,000	36
10,000	50
8,500	60

*Does not include light intensity ramp up time.

An erasure system should be calibrated periodically. The distance from lamp to device should be maintained at one inch. The erasure time increases as the square of the distance. Lamps lose intensity as they age. When a lamp has aged, the system should be checked to make certain that adequate UV dosages are being applied for full erasure.

Common symptoms of insufficient erasure are:

- Inability to be programmed.
- Operational malfunctions associated with V_{CC}, temperature, or clock frequency.
- Loss of data in program memory.
- A change in configuration values in the ECON register.

National Semiconductor

PRELIMINARY

COP87L20CJ/COP87L22CJ 8-Bit One-Time Programmable (OTP) Microcontrollers with Multi-Input Wake-Up and Brown Out Detector

General Description

The COP87L20CJ/COP87L22CJ are members of the COP8™ 8-bit OTP Microcontroller family. It is pin and software compatible to the mask ROM COP820CJ/COP822CJ product family. The device is a fully static Microcontroller, fabricated using double-metal silicon gate microCMOS technology. Features include an 8-bit memory mapped architecture, MICROWIRE/PLUS™ serial I/O, a 16-bit timer/counter with capture register, a multi-sourced interrupt, Comparator, Brown out protection and Multi-Input Wakeup. Each I/O pin has software selectable options to adapt the device to the specific application. The device operates over a voltage range of 2.7V to 5.5V. High throughput is achieved with an efficient, regular instruction set operating at a 1 μs per instruction rate.

The equivalent mask programmable device contains the Brown Out detection feature. This feature is not supported on these OTP devices.

Key Features

- Multi-Input Wake-Up (on the 8-bit Port L)
- Analog comparator
- Modulator/timer (high speed PWM timer for IR transmission)
- 16-bit multi-function timer supporting
  - — PWM mode
  - — External event counter mode
  - — Input capture mode
- Integrated capacitor for the R/C oscillator
- 1 kbyte on-board EPROM with security feature
- 64 bytes on-board RAM

I/O Features

- Memory mapped I/O
- Software selectable I/O options (TRI-STATE®, Push-Pull, Weak Pull-Up Input, High Impedance Input)
- High current outputs (8 pins)
- Schmitt trigger inputs on Port G
- MICROWIRE/PLUS serial I/O
- Packages:
  - — 20 DIP/SO with 16 I/O pins
  - — 28 DIP/SO with 24 I/O pins

CPU/Instruction Set Features

- 1 μs instruction cycle time
- Three multi-source interrupts servicing
  - — External interrupt with selectable edge
  - — Timer interrupt
  - — Software interrupt
- Versatile and easy to use instruction set
- 8-bit stack pointer (SP)—stack in RAM
- Two 8-bit Register Indirect Data Memory Pointers (B and X)

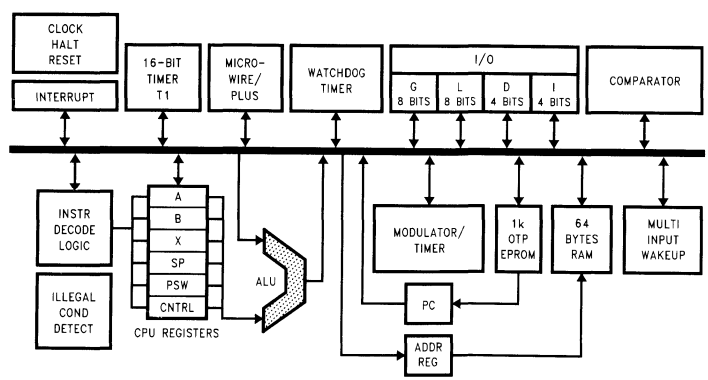
Fully Static CMOS

- Low current drain (typically < 1 μA)
- Single supply operation: 2.7V to 5.5V
- Temperature range: −40°C to +85°C

Development Support

- Emulation device for the COP820CJ/COP822CJ
- Real time emulation and full program debug offered by MetaLink Development Systems

Block Diagram



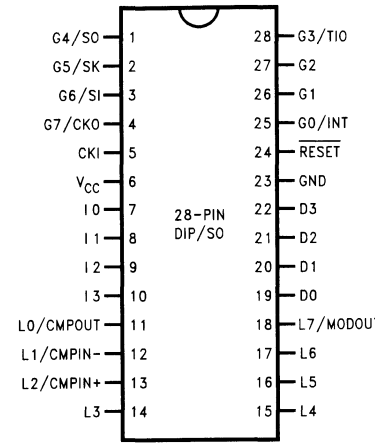
TL/DD/12528–1

FIGURE 1. Block Diagram

Pin Assignment

Port Pin	Typ	ALT Funct.	20 Pin	28 Pin
L0	I/O	MIWU/CMPOUT	7	11
L1	I/O	MIWU/CMPIN−	8	12
L2	I/O	MIWU/CMPIN+	9	13
L3	I/O	MIWU	10	14
L4	I/O	MIWU	11	15
L5	I/O	MIWU	12	16
L6	I/O	MIWU	13	17
L7	I/O	MIWU/MODOUT	14	18
G0	I/O	INTR	17	25
G1	I/O		18	26
G2	I/O		19	27
G3	I/O	TIO	20	28
G4	I/O	SO	1	1
G5	I/O	SK	2	2
G6	I	SI	3	3
G7	I	CKO	4	4
I0	I			7
I1	I			8
I2	I			9
I3	I			10
D0	O			19
D1	O			20
D2	O			21
D3	O			22
V_{CC}			6	6
GND			15	23
CKI			5	5
$\overline{\text{RESET}}$			16	24

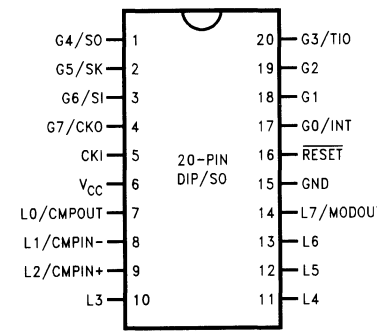
Connection Diagrams



TL/DD/12528–2

Top View

Order Number
COP87L20CJN (-1N, -2N, -3N)
COP87L20CJM (-1N, -2N, -3N)
See NS Package Number N28B or M28B



TL/DD/12528–3

Top View

Order Number
COP87L22CJN (-1N, -2N, -3N)
COP87L22CJM (-1N, -2N, -3N)
See NS Package Number N20A or M20B

Note: -1 Crystal Oscillator N - Brown out disabled
-2 External Oscillator
-3 R/C Oscillator

FIGURE 2. Connection Diagrams

Absolute Maximum Ratings

If Military/Aerospace specified devices are required, please contact the National Semiconductor Sales Office/Distributors for availability and specifications.

Supply Voltage (V_{CC})	7.0V
Voltage at any Pin	−0.3V to V_{CC} + 0.3V
Total Current into V_{CC} pin (Source)	80 mA
Total Current out of GND pin (sink)	80 mA
Storage Temperature Range	−65°C to +150°C

Note: *Absolute maximum ratings indicate limits beyond which damage to the device may occur.*

DC and AC electrical specifications are not ensured when operating the device at absolute maximum ratings.

DC Electrical Characteristics $-40°C \le T_A \le +85°C$ unless otherwise specified

Parameter	Conditions	Min	Typ	Max	Units
Operating Voltage	Peak to Peak	2.7		5.5	V
Power Supply Ripple 1 (Note 1)				0.1 V_{CC}	V
Supply Current (Note 2)					
CKI = 10 MHz	V_{CC} = 5.5V, tc = 1 μs			10	mA
CKI = 4 MHz	V_{CC} = 4.0V, tc = 2.5 μs			6.0	mA
HALT Current (Note 3)	V_{CC} = 5.5V, CKI = 0 MHz			12	μA
INPUT LEVELS (V_{IH}, V_{IL})					
Reset, CKI:					
Logic High		0.8 V_{CC}			V
Logic Low				0.2 V_{CC}	V
All Other Inputs					
Logic High		0.7 V_{CC}			V
Logic Low				0.2 V_{CC}	V
Hi-Z Input Leakage	V_{CC} = 5.5V	−2		+2	μA
Input Pullup Current	V_{CC} = 5.5V	−40		−250	μA
L- and G-Port Hysteresis (Note 6)			0.05 V_{CC}	0.35 V_{CC}	V
Output Current Levels					
D Outputs:					
Source	V_{CC} = 4.5V, V_{OH} = 3.8V	−0.4			mA
Sink (Note 4)	V_{CC} = 4.5V, V_{OL} = 1.0V	10			mA
L4–L7 Output Sink	V_{CC} = 4.5V, V_{OL} = 2.5V	15			mA
All Others					
Source (Weak Pull-up Mode)	V_{CC} = 4.5V, V_{OH} = 3.2V	−10		−110	μA
Source (Push-pull Mode)	V_{CC} = 4.5V, V_{OH} = 3.8V	−0.4			mA
Sink (Push-pull Mode)	V_{CC} = 4.5V, V_{OL} = 0.4V	1.6			mA
TRI-STATE Leakage	V_{CC} = 5.5V	−2.0		+2.0	μA
Allowable Sink/Source Current Per Pin					
D Outputs				15	mA
L4–L7 (Sink)				20	mA
All Others				3.0	mA
Maximum Input Current without Latchup (Note 5)	T_A = 25°C			±100	mA
RAM Retention Voltage, V_r	500 ns Rise and Fall Time (Min)	2.0			V
Input Capacitance				7.0	pF
Load Capacitance on D2				1000	pF

Note 1: Rate of voltage change must be less than 10 V/mS.

Note 2: Supply current is measured after running 2000 cycles with a square wave CKI input, CKO open, inputs at rails and outputs open.

Note 3: The HALT mode will stop CKI from oscillating in the RC and crystal configurations by bringing CKI high. HALT test conditions: All inputs tied to V_{CC}, L and G ports in the TRI-STATE mode are tied to ground, all outputs low are ties to ground. The comparator is disabled.

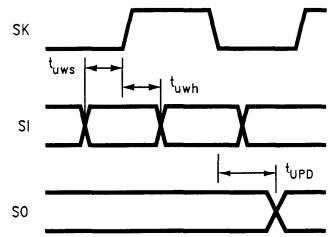
Note 4: The user must guarantee that D2 pin does not source more than 10 mA during RESET. If D2 sources more than 10 mA during reset, the device will go into programming mode.

Note 5: Pins G6 and $\overline{\text{RESET}}$ are designed with a high voltage input network. These pins allow input voltages greater than V_{CC} and the pins will have sink current to V_{CC} when biased at voltages greater than V_{CC} (the pins do not have source current when biased at a voltage below V_{CC}). The effective resistance to V_{CC} is 750Ω (typical). These two pins will not latch up. The voltage at the pins must be limited to less than 14V.

AC Electrical Characteristics $-40°C \le T_A \le +85°C$ unless otherwise specified

Parameter	Conditions	Min	Typ	Max	Units
Instruction Cycle Time (tc)					
Crystal/Resonator	$4.5V \le V_{CC} \le 5.5V$	1		DC	μs
R/C Oscillator	$4.5V \le V_{CC} \le 5.5V$	3		DC	μs
CKI Clock Duty Cycle (Note 6)	fr = Max	40		60	%
Rise Time (Note 6)	fr = 10 MHz ext. Clock			12	ns
Fall Time (Note 6)	fr = 10 MHz ext. Clock			8	ns
Inputs					
t_{Setup}	$4.5V \le V_{CC} \le 5.5V$	200			ns
t_{Hold}	$4.5V \le V_{CC} \le 5.5V$	60			ns
Output Propagation Delay	R_L = 2.2k, CL = 100 pF				
t_{PD1}, t_{PD0}					
SO, SK	$4.5V \le V_{CC} \le 5.5V$			0.7	μs
All Others	$4.5V \le V_{CC} \le 5.5V$			1	μs
Input Pulse Width					
Interrupt Input High Time		1			tc
Interrupt Input Low Time		1			tc
Timer Input High Time		1			tc
Timer Input Low Time		1			tc
MICROWIRE/PLUS Setup Time ($t_{\mu WS}$)		20			ns
MICROWIRE/PLUS Hold Time ($t_{\mu WH}$)		56			ns
MICROWIRE/PLUS Output Propagation Delay ($t_{\mu PD}$)				220	ns
Reset Pulse Width		1			μs

Note 6: Parameter characterized but not production tested.



TL/DD/12528–4

FIGURE 3. MICROWIRE/PLUS Timing

Pin Description

V_{CC} and **GND** are the power supply pins.

CKI is the clock input. This can come from an external source, a R/C generated oscillator or a crystal (in conjunction with CKO). See Oscillator description.

$\overline{\text{RESET}}$ is the master reset input. See Reset description.

PORT I is a 4-bit Hi-Z input port.

PORT L is an 8-bit I/O port.

There are two registers associated with the L port: a data register and a configuration register. Therefore, each L I/O bit can be individually configured under software control as shown below:

Port L Config.	Port L Data	Port L Setup
0	0	Hi-Z Input (TRI-STATE)
0	1	Input with Weak Pull-up
1	0	Push-pull Zero Output
1	1	Push-pull One Output

Three data memory address locations are allocated for this port, one each for data register [00D0], configuration register [00D1] and the input pins [00D2].

Port L has the following alternate features:

L0 MIWU or CMPOUT
L1 MIWU or CMPIN−
L2 MIWU or CMPIN+
L3 MIWU
L4 MIWU (high sink current capability)
L5 MIWU (high sink current capability)
L6 MIWU (high sink current capability)
L7 MIWU or MODOUT (high sink current capability)

The selection of alternate Port L functions is done through registers WKEN [00C9] to enable MIWU and CNTRL2 [00CC] to enable comparator and modulator.

All eight L-pins have Schmitt Triggers on their inputs.

PORT G is an 8-bit port with 6 I/O pins (G0–G5) and 2 input pins (G6, G7).

All eight G-pins have Schmitt Triggers on the inputs.

There are two registers associated with the G port: a data register and a configuration register. Therefore each G port bit can be individually configured under software control as shown below:

Port G Config.	Port G Data	Port G Setup
0	0	Hi-Z Input (TRI-STATE)
0	1	Input with Weak Pull-up
1	0	Push-pull Zero Output
1	1	Push-pull One Output

Three data memory address locations are allocated for this port, one for data register [00D3], one for configuration register [00D5] and one for the input pins [00D6]. Since G6 and G7 are Hi-Z input only pins, any attempt by the user to configure them as outputs by writing a one to the configuration register will be disregarded. Reading the G6 and G7 configuration bits will return zeros. Note that the device will be placed in the Halt mode by writing a "1" to the G7 data bit.

Six pins of Port G have alternate features:

G0 INTR (an external interrupt)
G3 TIO (timer/counter input/output)
G4 SO (MICROWIRE™ serial data output)
G5 SK (MICROWIRE clock I/O)
G6 SI (MICROWIRE serial data input)
G7 CKO crystal oscillator output (selected by mask option) or HALT restart input/general purpose input (if clock option is R/C or external clock)

Pins G1 and G2 currently do not have any alternate functions.

The selection of alternate Port G functions are done through registers PSW [00EF] to enable external interrupt and CNTRL1 [00EE] to select TIO and MICROWIRE operations.

PORT D is a four bit output port that is preset when RESET goes low. One data memory address location is allocated for the data register [00DC]. The user can tie two or more D port outputs (except D2 pin) together in order to get a higher drive.

Note: Care must be exercised with the D2 pin operation. At RESET, the external loads on this pin must ensure that the output voltages stay above 0.8 V_{CC} to prevent the chip from entering special modes. Also keep the external loading on D2 to less than 1000 pF.

Functional Description

ALU and CPU Registers

The ALU can do an 8-bit addition, subtraction, logical or shift operations in one cycle time. There are five CPU registers:

A is the 8-bit Accumulator register
PC is the 15-bit Program Counter register
PU is the upper 7 bits of the program counter (PC)
PL is the lower 8 bits of the program counter (PC)
B is the 8-bit address register and can be auto incremented or decremented.
X is the 8-bit alternate address register and can be auto incremented or decremented.
SP is the 8-bit stack pointer which points to the subroutine stack (in RAM).

B, X and SP registers are mapped into the on chip RAM. The B and X registers are used to address the on chip RAM. The SP register is used to address the stack in RAM during subroutine calls and returns. The SP must be preset by software before any subroutine call or interrupts occur.

Functional Description (Continued)

PROGRAM MEMORY

Program memory consists of 1 kByte of OTP EPROM. These bytes of ROM may be instructions or constant data. The memory is addressed by the 15-bit program counter (PC). ROM can be indirectly read by the LAID instruction for table lookup.

The device can be configured to inhibit external reads of the program memory. This is done by programming the Security Byte.

SECURITY FEATURE

The memory array has an associate Security Byte that is located outside of the program address range. This byte can be addressed only from programming mode by a programming tool.

Security is an optional feature and can only be asserted after the memory array has been programmed and verified. A secured part will read all 00(hex) by a programmer. The part will fail Blank Check and will fail Verify operations. A Read operation will fill the programmer's memory with 00(hex). The Security Byte itself is always readable with value of 00(hex) if unsecure and FF(hex) if secure.

DATA MEMORY

The data memory address space includes on chip RAM, I/O and registers. Data memory is addressed directly by the instruction or indirectly through B, X and SP registers. The device has 64 bytes of RAM. Sixteen bytes of RAM are mapped as "registers", these can be loaded immediately, decremented and tested. Three specific registers: X, B, and SP are mapped into this space, the other registers are available for general usage.

The instruction set permits any bit in memory to be directly set, reset or tested. All I/O and registers (except A and PC) are memory mapped; therefore, I/O bits and register bits can be directly and individually set, reset and tested, except the write once only bit (WDREN, WATCHDOG Reset Enable), and the unused and read only bits in CNTRL2 and WDREG registers.

Note: RAM contents are undefined upon power-up.

Reset

EXTERNAL RESET

The RESET input pin when pulled low initializes the microcontroller. The user must insure that the RESET pin is held low until V_{CC} is within the specified voltage range and the clock is stabilized. An R/C circuit with a delay 5x greater than the power supply rise time is recommended *(Figure 4)*. The device immediately goes into reset state when the RESET input goes low. When the RESET pin goes high the device comes out of reset state synchronously. The device will be running within two instruction cycles of the RESET pin going high. The following actions occur upon reset:

Port L	TRI-STATE
Port G	TRI-STATE
Port D	HIGH
PC	CLEARED
RAM Contents	RANDOM with Power-On-Reset UNAFFECTED with external Reset (power already applied)
B, X, SP	Same as RAM
PSW, CNTRL1, CNTRL2 and WDREG Reg.	CLEARED
Multi-Input Wakeup Reg. WKEDG, WKEN WKPND	 CLEARED UNKNOWN
Data and Configuration Registers for L and G	CLEARED
WATCHDOG Timer	Prescaler/Counter each loaded with FF

The device comes out of the HALT mode when the RESET pin is pulled low. In this case, the user has to ensure that the RESET signal is low long enough to allow the oscillator to restart. An internal 256 t_C delay is normally used in conjunction with the two pin crystal oscillator. When the device comes out of the HALT mode through Multi-Input Wakeup, this delay allows the oscillator to stabilize.

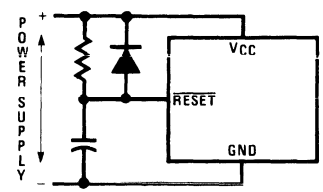
The following additional actions occur after the device comes out of the HALT mode through the RESET pin.

If a two pin crystal/resonator oscillator is being used:

RAM Contents	UNCHANGED
Timer T1 and A Contents	UNKNOWN
WATCHDOG Timer Prescaler/Counter	ALTERED

If the external or RC Clock option is being used:

RAM Contents	UNCHANGED
Timer T1 and A Contents	UNCHANGED
WATCHDOG Timer Prescaler/Counter	ALTERED



RC > 5 × Power Supply Rise Time TL/DD/12528–5

FIGURE 4. Recommended Reset Circuit

Functional Description (Continued)

WATCHDOG RESET

With WATCHDOG enabled, the WATCHDOG logic resets the device if the user program does not service the WATCHDOG timer within the selected service window. The WATCHDOG reset does not disable the WATCHDOG. Upon WATCHDOG reset, the WATCHDOG Prescaler/Counter are each initialized with FF Hex.

The following actions occur upon WATCHDOG reset that are different from external reset.

WDREN WATCHDOG Reset Enable bit UNCHANGED
WDUDF WATCHDOG Underflow bit UNCHANGED

Additional initialization actions that occur as a result of WATCHDOG reset are as follows:

Port L	TRI-STATE
Port G	TRI-STATE
Port D	HIGH
PC	CLEARED
Ram Contents	UNCHANGED
B, X, SP	UNCHANGED
PSW, CNTRL1 and CNTRL2 (except WDUDF Bit) Registers	CLEARED
Multi-Input Wakeup Registers WKEDG, WKEN WKPND	 CLEARED UNKNOWN
Data and Configuration Registers for L and G	CLEARED
WATCHDOG Timer	Prescalar/Counter each loaded with FF

Oscillator Circuits

EXTERNAL OSCILLATOR

CKI can be driven by an external clock signal provided it meets the specified duty cycle, rise and fall times, and input levels. CKO is available as a general purpose input G7 and/or Halt control.

CRYSTAL OSCILLATOR

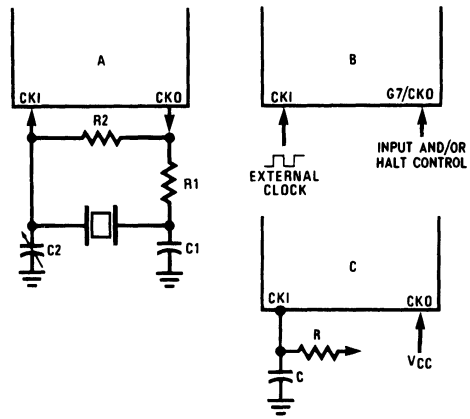
By selecting CKO as a clock output, CKI and CKO can be connected to create a crystal controlled oscillator. Table I shows the component values required for various standard crystal values.

R/C OSCILLATOR

By selecting CKI as a single pin oscillator, CKI can make a R/C oscillator. CKO is available as a general purpose input and/or HALT control. Table II shows variation in the oscillator frequencies as functions of the component (R and C) values.

The oscillator configuration is designated by -1, -2 and -3 at the end of the device ordering number.

-1 Crystal, -2 External and -3 R/C.



TL/DD/12528–6

FIGURE 5. Clock Oscillator Configurations

TABLE I. Crystal Oscillator Configuration

R1 (kΩ)	R2 (MΩ)	C1 (pF)	C2 (pF)	CKI Freq. (MHz)	Conditions
0	1	30	30–36	10	V_{CC} = 5V
0	1	30	30–36	4	V_{CC} = 5V
5.6	1	100	100–156	0.455	V_{CC} = 5V

TABLE II. RC Oscillator Configuration (Part-To-Part Variation)

R (kΩ)	C (pF)	CK1 Freq. (MHz)	Instr. Cycle (μs)	Conditions
3.3	82	1.6 to 1.4	6.1 to 7.0	V_{CC} = 5V
5.6	100	1.0 to 0.9	9.8 to 11.4	V_{CC} = 5V
6.8	100	0.8 to 0.7	12.1 to 14.3	V_{CC} = 5V

Halt Mode

The device is a fully static device. The device enters the HALT mode by writing a one to the G7 bit of the G data register. Once in the HALT mode, the internal circuitry does not receive any clock signal and is therefore frozen in the exact state it was in when halted. In this mode the chip will only draw leakage current.

The device supports three different methods of exiting the HALT mode. The first method is with a low to high transition on the CKO (G7) pin. This method precludes the use of the crystal clock configuration (since CKO is a dedicated output). It may be used either with an RC clock configuration or an external clock configuration. The second method of exiting the HALT mode is with the multi-Input Wakeup feature on the L port. The third method of exiting the HALT mode is by pulling the RESET input low.

If the two pin crystal/resonator oscillator is being used and Multi-Input Wakeup causes the device to exit the HALT mode, the WAKEUP signal does not allow the chip to start running immediately since crystal oscillators have a delayed start up time to reach full amplitude and freuqency stability. The WATCHDOG timer (consisting of an 8-bit prescaler followed by an 8-bit counter) is used to generate a fixed delay of 256tc to ensure that the oscillator has indeed stabilized before allowing instruction execution. In this case, upon detecting a valid WAKEUP signal only the oscillator circuitry is enabled. The WATCHDOG Counter and Prescaler are each loaded with a value of FF Hex. The WATCHDOG prescaler is clocked with the tc instruction cycle. (The tc clock is derived by dividing the oscillator clock down by a factor of 10).

The Schmitt trigger following the CKI inverter on the chip ensures that the WATCHDOG timer is clocked only when the oscillator has a sufficiently large amplitude to meet the Schmitt trigger specs. This Schmitt trigger is not part of the oscillator closed loop. The start-up timeout from the WATCHDOG timer enables the clock signals to be routed to the rest of the chip. The delay is not activated when the device comes out of HALT mode through RESET pin. Also, if the clock option is either RC or External clock, the delay is not used, but the WATCHDOG Prescaler/-Counter contents are changed. The Development System will not emulate the 256tc delay.

The $\overline{\text{RESET}}$ pin will cause the device to reset and start executing from address X'0000. A low to high transition on the G7 pin (if single pin oscillator is used) or Multi-Input Wakeup will cause the device to start executing from the address following the HALT instruction.

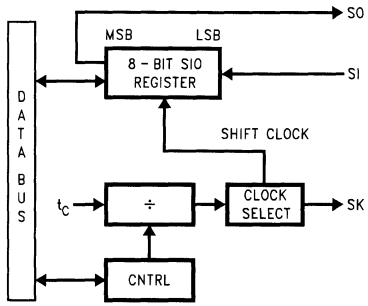
When $\overline{\text{RESET}}$ pin is used to exit the device from the HALT mode and the two pin crystal/resonator (CKI/CKO) clock option is selected, the contents of the Accumulator and the Timer T1 are undetermined following the reset. All other information except the WATCHDOG Prescaler/Counter contents is retained until continuing. All information except the WATCHDOG Prescaler/Counter contents is retained if the device exits the HALT mode through G7 pin or Multi-Input Wakeup.

G7 is the HALT-restart pin, but it can still be used as an input. If the device is not halted, G7 can be used as a general purpose input.

Note: To allow clock resynchronization, it is necessary to program two NOP's immediately after the device comes out of the HALT mode. The user must program two NOP's following the "enter HALT mode" (set G7 data bit) instruction.

MICROWIRE/PLUS

MICROWIRE/PLUS is a serial synchronous bidirectional communications interface. The MICROWIRE/PLUS capability enables the device to interface with any of National Semiconductor's MICROWIRE peripherals (i.e. A/D converters, display drivers, EEPROMS, etc.) and with other microcontrollers which support the MICROWIRE/PLUS interface. It consists of an 8-bit serial shift register (SIO) with serial data input (SI), serial data output (SO) and serial shift clock (SK). *Figure 6* shows the block diagram of the MICROWIRE/PLUS interface.



TL/DD/12528-7

FIGURE 6. MICROWIRE/PLUS Block Diagram

The shift clock can be selected from either an internal source or an external source. Operating the MICROWIRE/PLUS interface with the internal clock source is called the Master mode of operation. Operating the MICROWIRE/PLUS interface with an external shift clock is called the Slave mode of operation.

The CNTRL register is used to configure and control the MICROWIRE/PLUS mode. To use the MICROWIRE/PLUS, the MSEL bit in the CNTRL register is set to one. The SK clock rate is selected by the two bits, SL0 and SL1, in the CNTRL register. Table III details the different clock rates that may be selected.

TABLE III

SL1	SL0	SK Cycle Time
0	0	$2t_C$
0	1	$4t_C$
1	x	$8t_C$

where,

t_C is the instruction cycle time.

MICROWIRE/PLUS OPERATION

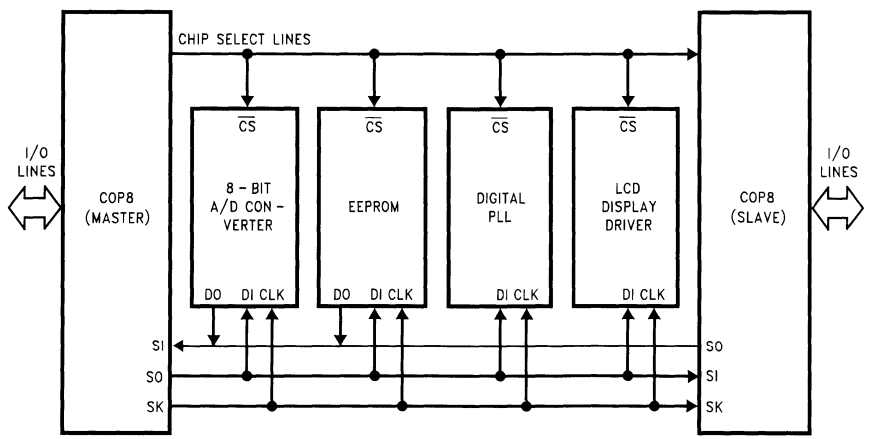
Setting the BUSY bit in the PSW register causes the MICROWIRE/PLUS arrangement to start shifting the data. It gets reset when eight data bits have been shifted. The user may reset the BUSY bit by software to allow less than 8 bits to shift. The device may enter the MICROWIRE/PLUS mode either as a Master or as a Slave. *Figure 7* shows how two device microcontrollers and several peripherals may be interconnected using the MICROWIRE/PLUS arrangement.

Master MICROWIRE/PLUS Operation

In the MICROWIRE/PLUS Master mode of operation the shift clock (SK) is generated internally by the device. The MICROWIRE/PLUS Master always initiates all data exchanges *(Figure 7)*. The MSEL bit in the CNTRL register must be set to enable the SO and SK functions on the G Port. The SO and SK pins must also be selected as outputs by setting appropriate bits in the Port G configuration register. Table IV summarizes the bit settings required for Master mode of operation.

Slave MICROWIRE/PLUS Operation

In the MICROWIRE/PLUS Slave mode of operation the SK clock is generated by an external source. Setting the MSEL bit in the CNTRL register enables the SO and SK functions on the G Port. The SK pin must be selected as an input and the SO pin selected as an output pin by appropriately setting up the Port G configuration register. Table IV summarizes the settings required to enter the Slave mode of operation.



TL/DD/12528-8

FIGURE 7. MICROWIRE/PLUS Application

MICROWIRE/PLUS (Continued)

The user must set the BUSY flag immediately upon entering the Slave mode. This will ensure that all data bits sent by the Master will be shifted properly. After eight clock pulses the BUSY flag will be cleared and the sequence may be repeated (see *Figure 7*).

TABLE IV

G4 Config. Bit	G5 Config. Bit	G4 Fun.	G5 Fun.	G6 Fun.	Operation
1	1	SO	Int. SK	SI	MICROWIRE Master
0	1	TRI-STATE	Int. SK	SI	MICROWIRE Master
1	0	SO	Ext. SK	SI	MICROWIRE Slave
0	0	TRI-STATE	Ext. SK	SI	MICROWIRE Slave

Timer/Counter

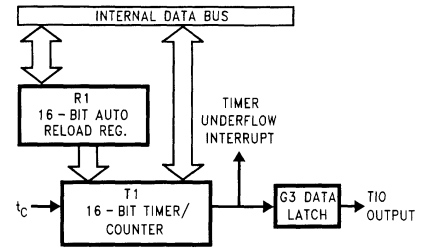
The device has a powerful 16-bit timer with an associated 16-bit register enabling it to perform extensive timer functions. The timer T1 and its register R1 are each organized as two 8-bit read/write registers. Control bits in the register CNTRL allow the timer to be started and stopped under software control. The timer-register pair can be operated in one of three possible modes. Table V details various timer operating modes and their requisite control settings.

MODE 1. TIMER WITH AUTO-LOAD REGISTER

In this mode of operation, the timer T1 counts down at the instruction cycle rate. Upon underflow the value in the register R1 gets automatically reloaded into the timer which continues to count down. The timer underflow can be programmed to interrupt the microcontroller. A bit in the control register CNTRL enables the TIO (G3) pin to toggle upon timer underflows. This allows the generation of square-wave outputs or pulse width modulated outputs under software control *(Figure 8)*.

MODE 2. EXTERNAL COUNTER

In this mode, the timer T1 becomes a 16-bit external event counter. The counter counts down upon an edge on the TIO pin. Control bits in the register CNTRL program the counter to decrement either on a positive edge or on a negative edge. Upon underflow the contents of the register R1 are automatically copied into the counter. The underflow can also be programmed to generate an interrupt *(Figure 8)*.



TL/DD/12528–9

FIGURE 8. Timer/Counter Auto Reload Mode Block Diagram

MODE 3. TIMER WITH CAPTURE REGISTER

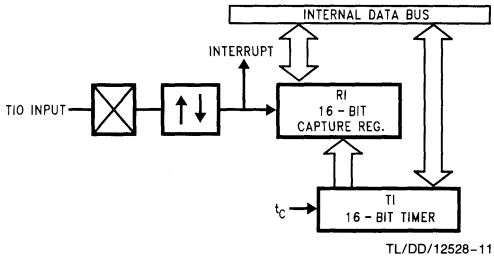
Timer T1 can be used to precisely measure external frequencies or events in this mode of operation. The timer T1 counts down at the instruction cycle rate. Upon the occurrence of a specified edge on the TIO pin the contents of the

TABLE V. Timer Operating Modes

CNTRL Bits 7 6 5	Operation Mode	T Interrupt	Timer Counter On
0 0 0	External Counter w/Auto-Load Reg.	Timer Underflow	TIO Pos. Edge
0 0 1	External Counter w/Auto-Load Reg.	Timer Underflow	TIO Neg. Edge
0 1 0	Not Allowed	Not Allowed	Not Allowed
0 1 1	Not Allowed	Not Allowed	Not Allowed
1 0 0	Timer w/Auto-Load Reg.	Timer Underflow	t_C
1 0 1	Timer w/Auto-Load Reg./Toggle TIO Out	Timer Underflow	t_C
1 1 0	Timer w/Capture Register	TIO Pos. Edge	t_C
1 1 1	Timer w/Capture Register	TIO Neg. Edge	t_C

Timer/Counter (Continued)

timer T1 are copied into the register R1. Bits in the control register CNTRL allow the trigger edge to be specified either as a positive edge or as a negative edge. In this mode the user can elect to be interrupted on the specified trigger edge *(Figure 9)*.

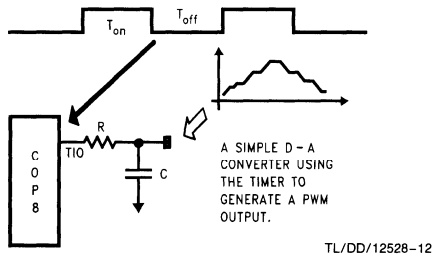


TL/DD/12528–11

FIGURE 9. Timer Capture Mode Block Diagram

TIMER PWM APPLICATION

Figure 10 shows how a minimal component D/A converter can be built out of the Timer-Register pair in the Auto-Reload mode. The timer is placed in the "Timer with auto reload" mode and the TIO pin is selected as the timer output. At the outset the TIO pin is set high, the timer T1 holds the on time and the register R1 holds the signal off time. Setting TRUN bit starts the timer which counts down at the instruction cycle rate. The underflow toggles the TIO output and copies the off time into the timer, which continues to run. By alternately loading in the on time and the off time at each successive interrupt a PWM frequency can be easily generated.



TL/DD/12528–12

FIGURE 10. Timer Application

WATCHDOG

The device has an on-board 8-bit WATCHDOG timer. The timer contains an 8-bit READ/WRITE down counter clocked by an 8-bit prescaler. Under software control the timer can be dedicated for the WATCHDOG or used as a general purpose counter. *Figure 11* shows the WATCHDOG timer block diagram.

MODE 1: WATCHDOG TIMER

The WATCHDOG is designed to detect user programs getting stuck in infinite loops resulting in loss of program control or "runaway" programs. The WATCHDOG can be enabled or disabled (only once) after the device is reset as a result of external reset. On power-up the WATCHDOG is disabled. The WATCHDOG is enabled by writing a "1" to WDREN bit (resides in WDREG register). Once enabled, the user program should write periodically into the 8-bit counter before the counter underflows. The 8-bit counter (WDCNT) is memory mapped at address 0CE Hex. The counter is loaded with n-1 to get n counts. The counter underflow resets the device, but does not disable the WATCHDOG. Loading the 8-bit counter initializes the prescaler with FF Hex and starts the prescaler/counter. Prescaler and counter are stopped upon counter underflow. Prescaler and counter are each loaded with FF Hex when the device goes into the HALT mode. The prescaler is used for crystal/resonator start-up when the device exits the HALT mode through Multi-Input Wakeup. In this case, the prescaler/counter contents are changed.

MODE 2: TIMER

In this mode, the prescaler/counter is used as a timer by keeping the WDREN (WATCHDOG reset enable) bit at 0. The counter underflow sets the WDUDF (underflow) bit and the underflow does not reset the device. Loading the 8-bit counter (load n-1 for n counts) sets the WDTEN bit (WATCHDOG Timer Enable) to "1", loads the prescaler with FF, and starts the timer. The counter underflow stops the timer. The WDTEN bit serves as a start bit for the WATCHDOG timer. This bit is set when the 8-bit counter is loaded by the user program. The load could be as a result of WATCHDOG service (WATCHDOG timer dedicated for WATCHDOG function) or write to the counter (WATCHDOG timer used as a general purpose counter). The bit is cleared upon Brown Out reset, WATCHDOG reset or external reset. The bit is not memory mapped and is transparent to the user program.

TABLE VI. WATCHDOG Control/Status

Parameter	HALT Mode	WD Reset	EXT Reset	Counter Load
8-Bit Prescaler	FF	FF	FF	FF
8-Bit WD Counter	FF	FF	FF	User Value
WDREN Bit	Unchanged	Unchanged	0	No Effect
WDUDF Bit	0	Unchanged	0	0
WDTEN Signal	Unchanged	0	0	1

WATCHDOG (Continued)

CONTROL/STATUS BITS

WDUDF: WATCHDOG Timer Underflow Bit

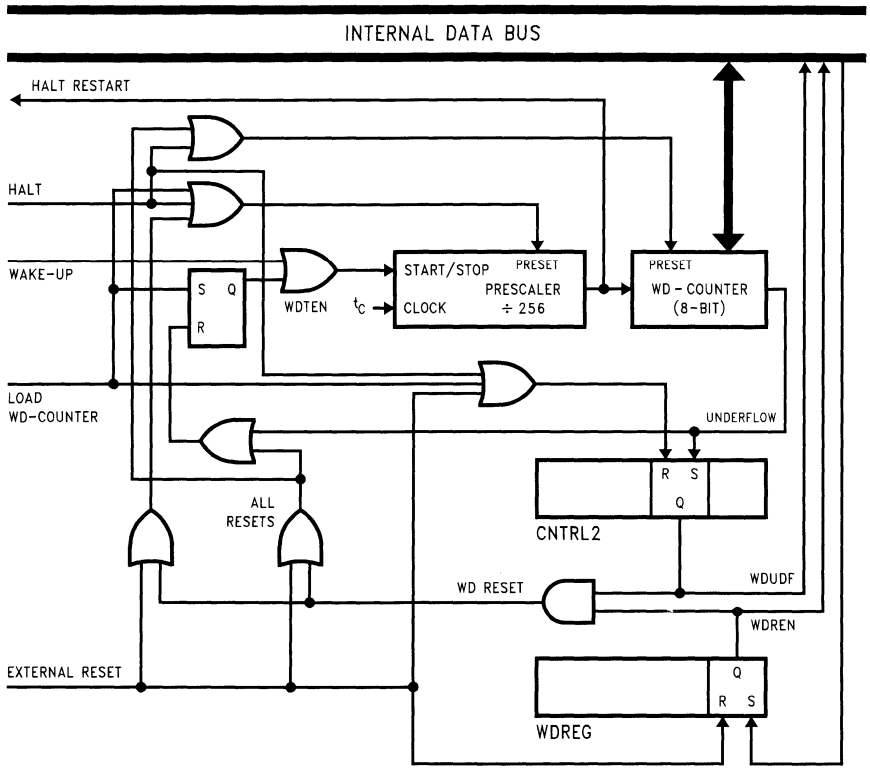
This bit resides in the CNTRL2 Register. The bit is set when the WATCHDOG timer underflows. The underflow resets the device if the WATCHDOG reset enable bit is set (WDREN = 1). Otherwise, WDUDF can be used as the timer underflow flag. The bit is cleared upon external reset, load to the 8-bit counter, or going into the HALT mode. It is a read only bit.

WDREN: WD Reset Enable

WDREN bit resides in a separate register (bit 0 of WDREG). This bit enables the WATCHDOG timer to generate a reset. The bit is cleared upon external reset. The bit under software control can be written to only once (once written to, the hardware does not allow the bit to be changed during program execution).

WDREN = 1 WATCHDOG reset is enabled.
WDREN = 0 WATCHDOG reset is disabled.

Table VI shows the impact of WATCHDOG Reset, and External Reset on the Control/Status bits.



TL/DD/12528-13

FIGURE 11. WATCHDOG Timer Block Diagram

Modulator/Timer

The Modulator/Timer contains an 8-bit counter and an 8-bit autoreload register (MODRL address 0CF Hex). The Modulator/Timer has two modes of operation, selected by the control bit MC3. The Modulator/Timer Control bits MC1, MC2 and MC3 reside in CNTRL2 Register.

MODE 1: MODULATOR

The Modulator is used to generate high frequency pulses on the modulator output pin (L7). The L7 pin should be configured as an output. The number of pulses is determined by the 8-bit down counter. Under software control the modulator input clock can be either CKI or tC. The tC clock is derived by dividing down the oscillator clock by a factor of 10. Three control bits (MC1, MC2, and MC3) are used for the Modulator/Timer output control. When MC2 = 1 and MC3 = 1, CKI is used as the modulator input clock. When MC2 = 0, and MC3 = 1, tC is used as the modulator input clock. The user loads the counter with the desired number of counts (256 max) and sets MC1 to start the counter. The modulator autoreload register is loaded with n-1 to get n pulses. CKI or tc pulses are routed to the modulator output (L7) until the counter underflows *(Figure 12).* Upon underflow the hardware resets MC1 and stops the counter. The L7 pin goes low and stays low until the counter is restarted by the user program. The user program has the responsibility to timeout the low time. Unless the number of counts is changed, the user program does not have to load the counter each time the counter is started. The counter can simply be started by setting the MC1 bit. Setting MC1 by software will load the counter with the value of the autoreload register. The software can reset MC1 to stop the counter.

MODE 2: PWM TIMER

The counter can also be used as a PWM Timer. In this mode, an 8-bit register is used to serve as an autoreload register (MODRL).

a. 50% Duty Cycle:

When MC1 is 1 and MC2, MC3 are 0, a 50% duty cycle free running signal is generated on the L7 output pin *(Figure 13).* The L7 pin must be configured as an output pin. In this mode the 8-bit counter is clocked by tC. Setting the MC1 control bit by software loads the counter with the value of the autoreload register and starts the counter. The counter underflow toggles the (L7) output pin. The 50% duty cycle signal will be continuously generated until MC1 is reset by the user program.

b. Variable Duty Cycle:

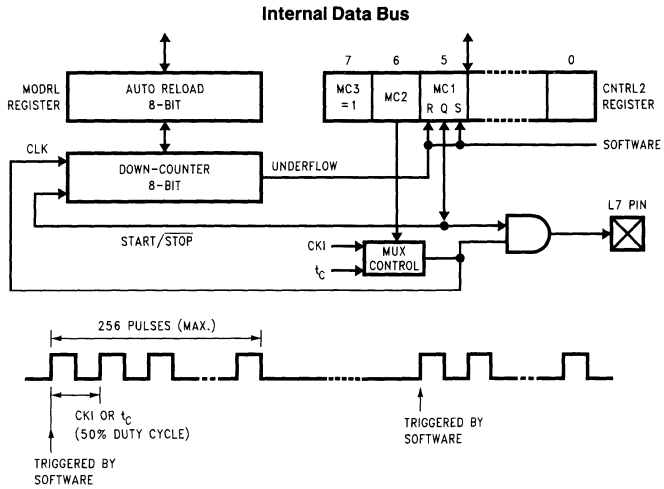
When MC3 = 0 and MC2 = 1, a variable duty cycle PWM signal is generated on the L7 output pin. The counter is clocked by tC. In this mode the 16-bit timer T1 along with the 8-bit down counter are used to generate a variable duty cycle PWM signal. The timer T1 underflow sets MC1 which starts the down counter and it also sets L7 high (L7 should be configured as an output).When the counter underflows the MC1 control bit is reset and the L7 output will go low until the next timer T1 underflow. Therefore, the width of the output pulse is controlled by the 8-bit counter and the pulse duration is controlled by the 16-bit timer T1 *(Figure 14).* Timer T1 must be configured in "PWM Mode/Toggle TIO Out" (CNTRL1 Bits 7,6,5 = 101).

Table VII shows the different operation modes for the Modulator/Timer.

TABLE VII. Modulator/Timer Modes

Control Bits in CNTRL2(00CC)			Operation Mode L7 Function
MC3	MC2	MC1	
0	0	0	Normal I/O
0	0	1	50% Duty Cycle Mode (Clocked by tc)
0	1	X	Variable Duty Cycle Mode (Clocked by tc) Using Timer 1 Underflow
1	0	X	Modulator Mode (Clocked by tc)
1	1	X	Modulator Mode (Clocked by CKI)

Note: MC1, MC2 and MC3 control bits are cleared upon reset.



TL/DD/12528-14

FIGURE 12. Mode 1: Modulator Block Diagram/Output Waveform

Modulator/Timer (Continued)

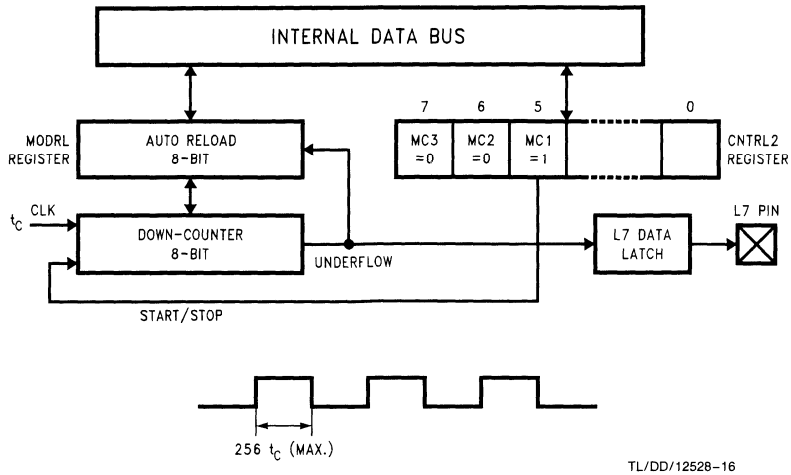


FIGURE 13. Mode 2a: 50% Duty Cycle Output

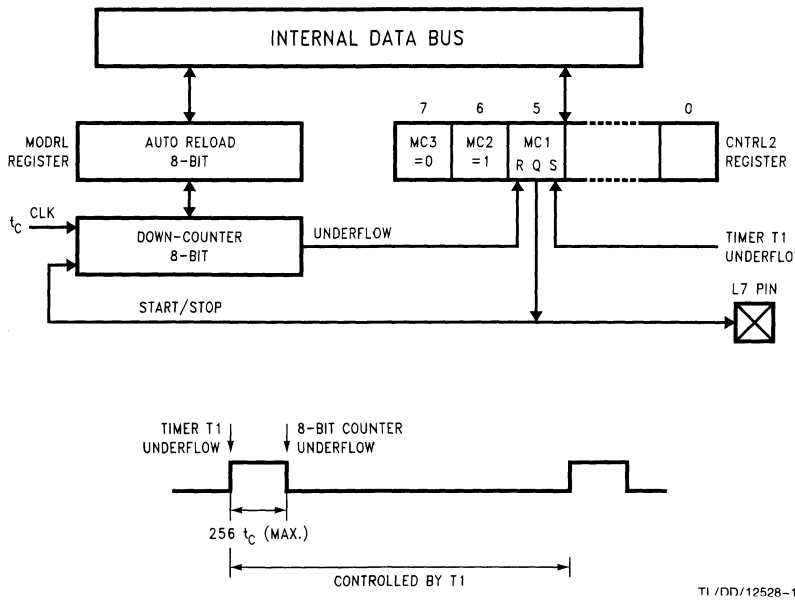


FIGURE 14. Mode 2b: Variable Duty Cycle Output

Comparator

The device has one differential comparator. Ports L0–L2 are used for the comparator. The output of the comparator is brought out to a pin. Port L has the following assignments:

L0 Comparator output
L1 Comparator negative input
L2 Comparator positive input

THE COMPARATOR STATUS/CONTROL BITS

These bits reside in the CNTRL2 Register (Address 0CC)

CMPEN Enables comparator ("1" = enable)

CMPRD Reads comparator output internally (CMPEN = 1, CMPOE=X)

CMPOE Enables comparator output to pin L0 ("1"=enable), CMPEN bit must be set to enable this function. If CMPEN=0, L0 will be 0.

The Comparator Select/Control bits are cleared on RESET (the comparator is disabled). To save power the program should also disable the comparator before the device enters the HALT mode.

The user program must set up L0, L1 and L2 ports correctly for comparator Inputs/Output: L1 and L2 need to be configured as inputs and L0 as output. Table X shows the DC amd AC characteristics for the comparator.

Multi-Input Wake Up

The Multi-Input Wakeup feature is used to return (wakeup) the device from the HALT mode. *Figure 15* shows the Multi-Input Wakeup logic.

This feature utilizes the L Port. The user selects which particular L port bit or combination of L Port bits will cause the device to exit the HALT mode. Three 8-bit memory mapped registers, Reg:WKEN, Reg:WKEDG, and Reg:WKPND are used in conjunction with the L port to implement the Multi-Input Wakeup feature.

All three registers Reg:WKEN, Reg:WKPND, and Reg:WKEDG are read/write registers, and are cleared at reset, except WKPND. WKPND is unknown on reset.

The user can select whether the trigger condition on the selected L Port pin is going to be either a positive edge (low to high transition) or a negative edge (high to low transition). This selection is made via the Reg:WKEDG, which is an 8-bit control register with a bit assigned to each L Port pin. Setting the control bit will select the trigger condition to be a negative edge on that particular L Port pin. Resetting the bit selects the trigger condition to be a positive edge. Changing an edge select entails several steps in order to avoid a pseudo Wakeup condition as a result of the edge change. First, the associated WKEN bit should be reset, followed by the edge select change in WKEDG. Next, the associated WKPND bit should be cleared, followed by the associated WKEN bit being re-enabled.

An example may serve to clarify this procedure. Suppose we wish to change the edge select from positive (low going high) to negative (high going low) for L port bit 5, where bit 5 has previously been enabled for an input. The program would be as follows:

```
RBIT 5,WKEN
SBIT 5,WKEDG
RBIT 5,WKPND
SBIT 5,WKEN
```

If the L port bits have been used as outputs and then changed to inputs with Multi-Input Wakeup, a safety procedure should also be followed to avoid inherited pseudo wakeup conditions. After the selected L port bits have been changed from output to input but before the associated WKEN bits are enabled, the associated edge select bits in WKEDG should be set or reset for the desired edge selects, followed by the associated WKPND bits being cleared. This same procedure should be used following RESET, since the L port inputs are left floating as a result of RESET.

The occurrence of the selected trigger condition for Multi-Input Wakeup is latched into a pending register called Reg:WKPND. The respective bits of the WKPND register will be set on the occurrence of the selected trigger edge on the corresponding Port L pin. The user has the responsibility of clearing these pending flags. Since the Reg:WKPND is a pending register for the occurrence of selected wakeup conditions, the device will not enter the HALT mode if any Wakeup bit is both enabled and pending. Setting the G7 data bit under this condition will not allow the device to enter the HALT mode. Consequently, the user has the responsibility of clearing the pending flags before attempting to enter the HALT mode.

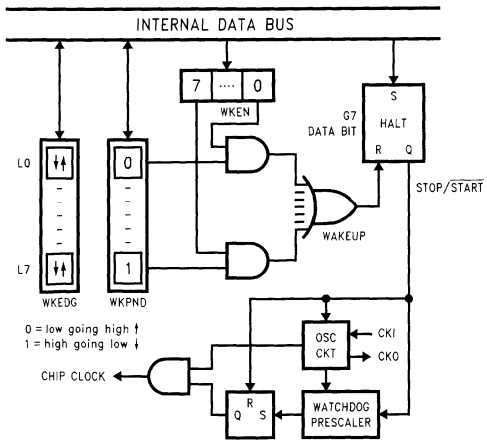
If a crystal oscillator is being used, the Wakeup signal will not start the chip running immediately since crystal oscillators have a finite start up time. The WATCHDOG timer prescaler generates a fixed delay to ensure that the oscillator has indeed stabilized before allowing the device to execute instructions. In this case, upon detecting a valid Wakeup signal only the oscillator circuitry and the WATCHDOG timer are enabled. The WATCHDOG timer prescaler is loaded with a value of FF Hex (256 counts) and is clocked from the tc instruction cycle clock. The tc clock is derived by dividing down the oscillator clock by a factor of 10. A Schmitt trigger following the CKI on chip inverter ensures that the WATCHDOG timer is clocked only when the oscillator has a sufficiently large amplitude to meet the Schmitt trigger specs. This Schmitt trigger is not part of the oscillator closed loop. The startup timeout from the WATCHDOG timer enables the clock signals to be routed to the rest of the chip.

Comparator DC and AC Characteristics $4.5V \le V_{CC} \le 5.5V$, $-40°C \le T_A \le +85°C$ (Note 1)

Parameters	Conditions	Min	Type	Max	Units
Input Offset Voltage	$0.4V < V_{IN} < V_{CC} - 1.5V$		±10	±25	mV
Input Common Mode Voltage Range		0.4		$V_{CC} - 1.5$	V
Voltage Gain			300k		V/V
DC Supply Current (when enabled)	$V_{CC} = 5.5V$			250	μA
Response Time	TBD mV Step, TBD mV Overdrive, 100 pF Load			1	μs

Note 1: For comparator output current characteristics see L-Port specs.

Multi-Input Wakeup (Continued)



TL/DD/12528-19

FIGURE 15. Multi-Input Wakeup Logic

INTERRUPTS

The device has a sophisticated interrupt structure to allow easy interface to the real world. There are three possible interrupt sources, as shown below.

A maskable interrupt on external G0 input (positive or negative edge sensitive under software control)

A maskable interrupt on timer carry or timer capture

A non-maskable software/error interrupt on opcode zero

INTERRUPT CONTROL

The GIE (global interrupt enable) bit enables the interrupt function. This is used in conjunction with ENI and ENTI to select one or both of the interrupt sources. This bit is reset when interrupt is acknowledged.

ENI and ENTI bits select external and timer interrupts respectively. Thus the user can select either or both sources to interrupt the microcontroller when GIE is enabled.

IEDG selects the external interrupt edge (0 = rising edge, 1 = falling edge). The user can get an interrupt on both rising and falling edges by toggling the state of IEDG bit after each interrupt.

IPND and TPND bits signal which interrupt is pending. After an interrupt is acknowledged, the user can check these two bits to determine which interrupt is pending. This permits the interrupts to be prioritized under software. The pending flags have to be cleared by the user. Setting the GIE bit high inside the interrupt subroutine allows nested interrupts.

The software interrupt does not reset the GIE bit. This means that the controller can be interrupted by other interrupt sources while servicing the software interrupt.

INTERRUPT PROCESSING

The interrupt, once acknowledged, pushes the program counter (PC) onto the stack and the stack pointer (SP) is decremented twice. The Global Interrupt Enable (GIE) bit is reset to disable further interrupts. The microcontroller then vectors to the address 00FFH and resumes execution from that address. This process takes 7 cycles to complete. At the end of the interrupt subroutine, any of the following three instructions return the processor back to the main program: RET, RETSK or RETI. Either one of the three instructions will pop the stack into the program counter (PC). The stack pointer is then incremented twice. The RETI instruction additionally sets the GIE bit to re-enable further interrupts.

Any of the three instructions can be used to return from a hardware interrupt subroutine. The RETSK instruction should be used when returning from a software interrupt subroutine to avoid entering an infinite loop.

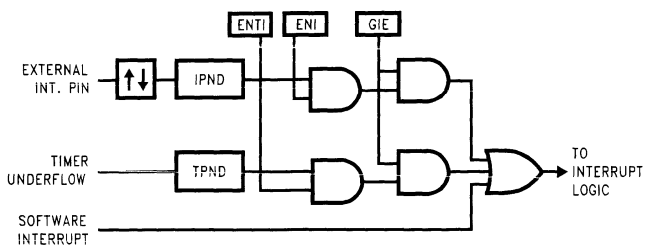
Note: There is always the possibility of an interrupt occurring during an instruction which is attempting to reset the GIE bit or any other interrupt enable bit. If this occurs when a single cycle instruction is being used to reset the interrupt enable bit, the interrupt enable bit will be reset but an interrupt may still occur. This is because interrupt processing is started at the same time as the interrupt bit is being reset. To avoid this scenario, the user should always use a two, three, or four cycle instruction to reset interrupt enable bits.

DETECTION OF ILLEGAL CONDITIONS

The device incorporates a hardware mechanism that allows it to detect illegal conditions which may occur from coding errors, noise, and "brown out" voltage drop situations. Specifically, it detects cases of executing out of undefined ROM area and unbalanced tack situations.

Reading an undefined ROM location returns 00 (hexadecimal) as its contents. The opcode for a software interrupt is also "00". Thus a program accessing undefined ROM will cause a software interrupt.

Reading an undefined RAM location returns an FF (hexadecimal). The subroutine stack on the device grows down for each subroutine call. By initializing the stack pointer to the top of RAM, the first unbalanced return instruction will cause the stack pointer to address undefined RAM. As a result the program will attempt to execute from FFFF (hexadecimal), which is an undefined ROM location and will trigger a software interrupt.



TL/DD/12528-20

FIGURE 16. Interrupt Block Diagram

Control Registers

CNTRL1 REGISTER (ADDRESS 00EE)

The Timer and MICROWIRE control register contains the following bits:

SL1 and SL0 Select the MICROWIRE clock divide-by (00 = 2, 01 = 4, 1x = 8)

IEDG External interrupt edge polarity select

MSEL Selects G5 and G4 as MICROWIRE signals SK and SO respectively

TRUN Used to start and stop the timer/counter (1 = run, 0 = stop)

TC1 Timer T1 Mode Control Bit

TC2 Timer T1 Mode Control Bit

TC3 Timer T1 Mode Control Bit

Bit 7							Bit 0
TC1	TC2	TC3	TRUN	MSEL	IEDG	SL1	SL0

PSW REGISTER (ADDRESS 00EF)

The PSW register contains the following select bits:

GIE Global interrupt enable (enables interrupts)

ENI External interrupt enable

BUSY MICROWIRE busy shifting flag

PND External interrupt pending

ENTI Timer T1 interrupt enable

TPND Timer T1 interrupt pending (timer Underflow or capture edge)

C Carry Flip/Flop

HC Half-Carry Flip/Flop

Bit 7							Bit 0
HC	C	TPND	ENTI	IPND	BUSY	ENI	GIE

The Half-Carry bit is also effected by all the instructions that effect the Carry flag. The flag values depend upon the instruction. For example, after executing the ADC instruction the values of the Carry and the Half-Carry flag depend upon the operands involved. However, instructions like SET C and RESET C will set and clear both the carry flags. Table XIII lists the instructions that effect the HC and the C flags.

TABLE XIII. Instructions Effecting HC and C Flags

Instr.	HC Flag	C Flag
ADC	Depends on Operands	Depends on Operands
SUBC	Depends on Operands	Depends on Operands
SET C	Set	Set
RESET C	Set	Set
RRC	Depends on Operands	Depends on Operands

CNTRL2 REGISTER (ADDRESS 00CC)

Bit 7							Bit 0
MC3	MC2	MC1	CMPEN	CMPRD	CMPOE	WDUDF	unused
R/W	R/W	R/W	R/W	R/O	R/W	R/O	

MC3 Modulator/Timer Control Bit

MC2 Modulator/Timer Control Bit

MC1 Modulator/Timer Control Bit

CMPEN Comparator Enable Bit

CMPRD Comparator Read Bit

CMPOE Comparator Output Enable Bit

WDUDF WATCHDOG Timer Underflow Bit (Read Only)

WDREN REGISTER (ADDRESS 00CD)

WDREN WATCHDOG Reset Enable Bit (Write Once Only)

Bit 7	Bit 0
UNUSED	WDREN

Memory Map

All RAM, ports and registers (except A and PC) are mapped into data memory address space.

TABLE IX. Memory Map

Address	Contents
00 to 2F	On-chip RAM bytes (48 bytes)
30 to 7F	Unused RAM Address Space (Reads as All Ones)
80 to BF	Expansion Space for On-Chip EERAM (Reads Undefined Data)
C0 to C7	Reserved
C8	MIWU Edge Select Register (Reg:WKEDG)
C9	MIWU Enable Register (Reg:WKEN)
CA	MIWU Pending Register (Reg:WKPND)
CB	Reserved
CC	Control2 Register (CNTRL2)
CD	WATCHDOG Register (WDREG)
CE	WATCHDOG Counter (WDCNT)
CF	Modulator Reload (MODRL)
D0	Port L Data Register
D1	Port L Configuration Register
D2	Port L Input Pins (Read Only)
D3	Reserved for Port L
D4	Port G Data Register
D5	Port G Configuration Register
D6	Port G Input Pins (Read Only)
D7	Port I Input Pins (Read Only)
D8 to DB	Reserved for Port C
DC	Port D Data Register
DD to DF	Reserved for Port D
E0 to EF	On-Chip Functions and Registers
E0 to E7	Reserved for Future Parts
E8	Reserved
E9	MICROWIRE Shift Register
EA	Timer Lower Byte
EB	Timer Upper Byte
EC	Timer1 Autoreload Register Lower Byte
ED	Timer1 Autoreload Register Upper Byte
EE	CNTRL1 Control Register
EF	PSW Register
F0 to FF	On-Chip RAM Mapped as Registers
FC	X Register
FD	SP Register
FE	B Register

Reading other unused memory locations will return undefined data.

Addressing Modes

There are ten addressing modes, six for operand addressing and four for transfer of control.

OPERAND ADDRESSING MODES

REGISTER INDIRECT

This is the "normal" addressing mode for the chip. The operand is the data memory addressed by the **B** or **X** pointer.

REGISTER INDIRECT WITH AUTO POST INCREMENT OR DECREMENT

This addressing mode is used with the LD and X instructions. The operand is the data memory addressed by the **B** or **X** pointer. This is a register indirect mode that automatically post increments or post decrements the **B** or **X** pointer after executing the instruction.

DIRECT

The instruction contains an 8-bit address field that directly points to the data memory for the operand.

IMMEDIATE

The instruction contains an 8-bit immediate field as the operand.

SHORT IMMEDIATE

This addressing mode issued with the LD B,# instruction, where the immediate # is less than 16. The instruction contains a 4-bit immediate field as the operand.

INDIRECT

This addressing mode is used with the LAID instruction. The contents of the accumulator are used as a partial address (lower 8 bits of PC) for accessing a data operand from the program memory.

TRANSFER OF CONTROL ADDRESSING MODES

RELATIVE

This mode is used for the JP instruction with the instruction field being added to the program counter to produce the next instruction address. JP has a range from −31 to +32 to allow a one byte relative jump (JP + 1 is implemented by a NOP instruction). There are no "blocks" or "pages" when using JP since all 15 bits of the PC are used.

ABSOLUTE

This mode is used with the JMP and JSR instructions with the instruction field of 12 bits replacing the lower 12 bits of the program counter (PC). This allows jumping to any location in the current 4k program memory segment.

ABSOLUTE LONG

This mode is used with the JMPL and JSRL instructions with the instruction field of 15 bits replacing the entire 15 bits of the program counter (PC). This allows jumping to any location up to 32k in the program memory space.

INDIRECT

This mode is used with the JID instruction. The contents of the accumulator are used as a partial address (lower 8 bits of PC) for accessing a location in the program memory. The contents of this program memory location serves as a partial address (lower 8 bits of PC) for the jump to the next instruction.

Note: The VIS is a special case of the Indirect Transfer of Control addressing mode, where the double byte vector associated with the interrupt is transferred from adjacent addresses in the program memory into the program counter (PC) in order to jump to the associated interrupt service routine.

Instruction Set

REGISTER AND SYMBOL DEFINITIONS

Registers

A	8-bit Accumulator register
B	8-bit Address register
X	8-bit Address register
SP	8-bit Stack pointer register
PC	15-bit Program counter register
PU	upper 7 bits of PC
PL	lower 8 bits of PC
C	1-bit of PSW register for carry
HC	Half Carry
GIE	1-bit of PSW register for global interrupt enable

Symbols

[B]	Memory indirectly addressed by B register
[X]	Memory indirectly addressed by X register
Mem	Direct address memory or [B]
Meml	Direct address memory or [B] or Immediate data
Imm	8-bit Immediate data
Reg	Register memory: addresses F0 to FF (Includes B, X and SP)
Bit	Bit number (0 to 7)
←	Loaded with
↔	Exchanged with

Instruction Set

ADD	add	A ← A + Meml
ADC	add with carry	A ← A + Meml + C, C ← Carry HC ← Half Carry
SUBC	subtract with carry	A ← A + $\overline{\text{Meml}}$ + C, C ← Carry HC ← Half Carry
AND	Logical AND	A ← A and Meml
OR	Logical OR	A ← A or Meml
XOR	Logical Exclusive-OR	A ← A xor Meml
IFEQ	IF equal	Compare A and Meml, Do next if A = Meml
IFGT	IF greater than	Compare A and Meml, Do next if A > Meml
IFBNE	IF B not equal	Do next if lower 4 bits of B ≠ Imm
DRSZ	Decrement Reg. ,skip if zero	Reg ← Reg − 1, skip if Reg goes to 0
SBIT	Set bit	1 to bit, Mem (bit= 0 to 7 immediate)
RBIT	Reset bit	0 to bit, Mem
IFBIT	If bit	If bit, Mem is true, do next instr.
X	Exchange A with memory	A ↔ Mem
LD A	Load A with memory	A ← Meml
LD mem	Load Direct memory Immed.	Mem ← Imm
LD Reg	Load Register memory Immed.	Reg ← Imm
X	Exchange A with memory [B]	A ↔ [B] (B ← B±1)
X	Exchange A with memory [X]	A ↔ [X] (X ← X±1)
LD A	Load A with memory [B]	A ← [B] (B ← B±1)
LD A	Load A with memory [X]	A ← [X] (X ← X±1)
LD M	Load Memory Immediate	[B] ← Imm (B ← B±1)
CLRA	Clear A	A ← 0
INCA	Increment A	A ← A + 1
DECA	Decrement A	A ← A − 1
LAID	Load A indirect from ROM	A ← ROM(PU,A)
DCORA	DECIMAL CORRECT A	A ← BCD correction (follows ADC, SUBC)
RRCA	ROTATE A RIGHT THRU C	C → A7 → ... → A0 → C
SWAPA	Swap nibbles of A	A7 ... A4 ↔ A3 ... A0
SC	Set C	C ← 1, HC ← 1
RC	Reset C	C ← 0, HC ← 0
IFC	If C	If C is true, do next instruction
IFNC	If not C	If C is not true, do next instruction
JMPL	Jump absolute long	PC ← ii (ii = 15 bits, 0 to 32k)
JMP	Jump absolute	PC11..0 ← i (i = 12 bits)
JP	Jump relative short	PC ← PC + r (r is −31 to +32, not 1)
JSRL	Jump subroutine long	[SP] ← PL,[SP-1] ← PU,SP-2,PC ← ii
JSR	Jump subroutine	[SP] ← PL,[SP-1] ← PU,SP-2,PC11.. 0 ← i
JID	Jump indirect	PL ← ROM(PU,A)
RET	Return from subroutine	SP+2,PL ← [SP],PU ← [SP-1]
RETSK	Return and Skip	SP+2,PL ← [SP],PU ← [SP-1],Skip next instruction
RETI	Return from Interrupt	SP+2,PL ← [SP],PU ← [SP-1],GIE ← 1
INTR	Generate an interrupt	[SP] ← PL,[SP−1] ← PU,SP-2,PC ← 0FF
NOP	No operation	PC ← PC + 1

OPCODE LIST

Bits 7–4

F	E	D	C	B	A	9	8	7	6	5	4	3	2	1	0	
JP -15	JP -31	LD 0F0,#i	DRSZ 0F0	RRCA	RC	ADC A, #i	ADC A, [B]	IFBIT 0,[B]	*	LD B, 0F	IFBNE 0	JSR 0000–00FF	JMP 0000–00FF	JP + 17	INTR	0
JP -14	JP -30	LD 0F1,#i	DRSZ 0F1	*	SC	SUBC A, #i	SUBC A,[B]	IFBIT 1,[B]	*	LD B, 0E	IFBNE 1	JSR 0100–01FF	JMP 0100–01FF	JP + 18	JP + 2	1
JP -13	JP -29	LD 0F2,#i	DRSZ 0F2	X A, [X+]	X A, [B+]	IFEQ A, #i	IFEQ A,[B]	IFBIT 2,[B]	*	LD B, 0D	IFBNE 2	JSR 0200–02FF	JMP 0200–02FF	JP + 19	JP + 3	2
JP -12	JP -28	LD 0F3,#i	DRSZ 0F3	X A, [X−]	X A, [B−]	IFGT A, #i	IFGT A,[B]	IFBIT 3,[B]	*	LD B, 0C	IFBNE 3	JSR 0300–03FF	JMP 0300–03FF	JP + 20	JP + 4	3
JP -11	JP -27	LD 0F4,#i	DRSZ 0F4	*	LAID	ADD A, #i	ADD A,[B]	IFBIT 4,[B]	CLRA	LD B, 0B	IFBNE 4	JSR 0400–04FF	JMP 0400–04FF	JP + 21	JP + 5	4
JP -10	JP -26	LD 0F5,#i	DRSZ 0F5	*	JID	AND A, #i	AND A,[B]	IFBIT 5,[B]	SWAPA	LD B, 0A	IFBNE 5	JSR 0500–05FF	JMP 0500–05FF	JP + 22	JP + 6	5
JP -9	JP -25	LD 0F6,#i	DRSZ 0F6	X A, [X]	X A, [B]	XOR A, #i	XOR A,[B]	IFBIT 6,[B]	DCORA	LD B, 9	IFBNE 6	JSR 0600–06FF	JMP 0600–06FF	JP + 23	JP + 7	6
JP -8	JP -24	LD 0F7,#i	DRSZ 0F7	*	*	OR A, #i	OR A,[B]	IFBIT 7,[B]	*	LD B, 8	IFBNE 7	JSR 0700–07FF	JMP 0700–07FF	JP + 24	JP + 8	7
JP -7	JP -23	LD 0F8,#i	DRSZ 0F8	NOP	*	LD A, #i	IFC	SBIT 0,[B]	RBIT 0,[B]	LD B, 7	IFBNE 8	JSR 0800–08FF	JMP 0800–08FF	JP + 25	JP + 9	8
JP -6	JP -22	LD 0F9,#i	DRSZ 0F9	*	*	*	IFNC	SBIT 1,[B]	RBIT 1,[B]	LD B, 6	IFBNE 9	JSR 0900–09FF	JMP 0900–09FF	JP + 26	JP + 10	9
JP -5	JP -21	LD 0FA,#i	DRSZ 0FA	LD A, [X+]	LD A, [B+]	LD [B+],#i	INCA	SBIT 2,[B]	RBIT 2,[B]	LD B, 5	IFBNE 0A	JSR 0A00–0AFF	JMP 0A00–0AFF	JP + 27	JP + 11	A
JP -4	JP -20	LD 0FB,#i	DRSZ 0FB	LD A, [X−]	LD A, [B−]	LD [B−],#i	DECA	SBIT 3,[B]	RBIT 3,[B]	LD B, 4	IFBNE 0B	JSR 0B00–0BFF	JMP 0B00–0BFF	JP + 28	JP + 12	B
JP -3	JP -19	LD 0FC,#i	DRSZ 0FC	LD Md, #i	JMPL	X A,Md	*	SBIT 4,[B]	RBIT 4,[B]	LD B, 3	IFBNE 0C	JSR 0C00–0CFF	JMP 0C00–0CFF	JP + 29	JP + 13	C
JP -2	JP -18	LD 0FD,#i	DRSZ 0FD	DIR	JSRL	LD A, Md	RETSK	SBIT 5,[B]	RBIT 5,[B]	LD B, 2	IFBNE 0D	JSR 0D00–0DFF	JMP 0D00–0DFF	JP + 30	JP + 14	D
JP -1	JP -17	LD 0FE,#	DRSZ 0FE	LD A, [X]	LD A, [B]	LD [B], #i	RET	SBIT 6, [B]	RBIT 6, [B]	LD B, 1	IFBNE 0E	JSR 0E00–0EFF	JMP 0E00–0EFF	JP + 31	JP + 15	E
JP -0	JP -16	LD 0FF,#	DRSZ 0FF	*	*	*	RETI	SBIT 7,[B]	RBIT 7,[B]	LD B, 0	IFBNE 0F	JSR 0F00–0FFF	JMP 0F00–0FFF	JP + 32	JP + 16	F

Bits 3–0

where, i is the immediate data Md is a directly addressed memory location * is an unused opcode (see following table)

Instruction Execution Time

Most instructions are single byte (with immediate addressing mode instruction taking two bytes).

Most single instructions take one cycle time to execute.

Skipped instructions require x number of cycles to be skipped, where x equals the number of bytes in the skipped instruction opcode.

See the BYTES and CYCLES per INSTRUCTION table for details.

BYTES and CYCLES per INSTRUCTION

The following table shows the number of bytes and cycles for each instruction in the format of byte/cycle.

Arithmetic Instructions (Bytes/Cycles)

	[B]	Direct	Immed.
ADD	1/1	3/4	2/2
ADC	1/1	3/4	2/2
SUBC	1/1	3/4	2/2
AND	1/1	3/4	2/2
OR	1/1	3/4	2/2
XOR	1/1	3/4	2/2
IFEQ	1/1	3/4	2/2
IFGT	1/1	3/4	2/2
IFBNE	1/1		
DRSZ		1/3	
SBIT	1/1	3/4	
RBIT	1/1	3/4	
IFBIT	1/1	3/4	

Memory Transfer Instructions (Bytes/Cycles)

	Register Indirect		Direct	Immed.	Register Indirect Auto Incr & Decr		
	[B]	[X]			[B+, B−]	[X+, X−]	
X A,*	1/1	1/3	2/3		1/2	1/3	
LD A,*	1/1	1/3	2/3	2/2	1/2	1/3	
LD B,Imm				1/1			(If B < 16)
LD B,Imm				2/3			(If B > 15)
LD Mem,Imm			3/3		2/2		
LD Reg,Imm				2/3			

* => Memory location addressed by B or X or directly.

Instructions Using A & C

Instructions	Bytes/Cycles
CLRA	1/1
INCA	1/1
DECA	1/1
LAID	1/3
DCORA	1/1
RRCA	1/1
SWAPA	1/1
SC	1/1
RC	1/1
IFC	1/1
IFNC	1/1

Transfer of Control Instructions

Instructions	Bytes/Cycles
JMPL	3/4
JMP	2/3
JP	1/3
JSRL	3/5
JSR	2/5
JID	1/3
RET	1/5
RETSK	1/5
RETI	1/5
INTR	1/7
NOP	1/1

Development Support

SUMMARY

- iceMASTER™: IM-COP8/400—Full feature in-circuit emulation for all COP8 products. A full set of COP8 Basic and Feature Family device and package specific probes are available.
- COP8 Debug Module: Moderate cost in-circuit emulation and development programming unit.
- COP8 Evaluation and Programming Unit: EPU-COP880C—low cost In-circuit simulation and development programming unit.
- Assembler: COP8-DEV-IBMA. A DOS installable cross development Assembler, Linker, Librarian and Utility Software Development Tool Kit.
- C Compiler: COP8C. A DOS installable cross development Software Tool Kit.
- OTP/EPROM Programmer Support: Covering needs from engineering prototype, pilot production to full production environments.

iceMASTER (IM) IN-CIRCUIT EMULATION

The iceMASTER IM-COP8/400 is a full feature, PC based, in-circuit emulation tool developed and marketed by MetaLink Corporation to support the whole COP8 family of products. National is a resale vendor for these products.

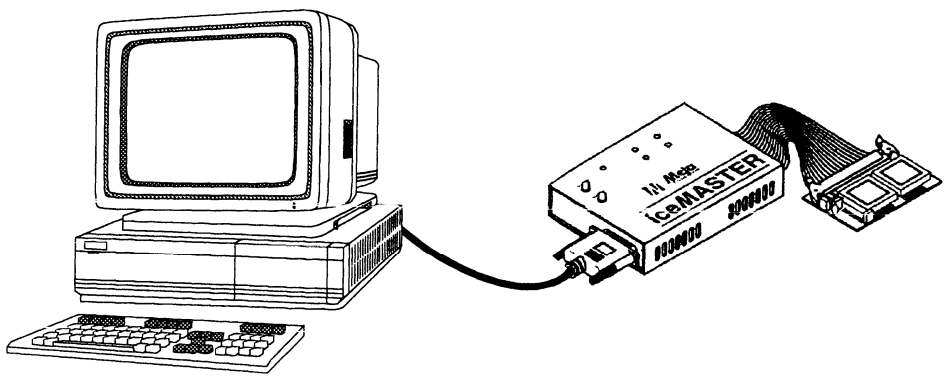
See *Figure 17* for configuration.

The iceMASTER IM-COP8/400 with its device specific COP8 Probe provides a rich feature set for developing, testing and maintaining product:

- Real-time in-circuit emulation; full 2.4V–5.5V DC operation range, full DC-10 MHz clock. Chip options are programmable or jumper selectable.
- Direct connection to application board by package compatible socket or surface mount assembly.
- Full 32 kbytes of loadable programming space that overlays (replaces) the on-chip ROM or EPROM. On-chip RAM and I/O blocks are used directly or recreated on the probe as necessary.
- Full 4k frame synchronous trace memory. Address, instruction, and 8 unspecified, circuit connectable trace lines. Display can be HLL source (e.g., C source), assembly or mixed.
- A full 64k hardware configurable break, trace on, trace off control, and pass count increment events.
- Tool set integrated interactive symbolic debugger—supports both assembler (COFF) and C Compiler (.COD) linked object formats.
- Real time performance profiling analysis; selectable bucket definition.
- Watch windows, content updated automatically at each execution break.
- Instruction by instruction memory/register changes displayed on source window when in single step operation.
- Single base unit and debugger software reconfigurable to support the entire COP8 family; only the probe personality needs to change. Debugger software is processor customized, and reconfigured from a master model file.
- Processor specific symbolic display of registers and bit level assignments, configured from master model file.
- Halt/Idle mode notification.
- On-line HELP customized to specific processor using master model file.
- Includes a copy of COP8-DEV-IBMA assembler and linker SDK.

IM Order Information

Base Unit	
IM-COP8/400-1	iceMASTER Base Unit, 110V Power Supply
IM-COP8/400-2	iceMASTER Base Unit, 220V Power Supply
iceMASTER Probe	
MHW-820CJ28DWPC	28 DIP
MHW-820CJ20DWPC	20 DIP
MHW-SOIC 28	28 SOIC Adaptor Kit
MHW-SOIC 20	20 SOIC Adaptor Kit



TL/DD/12528–21

FIGURE 17. COP8 iceMASTER Environment

Development Support (Continued)

iceMASTER DEBUG MODULE (DM)

The iceMASTER Debug Module is a PC based, combination in-circuit emulation tool and COP8 based OTP/EPROM programming tool developed and marketed by MetaLink Corporation to support the whole COP8 family of products. National is a resale vendor for these products.

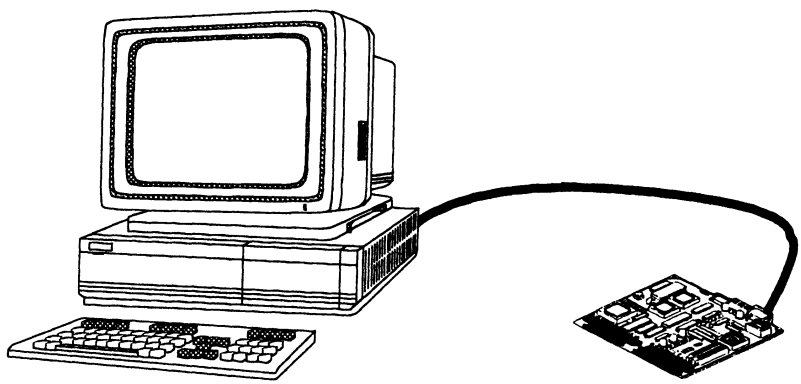
See *Figure 18* for configuration.

The iceMASTER Debug Module is a moderate cost development tool. It has the capability of in-circuit emulation for a specific COP8 microcontroller and in addition serves as a programming tool for COP8 OTP and EPROM product families. Summary of features is as follows:

- Real-time in-circuit emulation; full operating voltage range operation, full DC-10 MHz clock.
- All processor I/O pins can be cabled to an application development board with package compatible cable to socket and surface mount assembly.
- Full 32 kbytes of loadable programming space that overlays (replaces) the on-chip ROM or EPROM. On-chip RAM and I/O blocks are used directly or recreated as necessary.
- 100 frames of synchronous trace memory. The display can be HLL source (C source), assembly or mixed. The most recent history prior to a break is available in the trace memory.
- Configured break points; uses INTR instruction which is modestly intrusive.
- Software—only supported features are selectable.
- Tool set integrated interactive symbolic debugger—supports both assembler (COFF) and C Compiler (.COD) SDK linked object formats.
- Instruction by instruction memory/register changes displayed when in single step operation.
- Debugger software is processor customized, and reconfigured from a master model file.
- Processor specific symbolic display of registers and bit level assignments, configured from master model file.
- Halt/Idle mode notification.
- Programming menu supports full product line of programmable OTP and EPROM COP8 products. Program data is taken directly from the overlay RAM.
- Programming of 44 PLCC and 68 PLCC parts requires external programming adapters.
- Requires external power supply (5V lab supply connector provided).
- On-board V_{PP} generator from 5V input or connection to external supply supported. Requires V_{PP} level adjustment per the family programming specification (correct level is provided on an on-screen pop-down display).
- On-line HELP customized to specific processor using master model file.
- Includes a copy of COP8-DEV-IBMA assembler and linker SDK.

DM Order Information

Debug Model Unit	
COP8-DM/840CJ	
Cable Adapters	
DM-COP8/28D	28 DIP Cable
DM-COP8/28D-SO	28 DIP to 28 SOIC Adaptor
DM-COP8/20D	20 DIP Cable
DM-COP8/20D-SO	20 DIP to 20 SOIC Adaptor



TL/DD/12528–22

FIGURE 18. COP8-DM Environment

Development Support (Continued)

COP8 ASSEMBLER/LINKER SOFTWARE DEVELOPMENT TOOL KIT

National Semiconductor offers a relocatable COP8 macro cross assembler, linker, librarian and utility software development tool kit. Features are summarized as follows:

- Basic and Feature Family instruction set by "device" type.
- Nested macro capability.
- Extensive set of assembler directives.
- Supported on PC/DOS platform.
- Generates National standard COFF output files.
- Integrated Linker and Librarian.
- Integrated utilities to generate ROM code file outputs.
- DUMPCOFF utility.

This product is integrated as a part of MetaLink tools as a development kit, fully supported by the MetaLink debugger. It may be ordered separately or it is bundled with the MetaLink products at no additional cost.

Order Information

Assembler SDK	
COP8-DEV-IBMA	Assembler SDK on installable 3.5" PC/DOS Floppy Disk Drive format. Periodic upgrades and most recent version is available on National's BBS and Internet.

COP8 C COMPILER

A C Compiler is developed and marketed by Byte Craft Limited. The COP8C compiler is a fully integrated development tool specifically designed to support the compact embedded configuration of the COP8 family of products.

Features are summarized as follows:

- ANSI C with some restrictions and extensions that optimize development for the COP8 embedded application.
- BITS data type extension. Register declaration #pragma with direct bit level definitions.
- C language support for interrupt routines.
- Expert system, rule based code generation and optimization.
- Performs consistency checks against the architectural definitions of the target COP8 device.
- Generates program memory code.
- Supports linking of compiled object or COP8 assembled object formats.
- Global optimization of linked code.
- Symbolic debug load format fully source level supported by the MetaLink debugger.

OTP/EMULATOR SUPPORT

The COP8 family is supported by single chip OTP emulators.

Emulator Ordering Information

Device Number	Clock Option	Package	Emulates
COP87L20CJN-1N	Crystal, Brownout Dis	28 DIP	COP820CJ
COP87L20CJN-2N	External, Brownout Dis	28 DIP	COP820CJ
COP87L20CJN-3N	R/C, Brownout Dis	28 DIP	COP820CJ
COP87L20CJM-1N	Crystal, Brownout Dis	28 SO	COP820CJ
COP87L20CJM-2N	External, Brownout Dis	28 SO	COP820CJ
COP87L20CJM-3N	R/C, Brownout Dis	28 SO	COP820CJ
COP87L22CJN-1N	Crystal, Brownout Dis	20 DIP	COP822CJ
COP87L22CJN-2N	External, Brownout Dis	20 DIP	COP822CJ
COP87L22CJN-3N	R/C, Brownout Dis	20 DIP	COP822CJ
COP87L22CJM-1N	Crystal, Brownout Dis	20 SO	COP822CJ
COP87L22CJM-2N	External, Brownout Dis	20 SO	COP822CJ
COP87L22CJM-3N	R/C, Brownout Dis	20 SO	COP822CJ

Development Support (Continued)

Approved List

Manufacturer	North America	Europe	Asia
BP Microsystems	(800) 225-2102 (713) 688-4600 Fax: (713) 688-0920	+49-8152-4183 +49-8856-932616	+852-234-16611 +852-2710-8121
Data I/O	(800) 426-1045 (206) 881-6444 Fax: (206) 882-1043	+44-0734-440011	Call North America
HI-LO	(510) 623-8860	Call Asia	+886-2-764-0215 Fax: +886-2-756-6403
ICE Technology	(800) 624-8949 (919) 430-7915	+44-1226-767404 Fax: 0-1226-370-434	
MetaLink	(800) 638-2423 (602) 926-0797 Fax: (602) 693-0681	+49-80 9156 96-0 Fax: +49-80 9123 86	+852-737-1800
Systems General	(408) 263-6667	+41-1-9450300	+886-2-917-3005 Fax: +886-2-911-1283
Needhams	(916) 924-8037 Fax: (916) 924-8065		

Development Support (Continued)

INDUSTRY WIDE OTP/EPROM PROGRAMMING SUPPORT

Programming support, in addition to the MetaLink development tools, is provided by a full range of independent approved vendors to meet the needs from the engineering laboratory to full production.

AVAILABLE LITERATURE

For more information, please see the *COP8 Basic Family User's Manual,* Literature Number 620895, *COP8 Feature Family User's Manual,* Literature Number 620897 and *National's Family of 8-Bit Microcontrollers COP8 Selection Guide,* Literature Number 630009.

DIAL-A-HELPER SERVICE

Dial-A-Helper is a service provided by the Microcontroller Applications Group. The Dial-A-Helper is an Electronic Information System that may be accessed as a Bulletin Board System (BBS) via data modem, as an FTP site on the Internet via standard FTP client application or as an FTP site on the Internet using a standard Internet browser such as Netscape or Mosaic.

The Dial-A-Helper system provides access to an automated information storage and retrieval system. The system capabilities include a MESSAGE SECTION (electronic mail, when accessed as a BBS) for communications to and from the Microcontroller Applications Group and a FILE SECTION which consists of several file areas where valuable application software and utilities could be found.

DIAL-A-HELPER BBS VIA A STANDARD MODEM

Modem: CANADA/U.S.: (800) NSC-MICRO
(800) 672-6427
EUROPE: (+49) 0-8141-351332
Baud: 14.4k
Set-Up: Length: 8-Bit
Parity: None
Stop Bit: 1
Operation: 24 Hours, 7 Days

DIAL-A-HELPER VIA FTP

ftp nscmicro.nsc.com
user: anonymous
password: username@yourhost.site.domain

DIAL-A-HELPER VIA A WORLDWIDE WEB BROWSER

ftp://nscmicro.nsc.com

National Semiconductor on the WorldWide Web

See us on the WorldWide Web at: http://www.national.com

CUSTOMER RESPONSE CENTER

Complete product information and technical support is available from National's customer response centers.

CANADA/U.S.:	Tel:	(800) 272-9959
	email:	support@tevm2.nsc.com
EUROPE:	email:	europe.support.nsc.com
	Deutsch Tel:	+49 (0) 180-530 85 85
	English Tel:	+49 (0) 180-532 78 32
	Français Tel:	+49 (0) 180-532 93 58
	Italiano Tel:	+49 (0) 180-534 16 80
JAPAN:	Tel:	+81-043-299-2309
S.E. ASIA:	Beijing Tel:	(+86) 10-6856-8601
	Shanghai Tel:	(+86) 21-6415-4092
	Hong Kong Tel:	(+852) 2737-1600
	Korea Tel:	(+82) 2-3771-6909
	Malaysia Tel:	(+60-4) 644-9061
	Singapore Tel:	(+65) 255-2226
	Taiwan Tel:	+886-2-521-3288
AUSTRALIA:	Tel:	(+61) 3-9558-9999
INDIA:	Tel:	(+91) 80-559-9467

National Semiconductor

PRELIMINARY

COP87L40CJ/COP87L42CJ 8-Bit One-Time Programmable (OTP) Microcontrollers with Multi-Input Wake-Up and Brown Out Detector

General Description

The COP87L40CJ/COP87L42CJ are members of the COP8™ 8-bit OTP microcontroller family. It is pin and software compatible to the mask ROM COP840CJ/COP842CJ product family. They are fully static parts, fabricated using double-metal silicon gate microCMOS technology. Features include an 8-bit memory mapped architecture, MICROWIRE/PLUS™ serial I/O, a 16-bit timer/counter with capture register, a multi-sourced interrupt, Comparator, WATCHDOG™ Timer, Modulator/Timer and Multi-Input Wakeup. Each I/O pin has software selectable options to adapt the device to the specific application. The device operates over a voltage range of 2.7V to 5.5V. High throughput is achieved with an efficient, regular instruction set operating at a 1 μs per instruction rate.

The equivalent mask programmable device contains the Brown Out detection feature. This feature is not supported on these OTP devices.

Key Features

- Multi-Input Wakeup (on the 8-bit Port L)
- Analog comparator
- Modulator/Timer (high speed PWM timer for IR transmission)
- 16-bit multi-function timer supporting
  - — PWM mode
  - — External event counter mode
  - — Input capture mode
- Integrated capacitor for the R/C oscillator
- 1 kbyte on-board EPROM with security features
- 128 bytes on-chip RAM

I/O Features

- Memory mapped I/O
- Software selectable I/O options (TRI-STATE®, Push-Pull, Weak Pull-Up Input, High Impedance Input)
- High current outputs (8 pins)
- Schmitt trigger inputs on Port G
- MICROWIRE/PLUS serial I/O
- Packages:
  - — 20 DIP/SO with 16 I/O pins
  - — 28 DIP/SO with 24 I/O pins

CPU/Instruction Set Features

- 1 μs instruction cycle time
- Three multi-source interrupts servicing
  - — External interrupt with selectable edge
  - — Timer interrupt
  - — Software interrupt
- Versatile and easy to use instruction set
- 8-bit stack pointer (SP)—stack in RAM
- Two 8-bit Register Indirect Data Memory Pointers (B and X)

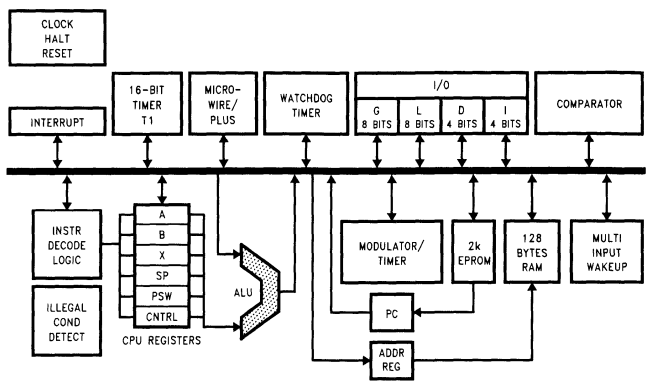
Fully Static CMOS

- Low current drain (typically <1 μA)
- Single supply operation: 2.7V to 5.5V
- Temperature range: $-40°C$ to $+85°C$

Development Support

- Emulation device for the COP840CJ/COP842CJ
- Real time emulation and full program debug offered by MetaLink Development Systems

Block Diagram



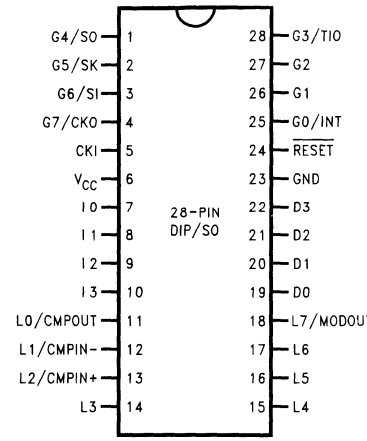
TL/DD/12529–1

FIGURE 1. Block Diagram

Pin Assignment

Port Pin	Typ	ALT Funct.	20 Pin	28 Pin
L0	I/O	MIWU/CMPOUT	7	11
L1	I/O	MIWU/CMPIN−	8	12
L2	I/O	MIWU/CMPIN+	9	13
L3	I/O	MIWU	10	14
L4	I/O	MIWU	11	15
L5	I/O	MIWU	12	16
L6	I/O	MIWU	13	17
L7	I/O	MIWU/MODOUT	14	18
G0	I/O	INTR	17	25
G1	I/O		18	26
G2	I/O		19	27
G3	I/O	TIO	20	28
G4	I/O	SO	1	1
G5	I/O	SK	2	2
G6	I	SI	3	3
G7	I	CKO	4	4
I0	I			7
I1	I			8
I2	I			9
I3	I			10
D0	O			19
D1	O			20
D2	O			21
D3	O			22
V_{CC}			6	6
GND			15	23
CKI			5	5
$\overline{\text{RESET}}$			16	24

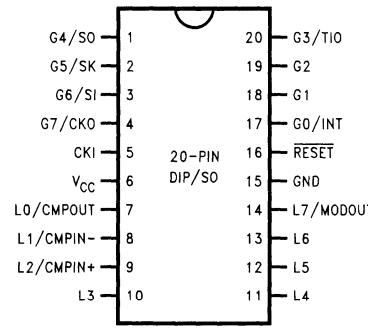
Connection Diagrams



TL/DD/12529-2

Top View

Order Number
COP87L40CJN (-1N, -2N, -3N)
COP87L40CJM (-1N, -2N, -3N)
See NS Package Number N28B or M28B



TL/DD/12529-3

Top View

Order Number
COP87L42CJN (-1N, -2N, -3N)
COP87L42CJM (-1N, -2N, -3N)
See NS Package Number N20A or M20B

Note: -1 Crystal Oscillator N - Brown out disabled
-2 External Oscillator
-3 R/C Oscillator

FIGURE 2. Connection Diagrams

Absolute Maximum Ratings

If Military/Aerospace specified devices are required, please contact the National Semiconductor Sales Office/Distributors for availability and specifications.

Supply Voltage (V_{CC})	7.0V
Voltage at any Pin	−0.3V to V_{CC} + 0.3V
Total Current into V_{CC} pin (Source)	80 mA
Total Current out of GND pin (sink)	80 mA
Storage Temperature Range	−65°C to +150°C

Note: *Absolute maximum ratings indicate limits beyond which damage to the device may occur.*

DC and AC electrical specifications are not ensured when operating the device at absolute maximum ratings.

DC Electrical Characteristics $-40°C \le T_A \le +85°C$ unless otherwise specified

Parameter	Conditions	Min	Typ	Max	Units
Operating Voltage		2.7		5.5	V
Power Supply Ripple 1 (Note 1)	Peak to Peak			0.1 V_{CC}	V
Supply Current (Note 2)					
CKI = 10 MHz	V_{CC} = 5.5V, tc = 1 μs			12	mA
CKI = 4 MHz	V_{CC} = 4.5V, tc = 2.5 μs			6.5	mA
HALT Current (Note 3)	V_{CC} = 5.5V, CKI = 0 MHz			12	μA
INPUT LEVELS (V_{IH}, V_{IL})					
Reset, CKI:					
Logic High		0.8 V_{CC}			V
Logic Low				0.2 V_{CC}	V
All Other Inputs					
Logic High		0.7 V_{CC}			V
Logic Low				0.2 V_{CC}	V
Hi-Z Input Leakage	V_{CC} = 5.5V	−2		+2	μA
Input Pullup Current	V_{CC} = 5.5V	−40		−250	μA
L- and G-Port Hysteresis (Note 6)				0.35 V_{CC}	V
Output Current Levels					
D Outputs:					
Source	V_{CC} = 4.5V, V_{OH} = 3.8V	−0.4			mA
Sink (Note 4)	V_{CC} = 4.5V, V_{OL} = 1.0V	10			mA
L4–L7 Output Sink	V_{CC} = 4.5V, V_{OL} = 2.5V	15			mA
All Others					
Source (Weak Pull-up Mode)	V_{CC} = 4.5V, V_{OH} = 3.2V	−10		−110	μA
Source (Push-pull Mode)	V_{CC} = 4.5V, V_{OH} = 3.8V	−0.4			mA
Sink (Push-pull Mode)	V_{CC} = 4.5V, V_{OL} = 0.4V	1.6			mA
TRI-STATE Leakage		−2.0		+2.0	μA
Allowable Sink/Source Current Per Pin					
D Outputs				15	mA
L4–L7 (Sink)				20	mA
All Others				3	mA
Maximum Input Current without Latchup (Note 5)	Room Temperature			±100	mA
RAM Retention Voltage, V_r	500 ns Rise and Fall Time (Min)	2.0			V
Input Capacitance				7	pF
Load Capacitance on D2				1000	pF

Note 1: Rate of voltage change must be less than 10 V/mS.

Note 2: Supply current is measured after running 2000 cycles with a square wave CKI input, CKO open, inputs at rails and outputs open.

Note 3: The HALT mode will stop CKI from oscillating in the RC and crystal configurations by bringing CKI high. HALT test conditions: L, and G0..G5 ports configured as outputs and set high. The D port set to zero. All inputs tied to V_{CC}. The comparator is disabled.

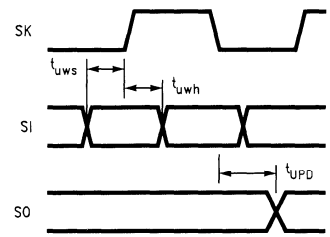
Note 4: The user must guarantee that D2 pin does not source more than 10 mA during RESET. If D2 sources more than 10 mA during reset, the device will go into programming mode.

Note 5: Pins G6 and $\overline{\text{RESET}}$ are designed with a high voltage input network. These pins allow input voltages greater than V_{CC} and the pins will have sink current to V_{CC} when biased at voltages greater than V_{CC} (the pins do not have source current when biased at a voltage below V_{CC}). The effective resistance to V_{CC} is 750Ω (typical). These two pins will not latch up. The voltage at the pins must be limited to less than 14V.

AC Electrical Characteristics $-40°C \leq T_A \leq +85°C$ unless otherwise specified

Parameter	Conditions	Min	Typ	Max	Units
Instruction Cycle Time (tc)					
Crystal/Resonator	$4.5V \leq V_{CC} \leq 5.5V$	1		DC	μs
R/C Oscillator	$4.5V \leq V_{CC} \leq 5.5V$	2		DC	μs
CKI Clock Duty Cycle (Note 6)	fr = Max	40		60	%
Rise Time (Note 6)	fr = 10 MHz ext. Clock			12	ns
Fall Time (Note 6)	fr = 10 MHz ext. Clock			8	ns
Inputs					
t_{Setup}	$4.5V \leq V_{CC} \leq 5.5V$	200			ns
t_{Hold}	$4.5V \leq V_{CC} \leq 5.5V$	60			ns
Output Propagation Delay	R_L = 2.2k, CL = 100 pF				
t_{PD1}, t_{PD0}					
SO, SK	$4.5V \leq V_{CC} \leq 5.5V$			0.7	μs
All Others	$4.5V \leq V_{CC} \leq 5.5V$			1	μs
Input Pulse Width					
Interrupt Input High Time		1			tc
Interrupt Input Low Time		1			tc
Timer Input High Time		1			tc
Timer Input Low Time		1			tc
MICROWIRE™ Setup Time ($t_{\mu WS}$)		20			ns
MICROWIRE Hold Time ($t_{\mu WH}$)		56			ns
MICROWIRE Output Propagation Delay ($t_{\mu PD}$)				220	ns
Reset Pulse Width		1			μs

Note 6: Parameter characterized but not production tested.



TL/DD/12529–4

FIGURE 3. MICROWIRE/PLUS Timing

Pin Description

V_{CC} and **GND** are the power supply pins.

CKI is the clock input. This can come from an external source, a R/C generated oscillator or a crystal (in conjunction with CKO). See Oscillator description.

$\overline{\text{RESET}}$ is the master reset input. See Reset description.

PORT I is a 4-bit Hi-Z input port.

PORT L is an 8-bit I/O port.

There are two registers associated with the L port: a data register and a configuration register. Therefore, each L I/O bit can be individually configured under software control as shown below:

Port L Config.	Port L Data	Port L Setup
0	0	Hi-Z Input (TRI-STATE)
0	1	Input with Weak Pull-up
1	0	Push-pull Zero Output
1	1	Push-pull One Output

Three data memory address locations are allocated for this port, one each for data register [00D0], configuration register [00D1] and the input pins [00D2].

Port L has the following alternate features:

L0 MIWU or CMPOUT
L1 MIWU or CMPIN−
L2 MIWU or CMPIN+
L3 MIWU
L4 MIWU (high sink current capability)
L5 MIWU (high sink current capability)
L6 MIWU (high sink current capability)
L7 MIWU or MODOUT (high sink current capability)

The selection of alternate Port L functions is done through registers WKEN [00C9] to enable MIWU and CNTRL2 [00CC] to enable comparator and modulator.

All eight L-pins have Schmitt Triggers on their inputs.

PORT G is an 8-bit port with 6 I/O pins (G0–G5) and 2 input pins (G6, G7).

All eight G-pins have Schmitt Triggers on the inputs.

There are two registers associated with the G port: a data register and a configuration register. Therefore each G port bit can be individually configured under software control as shown below:

Port G Config.	Port G Data	Port G Setup
0	0	Hi-Z Input (TRI-STATE)
0	1	Input with Weak Pull-up
1	0	Push-pull Zero Output
1	1	Push-pull One Output

Three data memory address locations are allocated for this port, one for data register [00D4], one for configuration register [00D5] and one for the input pins [00D6]. Since G6 and G7 are Hi-Z input only pins, any attempt by the user to configure them as outputs by writing a one to the configuration register will be disregarded. Reading the G6 and G7 configuration bits will return zeros. Note that the device will be placed in the Halt mode by writing a "1" to the G7 data bit.

Six pins of Port G have alternate features:

G0 INTR (an external interrupt)
G3 TIO (timer/counter input/output)
G4 SO (MICROWIRE serial data output)
G5 SK (MICROWIRE clock I/O)
G6 SI (MICROWIRE serial data input)
G7 CKO crystal oscillator output (selected by mask option) or HALT restart input/general purpose input (if clock option is R/C or external clock)

Pins G1 and G2 currently do not have any alternate functions.

The selection of alternate Port G functions are done through registers PSW [00EF] to enable external interrupt and CNTRL1 [00EE] to select TIO and MICROWIRE operations.

PORT D is a four bit output port that is preset when RESET goes low. One data memory address location is allocated for the data register [00DC]. The user can tie two or more D port outputs (except D2 pin) together in order to get a higher drive.

Note: Care must be exercised with the D2 pin operation. At RESET, the external loads on this pin must ensure that the output voltages stay above 0.8 V_{CC} to prevent the chip from entering special modes. Also keep the external loading on D2 to less than 1000 pF.

Functional Description

The internal architecture is shown in the block diagram. Data paths are illustrated in simplified form to depict how the various logic elements communicate with each other in implementing the instruction set of the device.

ALU and CPU Registers

The ALU can do an 8-bit addition, subtraction, logical or shift operations in one cycle time. There are five CPU registers:

A is the 8-bit Accumulator register

PC is the 15-bit Program Counter register

PU is the upper 7 bits of the program counter (PC)

PL is the lower 8 bits of the program counter (PC)

B is the 8-bit address register and can be auto incremented or decremented.

X is the 8-bit alternate address register and can be auto incremented or decremented.

SP is the 8-bit stack pointer which points to the subroutine stack (in RAM).

B, X and SP registers are mapped into the on chip RAM. The B and X registers are used to address the on chip RAM. The SP register is used to address the stack in RAM during subroutine calls and returns. The SP must be initialized by software before any subroutine call or interrupts occurs.

Memory

The memory is separated into two memory spaces: program and data.

PROGRAM MEMORY

Program memory consists of 2 kBytes of OTP EPROM. These bytes of ROM may be instructions or constant data. The memory is addressed by the 15-bit program counter (PC). ROM can be indirectly read by the LAID instruction for table lookup.

The device can be configured to inhibit external reads of the program memory. This is done by programming the Security Byte.

SECURITY FEATURE

The memory array has an associate Security Byte that is located outside of the program address range. This byte can be addressed only from programming mode by a programmer tool.

Security is an optional feature and can only be asserted after the memory arrary has been programmed and verified. A secured part will read all 00(hex) by a programmer. The part will fail Blank Check and will fail Verify operations. A Read operation will fill the programmer's memory with 00(hex). The Security Byte itself is always readable with value of 00(hex) if unsecure and FF(hex) if secure.

DATA MEMORY

The data memory address space includes on chip RAM, I/O and registers. Data memory is addressed directly by the instruction or indirectly through B, X and SP registers. The device has 128 bytes of RAM. Sixteen bytes of RAM are mapped as "registers", these can be loaded immediateiy, decremented and tested. Three specific registers: X, B, and SP are mapped into this space, the other registers are available for general usage.

Any bit of data memory can be directly set, reset or tested. All I/O and registers (except A and PC) are memory mapped; therefore, I/O bits and register bits can be directly and individually set, reset and tested, except the write once only bit (WDREN, WATCHDOG Reset Enable), and the unused and read only bits in CNTRL2 and WDREG registers.

Note: RAM contents are undefined upon power-up.

Reset

EXTERNAL RESET

The RESET input pin when pulled low initializes the microcontroller. The user must insure that the RESET pin is held low until V_{CC} is within the specified voltage range and the clock is stabilized. An R/C circuit with a delay 5x greater than the power supply rise time is recommended *(Figure 4)*. The device immediately goes into reset state when the RESET input goes low. When the RESET pin goes high the device comes out of reset state synchronously. The device will be running within two instruction cycles of the RESET pin going high. The following actions occur upon reset:

Port L	TRI-STATE
Port G	TRI-STATE
Port D	HIGH
PC	CLEARED
RAM Contents	RANDOM with Power-On-Reset UNAFFECTED with external Reset (power already applied)
B, X, SP	Same as RAM
PSW, CNTRL1, CNTRL2 and WDREG Reg.	CLEARED
Multi-Input Wakeup Reg. WKEDG, WKEN WKPND	 CLEARED UNKNOWN
Data and Configuration Registers for L & G	CLEARED
WATCHDOG Timer	Prescaler/Counter each loaded with FF

The device comes out of the HALT mode when the RESET pin is pulled low. In this case, the user has to ensure that the RESET signal is low long enough to allow the oscillator to restart. An internal 256 t_C delay is normally used in conjunction with the two pin crystal oscillator. When the device comes out of the HALT mode through Multi-Input Wakeup, this delay allows the oscillator to stabilize.

The following additional actions occur after the device comes out of the HALT mode through the RESET pin.

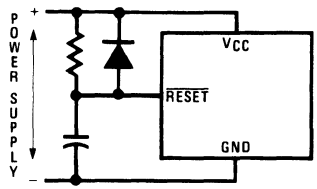
If a two pin crystal/resonator oscillator is being used:

RAM Contents	UNCHANGED
Timer T1 and A Contents	UNKNOWN
WATCHDOG Timer Prescaler/Counter	ALTERED

Functional Description (Continued)

If the external or RC Clock option is being used:

RAM Contents	UNCHANGED
Timer T1 and A Contents	UNCHANGED
WATCHDOG Timer Prescaler/Counter	ALTERED



RC > 5 × Power Supply Rise Time TL/DD/12529–5

FIGURE 4. Recommended Reset Circuit

WATCHDOG RESET

With WATCHDOG enabled, the WATCHDOG logic resets the device if the user program does not service the WATCHDOG timer within the selected service window. The WATCHDOG reset does not disable the WATCHDOG. Upon WATCHDOG reset, the WATCHDOG Prescaler/Counter are each initialized with FF Hex.

The following actions occur upon WATCHDOG reset that are different from external reset.

WDREN	WATCHDOG Reset Enable bit	UNCHANGED
WDUDF	WATCHDOG Underflow bit	UNCHANGED

Additional initialization actions that occur as a result of WATCHDOG reset are as follows:

Port L	TRI-STATE
Port G	TRI-STATE
Port D	HIGH
PC	CLEARED
Ram Contents	UNCHANGED
B, X, SP	UNCHANGED
PSW, CNTRL1 and CNTRL2 (except WDUDF Bit) Registers	CLEARED
Multi-Input Wakeup Registers WKEDG, WKEN WKPND	 CLEARED UNKNOWN
Data and Configuration Registers for L & G	CLEARED
WATCHDOG Timer	Prescalar/Counter each loaded with FF

Oscillator Circuits

EXTERNAL OSCILLATOR

By selecting the external oscillator option, the CKI pin can be driven by an external clock signal provided it meets the specified duty cycle, rise and fall times, and input levels. The G7/CKO is available as a general purpose input G7 and/or HALT control.

CRYSTAL OSCILLATOR

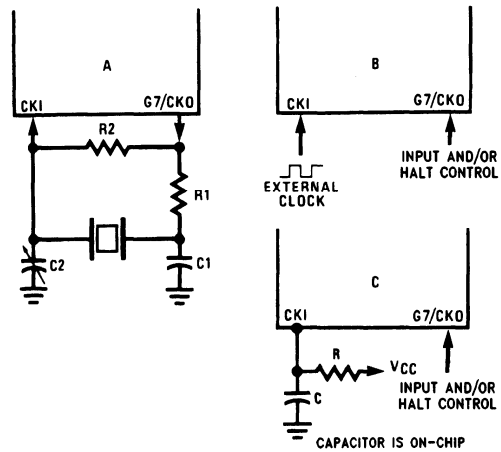
By selecting the crystal oscillator option, the G7/CKO pin is connected as a clock output, CKI and G7/CKO can be connected to make a crystal controlled oscillator. Table I shows the clock frequency for different component values. See *Figure 5* for the connections.

R/C OSCILLATOR

By selecting R/C oscillator option, connecting a resistor from the CKI pin to V_{CC} makes a R/C oscillator. The capacitor is on-chip. The G7/CKO pin is available as a general purpose input G7 and/or HALT control. Adding an external capacitor will jeopardize the clock frequency tolerance and increase EMI emissions.

Table II shows the clock frequency for the different resistor values. The capacitor is on-chip. See *Figure 5* for the connections.

Oscillator Circuits (Continued)



TL/DD/12529–6

FIGURE 5. Clock Oscillator Configurations

TABLE I. Crystal Oscillator Configuration

R1 (kΩ)	R2 (MΩ)	C1 (pF)	C2 (pF)	CKI Freq. (MHz)	Conditions
0	1	30	30–36	10	V_{CC} = 5V
0	1	30	30–36	4	V_{CC} = 5V
5.6	1	100	100–156	0.455	V_{CC} = 5V

TABLE II. RC Oscillator Configuration (Part-To-Part Variation) T_A = 25°C

R (kΩ)	CK1 Freq. (MHz)	Instr. Cycle (μs)	Conditions
8.2	3.3 ±10%	3.0 ±10%	V_{CC} = 5V
2.2	1.3 ±10%	7.7 ±10%	V_{CC} = 5V
3.9	0.75 ±10%	13.3 ±10%	V_{CC} = 5V

Halt Mode

The device is a fully static device. The device enters the HALT mode by writing a one to the G7 bit of the G data register. Once in the HALT mode, the internal circuitry does not receive any clock signal and is therefore frozen in the exact state it was in when halted. In this mode the chip will only draw leakage current.

The device supports three different methods of exiting the HALT mode. The first method is with a low to high transition on the CKO (G7) pin. This method precludes the use of the crystal clock configuration (since CKO is a dedicated output). It may be used either with an RC clock configuration or an external clock configuration. The second method of exiting the HALT mode is with the multi-Input Wakeup feature on the L port. The third method of exiting the HALT mode is by pulling the RESET input low.

If the two pin crystal/resonator oscillator is being used and Multi-Input Wakeup causes the device to exit the HALT mode, the WAKEUP signal does not allow the chip to start running immediately since crystal oscillators have a delayed start up time to reach full amplitude and freuqency stability. The WATCHDOG timer (consisting of an 8-bit prescaler followed by an 8-bit counter) is used to generate a fixed delay of 256tc to ensure that the oscillator has indeed stabilized before allowing instruction execution. In this case, upon detecting a valid WAKEUP signal only the oscillator circuitry is enabled. The WATCHDOG Counter and Prescaler are each loaded with a value of FF Hex. The WATCHDOG prescaler is clocked with the tc instruction cycle. (The tc clock is derived by dividing the oscillator clock down by a factor of 10).

The Schmitt trigger following the CKI inverter on the chip ensures that the WATCHDOG timer is clocked only when the oscillator has a sufficiently large amplitude to meet the Schmitt trigger specs. This Schmitt trigger is not part of the oscillator closed loop. The start-up timeout from the WATCHDOG timer enables the clock signals to be routed to the rest of the chip. The delay is not activated when the device comes out of HALT mode through RESET pin. Also, if the clock option is either RC or External clock, the delay is not used, but the WATCHDOG Prescaler/-Counter contents are changed. The Development System will not emulate the 256tc delay.

The $\overline{\text{RESET}}$ pin will cause the device to reset and start executing from address X'0000. A low to high transition on the G7 pin (if single pin oscillator is used) or Multi-Input Wakeup will cause the device to start executing from the address following the HALT instruction.

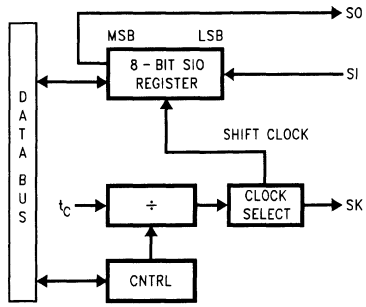
When $\overline{\text{RESET}}$ pin is used to exit the device from the HALT mode and the two pin crystal/resonator (CKI/CKO) clock option is selected, the contents of the Accumulator and the Timer T1 are undetermined following the reset. All other information except the WATCHDOG Prescaler/Counter contents is retained until continuing. All information except the WATCHDOG Prescaler/Counter contents is retained if the device exits the HALT mode through G7 pin or Multi-Input Wakeup.

G7 is the HALT-restart pin, but it can still be used as an input. If the device is not halted, G7 can be used as a general purpose input.

Note: To allow clock resynchronization, it is necessary to program two NOP's immediately after the device comes out of the HALT mode. The user must program two NOP's following the "enter HALT mode" (set G7 data bit) instruction.

MICROWIRE/PLUS

MICROWIRE/PLUS is a serial synchronous bidirectional communications interface. The MICROWIRE/PLUS capability enables the device to interface with any of National Semiconductor's MICROWIRE peripherals (i.e. A/D converters, display drivers, EEPROMS, etc.) and with other microcontrollers which support the MICROWIRE/PLUS interface. It consists of an 8-bit serial shift register (SIO) with serial data input (SI), serial data output (SO) and serial shift clock (SK). *Figure 6* shows the block diagram of the MICROWIRE/PLUS interface.



TL/DD/12529-7

FIGURE 6. MICROWIRE/PLUS Block Diagram

The shift clock can be selected from either an internal source or an external source. Operating the MICROWIRE/PLUS interface with the internal clock source is called the Master mode of operation. Operating the MICROWIRE/PLUS interface with an external shift clock is called the Slave mode of operation.

The CNTRL register is used to configure and control the MICROWIRE/PLUS mode. To use the MICROWIRE/PLUS, the MSEL bit in the CNTRL register is set to one. The SK clock rate is selected by the two bits, SL0 and SL1, in the CNTRL register. Table III details the different clock rates that may be selected.

TABLE III

SL1	SL0	SK Cycle Time
0	0	$2t_C$
0	1	$4t_C$
1	x	$8t_C$

where,

t_C is the instruction cycle time.

MICROWIRE/PLUS OPERATION

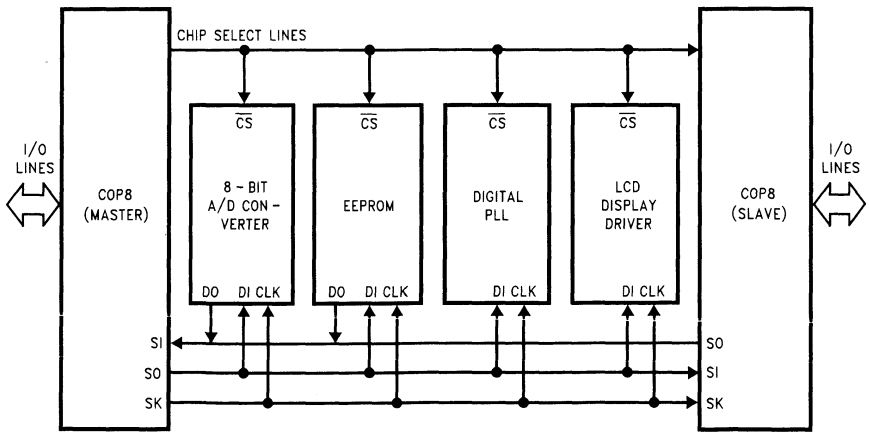
Setting the BUSY bit in the PSW register causes the MICROWIRE/PLUS arrangement to start shifting the data. It gets reset when eight data bits have been shifted. The user may reset the BUSY bit by software to allow less than 8 bits to shift. The device may enter the MICROWIRE/PLUS mode either as a Master or as a Slave. *Figure 7* shows how two device microcontrollers and several peripherals may be interconnected using the MICROWIRE/PLUS arrangement.

Master MICROWIRE/PLUS Operation

In the MICROWIRE/PLUS Master mode of operation the shift clock (SK) is generated internally by the device. The MICROWIRE/PLUS Master always initiates all data exchanges *(Figure 7)*. The MSEL bit in the CNTRL register must be set to enable the SO and SK functions on the G Port. The SO and SK pins must also be selected as outputs by setting appropriate bits in the Port G configuration register. Table IV summarizes the bit settings required for Master mode of operation.

Slave MICROWIRE/PLUS Operation

In the MICROWIRE/PLUS Slave mode of operation the SK clock is generated by an external source. Setting the MSEL bit in the CNTRL register enables the SO and SK functions on the G Port. The SK pin must be selected as an input and the SO pin selected as an output pin by appropriately setting up the Port G configuration register. Table IV summarizes the settings required to enter the Slave mode of operation.



TL/DD/12529-8

FIGURE 7. MICROWIRE/PLUS Application

MICROWIRE/PLUS (Continued)

The user must set the BUSY flag immediately upon entering the Slave mode. This will ensure that all data bits sent by the Master will be shifted properly. After eight clock pulses the BUSY flag will be cleared and the sequence may be repeated (see *Figure 7*).

TABLE IV

G4 Config. Bit	G5 Config. Bit	G4 Fun.	G5 Fun.	G6 Fun.	Operation
1	1	SO	Int. SK	SI	MICROWIRE Master
0	1	TRI-STATE	Int. SK	SI	MICROWIRE Master
1	0	SO	Ext. SK	SI	MICROWIRE Slave
0	0	TRI-STATE	Ext. SK	SI	MICROWIRE Slave

Timer/Counter

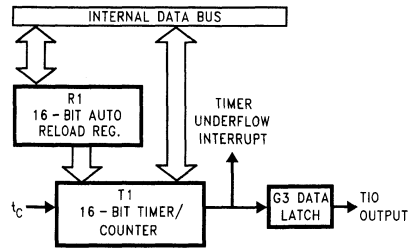
The device has a powerful 16-bit timer with an associated 16-bit register enabling it to perform extensive timer functions. The timer T1 and its register R1 are each organized as two 8-bit read/write registers. Control bits in the register CNTRL allow the timer to be started and stopped under software control. The timer-register pair can be operated in one of three possible modes. Table V details various timer operating modes and their requisite control settings.

MODE 1. TIMER WITH AUTO-LOAD REGISTER

In this mode of operation, the timer T1 counts down at the instruction cycle rate. Upon underflow the value in the register R1 gets automatically reloaded into the timer which continues to count down. The timer underflow can be programmed to interrupt the microcontroller. A bit in the control register CNTRL enables the TIO (G3) pin to toggle upon timer underflows. This allows the generation of square-wave outputs or pulse width modulated outputs under software control *(Figure 8)*.

MODE 2. EXTERNAL COUNTER

In this mode, the timer T1 becomes a 16-bit external event counter. The counter counts down upon an edge on the TIO pin. Control bits in the register CNTRL program the counter to decrement either on a positive edge or on a negative edge. Upon underflow the contents of the register R1 are automatically copied into the counter. The underflow can also be programmed to generate an interrupt *(Figure 8)*.



TL/DD/12529–9

FIGURE 8. Timer/Counter Auto Reload Mode Block Diagram

MODE 3. TIMER WITH CAPTURE REGISTER

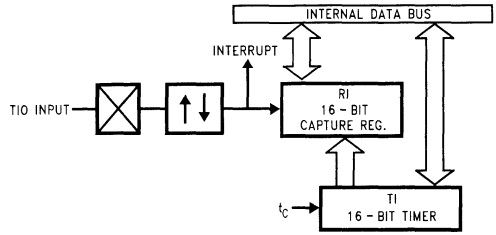
Timer T1 can be used to precisely measure external frequencies or events in this mode of operation. The timer T1 counts down at the instruction cycle rate. Upon the occurrence of a specified edge on the TIO pin the contents of the

TABLE V. Timer Operating Modes

CNTRL Bits 7 6 5	Operation Mode	T Interrupt	Timer Counter On
0 0 0	External Counter w/Auto-Load Reg.	Timer Underflow	TIO Pos. Edge
0 0 1	External Counter w/Auto-Load Reg.	Timer Underflow	TIO Neg. Edge
0 1 0	Not Allowed	Not Allowed	Not Allowed
0 1 1	Not Allowed	Not Allowed	Not Allowed
1 0 0	Timer w/Auto-Load Reg.	Timer Underflow	t_C
1 0 1	Timer w/Auto-Load Reg./Toggle TIO Out	Timer Underflow	t_C
1 1 0	Timer w/Capture Register	TIO Pos. Edge	t_C
1 1 1	Timer w/Capture Register	TIO Neg. Edge	t_C

Timer/Counter (Continued)

timer T1 are copied into the register R1. Bits in the control register CNTRL allow the trigger edge to be specified either as a positive edge or as a negative edge. In this mode the user can elect to be interrupted on the specified trigger edge *(Figure 9)*.

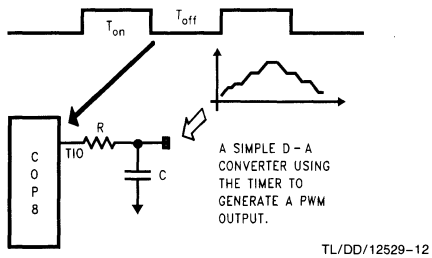


TL/DD/12529–11

FIGURE 9. Timer Capture Mode Block Diagram

TIMER PWM APPLICATION

Figure 10 shows how a minimal component D/A converter can be built out of the Timer-Register pair in the Auto-Reload mode. The timer is placed in the "Timer with auto reload" mode and the TIO pin is selected as the timer output. At the outset the TIO pin is set high, the timer T1 holds the on time and the register R1 holds the signal off time. Setting TRUN bit starts the timer which counts down at the instruction cycle rate. The underflow toggles the TIO output and copies the off time into the timer, which continues to run. By alternately loading in the on time and the off time at each successive interrupt a PWM frequency can be easily generated.



TL/DD/12529–12

FIGURE 10. Timer Application

WATCHDOG

The device has an on-board 8-bit WATCHDOG timer. The timer contains an 8-bit READ/WRITE down counter clocked by an 8-bit prescaler. Under software control the timer can be dedicated for the WATCHDOG or used as a general purpose counter. *Figure 11* shows the WATCHDOG timer block diagram.

MODE 1: WATCHDOG TIMER

The WATCHDOG is designed to detect user programs getting stuck in infinite loops resulting in loss of program control or "runaway" programs. The WATCHDOG can be enabled or disabled (only once) after the device is reset as a result of external reset. On power-up the WATCHDOG is disabled. The WATCHDOG is enabled by writing a "1" to WDREN bit (resides in WDREG register). Once enabled, the user program should write periodically into the 8-bit counter before the counter underflows. The 8-bit counter (WDCNT) is memory mapped at address 0CE Hex. The counter is loaded with n-1 to get n counts. The counter underflow resets the device, but does not disable the WATCHDOG. Loading the 8-bit counter initializes the prescaler with FF Hex and starts the prescaler/counter. Prescaler and counter are stopped upon counter underflow. Prescaler and counter are each loaded with FF Hex when the device goes into the HALT mode. The prescaler is used for crystal/resonator start-up when the device exits the HALT mode through Multi-Input Wakeup. In this case, the prescaler/counter contents are changed.

MODE 2: TIMER

In this mode, the prescaler/counter is used as a timer by keeping the WDREN (WATCHDOG reset enable) bit at 0. The counter underflow sets the WDUDF (underflow) bit and the underflow does not reset the device. Loading the 8-bit counter (load n-1 for n counts) sets the WDTEN bit (WATCHDOG Timer Enable) to "1", loads the prescaler with FF, and starts the timer. The counter underflow stops the timer. The WDTEN bit serves as a start bit for the WATCHDOG timer. This bit is set when the 8-bit counter is loaded by the user program. The load could be as a result of WATCHDOG service (WATCHDOG timer dedicated for WATCHDOG function) or write to the counter (WATCHDOG timer used as a general purpose counter). The bit is cleared upon Brown Out reset, WATCHDOG reset or external reset. The bit is not memory mapped and is transparent to the user program.

TABLE VI. WATCHDOG Control/Status

Parameter	HALT Mode	WD Reset	EXT Reset	Counter Load
8-Bit Prescaler	FF	FF	FF	FF
8-Bit WD Counter	FF	FF	FF	User Value
WDREN Bit	Unchanged	Unchanged	0	No Effect
WDUDF Bit	0	Unchanged	0	0
WDTEN Signal	Unchanged	0	0	1

WATCHDOG (Continued)

CONTROL/STATUS BITS

WDUDF: WATCHDOG Timer Underflow Bit

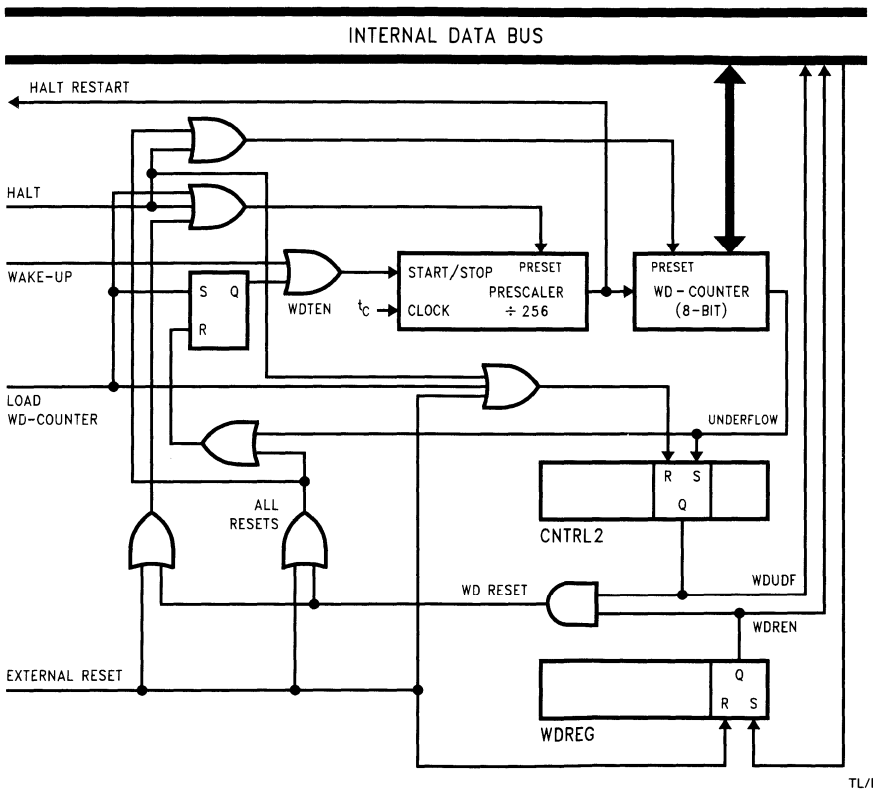
This bit resides in the CNTRL2 Register. The bit is set when the WATCHDOG timer underflows. The underflow resets the device if the WATCHDOG reset enable bit is set (WDREN = 1). Otherwise, WDUDF can be used as the timer underflow flag. The bit is cleared upon external reset, load to the 8-bit counter, or going into the HALT mode. It is a read only bit.

WDREN: WD Reset Enable

WDREN bit resides in a separate register (bit 0 of WDREG). This bit enables the WATCHDOG timer to generate a reset. The bit is cleared upon external reset. The bit under software control can be written to only once (once written to, the hardware does not allow the bit to be changed during program execution).

WDREN = 1 WATCHDOG reset is enabled.
WDREN = 0 WATCHDOG reset is disabled.

Table VI shows the impact of WATCHDOG Reset and External Reset on the Control/Status bits.



TL/DD/12529–13

FIGURE 11. WATCHDOG Timer Block Diagram

Modulator/Timer

The Modulator/Timer contains an 8-bit counter and an 8-bit autoreload register (MODRL address 0CF Hex). The Modulator/Timer has two modes of operation, selected by the control bit MC3. The Modulator/Timer Control bits MC1, MC2 and MC3 reside in CNTRL2 Register.

MODE 1: MODULATOR

The Modulator is used to generate high frequency pulses on the modulator output pin (L7). The L7 pin should be configured as an output. The number of pulses is determined by the 8-bit down counter. Under software control the modulator input clock can be either CKI or tC. The tC clock is derived by dividing down the oscillator clock by a factor of 10. Three control bits (MC1, MC2, and MC3) are used for the Modulator/Timer output control. When MC2 = 1 and MC3 = 1, CKI is used as the modulator input clock. When MC2 = 0, and MC3 = 1, tC is used as the modulator input clock. The user loads the counter with the desired number of counts (256 max) and sets MC1 to start the counter. The modulator autoreload register is loaded with n-1 to get n pulses. CKI or tc pulses are routed to the modulator output (L7) until the counter underflows *(Figure 12)*. Upon underflow the hardware resets MC1 and stops the counter. The L7 pin goes low and stays low until the counter is restarted by the user program. The user program has the responsibility to timeout the low time. Unless the number of counts is changed, the user program does not have to load the counter each time the counter is started. The counter can simply be started by setting the MC1 bit. Setting MC1 by software will load the counter with the value of the autoreload register. The software can reset MC1 to stop the counter.

MODE 2: PWM TIMER

The counter can also be used as a PWM Timer. In this mode, an 8-bit register is used to serve as an autoreload register (MODRL).

a. 50% Duty Cycle:

When MC1 is 1 and MC2, MC3 are 0, a 50% duty cycle free running signal is generated on the L7 output pin *(Figure 13)*. The L7 pin must be configured as an output pin. In this mode the 8-bit counter is clocked by tC. Setting the MC1 control bit by software loads the counter with the value of the autoreload register and starts the counter. The counter underflow toggles the (L7) output pin. The 50% duty cycle signal will be continuously generated until MC1 is reset by the user program.

b. Variable Duty Cycle:

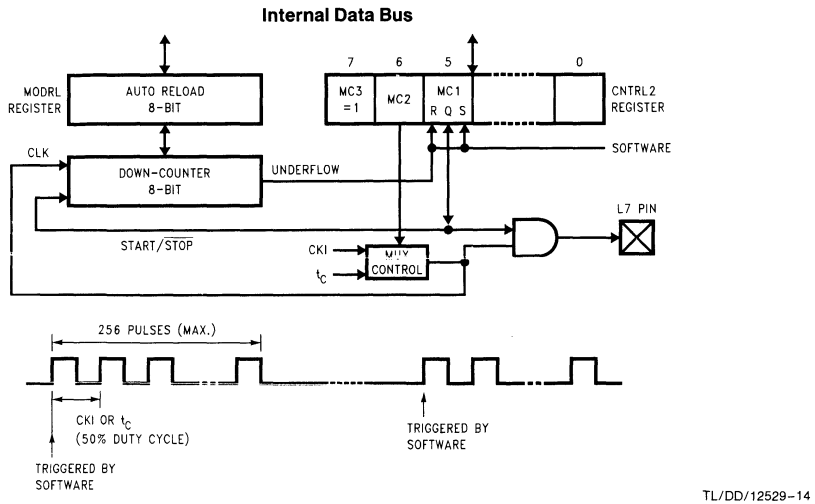
When MC3 = 0 and MC2 = 1, a variable duty cycle PWM signal is generated on the L7 output pin. The counter is clocked by tC. In this mode the 16-bit timer T1 along with the 8-bit down counter are used to generate a variable duty cycle PWM signal. The timer T1 underflow sets MC1 which starts the down counter and it also sets L7 high (L7 should be configured as an output).When the counter underflows the MC1 control bit is reset and the L7 output will go low until the next timer T1 underflow. Therefore, the width of the output pulse is controlled by the 8-bit counter and the pulse duration is controlled by the 16-bit timer T1 *(Figure 14)*. Timer T1 must be configured in "PWM Mode/Toggle TIO Out" (CNTRL1 Bits 7,6,5 = 101).

Table VII shows the different operation modes for the Modulator/Timer.

TABLE VII. Modulator/Timer Modes

Control Bits in CNTRL2(00CC)			Operation Mode L7 Function
MC3	MC2	MC1	
0	0	0	Normal I/O
0	0	1	50% Duty Cycle Mode (Clocked by tc)
0	1	X	Variable Duty Cycle Mode (Clocked by tc) Using Timer 1 Underflow
1	0	X	Modulator Mode (Clocked by tc)
1	1	X	Modulator Mode (Clocked by CKI)

Note: MC1, MC2 and MC3 control bits are cleared upon reset.



TL/DD/12529-14

FIGURE 12. Mode 1: Modulator Block Diagram/Output Waveform

Modulator/Timer (Continued)

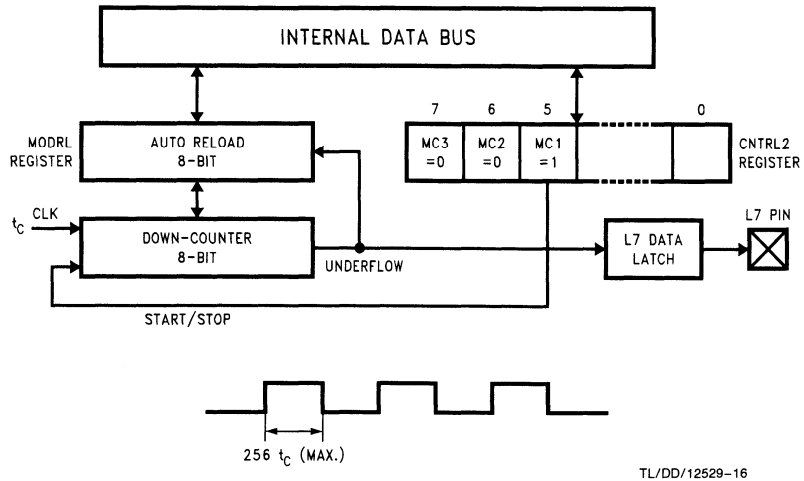


FIGURE 13. Mode 2a: 50% Duty Cycle Output

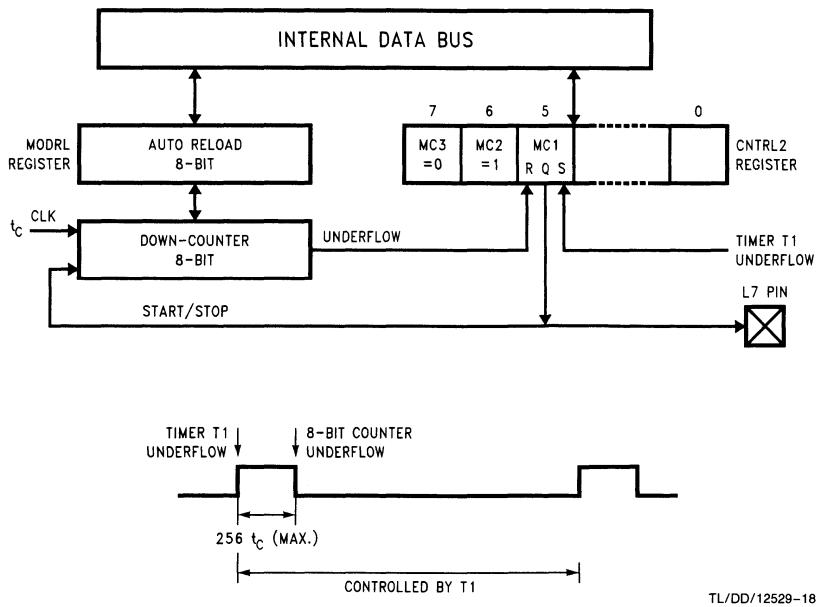


FIGURE 14. Mode 2b: Variable Duty Cycle Output

Comparator

The device has one differential comparator. Ports L0–L2 are used for the comparator. The output of the comparator is brought out to a pin. Port L has the following assignments:

L0 Comparator output
L1 Comparator negative input
L2 Comparator positive input

THE COMPARATOR STATUS/CONTROL BITS

These bits reside in the CNTRL2 Register (Address 0CC)

CMPEN Enables comparator ("1" = enable)

CMPRD Reads comparator output internally (CMPEN = 1, CMPOE = X)

CMPOE Enables comparator output to pin L0 ("1" = enable), CMPEN bit must be set to enable this function. If CMPEN = 0, L0 will be 0.

The Comparator Select/Control bits are cleared on RESET (the comparator is disabled). To save power the program should also disable the comparator before the device enters the HALT mode.

The user program must set up L0, L1 and L2 ports correctly for comparator Inputs/Output: L1 and L2 need to be configured as inputs and L0 as output. Table VIII shows the DC and AC characteristics for the comparator.

Multi-Input Wake Up

The Multi-Input Wakeup feature is used to return (wakeup) the device from the HALT mode. *Figure 15* shows the Multi-Input Wakeup logic.

This feature utilizes the L Port. The user selects which particular L port bit or combination of L Port bits will cause the device to exit the HALT mode. Three 8-bit memory mapped registers, Reg:WKEN, Reg:WKEDG, and Reg:WKPND are used in conjunction with the L port to implement the Multi-Input Wakeup feature.

All three registers Reg:WKEN, Reg:WKPND, and Reg:WKEDG are read/write registers, and are cleared at reset, except WKPND. WKPND is unknown on reset.

The user can select whether the trigger condition on the selected L Port pin is going to be either a positive edge (low to high transition) or a negative edge (high to low transition). This selection is made via the Reg:WKEDG, which is an 8-bit control register with a bit assigned to each L Port pin. Setting the control bit will select the trigger condition to be a negative edge on that particular L Port pin. Resetting the bit selects the trigger condition to be a positive edge. Changing an edge select entails several steps in order to avoid a pseudo Wakeup condition as a result of the edge change. First, the associated WKEN bit should be reset, followed by the edge select change in WKEDG. Next, the associated WKPND bit should be cleared, followed by the associated WKEN bit being re-enabled.

An example may serve to clarify this procedure. Suppose we wish to change the edge select from positive (low going high) to negative (high going low) for L port bit 5, where bit 5 has previously been enabled for an input. The program would be as follows:

```
RBIT  5,WKEN
SBIT  5,WKEDG
RBIT  5,WKPND
SBIT  5,WKEN
```

If the L port bits have been used as outputs and then changed to inputs with Multi-Input Wakeup, a safety procedure should also be followed to avoid inherited pseudo wakeup conditions. After the selected L port bits have been changed from output to input but before the associated WKEN bits are enabled, the associated edge select bits in WKEDG should be set or reset for the desired edge selects, followed by the associated WKPND bits being cleared. This same procedure should be used following RESET, since the L port inputs are left floating as a result of RESET.

The occurrence of the selected trigger condition for Multi-Input Wakeup is latched into a pending register called Reg:WKPND. The respective bits of the WKPND register will be set on the occurrence of the selected trigger edge on the corresponding Port L pin. The user has the responsibility of clearing these pending flags. Since the Reg:WKPND is a pending register for the occurrence of selected wakeup conditions, the device will not enter the HALT mode if any Wakeup bit is both enabled and pending. Setting the G7 data bit under this condition will not allow the device to enter the HALT mode. Consequently, the user has the responsibility of clearing the pending flags before attempting to enter the HALT mode.

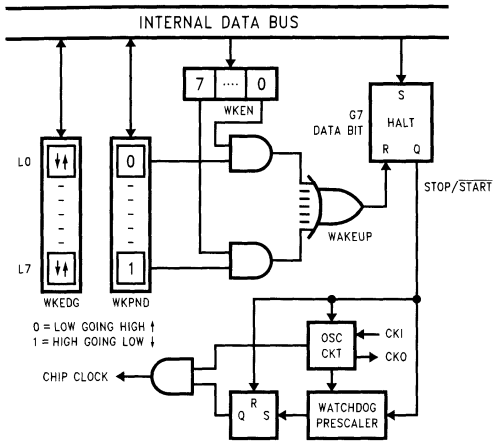
If a crystal oscillator is being used, the Wakeup signal will not start the chip running immediately since crystal oscillators have a finite start up time. The WATCHDOG timer prescaler generates a fixed delay to ensure that the oscillator has indeed stabilized before allowing the device to execute instructions. In this case, upon detecting a valid Wakeup signal only the oscillator circuitry and the WATCHDOG timer are enabled. The WATCHDOG timer prescaler is loaded with a value of FF Hex (256 counts) and is clocked from the tc instruction cycle clock. The tc clock is derived by dividing down the oscillator clock by a factor of 10. A Schmitt trigger following the CKI on chip inverter ensures that the WATCHDOG timer is clocked only when the oscillator has a sufficiently large amplitude to meet the Schmitt trigger specs. This Schmitt trigger is not part of the oscillator closed loop. The startup timeout from the WATCHDOG timer enables the clock signals to be routed to the rest of the chip.

TABLE VIII. DC and AC Characteristics (Note 1) $4.5V \le V_{CC} \le 5.5V$, $-40°C \le T_A \le +85°C$

Parameters	Conditioins	Min	Typ	Max	Units
Input Offset Voltage	$0.4V < V_{IN} < V_{CC} - 1.5V$		±10	±25	mV
Input Common Mode Voltage Range		0.4		$V_{CC} - 1.5$	V
Voltage Gain			300k		V/V
DC Supply Current (when enabled)	$V_{CC} = 5.5V$			250	μA
Response Time	TBD mV Step, TBD mV Overdrive, 100 pF Load			1	μs

Note 1: For comparator output current characteristics see L-Port specs.

Multi-Input Wakeup (Continued)



TL/DD/12529–19

FIGURE 15. Multi-Input Wakeup Logic

INTERRUPTS

The device has a sophisticated interrupt structure to allow easy interface to the real world. There are three possible interrupt sources, as shown below.

A maskable interrupt on external G0 input (positive or negative edge sensitive under software control)

A maskable interrupt on timer carry or timer capture

A non-maskable software/error interrupt on opcode zero

INTERRUPT CONTROL

The GIE (global interrupt enable) bit enables the interrupt function. This is used in conjunction with ENI and ENTI to select one or both of the interrupt sources. This bit is reset when interrupt is acknowledged.

ENI and ENTI bits select external and timer interrupts respectively. Thus the user can select either or both sources to interrupt the microcontroller when GIE is enabled.

IEDG selects the external interrupt edge (0 = rising edge, 1 = falling edge). The user can get an interrupt on both rising and falling edges by toggling the state of IEDG bit after each interrupt.

IPND and TPND bits signal which interrupt is pending. After an interrupt is acknowledged, the user can check these two bits to determine which interrupt is pending. This permits the interrupts to be prioritized under software. The pending flags have to be cleared by the user. Setting the GIE bit high inside the interrupt subroutine allows nested interrupts.

The software interrupt does not reset the GIE bit. This means that the controller can be interrupted by other interrupt sources while servicing the software interrupt.

INTERRUPT PROCESSING

The interrupt, once acknowledged, pushes the program counter (PC) onto the stack and the stack pointer (SP) is decremented twice. The Global Interrupt Enable (GIE) bit is reset to disable further interrupts. The microcontroller then vectors to the address 00FFH and resumes execution from that address. This process takes 7 cycles to complete. At the end of the interrupt subroutine, any of the following three instructions return the processor back to the main program: RET, RETSK or RETI. Either one of the three instructions will pop the stack into the program counter (PC). The stack pointer is then incremented twice. The RETI instruction additionally sets the GIE bit to re-enable further interrupts.

Any of the three instructions can be used to return from a hardware interrupt subroutine. The RETSK instruction should be used when returning from a software interrupt subroutine to avoid entering an infinite loop.

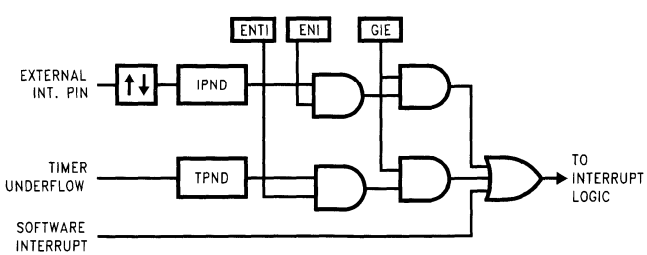
Note: There is always the possibility of an interrupt occurring during an instruction which is attempting to reset the GIE bit or any other interrupt enable bit. If this occurs when a single cycle instruction is being used to reset the interrupt enable bit, the interrupt enable bit will be reset but an interrupt may still occur. This is because interrupt processing is started at the same time as the interrupt bit is being reset. To avoid this scenario, the user should always use a two, three, or four cycle instruction to reset interrupt enable bits.

DETECTION OF ILLEGAL CONDITIONS

The device incorporates a hardware mechanism that allows it to detect illegal conditions which may occur from coding errors, noise, and "brown out" voltage drop situations. Specifically, it detects cases of executing out of undefined ROM area and unbalanced tack situations.

Reading an undefined ROM location returns 00 (hexadecimal) as its contents. The opcode for a software interrupt is also "00". Thus a program accessing undefined ROM will cause a software interrupt.

Reading an undefined RAM location returns an FF (hexadecimal). The subroutine stack on the device grows down for each subroutine call. By initializing the stack pointer to the top of RAM, the first unbalanced return instruction will cause the stack pointer to address undefined RAM. As a result the program will attempt to execute from FFFF (hexadecimal), which is an undefined ROM location and will trigger a software interrupt.



TL/DD/12529–20

FIGURE 16. Interrupt Block Diagram

Control Registers

CNTRL1 REGISTER (ADDRESS 00EE)

The Timer and MICROWIRE control register contains the following bits:

SL1 and SL0 Select the MICROWIRE clock divide-by (00 = 2, 01 = 4, 1x = 8)

IEDG External interrupt edge polarity select

MSEL Selects G5 and G4 as MICROWIRE signals SK and SO respectively

TRUN Used to start and stop the timer/counter (1 = run, 0 = stop)

TC1 Timer T1 Mode Control Bit

TC2 Timer T1 Mode Control Bit

TC3 Timer T1 Mode Control Bit

Bit 7							Bit 0
TC1	TC2	TC3	TRUN	MSEL	IEDG	SL1	SL0

PSW REGISTER (ADDRESS 00EF)

The PSW register contains the following select bits:

GIE Global interrupt enable (enables interrupts)

ENI External interrupt enable

BUSY MICROWIRE busy shifting flag

IPND External interrupt pending

ENTI Timer T1 interrupt enable

TPND Timer T1 interrupt pending (timer Underflow or capture edge)

C Carry Flip/Flop

HC Half-Carry Flip/Flop

Bit 7							Bit 0
HC	C	TPND	ENTI	IPND	BUSY	ENI	GIE

The Half-Carry bit is also effected by all the instructions that effect the Carry flag. The flag values depend upon the instruction. For example, after executing the ADC instruction the values of the Carry and the Half-Carry flag depend upon the operands involved. However, instructions like SET C and RESET C will set and clear both the carry flags. Table XIII lists the instructions that effect the HC and the C flags.

TABLE XIII. Instructions Effecting HC and C Flags

Instr.	HC Flag	C Flag
ADC	Depends on Operands	Depends on Operands
SUBC	Depends on Operands	Depends on Operands
SET C	Set	Set
RESET C	Set	Set
RRC	Depends on Operands	Depends on Operands

CNTRL2 REGISTER (ADDRESS 00CC)

Bit 7							Bit 0
MC3	MC2	MC1	CMPEN	CMPRD	CMPOE	WDUDF	unused
R/W	R/W	R/W	R/W	R/O	R/W	R/O	

MC3 Modulator/Timer Control Bit

MC2 Modulator/Timer Control Bit

MC1 Modulator/Timer Control Bit

CMPEN Comparator Enable Bit

CMPRD Comparator Read Bit

CMPOE Comparator Output Enable Bit

WDUDF WATCHDOG Timer Underflow Bit (Read Only)

WDREN REGISTER (ADDRESS 00CD)

WDREN WATCHDOG Reset Enable Bit (Write Once Only)

Bit 7							Bit 0
UNUSED							WDREN

Memory Map

All RAM, ports and registers (except A and PC) are mapped into data memory address space.

TABLE IX. Memory Map

Address	Contents
00 to 6F	On-chip RAM bytes (112 bytes)
70 to 7F	Unused RAM Address Space (Reads as All Ones)
80 to BF	Expansion Space for On-Chip EERAM (Reads Undefined Data)
C0 to C7	Reserved
C8	MIWU Edge Select Register (Reg:WKEDG)
C9	MIWU Enable Register (Reg:WKEN)
CA	MIWU Pending Register (Reg:WKPND)
CB	Reserved
CC	Control2 Register (CNTRL2)
CD	WATCHDOG Register (WDREG)
CE	WATCHDOG Counter (WDCNT)
CF	Modulator Reload (MODRL)
D0	Port L Data Register
D1	Port L Configuration Register
D2	Port L Input Pins (Read Only)
D3	Reserved for Port L
D4	Port G Data Register
D5	Port G Configuration Register
D6	Port G Input Pins (Read Only)
D7	Port I Input Pins (Read Only)
D8 to DB	Reserved for Port C
DC	Port D Data Register
DD to DF	Reserved for Port D
E0 to EF	On-Chip Functions and Registers
E0 to E7	Reserved for Future Parts
E8	Reserved
E9	MICROWIRE Shift Register
EA	Timer Lower Byte
EB	Timer Upper Byte
EC	Timer1 Autoreload Register Lower Byte
ED	Timer1 Autoreload Register Upper Byte
EE	CNTRL1 Control Register
EF	PSW Register
F0 to FF	On-Chip RAM Mapped as Registers
FC	X Register
FD	SP Register
FE	B Register

Reading other unused memory locations will return undefined data.

Addressing Modes

There are ten addressing modes, six for operand addressing and four for transfer of control.

OPERAND ADDRESSING MODES

REGISTER INDIRECT

This is the "normal" addressing mode for the chip. The operand is the data memory addressed by the **B** or **X** pointer.

REGISTER INDIRECT WITH AUTO POST INCREMENT OR DECREMENT

This addressing mode is used with the LD and X instructions. The operand is the data memory addressed by the **B** or **X** pointer. This is a register indirect mode that automatically post increments or post decrements the **B** or **X** pointer after executing the instruction.

DIRECT

The instruction contains an 8-bit address field that directly points to the data memory for the operand.

IMMEDIATE

The instruction contains an 8-bit immediate field as the operand.

SHORT IMMEDIATE

This addressing mode issued with the LD B,# instruction, where the immediate # is less than 16. The instruction contains a 4-bit immediate field as the operand.

INDIRECT

This addressing mode is used with the LAID instruction. The contents of the accumulator are used as a partial address (lower 8 bits of PC) for accessing a data operand from the program memory.

TRANSFER OF CONTROL ADDRESSING MODES

RELATIVE

This mode is used for the JP instruction with the instruction field being added to the program counter to produce the next instruction address. JP has a range from −31 to +32 to allow a one byte relative jump (JP + 1 is implemented by a NOP instruction). There are no "blocks" or "pages" when using JP since all 15 bits of the PC are used.

ABSOLUTE

This mode is used with the JMP and JSR instructions with the instruction field of 12 bits replacing the lower 12 bits of the program counter (PC). This allows jumping to any location in the current 4k program memory segment.

ABSOLUTE LONG

This mode is used with the JMPL and JSRL instructions with the instruction field of 15 bits replacing the entire 15 bits of the program counter (PC). This allows jumping to any location up to 32k in the program memory space.

INDIRECT

This mode is used with the JID instruction. The contents of the accumulator are used as a partial address (lower 8 bits of PC) for accessing a location in the program memory. The contents of this program memory location serves as a partial address (lower 8 bits of PC) for the jump to the next instruction.

Instruction Set

REGISTER AND SYMBOL DEFINITIONS

Registers

A	8-bit Accumulator register
B	8-bit Address register
X	8-bit Address register
SP	8-bit Stack pointer register
PC	15-bit Program counter register
PU	upper 7 bits of PC
PL	lower 8 bits of PC
C	1-bit of PSW register for carry
HC	Half Carry
GIE	1-bit of PSW register for global interrupt enable

Symbols

[B]	Memory indirectly addressed by B register
[X]	Memory indirectly addressed by X register
Mem	Direct address memory or [B]
Meml	Direct address memory or [B] or Immediate data
Imm	8-bit Immediate data
Reg	Register memory: addresses F0 to FF (Includes B, X and SP)
Bit	Bit number (0 to 7)
←	Loaded with
↔	Exchanged with

Instruction Set

ADD	add	A ← A + Meml
ADC	add with carry	A ← A + Meml + C, C ← Carry HC ← Half Carry
SUBC	subtract with carry	A ← A + $\overline{\text{Meml}}$ + C, C ← Carry HC ← Half Carry
AND	Logical AND	A ← A and Meml
OR	Logical OR	A ← A or Meml
XOR	Logical Exclusive-OR	A ← A xor Meml
IFEQ	IF equal	Compare A and Meml, Do next if A = Meml
IFGT	IF greater than	Compare A and Meml, Do next if A > Meml
IFBNE	IF B not equal	Do next if lower 4 bits of B ≠ Imm
DRSZ	Decrement Reg. ,skip if zero	Reg ← Reg − 1, skip if Reg goes to 0
SBIT	Set bit	1 to bit, Mem (bit= 0 to 7 immediate)
RBIT	Reset bit	0 to bit, Mem
IFBIT	If bit	If bit, Mem is true, do next instr.
X	Exchange A with memory	A ↔ Mem
LD A	Load A with memory	A ← Meml
LD mem	Load Direct memory Immed.	Mem ← Imm
LD Reg	Load Register memory Immed.	Reg ← Imm
X	Exchange A with memory [B]	A ↔ [B] (B ← B±1)
X	Exchange A with memory [X]	A ↔ [X] (X ← X±1)
LD A	Load A with memory [B]	A ← [B] (B ← B±1)
LD A	Load A with memory [X]	A ← [X] (X ← X±1)
LD M	Load Memory Immediate	[B] ← Imm (B ← B±1)
CLRA	Clear A	A ← 0
INCA	Increment A	A ← A + 1
DECA	Decrement A	A ← A − 1
LAID	Load A indirect from ROM	A ← ROM(PU,A)
DCORA	DECIMAL CORRECT A	A ← BCD correction (follows ADC, SUBC)
RRCA	ROTATE A RIGHT THRU C	C → A7 → . . . → A0 → C
SWAPA	Swap nibbles of A	A7 . . . A4 ↔ A3 . . . A0
SC	Set C	C ← 1, HC ← 1
RC	Reset C	C ← 0, HC ← 0
IFC	If C	If C is true, do next instruction
IFNC	If not C	If C is not true, do next instruction
JMPL	Jump absolute long	PC ← ii (ii = 15 bits, 0 to 32k)
JMP	Jump absolute	PC11..0 ← i (i = 12 bits)
JP	Jump relative short	PC ← PC + r (r is −31 to +32, not 1)
JSRL	Jump subroutine long	[SP] ← PL,[SP-1] ← PU,SP-2,PC ← ii
JSR	Jump subroutine	[SP] ← PL,[SP-1] ← PU,SP-2,PC11..0 ← i
JID	Jump indirect	PL ← ROM(PU,A)
RET	Return from subroutine	SP+2,PL ← [SP],PU ← [SP-1]
RETSK	Return and Skip	SP+2,PL ← [SP],PU ← [SP-1],Skip next instruction
RETI	Return from Interrupt	SP+2,PL ← [SP],PU ← [SP-1],GIE ← 1
INTR	Generate an interrupt	[SP] ← PL,[SP−1] ← PU,SP-2,PC ← 0FF
NOP	No operation	PC ← PC + 1

Bits 7–4

F	E	D	C	B	A	9	8	7	6	5	4	3	2	1	0	
JP -15	JP -31	LD 0F0,#i	DRSZ 0F0	RRCA	RC	ADC A, #i	ADC A, [B]	IFBIT 0,[B]	*	LD B, 0F	IFBNE 0	JSR 0000–00FF	JMP 0000–00FF	JP + 17	INTR	0
JP -14	JP -30	LD 0F1,#i	DRSZ 0F1	*	SC	SUBC A, #i	SUBC A,[B]	IFBIT 1,[B]	*	LD B, 0E	IFBNE 1	JSR 0100–01FF	JMP 0100–01FF	JP + 18	JP + 2	1
JP -13	JP -29	LD 0F2,#i	DRSZ 0F2	X A, [X+]	X A, [B+]	IFEQ A, #i	IFEQ A,[B]	IFBIT 2,[B]	*	LD B, 0D	IFBNE 2	JSR 0200–02FF	JMP 0200–02FF	JP + 19	JP + 3	2
JP -12	JP -28	LD 0F3,#i	DRSZ 0F3	X A, [X−]	X A, [B−]	IFGT A, #i	IFGT A,[B]	IFBIT 3,[B]	*	LD B, 0C	IFBNE 3	JSR 0300–03FF	JMP 0300–03FF	JP + 20	JP + 4	3
JP -11	JP -27	LD 0F4,#i	DRSZ 0F4	*	LAID	ADD A, #i	ADD A,[B]	IFBIT 4,[B]	CLRA	LD B, 0B	IFBNE 4	JSR 0400–04FF	JMP 0400–04FF	JP + 21	JP + 5	4
JP -10	JP -26	LD 0F5,#i	DRSZ 0F5	*	JID	AND A, #i	AND A,[B]	IFBIT 5,[B]	SWAPA	LD B, 0A	IFBNE 5	JSR 0500–05FF	JMP 0500–05FF	JP + 22	JP + 6	5
JP -9	JP -25	LD 0F6,#i	DRSZ 0F6	X A, [X]	X A, [B]	XOR A, #i	XOR A,[B]	IFBIT 6,[B]	DCORA	LD B, 9	IFBNE 6	JSR 0600–06FF	JMP 0600–06FF	JP + 23	JP + 7	6
JP -8	JP -24	LD 0F7,#i	DRSZ 0F7	*	*	OR A, #i	OR A,[B]	IFBIT 7,[B]	*	LD B, 8	IFBNE 7	JSR 0700–07FF	JMP 0700–07FF	JP + 24	JP + 8	7
JP -7	JP -23	LD 0F8,#i	DRSZ 0F8	NOP	*	LD A, #i	IFC	SBIT 0,[B]	RBIT 0,[B]	LD B, 7	IFBNE 8	JSR 0800–08FF	JMP 0800–08FF	JP + 25	JP + 9	8
JP -6	JP -22	LD 0F9,#i	DRSZ 0F9	*	*	*	IFNC	SBIT 1,[B]	RBIT 1,[B]	LD B, 6	IFBNE 9	JSR 0900–09FF	JMP 0900–09FF	JP + 26	JP + 10	9
JP -5	JP -21	LD 0FA,#i	DRSZ 0FA	LD A, [X+]	LD A, [B+]	LD [B+],#i	INCA	SBIT 2,[B]	RBIT 2,[B]	LD B, 5	IFBNE 0A	JSR 0A00–0AFF	JMP 0A00–0AFF	JP + 27	JP + 11	A
JP -4	JP -20	LD 0FB,#i	DRSZ 0FB	LD A, [X−]	LD A, [B−]	LD [B−],#i	DECA	SBIT 3,[B]	RBIT 3,[B]	LD B, 4	IFBNE 0B	JSR 0B00–0BFF	JMP 0B00–0BFF	JP + 28	JP + 12	B
JP -3	JP -19	LD 0FC,#i	DRSZ 0FC	LD Md, #i	JMPL	X A,Md	*	SBIT 4,[B]	RBIT 4,[B]	LD B, 3	IFBNE 0C	JSR 0C00–0CFF	JMP 0C00–0CFF	JP + 29	JP +13	C
JP -2	JP -18	LD 0FD,#i	DRSZ 0FD	DIR	JSRL	LD A, Md	RETSK	SBIT 5,[B]	RBIT 5,[B]	LD B, 2	IFBNE 0D	JSR 0D00–0DFF	JMP 0D00–0DFF	JP + 30	JP +14	D
JP -1	JP -17	LD 0FE,#i	DRSZ 0FE	LD A, [X]	LD A, [B]	LD [B], #i	RET	SBIT 6, [B]	RBIT 6, [B]	LD B, 1	IFBNE 0E	JSR 0E00–0EFF	JMP 0E00–0EFF	JP + 31	JP +15	E
JP -0	JP -16	LD 0FF,#1	DRSZ 0FF	*	*	*	RETI	SBIT 7,[B]	RBIT 7,[B]	LD B, 0	IFBNE 0F	JSR 0F00–0FFF	JMP 0F00–0FFF	JP + 32	JP + 16	F

Bits 3–0

where, i is the immediate data Md is a directly addressed memory location * is an unused opcode (see following table)

Instruction Execution Time

Most instructions are single byte (with immediate addressing mode instruction taking two bytes).

Most single instructions take one cycle time to execute.

Skipped instructions require x number of cycles to be skipped, where x equals the number of bytes in the skipped instruction opcode.

See the BYTES and CYCLES per INSTRUCTION table for details.

BYTES and CYCLES per INSTRUCTION

The following table shows the number of bytes and cycles for each instruction in the format of byte/cycle.

Arithmetic Instructions (Bytes/Cycles)

	[B]	Direct	Immed.
ADD	1/1	3/4	2/2
ADC	1/1	3/4	2/2
SUBC	1/1	3/4	2/2
AND	1/1	3/4	2/2
OR	1/1	3/4	2/2
XOR	1/1	3/4	2/2
IFEQ	1/1	3/4	2/2
IFGT	1/1	3/4	2/2
IFBNE	1/1		
DRSZ		1/3	
SBIT	1/1	3/4	
RBIT	1/1	3/4	
IFBIT	1/1	3/4	

Memory Transfer Instructions (Bytes/Cycles)

	Register Indirect		Direct	Immed.	Register Indirect Auto Incr & Decr		
	[B]	[X]			[B+, B−]	[X+, X−]	
X A,*	1/1	1/3	2/3		1/2	1/3	
LD A,*	1/1	1/3	2/3	2/2	1/2	1/3	
LD B,Imm				1/1			(If B < 16)
LD B,Imm				2/3			(If B > 15)
LD Mem,Imm			3/3		2/2		
LD Reg,Imm				2/3			

* => Memory location addressed by B or X or directly.

Instructions Using A & C

Instructions	Bytes/Cycles
CLRA	1/1
INCA	1/1
DECA	1/1
LAID	1/3
DCORA	1/1
RRCA	1/1
SWAPA	1/1
SC	1/1
RC	1/1
IFC	1/1
IFNC	1/1

Transfer of Control Instructions

Instructions	Bytes/Cycles
JMPL	3/4
JMP	2/3
JP	1/3
JSRL	3/5
JSR	2/5
JID	1/3
RET	1/5
RETSK	1/5
RETI	1/5
INTR	1/7
NOP	1/1

Development Support

SUMMARY

- iceMASTER™: IM-COP8/400—Full feature in-circuit emulation for all COP8 products. A full set of COP8 Basic and Feature Family device and package specific probes are available.
- COP8 Debug Module: Moderate cost in-circuit emulation and development programming unit.
- COP8 Evaluation and Programming Unit: EPU-COP880C—low cost In-circuit simulation and development programming unit.
- Assembler: COP8-DEV-IBMA. A DOS installable cross development Assembler, Linker, Librarian and Utility Software Development Tool Kit.
- C Compiler: COP8C. A DOS installable cross development Software Tool Kit.
- OTP/EPROM Programmer Support: Covering needs from engineering prototype, pilot production to full production environments.

iceMASTER (IM) IN-CIRCUIT EMULATION

The iceMASTER IM-COP8/400 is a full feature, PC based, in-circuit emulation tool developed and marketed by Meta-Link Corporation to support the whole COP8 family of products. National is a resale vendor for these products.

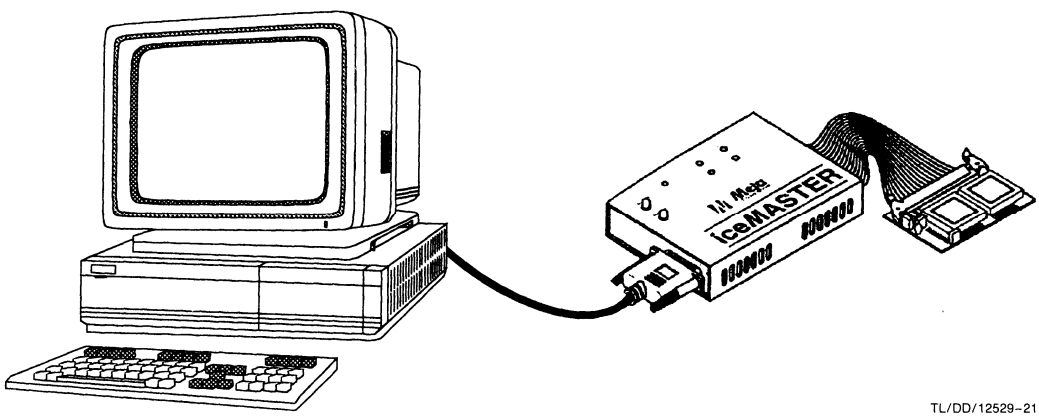
See *Figure 17* for configuration.

The iceMASTER IM-COP8/400 with its device specific COP8 Probe provides a rich feature set for developing, testing and maintaining product:

- Real-time in-circuit emulation; full 2.4V–5.5V operation range, full DC-10 MHz clock. Chip options are programmable or jumper selectable.
- Direct connection to application board by package compatible socket or surface mount assembly.
- Full 32 kbytes of loadable programming space that overlays (replaces) the on-chip ROM or EPROM. On-chip RAM and I/O blocks are used directly or recreated on the probe as necessary.
- Full 4k frame synchronous trace memory. Address, instruction, and 8 unspecified, circuit connectable trace lines. Display can be HLL source (e.g., C source), assembly or mixed.
- A full 64k hardware configurable break, trace on, trace off control, and pass count increment events.
- Tool set integrated interactive symbolic debugger—supports both assembler (COFF) and C Compiler (.COD) linked object formats.
- Real time performance profiling analysis; selectable bucket definition.
- Watch windows, content updated automatically at each execution break.
- Instruction by instruction memory/register changes displayed on source window when in single step operation.
- Single base unit and debugger software reconfigurable to support the entire COP8 family; only the probe personality needs to change. Debugger software is processor customized, and reconfigured from a master model file.
- Processor specific symbolic display of registers and bit level assignments, configured from master model file.
- Halt/Idle mode notification.
- On-line HELP customized to specific processor using master model file.
- Includes a copy of COP8-DEV-IBMA assembler and linker SDK.

IM Order Information

Base Unit	
IM-COP8/400-1	iceMASTER Base Unit, 110V Power Supply
IM-COP8/400-2	iceMASTER Base Unit, 220V Power Supply
iceMASTER Probe	
MHW-840CJ28DWPC MHW-840CJ20DWPC	28 DIP 20 DIP
MHW-SOIC28 MHW-SOIC20	28 SOIC Adaptor Kit 20 SOIC Adaptor Kit



TL/DD/12529–21

FIGURE 17. COP8 iceMASTER Environment

Development Support (Continued)

iceMASTER DEBUG MODULE (DM)

The iceMASTER Debug Module is a PC based, combination in-circuit emulation tool and COP8 based OTP/EPROM programming tool developed and marketed by MetaLink Corporation to support the whole COP8 family of products. National is a resale vendor for these products.

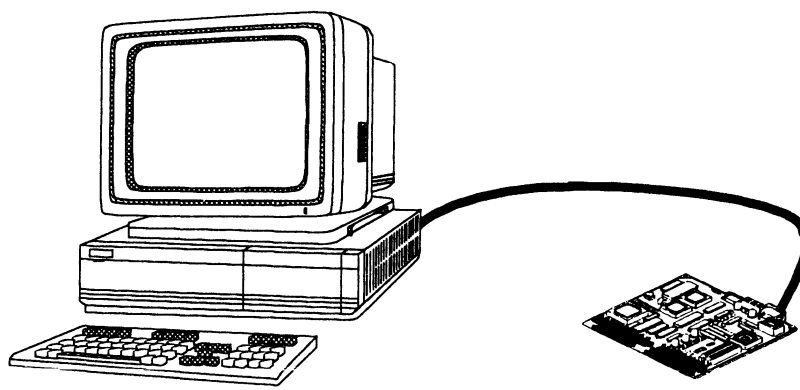
See *Figure 18* for configuration.

The iceMASTER Debug Module is a moderate cost development tool. It has the capability of in-circuit emulation for a specific COP8 microcontroller and in addition serves as a programming tool for COP8 OTP and EPROM product families. Summary of features is as follows:

- Real-time in-circuit emulation; full operating voltage range operation, full DC-10 MHz clock.
- All processor I/O pins can be cabled to an application development board with package compatible cable to socket and surface mount assembly.
- Full 32 kbytes of loadable programming space that overlays (replaces) the on-chip ROM or EPROM. On-chip RAM and I/O blocks are used directly or recreated as necessary.
- 100 frames of synchronous trace memory. The display can be HLL source (C source), assembly or mixed. The most recent history prior to a break is available in the trace memory.
- Configured break points; uses INTR instruction which is modestly intrusive.
- Software—only supported features are selectable.
- Tool set integrated interactive symbolic debugger—supports both assembler (COFF) and C Compiler (.COD) SDK linked object formats.
- Instruction by instruction memory/register changes displayed when in single step operation.
- Debugger software is processor customized, and reconfigured from a master model file.
- Processor specific symbolic display of registers and bit level assignments, configured from master model file.
- Halt/Idle mode notification.
- Programming menu supports full product line of programmable OTP and EPROM COP8 products. Program data is taken directly from the overlay RAM.
- Programming of 44 PLCC and 68 PLCC parts requires external programming adapters.
- Includes wall mount power supply.
- On-board V_{PP} generator from 5V input or connection to external supply supported. Requires V_{PP} level adjustment per the family programming specification (correct level is provided on an on-screen pop-down display).
- On-line HELP customized to specific processor using master model file.
- Includes a copy of COP8-DEV-IBMA assembler and linker SDK.

DM Order Information

Debug Model Unit	
COP8-DM/840CJ	
Cable Adapters	
DM-COP8/28D DM-COP8/28D-SO	28 DIP Cable 28 DIP to 28 SOIC Adaptor
DM-COP8/20D DM-COP8/20D-SO	20 DIP Cable 20 DIP to 20 SOIC Adaptor



TL/DD/12529–22

FIGURE 18. COP8-DM Environment

Development Support (Continued)

COP8 ASSEMBLER/LINKER SOFTWARE DEVELOPMENT TOOL KIT

National Semiconductor offers a relocatable COP8 macro cross assembler, linker, librarian and utility software development tool kit. Features are summarized as follows:

- Basic and Feature Family instruction set by "device" type.
- Nested macro capability.
- Extensive set of assembler directives.
- Supported on PC/DOS platform.
- Generates National standard COFF output files.
- Integrated Linker and Librarian.
- Integrated utilities to generate ROM code file outputs.
- DUMPCOFF utility.

This product is integrated as a part of MetaLink tools as a development kit, fully supported by the MetaLink debugger. It may be ordered separately or it is bundled with the MetaLink products at no additional cost.

Order Information

Assembler SDK	
COP8-DEV-IBMA	Assembler SDK on installable 3.5" PC/DOS Floppy Disk Drive format. Periodic upgrades and most recent version is available on National's BBS and Internet.

COP8 C COMPILER

A C Compiler is developed and marketed by Byte Craft Limited. The COP8C compiler is a fully integrated development tool specifically designed to support the compact embedded configuration of the COP8 family of products.

Features are summarized as follows:

- ANSI C with some restrictions and extensions that optimize development for the COP8 embedded application.
- BITS data type extension. Register declaration #pragma with direct bit level definitions.
- C language support for interrupt routines.
- Expert system, rule based code generation and optimization.
- Performs consistency checks against the architectural definitions of the target COP8 device.
- Generates program memory code.
- Supports linking of compiled object or COP8 assembled object formats.
- Global optimization of linked code.
- Symbolic debug load format fully sourced level supported by the MetaLink debugger.

OTP/EMULATOR SUPPORT

The COP8 family is supported by single chip OTP emulators.

Emulator Ordering Information

Device Number	Clock Option	Package	Emulates
COP87L40CJN-1N	Crystal, Brownout Dis	28 DIP	COP840CJ
COP87L40CJN-2N	External, Brownout Dis	28 DIP	COP840CJ
COP87L40CJN-3N	R/C, Brownout Dis	28 DIP	COP840CJ
COP87L40CJM-1N	Crystal, Brownout Dis	28 SO	COP840CJ
COP87L40CJM-2N	External, Brownout Dis	28 SO	COP840CJ
COP87L40CJM-3N	R/C, Brownout Dis	28 SO	COP840CJ
COP87L42CJN-1N	Crystal, Brownout Dis	20 DIP	COP842CJ
COP87L42CJN-2N	External, Brownout Dis	20 DIP	COP842CJ
COP87L42CJN-3N	R/C, Brownout Dis	20 DIP	COP842CJ
COP87L42CJM-1N	Crystal, Brownout Dis	20 SO	COP842CJ
COP87L42CJM-2N	External, Brownout Dis	20 SO	COP842CJ
COP87L42CJM-3N	R/C, Brownout Dis	20 SO	COP842CJ

Development Support (Continued)

Approved List

Manufacturer	North America	Europe	Asia
BP Microsystems	(800) 225-2102 (713) 688-4600 Fax: (713) 688-0920	+49-8152-4183 +49-8856-932616	+852-234-16611 +852-2710-8121
Data I/O	(800) 426-1045 (206) 881-6444 Fax: (206) 882-1043	+44-0734-440011	Call North America
HI-LO	(510) 623-8860	Call Asia	+886-2-764-0215 Fax: +886-2-756-6403
ICE Technology	(800) 624-8949 (919) 430-7915	+44-1226-767404 Fax: 0-1226-370-434	
MetaLink	(800) 638-2423 (602) 926-0797 Fax: (602) 693-0681	+49-80 9156 96-0 Fax: +49-80 9123 86	+852-737-1800
Systems General	(408) 263-6667	+41-1-9450300	+886-2-917-3005 Fax: +886-2-911-1283
Needhams	(916) 924-8037 Fax: (916) 924-8065		

Development Support (Continued)

INDUSTRY WIDE OTP/EPROM PROGRAMMING SUPPORT

Programming support, in addition to the MetaLink development tools, is provided by a full range of independent approved vendors to meet the needs from the engineering laboratory to full production.

AVAILABLE LITERATURE

For more information, please see the *COP8 Basic Family User's Manual,* Literature Number 620895, *COP8 Feature Family User's Manual,* Literature Number 620897 and *National's Family of 8-Bit Microcontrollers COP8 Selection Selection Guide,* Literature Number 630009.

DIAL-A-HELPER SERVICE

Dial-A-Helper is a service provided by the Microcontroller Applications Group. The Dial-A-Helper is an Electronic Information System that may be accessed as a Bulletin Board System (BBS) via data modem, as an FTP site on the Internet via standard FTP client application or as an FTP site on the Internet using a standard Internet browser such as Netscape or Mosaic.

The Dial-A-Helper system provides access to an automated information storage and retrieval system. The system capabilities include a MESSAGE SECTION (electronic mail, when accessed as a BBS) for communications to and from the Microcontroller Applications Group and a FILE SECTION which consists of several file areas where valuable application software and utilities could be found.

DIAL-A-HELPER BBS VIA A STANDARD MODEM

Modem: CANADA/U.S.: (800) NSC-MICRO
(800) 672-6427
EUROPE: (+49) 0-8141-351332
Baud: 14.4k
Set-Up: Length: 8-Bit
Parity: None
Stop Bit: 1
Operation: 24 Hours, 7 Days

DIAL-A-HELPER VIA FTP

ftp nscmicro.nsc.com
user: anonymous
password: username@yourhost.site.domain

DIAL-A-HELPER VIA A WORLDWIDE WEB BROWSER

ftp://nscmicro.nsc.com

National Semiconductor on the WorldWide Web

See us on the WorldWide Web at: http://www.national.com

CUSTOMER RESPONSE CENTER

Complete product information and technical support is available from National's customer response centers.

CANADA/U.S.:	Tel:	(800) 272-9959
	email:	support@tevm2.nsc.com
EUROPE:	email:	europe.support@nsc.com
	Deutsch Tel:	+49 (0) 180-530 85 85
	English Tel:	+49 (0) 180-532 78 32
	Français Tel:	+49 (0) 180-532 93 58
	Italiano Tel:	+49 (0) 180-534 16 80
JAPAN:	Tel:	+81-043-299-2309
S.E. ASIA:	Beijing Tel:	(+86) 10-6856-8601
	Shanghai Tel:	(+86) 21-6415-4092
	Hong Kong Tel:	(+852) 2737-1600
	Korea Tel:	(+82) 2-3771-6909
	Malaysia Tel:	(+60-4) 644-9061
	Singapore Tel:	(+65) 255-2226
	Taiwan Tel:	+886-2-521-3288
AUSTRALIA:	Tel:	(+61) 3-9558-9999
INDIA:	Tel:	(+91) 80-559-9467

National Semiconductor

PRELIMINARY

COP87L84BC 8-Bit One-Time Programmable (OTP) Microcontroller with CAN Interface

General Description

The COP87L84BC OTP microcontroller is a member of the COP8™ feature family using an 8-bit core architecture. It is pin and software compatible to the mask ROM COP884BC product family. (Continued)

Key Features

- CAN Interface
- On chip reset
- One 16-bit timer, with two 16-bit registers supporting:
  - — Processor Independent PWM mode
  - — External Event counter mode
  - — Input Capture mode
- High speed, constant resolution 8-bit PWM/frequency monitor timer with 2 output pins
- 2 kbytes on-board OTP EPROM with security feature
- 64 bytes on-board RAM

Additional Peripheral Features

- Idle timer
- Multi-Input Wake-Up (MIWU) with optional interrupts (7)
- Two analog comparators
- MICROWIRE/PLUS™ serial I/O

I/O Features

- Memory mapped I/O
- Software selectable I/O options (TRI-STATE® output, push-pull output, weak pull-up input, high impedance input)
- Schmitt trigger inputs on ports G and L
- Packages: 28 SO with 18 I/O pins

CPU/Instruction Set Features

- 1 μs instruction cycle time
- Eleven multi-source vectored interrupts servicing
  - — External Interrupt
  - — Idle Timer T0
  - — Timer T1 (with 2 Interrupts)
  - — MICROWIRE/PLUS
  - — Multi-Input Wake-Up
  - — Software Trap
  - — PWM Timer
  - — CAN Interface (with 3 interrupts)
- Versatile and easy to use instruction set
- 8-bit Stack Pointer (SP)—stack in RAM
- Two 8-bit Register Indirect Data Memory Pointers (B, X)

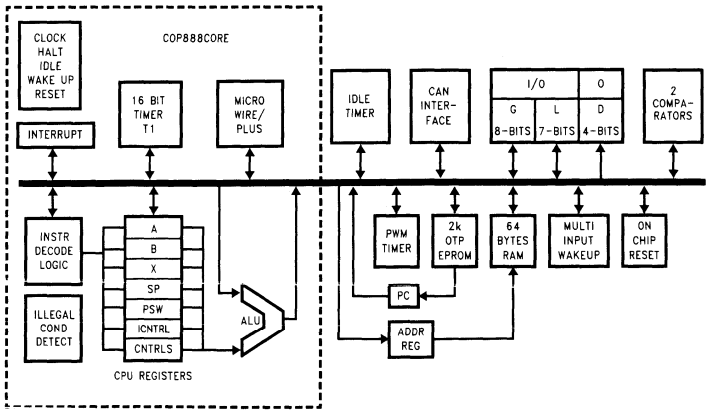
Fully Static CMOS

- Two power saving modes: HALT and IDLE
- Single supply operation: 4.5V to 5.5V
- Temperature ranges: −40°C to +85°C

Development Support

- Emulation device for the COP884BC
- Real time emulation and full program debug offered by MetaLink Development System

Block Diagram



TL/DD/12522–1

FIGURE 1

General Description (Continued)

It is a fully static part, fabricated using double-metal silicon gate microCMOS technology. Features include an 8-bit memory mapped architecture, MICROWIRE/PLUS serial I/O, a 16-bit timer/counter supporting three modes (Processor Independent PWM generation, External Event counter, and Input Capture mode capabilities), a CAN interface, two comparators, 8-bit, high speed, constant resolution PWM/ frequency monitor timer, and two power savings modes (HALT and IDLE), both with a multi-sourced wake up/ interrupt capability. This multi-sourced interrupt capability may also be used independent of the HALT or IDLE modes. Each I/O pin has software selectable configurations. The device operates over a voltage range of 4.5V to 5.5V. High throughput is achieved with an efficient, regular instruction set operating at a maximum of 1 μs per instruction rate.

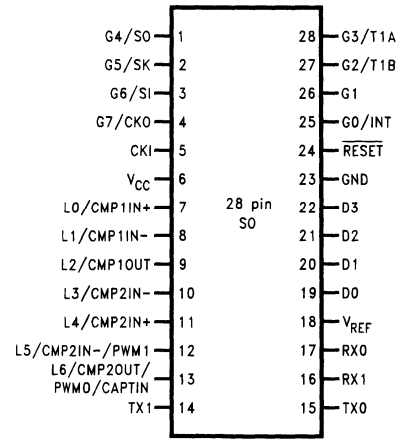
Connection Diagram

Pinouts for 28-Pin SO Package

Port Pin	Type	Alt. Function	28-Pin SO
G0	I/O	INTR	25
G1	I/O		26
G2	I/O	T1B	27
G3	I/O	T1A	28
G4	I/O	SO	1
G5	I/O	SK	2
G6	I	SI	3
G7	I	CKO	4
L0	I/O	CMP1IN+/MIWU	7
L1	I/O	CMP1IN−/MIWU	8
L2	I/O	CMP1OUT/MIWU	9
L3	I/O	CMP2IN−/MIWU	10
L4	I/O	CMP2IN+/MIWU	11
L5	I/O	CMP2IN−/PWM1/MIWU	12
L6	I/O	CMP2OUT/PWM0/ CAPTIN/MIWU	13
D0	O		19
D1	O		20
D2	O		21
D3	O		22
CAN V_{REF}			18
CAN Tx0	O		15
CAN Tx1	O		14
CAN Rx0	I	MIWU (Note 1)	17
CAN Rx1	I	MIWU	16
V_{CC}			6
GND			23
CKI	I		5
RESET	I		24

Note 1: The MIWU function for the CAN interface is internal (see CAN interface block diagram).

Dual-In-Line Package



TL/DD/12522–2

Top View

28-Lead (0.300" Wide)
Molded Small Outline Package, JEDEC
Order Number COP87L84BCM-XEN
See NS Package Number M28B

X = Crystal
E = HALT Enable
N = Power On Reset Disable

FIGURE 2. Connection Diagram

Absolute Maximum Ratings (Note)

If Military/Aerospace specified devices are required, please contact the National Semiconductor Sales Office/Distributors for availability and specifications.

Supply Voltage (V_{CC})	6V
Voltage at Any Pin	−0.3V to V_{CC} +0.3V
Total Current into V_{CC} Pin (Source)	90 mA
Total Current out of GND Pin (Sink)	100 mA
Storage Temperature Range	−65°C to +150°C

Note: *Absolute maximum ratings indicate limits beyond which damage to the device may occur. DC and AC electrical specifications are not ensured when operating the device at absolute maximum ratings.*

DC Electrical Characteristics $-40°C \le T_A \le +85°C$

Parameter	Conditions	Min	Typ	Max	Units
Operating Voltage		4.5		5.5	V
Power Supply Ripple (Note 1)	Peak-to-Peak			0.1 V_{CC}	V
Supply Current CKI = 10 MHz (Note 2)	V_{CC} = 5.5V, t_c = 1 μs			19	mA
HALT Current (Notes 3, 4)	V_{CC} = 5.5V, CKI = 0 MHz				
	Power-On Reset Enabled			480	μA
	Power-On Reset Disabled			380	μA
IDLE Current (Note 4) CKI = 10 MHz	V_{CC} = 5.5V, t_c = 1 μs			5.5	mA
Input Levels (V_{IH}, V_{IL})					
Reset, CKI					
Logic High		0.8 V_{CC}			V
Logic Low				0.2 V_{CC}	V
All Other Inputs					
Logic High		0.7 V_{CC}			V
Logic Low				0.2 V_{CC}	V
Hi-Z Input Leakage	V_{CC} = 5.5V			±2	μA
Input Pull-up Current	V_{CC} = 5.5V, V_{IN} = 0V	−40		−250	μA
G and L Port Input Hysteresis	(Note 8)		0.05 V_{CC}		V
Output Current Levels D Outputs					
Source	V_{CC} = 4.5V, V_{OH} = 3.3V	−0.4			mA
Sink (Note 5)	V_{CC} = 4.5V, V_{OL} = 1.0V	10			mA
Comparator Output (L2, L6)					
Source (Push-Pull)	V_{CC} = 4.5V, V_{OH} = 3.3V	−1.6			mA
Sink (Push-Pull)	V_{CC} = 4.5V, V_{OL} = 0.4V	1.6			mA
CAN Transmitter Outputs					
Source (T × 1)	V_{CC} = 4.5V, V_{OH} = V_{CC} −0.1V	−1.5			mA
	V_{CC} = 4.5V, V_{OH} = V_{CC} −0.6V	−10			mA
Sink (T × 0)	V_{CC} = 4.5V, V_{OL} = 0.1V	1.5			mA
	V_{CC} = 4.5V, V_{OL} = 0.6V	1.0			mA
All Others					
Source (Weak Pull-Up)	V_{CC} = 4.5V, V_{OH} = 2.7V	−10		110	μA
Source (Push-Pull)	V_{CC} = 4.5V, V_{OH} = 3.3V	−0.4			mA
Sink (Push-Pull)	V_{CC} = 4.5V, V_{OL} = 0.4V	1.6			mA
TRI-STATE Leakage	V_{CC} = 5.5V			±2.0	μA
Allowable Sink/Source Current per Pin					
D Outputs (Sink)				15	mA
T × 0 (Sink) (Note 7)				30	mA
T × 1 (Source) (Note 7)				30	mA
All Other				3	mA

Note 1: Maximum rate of voltage change must be less than 0.5V/ms

Note 2: Supply current is measured after running 2000 cycles with a square wave CKI input, CKO open, inputs at V_{CC} or GND, and outputs open.

Note 3: The HALT mode will stop CKI from oscillating in the Crystal configurations. Halt test conditions: All inputs tied to V_{CC}; L, and G port I/Os configured as outputs and programmed low; D outputs programmed low. Parameter refers to HALT mode entered via setting bit 7 of the G Port data register. Part will pull up CKI during HALT in crystal clock mode.

Note 4: HALT and IDLE current specifications assume CAN block and comparators are disabled.

Note 5: The user must guarantee that D2 pin does not source more than 10 mA during RESET. If D2 sources more than 10 mA during reset, the device will go into programming mode.

DC Electrical Characteristics $-40°C \le T_A \le +85°C$ (Continued)

Parameter	Conditions	Min	Typ	Max	Units
Maximum Input Current without Latchup (Notes 6, 8)	Room Temp			±100	mA
RAM Retention Voltage, V_r (Note 7)	500 ns Rise and Fall Time	2.0			V
Input Capacitance	(Note 8)			7	pF
Load Capacitance on D2				1000	pF

AC Electrical Characteristics $-40°C \le T_A \le +85°C$

Parameter	Conditions	Min	Typ	Max	Units
Instruction Cycle Time (t_c) Crystal/Resonator	$V_{CC} \ge 4.5V$	1.0		DC	μs
Inputs					
t_{SETUP}	$V_{CC} \ge 4.5V$	200			ns
t_{HOLD}	$V_{CC} \ge 4.5V$	60			ns
PWM Capture Input					
t_{SETUP}	$V_{CC} \ge 4.5V$	30			ns
t_{HOLD}	$V_{CC} \ge 4.5V$	70			ns
Output Propagation Delay (t_{PD1}, t_{PD0}) (Note 6)	$C_L = 100$ pF, $R_L = 2.2$ kΩ				
SK, SO	$V_{CC} \ge 4.5V$			0.7	μs
PWM Outputs	$V_{CC} \ge 4.5V$			75	ns
All Others	$V_{CC} \ge 4.5V$			1	μs
MICROWIRE™					
Setup Time (t_{UWS}) (Note 7)		20			ns
Hold Time (t_{UWH}) (Note 7)		56			ns
Output Prop Delay (t_{UPD})				220	ns
Input Pulse Width (Note 8)					
Interrupt High Time		1			t_c
Interrupt Low Time		1			t_c
Timer 1, 2 High Time		1			t_c
Timer 1, 2 Low Time		1			t_c
Reset Pulse Width (Note 7)		1.0			μs
Power Supply Rise Time for Proper Operation of On-Chip RESET		50 μs		$256*t_c$	

Note: For device testing purposes of all AC parameters, V_{OH} will be tested at $0.5*V_{CC}$.

Note 6: The output propagation is referenced to the end of the instruction cycle where the output change occurs.

Note 7: Parameter not tested.

Note 8: t_c = Instruction Cycle Time.

On-Chip Voltage Reference $-40°C \le T_A \le +85°C$

Parameter	Conditions	Min	Max	Units
Reference Voltage V_{REF}	$I_{OUT} < 80$ μA, $V_{CC} = 5V$	$0.5\ V_{CC} - 0.12$	$0.5\ V_{CC} + 0.12$	V
Reference Supply Current I_{DD}	$I_{OUT} = 0A$ (No Load) $V_{CC} = 5V$ (Note 1)		120	μA

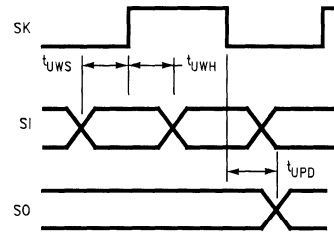
Note 1: Reference supply I_{DD} is supplied for information purposes only, it is not tested.

Comparator DC/AC Characteristics $4.5V \le V_{CC} \le 5.5V$, $-55°C \le T_A \le +125°C$

Parameter	Conditions	Min	Typ	Max	Units
Input Offset Voltage	$0.4V < V_{IN} < V_{CC} - 1.5V$		±10	±25	mV
Input Common Mode Voltage Range		0.4		$V_{CC} - 1.5$	V
Voltage Gain			300k		V/V
Outputs Sink/Source	See I/O-Port DC Specifications				
DC Supply Current (when enabled)	$V_{CC} = 6.0V$			250	μA
Response Time	TBD mV Step, TBD mV Overdrive, 100 pF Load		1		μs

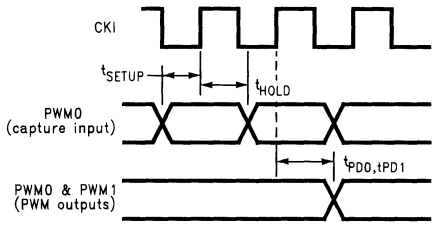
CAN Comparator DC and AC Characteristics: $4.8V \le V_{CC} \le 5.2V$, $-40°C \le T_A \le +125°C$

Parameter	Conditions	Min	Typ	Max	Units
Differential Input Voltage				±25	mV
Input Offset Voltage	$1.5V < V_{IN} < V_{CC} - 1.5V$			±10	mV
Input Common Mode Voltage Range		1.5		$V_{CC} - 1.5$	V
Input Hysteresis		8			mV



TL/DD/12522–3

FIGURE 3. MICROWIRE/PLUS Timing Diagram



TL/DD/12522–4

FIGURE 4. PWM/CAPTURE Timer Input/Output Timing Diagram

Pin Descriptions

V_{CC} and GND are the power supply pins.

CKI is the clock input. The clock can come from a crystal oscillator (in conjunction with CKO). See Oscillator Description section.

$\overline{\text{RESET}}$ is the master reset input. See Reset Description section.

The device contains one bidirectional 8-bit I/O port (G), and one 7-bit bidirectional I/O port (L) where each individual bit may be independently configured as an input (Schmitt trigger inputs on ports G and L), output or TRI-STATE under program control. Three data memory address locations are allocated for each of these I/O ports. Each I/O port has two associated 8-bit memory mapped registers, the CONFIGURATION register and the output DATA register. A memory mapped address is also reserved for the input pins of each I/O port. (See the memory map for the various addresses associated with the I/O ports.) *Figure 5* shows the I/O port configurations for the device. The DATA and CONFIGURATION registers allow for each port bit to be individually configured under software control as shown below:

Configuration Register	Data Register	Port Set-Up
0	0	Hi-Z Input (TRI-STATE Output)
0	1	Input with Weak Pull-Up
1	0	Push-Pull Zero Output
1	1	Push-Pull One Output

PORT L is a 7-bit I/O port. All L-pins have Schmitt triggers on the inputs.

Port L supports Multi-Input Wake Up (MIWU) on all seven pins.

Port L has the following alternate features:

L0 MIWU or CMP1IN+
L1 MIWU or CMP1IN−
L2 MIWU or CMP1OUT
L3 MIWU or CMP2IN−
L4 MIWU or CMP2IN+
L5 MIWU or CMP2IN− or PWM1
L6 MIWU or CMP2OUT or PWM0 or CAPTIN

Port G is an 8-bit port with 5 I/O pins (G0–G5), an input pin (G6), and one dedicated output pin (G7). Pins G0–G6 all have Schmitt Triggers on their inputs. G7 serves as the dedicated output pin for the CKO clock output. There are two registers associated with the G Port, a data register and a configuration register. Therefore, each of the 6 I/O bits (G0–G5) can be individually configured under software control.

Since G6 is an input only pin and G7 is the dedicated CKO clock output pin the associated bits in the data and configuration registers for G6 and G7 are used for special purpose functions as outlined below. Reading the G6 and G7 data bits will return zeroes.

Note that the chip will be placed in the HALT mode by writing a "1" to bit 7 of the Port G Data Register. Similarly the chip will be placed in the IDLE mode by writing a "1" to bit 6 of the Port G Data Register.

Writing a "1" to bit 6 of the Port G Configuration Register enables the MICROWIRE/PLUS to operate with the alternate phase of the SK clock.

	Config. Register	Data Register
G7		HALT
G6	Alternate SK	IDLE

CAN pins: For the on-chip CAN interface this device has five dedicated pins with the following features:

V_{REF} On-chip reference voltage with the value of $V_{CC}/2$

Rx0 CAN receive data input pin.

Rx1 CAN receive data input pin.

Tx0 CAN transmit data output pin. This pin may be put in the TRI-STATE mode with the TXEN0 bit in the CAN Bus control register.

Tx1 CAN transmit data output pin. This pin may be put in the TRI-STATE mode with the TXEN1 bit in the CAN Bus control register.

Port G has the following alternate features:

G0 INTR (External Interrupt Input)
G2 T1B (Timer T1 Capture Input)
G3 T1A (Timer T1 I/O)
G4 SO (MICROWIRE Serial Data Output)
G5 SK (MICROWIRE Serial Clock)
G6 SI (MICROWIRE Serial Data Input)

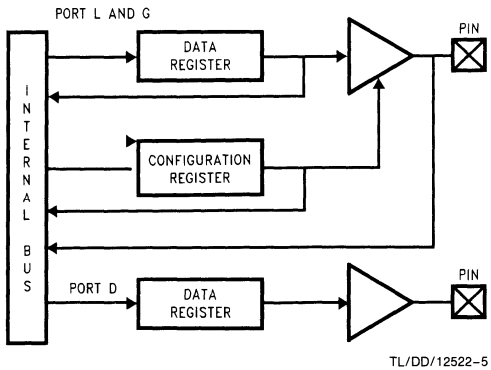
Port G has the following dedicated function:

G7 CKO Oscillator dedicated output

Port D is a 4-bit output port that is preset high when RESET goes low. The user can tie two or more D port outputs (except D2) together in order to get a higher drive.

Note: Care must be exercised with the D2 pin operation. At RESET, the external loads on this pin must ensure that the output voltages stay above 0.8 V_{CC} to prevent the chip from entering special modes. Also keep the external loading on D2 to less than 1000 pF.

Pin Descriptions (Continued)



TL/DD/12522–5

FIGURE 5. I/O Port Configurations

Functional Description

The architecture of the device utilizes a modified Harvard architecture. With the Harvard architecture, the control store program memory (ROM) is separated from the data store memory (RAM). Both ROM and RAM have their own separate addressing space with separate address buses. The architecture, though based on Harvard architecture, permits transfer of data from ROM to RAM.

CPU REGISTERS

The CPU can do an 8-bit addition, subtraction, logical or shift operation in one instruction (t_C) cycle time.

There are five CPU registers:

A is the 8-bit Accumulator Register

PC is the 15-bit Program Counter Register

PU is the upper 7 bits of the program counter (PC)

PL is the lower 8 bits of the program counter (PC)

B is an 8-bit RAM address pointer, which can be optionally post auto incremented or decremented.

X is an 8-bit alternate RAM address pointer, which can be optionally post auto incremented or decremented.

SP is the 8-bit stack pointer, which points to the subroutine/interrupt stack (in RAM). The SP is initialized to RAM address 02F with reset.

All the CPU registers are memory mapped with the exception of the Accumulator (A) and the Program Counter (PC).

PROGRAM MEMORY

Program memory for the device consists of 2 kbytes of OTP EPROM. These bytes may hold program instructions or constant data (data tables tor the LAID instruction, jump vectors for the JID instruction, and interrupt vectors for the VIS instruction). The program memory is addressed by the 15-bit program counter (PC). All interrupts in the device vector to program memory location 0FF Hex.

The device can be configured to inhibit external reads of the program memory. This is done by programming the Security Byte.

SECURITY FEATURE

The program memory array has an associate Security Byte that is located outside of the program address range. This byte can be addressed only from programming mode by a programmer tool.

Security is an optional feature and can only be asserted after the memory array has been programmed and verified. A secured part will read all 00(hex) by a programmer. The part will fail Blank Check and will fail Verify operations. A Read operation will fill the programmer's memory with 00(hex). The Security Byte itself is always readable with value of 00(hex) if unsecure and FF(hex) if secure.

DATA MEMORY

The data memory address space includes the on-chip RAM and data registers, the I/O registers (Configuration, Data and Pin), the control registers, the MICROWIRE/PLUS SIO shift register, and the various registers, and counters associated with the timers (with the exception of the IDLE timer). Data memory is addressed directly by the instruction or indirectly by the B, X and SP pointers.

The device has 64 bytes of RAM. Sixteen bytes of RAM are mapped as "registers" at addresses 0F0 to 0FF Hex. These registers can be loaded immediately, and also decremented and tested with the DRSZ (decrement register and skip if zero) instruction. The memory pointer registers X, SP, and B are memory mapped into this space at address locations 0FC to 0FE Hex respectively, with the other registers (other than reserved register 0FF) being available for general usage.

The instruction set permits any bit in memory to be set, reset or tested. All I/O and registers (except A and PC) are memory mapped; therefore, I/O bits and register bits can be directly and individually set, reset and tested. The accumulator (A) bits can also be directly and individually tested.

Note: RAM contents are undefined upon power-up.

RESET

The $\overline{\text{RESET}}$ input when pulled low initializes the microcontroller. Initialization will occur whenever the $\overline{\text{RESET}}$ input is pulled low. Upon initialization, the data and configuration registers for Ports L and G, are cleared, resulting in these Ports being initialized to the TRI-STATE mode. Port D is initialized high with $\overline{\text{RESET}}$. The PC, PSW, CNTRL, and ICNTRL control registers are cleared. The Multi-Input Wake Up registers WKEN, WKEDG, and WKPND are cleared. The Stack Pointer, SP, is initialized to 02F Hex.

The following initializations occur with $\overline{\text{RESET}}$:

Port L: TRI-STATE

Port G: TRI-STATE

Port D: HIGH

PC: CLEARED

PSW, CNTRL and ICNTRL registers: CLEARED

Accumulator and Timer 1:

RANDOM after RESET with power already applied

RANDOM after RESET at power-on

SP (Stack Pointer): Loaded with 2F Hex

CMPSL (Comparator control register): CLEARED

PWMCON (PWM control register): CLEARED

B and X Pointers:

UNAFFECTED after RESET with power already applied

RANDOM after RESET at power-up

RAM:

UNAFFECTED after RESET with power already applied

RANDOM after RESET at power-up

Functional Description (Continued)

CAN:

The CAN Interface comes out of external reset in the "error-active" state and waits until the user's software sets either one or both of the TXEN0, TXEN1 bits to "1". After that, the device will not start transmission or reception of a frame until eleven consecutive "recessive" (undriven) bits have been received. This is done to ensure that the output drivers are not enabled during an active message on the bus.

CSCAL, CTIM, TCNTL, TEC, REC: CLEARED

RTSTAT: CLEARED with the exception of the TBE bit which is set to 1

RID, RIDL, TID, TDLC: RANDOM

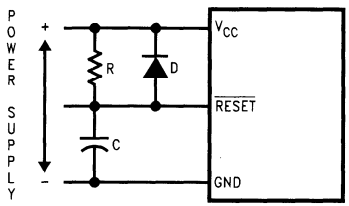
ON-CHIP POWER-ON RESET

The device is designed with an on-chip power-on reset circuit which will trigger a 256 t_c delay as V_{CC} rises above the minimum RAM retention voltage (V_r). This delay allows the oscillator to stabilize before the device exits the reset state. The contents of data registers and RAM are unknown following an on-chip power-on reset. The external reset takes priority over the on-chip reset and will deactivate the 256 t_c delay if in progress.

When using external reset, the external RC network shown in *Figure 6* should be used to ensure that the $\overline{\text{RESET}}$ pin is held low until the power supply to the chip stabilizes.

Under no circumstances should the $\overline{\text{RESET}}$ pin be allowed to float. If the on-chip power-on reset feature is being used, $\overline{\text{RESET}}$ should be connected directly to V_{CC}. Be aware of the Power Supply Rise Time requirements specified in the DC Specifications Table. These requirements must be met for the on-chip power-on reset to function properly.

The on-chip power-on reset circuit may reset the device if the operating voltage (V_{CC}) goes below V_r.



TL/DD/12522–6

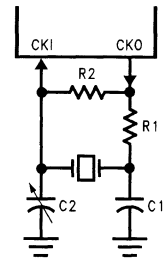
RC > 5 x Power Supply Rise Time

FIGURE 6. Recommended Reset Circuit

Oscillator Circuits

The chip can be driven by a clock input on the CKI input pin which can be between DC and 10 MHz. The CKO output clock is on pin G7. The CKI input frequency is divided by 10 to produce the instruction cycle clock ($1/t_c$).

Figure 7 shows the Crystal diagram.



TL/DD/12522–7

FIGURE 7. Crystal Oscillator Diagram

CRYSTAL OSCILLATOR

CKI and CKO can be connected to make a closed loop crystal (or resonator) controlled oscillator.

Table I shows the component values required for various standard crystal values.

TABLE I. Crystal Oscillator Configuration, T_A = 25°C

R1 (kΩ)	R2 (MΩ)	C1 (pF)	C2 (pF)	CKI Freq. (MHz)	Conditions
0	1	30	30–36	10	V_{CC} = 5V
0	1	30	30–36	4	V_{CC} = 5V
0	1	200	100–150	0.455	V_{CC} = 5V

Control Registers

CNTRL Register (Address X'00EE)

The Timer1 (T1) and MICROWIRE/PLUS control register contains the following bits:

SL1 & SL0 Select the MICROWIRE/PLUS clock divide by (00 = 2, 01 = 4, 1x = 8)

IEDG External interrupt edge polarity select (0 = Rising edge, 1 = Falling edge)

MSEL Selects G5 and G4 as MICROWIRE/PLUS signals SK and SO respectively

T1C0 Timer T1 Start/Stop control in timer

Timer T1 Underflow Interrupt Pending Flag in timer mode 3

T1C1 Timer T1 mode control bit

T1C2 Timer T1 mode control bit

T1C3 Timer T1 mode control bit

T1C3	T1C2	T1C1	T1C0	MSEL	IEDG	SL1	SL0

Bit 7 Bit 0

PSW Register (Address X'00EF)

The PSW register contains the following select bits:

GIE Global interrupt enable (enables interrupts)

EXEN Enable external interrupt

BUSY MICROWIRE/PLUS busy shifting flag

EXPND External interrupt pending

T1ENA Timer T1 Interrupt Enable for Timer Underflow or T1A Input capture edge

T1PNDA Timer T1 Interrupt Pending Flag (Autoreload RA in mode 1, T1 Underflow in Mode 2, T1A capture edge in mode 3)

C Carry Flag

HC Half Carry Flag

HC	C	T1PNDA	T1ENA	EXPND	BUSY	EXEN	GIE

Bit 7 Bit 0

The Half-Carry bit is also affected by all the instructions that affect the Carry flag. The SC (Set Carry) and RC (Reset Carry) instructions will respectively set or clear both the carry flags. In addition to the SC and RC instructions, ADC, SUBC, RRC and RLC instructions affect the carry and Half Carry flags.

ICNTRL Register (Address X'00E8)

The ICNTRL register contains the following bits:

T1ENB Timer T1 Interrupt Enable for T1B Input capture edge

T1PNDB Timer T1 Interrupt Pending Flag for T1B capture edge

μWEN Enable MICROWIRE/PLUS interrupt

μWPND MICROWIRE/PLUS interrupt pending

T0EN Timer T0 Interrupt Enable (Bit 12 toggle)

T0PND Timer T0 Interrupt pending

LPEN L Port Interrupt Enable (Multi-Input Wake Up/ Interrupt)

Bit 7 could be used as a flag

Unused	LPEN	T0PND	T0EN	μWPND	μWEN	T1PNDB	T1ENB

Bit 7 Bit 0

Timers

The device contains a very versatile set of timers (T0, T1, and an 8-bit PWM timer). All timers and associated autoreload/capture registers power up containing random data.

Figure 8 shows a block diagram for timers T1 and T0 on the device.

TIMER T0 (IDLE TIMER)

The device supports applications that require maintaining real time and low power with the IDLE mode. This IDLE mode support is furnished by the IDLE timer T0, which is a 16-bit timer. The Timer T0 runs continuously at the fixed rate of the instruction cycle clock, t_C. The user cannot read or write to the IDLE Timer T0, which is a count down timer.

The Timer T0 supports the following functions:

Exit out of the Idle Mode (See Idle Mode description)

Start up delay out of the HALT mode

The IDLE Timer T0 can generate an interrupt when the thirteenth bit toggles. This toggle is latched into the T0PND pending flag, and will occur every 4.096 ms at the maximum clock frequency (t_C = 1 μs). A control flag T0EN allows the interrupt from the thirteenth bit of Timer T0 to be enabled or disabled. Setting T0EN will enable the interrupt, while resetting it will disable the interrupt.

TIMER T1

The device has a powerful timer/counter block, T1.

The timer block consists of a 16-bit timer, T1, and two supporting 16-bit autoreload/capture registers, R1A and R1B. The timer block has two pins associated with it, T1A and T1B. The pin T1A supports I/O required by the timer block, while the pin T1B is an input to the timer block. The powerful and flexible timer block allows the device to easily perform all timer functions with minimal software overhead. The timer block has three operating modes: Processor Independent PWM mode, External Event Counter mode, and Input Capture mode.

The control bits T1C3, T1C2, and T1C1 allow selection of the different modes of operation.

Mode 1. Processor Independent PWM Mode

As the name suggests, this mode allows the device to generate a PWM signal with very minimal user intervention.

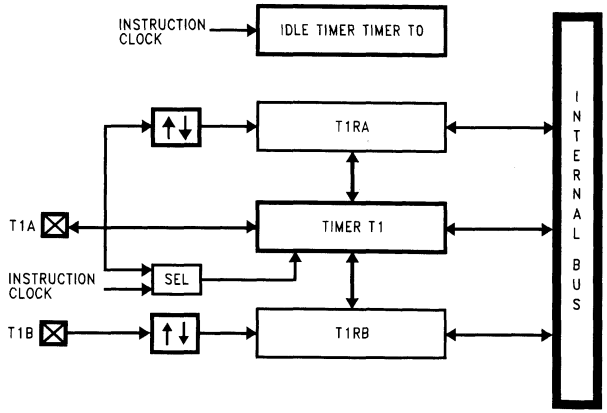
The user only has to define the parameters of the PWM signal (ON time and OFF time). Once begun, the timer block will continuously generate the PWM signal completely independent of the microcontroller. The user software services the timer block only when the PWM parameters require updating.

In this mode the timer T1 counts down at a fixed rate of t_C. Upon every underflow the timer is alternately reloaded with the contents of supporting registers, R1A and R1B. The very first underflow of the timer causes the timer to reload from the register R1A. Subsequent underflows cause the timer to be reloaded from the registers alternately beginning with the register R1B.

The T1 Timer control bits, T1C3, T1C2 and T1C1 set up the timer for PWM mode operation.

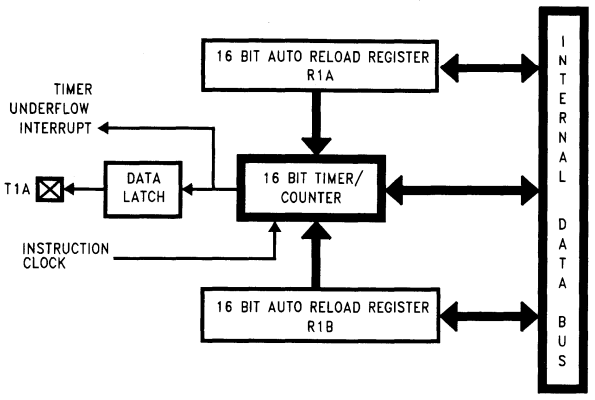
Figure 9 shows a block diagram of the timer in PWM mode. The underflows can be programmed to toggle the T1A output pin. The underflows can also be programmed to generate interrupts.

Timers (Continued)



TL/DD/12522–8

FIGURE 8. Timers T1 and T0



TL/DD/12522–9

FIGURE 9. Timer 1 in PWM MODE

Underflows from the timer are alternately latched into two pending flags, T1PNDA and T1PNDB. The user must reset these pending flags under software control. Two control enable flags, T1ENA and T1ENB, allow the interrupts from the timer underflow to be enabled or disabled. Setting the timer enable flag T1ENA will cause an interrupt when a timer underflow causes the R1A register to be reloaded into the timer. Setting the timer enable flag T1ENB will cause an interrupt when a timer underflow causes the R1B register to be reloaded into the timer. Resetting the timer enable flags will disable the associated interrupts.

Either or both of the timer underflow interrupts may be enabled. This gives the user the flexibility of interrupting once per PWM period on either the rising or falling edge of the PWM output. Alternatively, the user may choose to interrupt on both edges of the PWM output.

Mode 2. External Event Counter Mode

This mode is quite similar to the processor independent PWM mode described above. The main difference is that the timer, T1, is clocked by the input signal from the T1A pin. The T1 timer control bits, T1C3, T1C2 and T1C1 allow the timer to be clocked either on a positive or negative edge from the T1A pin. Underflows from the timer are latched into the T1PNDA pending flag. Setting the T1ENA control flag will cause an interrupt when the timer underflows.

In this mode the input pin T1B can be used as an independent positive edge sensitive interrupt input if the T1ENB control flag is set. The occurrence of a positive edge on the T1B input pin is latched into the T1PNDB flag.

Figure 10 shows a block diagram of the timer in External Event Counter mode.

Note: The PWM output is not available in this mode since the T1A pin is being used as the counter input clock.

Mode 3. Input Capture Mode

The device can precisely measure external frequencies or time external events by placing the timer block, T1, in the input capture mode.

In this mode, the timer T1 is constantly running at the fixed t_C rate. The two registers, R1A and R1B, act as capture registers. Each register acts in conjunction with a pin. The register R1A acts in conjunction with the T1A pin and the register R1B acts in conjunction with the T1B pin.

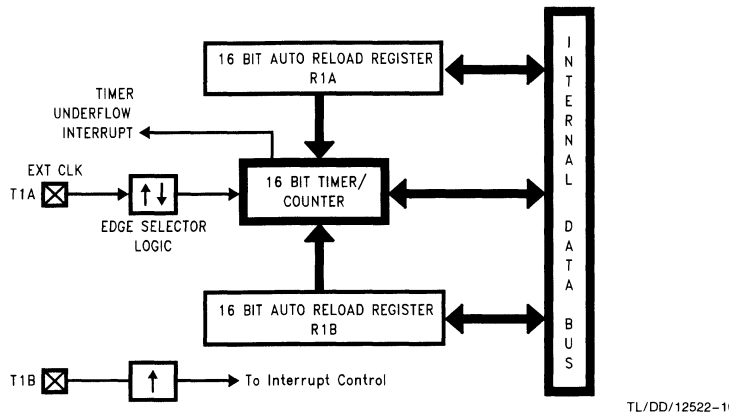
Timers (Continued)

The timer value gets copied over into the register when a trigger event occurs on its corresponding pin. Control bits, T1C3, T1C2 and T1C1, allow the trigger events to be specified either as a positive or a negative edge. The trigger condition for each input pin can be specified independently.

The trigger conditions can also be programmed to generate interrupts. The occurrence of the specified trigger condition on the T1A and T1B pins will be respectively latched into the pending flags, T1PNDA and T1PNDB. The control flag T1ENA allows the interrupt on T1A to be either enabled or disabled. Setting the T1ENA flag enables interrupts to be generated when the selected trigger condition occurs on the T1A pin. Similarly, the flag T1ENB controls the interrupts from the T1B pin.

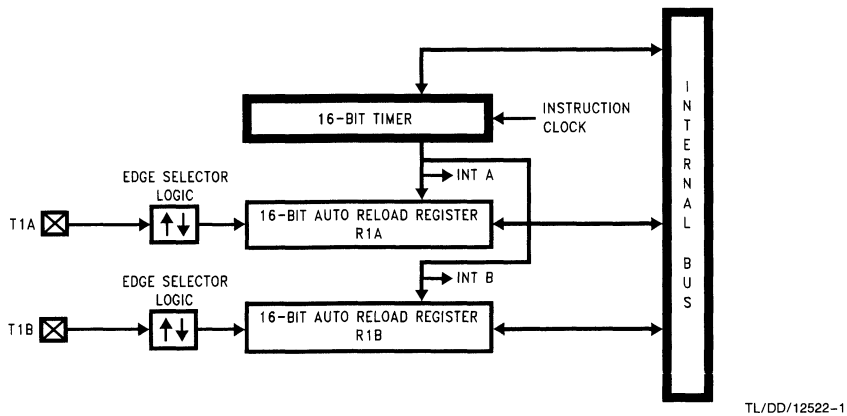
Underflows from the timer can also be programmed to generate interrupts. Underflows are latched into the timer T1C0 pending flag (the T1C0 control bit serves as the timer underflow interrupt pending flag in the Input Capture mode). Consequently, the T1C0 control bit should be reset when entering the Input Capture mode. The timer underflow interrupt is enabled with the T1ENA control flag. When a T1A interrupt occurs in the Input Capture mode, the user must check both the T1PNDA and T1C0 pending flags in order to determine whether a T1A input capture or a timer underflow (or both) caused the interrupt.

Figure 11 shows a block diagram of the timer in Input Capture mode.



TL/DD/12522–10

FIGURE 10. Timer 1 in External Event Counter Mode



TL/DD/12522–11

FIGURE 11. Timer 1 in Input Capture Mode

Timers (Continued)

TIMER CONTROL FLAGS

The control bits and their functions are summarized below.

T1C0 Timer Start/Stop control in Modes 1 and 2 (Processor Independent PWM and External Event Counter), where 1 = Start, 0 = Stop
Timer Underflow Interrupt Pending Flag in Mode 3 (Input Capture)
T1PNDA Timer Interrupt Pending Flag
T1PNDB Timer Interrupt Pending Flag
T1ENA Timer Interrupt Enable Flag
T1ENB Timer Interrupt Enable Flag
1 = Timer Interrupt Enabled
0 = Timer Interrupt Disabled
T1C3 Timer mode control
T1C2 Timer mode control
T1C1 Timer mode control

The timer mode control bits (T1C3, T1C2 and T1C1) are detailed below:

T1C3	T1C2	T1C1	Timer Mode	Interrupt A Source	Interrupt B Source	Timer Counts On
0	0	0	MODE 2 (External Event Counter)	Timer Underflow	Positive T1B Edge	T1A Positive Edge
0	0	1	MODE 2 (External Event Counter)	Timer Underflow	Positive T1B Edge	T1A Negative Edge
1	0	1	MODE 1 (PWM) T1A Toggle	Autoreload RA	Autoreload RB	t_C
1	0	0	MODE 1 (PWM) No T1A Toggle	Autoreload RA	Autoreload RB	t_C
0	1	0	MODE 3 (Capture) Captures: T1A Positive Edge T1B Positive Edge	Positive T1A Edge or Timer Underflow	Positive T1B Edge	t_C
1	1	0	MODE 3 (Capture) Captures: T1A Positive Edge T1B Negative Edge	Positive T1A Edge or Timer Underflow	Negative T1B Edge	t_C
0	1	1	MODE 3 (Capture) Captures: T1A Negative Edge T1B Positive Edge	Negative T1A Edge or Timer Underflow	Positive T1B Edge	t_C
1	1	1	MODE 3 (Capture) Captures: T1A Negative Edge T1B Negative Edge	Negative T1A Edge or Timer Underflow	Negative T1B Edge	t_C

HIGH SPEED, CONSTANT RESOLUTION PWM TIMER

The device has one processor independent PWM timer. The PWM timer operates in two modes: PWM mode and capture mode. In PWM mode the timer outputs can be programmed to two pins PWM0 and PWM1. In capture mode, pin PWM0 functions as the capture input. *Figure 12* shows a block diagram for this timer in capture mode and *Figure 13* shows a block diagram for the timer in PWM mode.

PWM Timer Registers

The PWM Timer has three registers: PWMCON, the PWM control register, RLON, the PWM on-time register and PSCAL, the prescaler register.

PWM Prescaler Register (PSCAL) (Address X'00A0)

The prescaler is the clock source for the counter in both PWM mode and in frequency monitor mode.

PSCAL is a read/write register that can be used to program the prescaler. The clock source to the timer in both PWM and capture modes can be programmed to CKI/N where N = PSCAL + 1, so the maximum PWM clock frequency = CKI and the minimum PWM clock frequency = CKI/256. The processor is able to modify the PSCAL register regardless of whether the counter is running or not and the change in frequency occurs with the next underflow of the prescaler (CK-PWM).

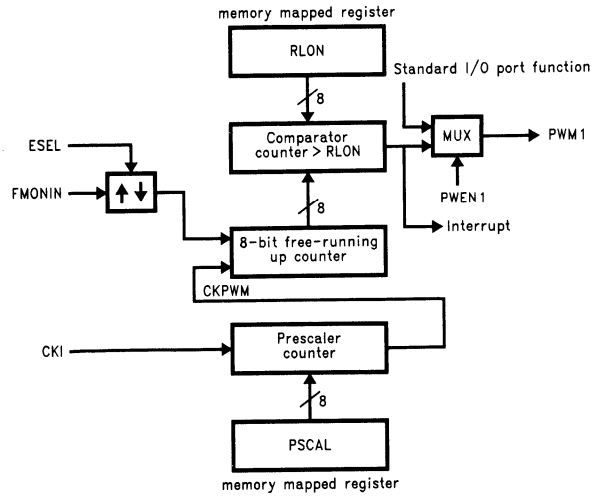
PWM On-time Register (RLON) (Address X'00A1)

RLON is a read/write register. In PWM mode the timer output will be a "1" for RLON counts out of a total cycle of 255 PWM clocks. In capture mode it is used to program the threshold frequency.

The PWM timer is specially designed to have a resolution of 255 PWM clocks. This allows the duty cycle of the PWM output to be selected between 1/255 and 254/255. A value of 0 in the RLON register will result in the PWM output being continuously low and a value of 255 will result in the PWM output being continuously high.

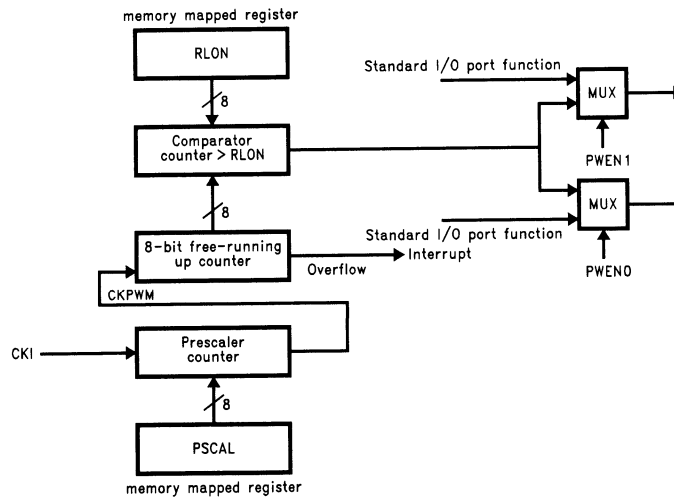
Note: The effect of changing the RLON register during active PWM mode operation is delayed until the boundary of a PWM cycle. In capture mode the effect takes place immediately.

Timers (Continued)



TL/DD/12522-12

FIGURE 12. PWM Timer Capture Mode Block Diagram



TL/DD/12522-13

FIGURE 13. PWM Timer PWM Mode Block Diagram

Timers (Continued)

PWM Control Register (PWMCON) (Address X'00A2)

The PWMCON Register Bits are:

PWEN0 Enable PWM0 output/input function on I/O port.

PWEN1 Enable PWM1 output function on I/O port.

Note: The associated bits in the configuration and data register of the I/O-port have to be setup as outputs and/or inputs in addition to setting the PWEN bits.

PWON PWM start Bit, "1" to start timer, "0" to stop timer.

PWMD PWM Mode bit, "1" for PWM mode, "0" frequency monitor mode.

PWIE PWM interrupt enable bit.

PWPND PWM interrupt pending bit.

ESEL Edge select bit, "1" for falling edge, "0" for rising edge.

unused	ESEL	PWPND	PWIE	PWMD	PWON	PWEN1	PWEN0

Bit 7 Bit 0

PWM Mode

The PWM timer can generate PWM signals at frequencies up to 39 kHz (@ t_C = 1 μs) with a resolution of 255 parts. Lower PWM frequencies can be programmed via the prescaler.

If the PWM mode bit (PWMD) in the PWM configuration register (PWMCON) is set to "1" the timer operates in PWM mode. In this mode, the timer generates a PWM signal with a fixed, non-programmable repetition rate of 255 PWM clock cycles. The timer is clocked by the output of an 8-bit, programmable prescaler, which is clocked with the chip's CKI frequency. Thus the PWM signal frequency can be calculated with the formula:

$$fpwm = \frac{CKI}{(1 + (PSCAL\text{-}contents)) \times 255}$$

Selecting the PWM mode by setting PWMD to "1", but not yet starting the timer (PWON is "0"), will set the timer output to "1".

The contents of an 8-bit register, RLON, multiplied by the clock cycle of the prescaler output defines the time between overflow (or starting) and the falling edge of the PWM output.

Once the timer is started, the timer output goes low after RLON cycles and high after a total of 255 cycles. The procedure is continually repeated. In PWM mode the timer is available at pins PWM0 and/or PWM1, provided the port configuration bits for those pins are defined as outputs and the PWEN0 and/or PWEN1 bits in the PWMCON register are set.

The PWM timer is started by the software setting the PWON bit to "1". Starting the timer initializes the timer register. From this point, the timer will continually generate the PWM signal, independent of any processor activity, until the timer is stopped by software setting the PWON bit to "0". The processor is able to modify the RLON register regardless of whether the timer is running. If RLON is changed while the timer is running, the previous value of RLON is used for comparison until the next overflow occurs, when the new value of RLON is latched into the comparator inputs.

When the timer overflows, the PWM pending flag (PWPND) is set to "1". If the PWM interrupt enable bit (PWIE) is also set to "1", timer overflow will generate an interrupt. The PWPND bit remains set until the user's software writes a "0" to it. If the software writes a "1" to the PWPND bit, this has no effect. If the software writes a "0" to the PWPND bit at the same time as the hardware writes to the bit, the hardware has precedence.

Note: The software controlling the duty cycle is able to change the PWM duty cycle without having to wait for the timer overflow.

Figure 14 shows how the PWM output is implemented. The PWM Timer output is set to "1" on an overflow of the timer and set to "0" when the timer is greater than RLON. The output can be multiplexed to two pins.

Capture Mode

If the PWM mode bit (PWMD) is set to "0" the PWM Timer operates in capture mode. Capture mode allows the programmer to test whether the frequency of an external source exceeds a certain threshold.

If PWMD is "0" and PWON is "0", the timer output is set to "0". In capture mode the timer output is available at pin PWM1, provided the port configuration register bit for that pin is set up as an output and the PWEN1 bit in the PWMCON register is set. Setting PWON to "1" will initialize the timer register and start the counter. A rising edge, or if selected, a falling edge, on the FMONIN input pin will initialize the timer register and clear the timer output. The counter continues to count up after being initialized. The ESEL bit determines whether the active edge is a rising or a falling edge.

Timers (Continued)

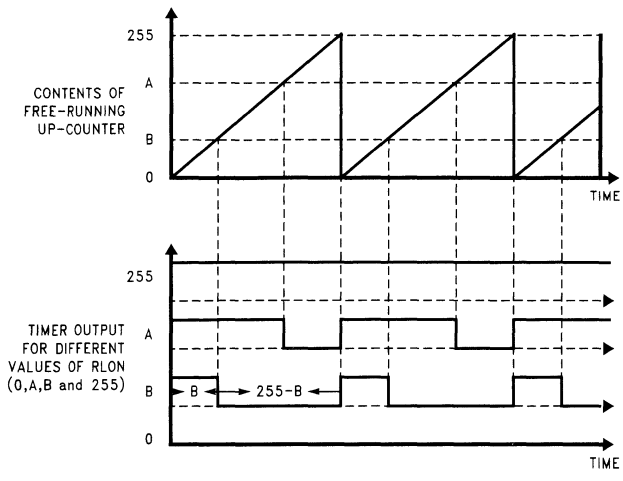


FIGURE 14. PWM Mode Operation

If, in capture mode PWM0 is configured incorrectly as an output and is enabled via the PWEN0 bit, the timer output will feedback into the PWM block as the timer input.

The contents of the counter are continually compared with the RLON register. If the frequency of the input edges is sufficiently high, the contents of the counter will always be less than the value in RLON. However, if the frequency of the input edges is too low, the free-running counter value will count up beyond the value in RLON.

When the counter is greater than RLON, the PWM timer output is set to "1". It is set to "0" by a detected edge on the timer input or when the counter overflows. When the counter becomes greater than RLON, the PWPND bit in the PWM control register is set to "1". If the PWIE bit is also set to "1", the PWPND bit is enabled to request an interrupt.

It should be noted that two other conditions could also set the PWPND bit:

1. If the mode of operation is changed on the fly the timer output will toggle. If frequency monitor mode is entered on the fly such that the timer output changes from 0 to 1, PWPND will be set.
2. If the timer is operating in frequency monitor mode and the RLON value is changed on the fly so that RLON becomes less than the current timer value, PWPND will be set.

The PWPND bit remains set until the user's software writes a "0" to it. If the software writes a "1" to the PWPND bit, this has no effect. If the software writes a "0" to the PWPND bit at the same time as the hardware writes to the bit, the hardware has precedence. (See *Figure 15* for Frequency Monitor Mode Operation.)

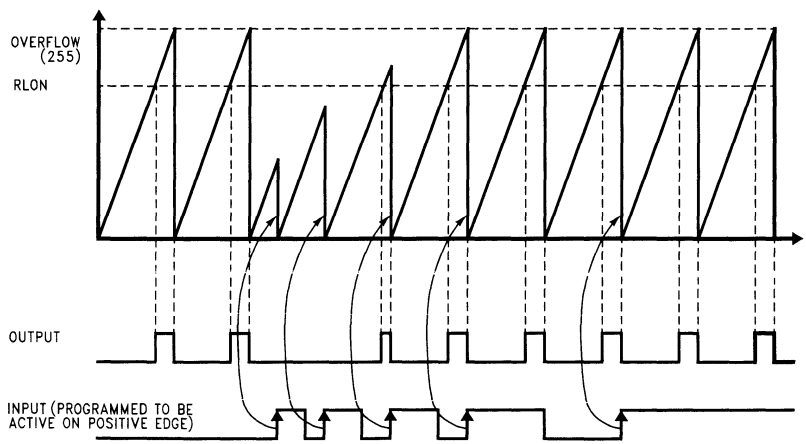


FIGURE 15. Frequency Monitor Mode Operation

Power Save Modes

The device offers the user two power save modes of operation: HALT and IDLE. In the HALT mode, all microcontroller activities are stopped. In the IDLE mode, the on-board oscillator circuitry and timer T0 are active but all other microcontroller activities are stopped. In either mode, all on-board RAM, registers, I/O states, and timers (with the exception of T0) are unaltered.

HALT MODE

The contents of all PWM Timer registers are frozen during HALT mode and are left unchanged when exiting HALT mode. The PWM timer resumes its previous mode of operation when exiting HALT mode.

The device is placed in the HALT mode by writing a "1" to the HALT flag (G7 data bit). All microcontroller activities, including the clock, and timers, are stopped. In the HALT mode, the power requirements of the device are minimal and the applied voltage (V_{CC}) may be decreased to V_r (V_r = 2.0V) without altering the state of the machine.

The device supports two different ways of exiting the HALT mode. The first method of exiting the HALT mode is with the Multi-Input Wake Up feature on the L port. The second method of exiting the HALT mode is by pulling the RESET pin low.

Since a crystal or ceramic resonator may be selected as the oscillator, the Wake Up signal is not allowed to start the chip running immediately since crystal oscillators and ceramic resonators have a delayed start up time to reach full amplitude and frequency stability. The IDLE timer is used to generate a fixed delay to ensure that the oscillator has indeed stabilized before allowing instruction execution. In this case, upon detecting a valid Wake Up signal, only the oscillator circuitry is enabled. The IDLE timer is loaded with a value of 256 and is clocked with the t_c instruction cycle clock. The t_c clock is derived by dividing the oscillator clock down by a factor of 10. The Schmitt trigger following the CKI inverter on the chip ensures that the IDLE timer is clocked only when the oscillator has a sufficiently large amplitude to meet the Schmitt trigger specifications. This Schmitt trigger is not part of the oscillator closed loop. The start-up time-out from the IDLE timer enables the clock signals to be routed to the rest of the chip.

The device has two mask options associated with the HALT mode. The first mask option enables the HALT mode feature, while the second mask option disables the HALT mode. With the HALT mode enable mask option, the device will enter and exit the HALT mode as described above. With the HALT disable mask option, the device cannot be placed in the HALT mode (writing a "1" to the HALT flag will have no effect).

IDLE MODE

The device is placed in the IDLE mode by writing a "1" to the IDLE flag (G6 data bit). In this mode, all activities, except the associated on-board oscillator circuitry, and the IDLE Timer T0, are stopped. The power supply requirements of the microcontroller in this mode of operation are typically around 30% of normal power requirement of the microcontroller.

As with the HALT mode, the device can be returned to normal operation with a reset, or with a Multi-Input Wake Up from the L Port or CAN Interface. Alternately, the microcontroller resumes normal operation from the IDLE mode when the thirteenth bit (representing 4.096 ms at internal clock frequency of 1 MHz, t_c = 1 μs) of the IDLE Timer toggles.

This toggle condition of the thirteenth bit of the IDLE Timer T0 is latched into the T0PND pending flag.

The user has the option of being interrupted with a transition on the thirteenth bit of the IDLE Timer T0. The interrupt can be enabled or disabled via the T0EN control bit. Setting the T0EN flag enables the interrupt and vice versa.

The user can enter the IDLE mode with the Timer T0 interrupt enabled. In this case, when the T0PND bit gets set, the device will first execute the Timer T0 interrupt service routine and then return to the instruction following the "Enter Idle Mode" instruction.

Alternatively, the user can enter the IDLE mode with the IDLE Timer T0 interrupt disabled. In this case, the device will resume normal operation with the instruction immediately following the "Enter IDLE Mode" instruction.

Note: It is necessary to program two NOP instructions following both the set HALT mode and set IDLE mode instructions. These NOP instructions are necessary to allow clock resynchronization following the HALT or IDLE modes.

Multi-Input Wake Up

The Multi-Input Wake Up feature is used to return (wake up) the device from either the HALT or IDLE modes. Alternately, the Multi-Input Wake Up/Interrupt feature may also be used to generate up to 7 edge selectable external interrupts.

Figure 16 shows the Multi-Input Wake Up logic for the microcontroller. The Multi-Input Wake Up feature utilizes the L Port. The user selects which particular L port bit (or combination of L Port bits) will cause the device to exit the HALT or IDLE modes. The selection is done through the Reg: WKEN. The Reg: WKEN is an 8-bit read/write register, which contains a control bit for every L port bit. Setting a particular WKEN bit enables a Wake Up from the associated port pin.

The user can select whether the trigger condition on the selected L Port pin is going to be either a positive edge (low to high transition) or a negative edge (high to low transition). This selection is made via the Reg: WKEDG, which is an 8-bit control register with a bit assigned to each L Port pin. Setting the control bit will select the trigger condition to be a negative edge on that particular L Port pin. Resetting the bit selects the trigger condition to be a positive edge. Changing an edge select entails several steps in order to avoid a pseudo Wake Up condition as a result of the edge change. First, the associated WKEN bit should be reset, followed by the edge select change in WKEDG. Next, the associated WKPND bit should be cleared, followed by the associated WKEN bit being re-enabled.

An example may serve to clarify this procedure. Suppose we wish to change the edge select from positive (low going high) to negative (high going low) for L Port bit 5, where bit 5 has previously been enabled for an input interrupt. The program would be as follows:

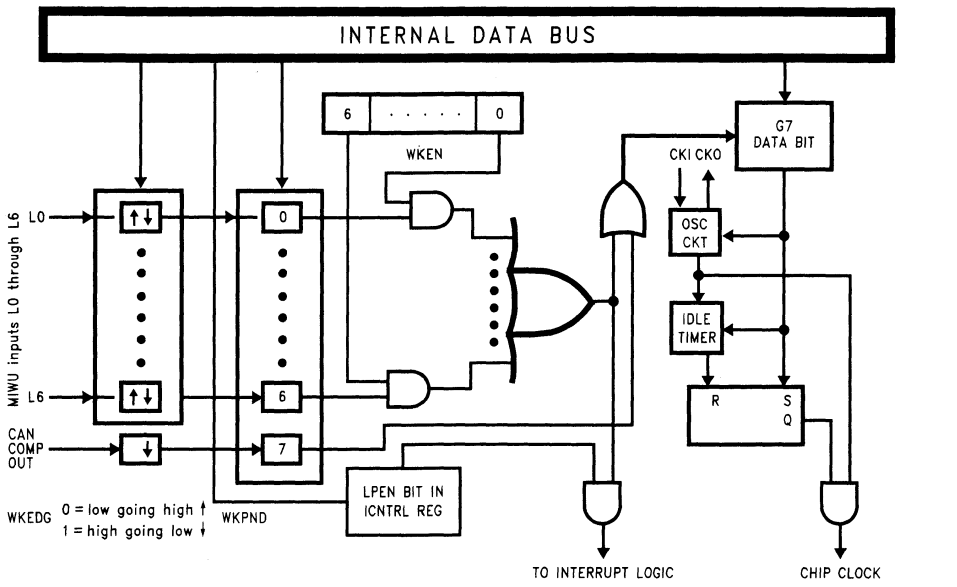
```
RBIT  5, WKEN
SBIT  5, WKEDG
RBIT  5, WKPND
SBIT  5, WKEN
```

Multi-Input Wake Up (Continued)

If the L port bits have been used as outputs and then changed to inputs with Multi-Input Wake Up/Interrupt, a safety procedure should also be followed to avoid inherited pseudo wake up conditions. After the selected L port bits have been changed from output to input but before the associated WKEN bits are enabled, the associated edge select bits in WKEDG should be set or reset for the desired edge selects, followed by the associated WKPND bits being cleared.

This same procedure should be used following reset, since the L port inputs are left floating as a result of reset. The occurrence of the selected trigger condition for Multi-Input Wake Up is latched into a pending register called WKPND. The respective bits of the WKPND register will be set on the occurrence of the selected trigger edge on the corresponding Port L pin. The user has the responsibility of clearing these pending flags. Since WKPND is a pending register for the occurrence of selected wake up conditions, the device will not enter the HALT mode if any Wake Up bit is both enabled and pending. Consequently, the user has the responsibility of clearing the pending flags before attempting to enter the HALT mode.

The WKEN, WKPND and WKEDG are all read/write registers, and are cleared at reset.



TL/DD/12522-16

FIGURE 16. Multi-Input Wake Up Logic

Multi-Input Wake Up (Continued)

CAN RECEIVE WAKE UP

The CAN Receive Wake Up source is always enabled and is always active on a falling edge of the CAN comparator output. There is no specific enable bit for the CAN Wake Up feature. Although the wake up feature on pins L0..L6 can be programmed to generate an interrupt (L-port interrupt), no interrupt is generated upon a CAN receive wake up condition. The CAN block has its own, dedicated receiver interrupt upon receive buffer full.

PORT L INTERRUPTS

Port L provides the user with an additional seven fully selectable, edge sensitive interrupts which are all vectored into the same service subroutine.

The interrupt from Port L shares logic with the wake up circuitry. The register WKEN allows interrupts from Port L to be individually enabled or disabled. The register WKEDG specifies the trigger condition to be either a positive or a negative edge. Finally, the register WKPND latches in the pending trigger conditions.

The GIE (global interrupt enable) bit enables the interrupt function. A control flag, LPEN, functions as a global interrupt enable for Port L interrupts. Setting the LPEN flag will enable interrupts and vice versa. A separate global pending flag is not needed since the register WKPND is adequate.

Since Port L is also used for waking the device out of the HALT or IDLE modes, the user can elect to exit the HALT or IDLE modes either with or without the interrupt enabled. If he elects to disable the interrupt, then the device will restart execution from the instruction immediately following the instruction that placed the microcontroller in the HALT or IDLE modes. In the other case, the device will first execute the interrupt service routine and then revert to normal operation.

The Wake Up signal will not start the chip running immediately since crystal oscillators or ceramic resonators have a finite start up time. The IDLE Timer (T0) generates a fixed delay to ensure that the oscillator has indeed stabilized before allowing the device to execute instructions. In this case, upon detecting a valid Wake Up signal, only the oscillator circuitry and the IDLE Timer T0 are enabled. The IDLE Timer is loaded with a value of 256 and is clocked from the t_c instruction cycle clock. The t_c clock is derived by dividing down the oscillator clock by a factor of 10. A Schmitt trigger following the CKI on-chip inverter ensures that the IDLE timer is clocked only when the oscillator has a sufficiently large amplitude to meet the Schmitt trigger specifications. This Schmitt trigger is not part of the oscillator closed loop. The start-up time-out from the IDLE timer enables the clock signals to be routed to the rest of the chip.

Interrupts

The device supports a vectored interrupt scheme. It supports a total of eleven interrupt sources. The following table lists all the possible device interrupt sources, their arbitration ranking and the memory locations reserved for the interrupt vector for each source.

Two bytes of program memory space are reserved for each interrupt source. All interrupt sources except the software interrupt are maskable. Each of the maskable interrupts have an Enable bit and a Pending bit. A maskable interrupt is active if its associated enable and pending bits are set. If GIE = 1 and an interrupt is active, then the processor will be interrupted as soon as it is ready to start executing an instruction except if the above conditions happen during the Software Trap service routine. This exception is described in the Software Trap sub-section.

The interruption process is accomplished with the INTR instruction (opcode 00), which is jammed inside the Instruction Register and replaces the opcode about to be executed. The following steps are performed for every interrupt:

1. The GIE (Global Interrupt Enable) bit is reset.
2. The address of the instruction about to be executed is pushed into the stack.
3. The PC (Program Counter) branches to address 00FF. This procedure takes 7 t_c cycles to execute.

At this time, since GIE = 0, other maskable interrupts are disabled. The user is now free to do whatever context switching is required by saving the context of the machine in the stack with PUSH instructions. The user would then program a VIS (Vector Interrupt Select) instruction in order to branch to the interrupt service routine of the highest priority interrupt enabled and pending at the time of the VIS. Note that this is not necessarily the interrupt that caused the branch to address location 00FF Hex prior to the context switching.

Thus, if an interrupt with a higher rank than the one which caused the interruption becomes active before the decision of which interrupt to service is made by the VIS, then the interrupt with the higher rank will override any lower ones and will be acknowledged. The lower priority interrupt(s) are still pending, however, and will cause another interrupt immediately following the completion of the interrupt service routine associated with the higher priority interrupt just serviced. This lower priority interrupt will occur immediately following the RETI (Return from Interrupt) instruction at the end of the interrupt service routine just completed.

Inside the interrupt service routine, the associated pending bit has to be cleared by software. The RETI (Return from Interrupt) instruction at the end of the interrupt service routine will set the GIE (Global Interrupt Enable) bit, allowing the processor to be interrupted again ff another interrupt is active and pending.

The VIS instruction looks at all the active interrupts at the time it is executed and performs an indirect jump to the beginning of the service routine of the one with the highest rank.

The addresses of the different interrupt service routines, called vectors, are chosen by the user and stored in ROM in a table starting at 01E0 (assuming that VIS is located between 00FF and 01DF). The vectors are 15-bit wide and therefore occupy 2 ROM locations.

VIS and the vector table must be located in the same 256-byte block (0y00 to 0yFF) except if VIS is located at the last address of a block. In this case, the table must be in the next block. The vector table cannot be inserted in the first 256-byte block.

Interrupts (Continued)

The vector of the maskable interrupt with the lowest rank is located at 0yE0 (Hi-Order byte) and 0yE1 (Lo-Order byte) and so forth in increasing rank number. The vector of the maskable interrupt with the highest rank is located at 0yFA (Hi-Order byte) and 0yFB (Lo-Order byte).

The Software Trap has the highest rank and its vector is located at 0yFE and 0yFF.

Arbitration Ranking	Source	Vector Address Hi-Low Byte
1	Software Trap	0yFE–0yFF
2	Reserved	0yFC–0yFD
3	CAN Receive	0yFA–0yFB
4	CAN Error (transmit/receive)	0yF9–0yF9
5	CAN Transmit	0yF6–0yF7
6	Pin G0 Edge	0yF4–0yF5
7	IDLE Timer Underflow	0yF2–0yF3
8	Timer T1A/Underflow	0yF0–0yF1
9	Timer T1B	0yEE–0yEF
10	MICROWIRE/PLUS	0yEC–0yED
11	PWM timer	0YEA–0yEB
12	Reserved	0yE8–0yE9
13	Reserved	0yE6–0yE7
14	Reserved	0yE4–0yE5
15	Port L/Wake Up	0yE2–0yE3
16	Default VIS Interrupt	0yE0–0yE1

y is VIS page, y ≠ 0

If, by accident, a VIS gets executed and no interrupt is active, then the PC (Program Counter) will branch to a vector located at 0yE0-0yE1.

WARNING

A Default VIS interrupt handler routine must be present. As a minimum, this handler should confirm that the GIE bit is cleared (this indicates that the interrupt sequence has been taken), take care of any required housekeeping, restore context and return. Some sort of Warm Restart procedure should be implemented. These events can occur without any error on the part of the system designer or programmer.

Note: There is always the possibility of an interrupt occurring during an instruction which is attempting to reset the GIE bit or any other interrupt enable bit. If this occurs when a single cycle instruction is being used to reset the interrupt enable bit, the interrupt enable bit will be reset but an interrupt may still occur. This is because interrupt processing is started at the same time as the interrupt bit is being reset. To avoid this scenario, the user should always use a two, three, or four cycle instruction to reset interrupt enable bits.

Figure 17 shows the Interrupt Block diagram.

SOFTWARE TRAP

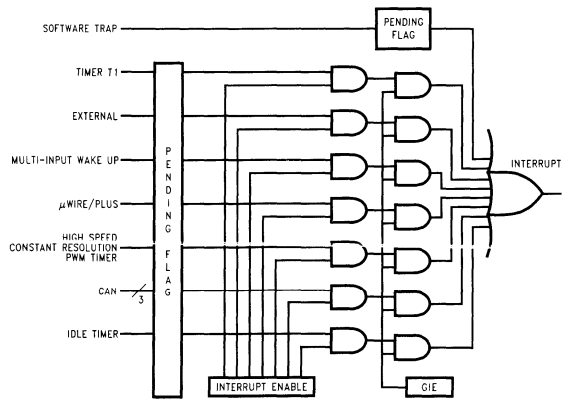
The Software Trap (ST) is a special kind of non-maskable interrupt which occurs when the INTR instruction (used to acknowledge interrupts) is fetched from ROM and placed inside the instruction register. This may happen when the PC is pointing beyond the available ROM address space or when the stack is over-popped.

When an ST occurs, the user can re-initialize the stack pointer and do a recovery procedure (similar to RESET, but not necessarily containing all of the same initialization procedures) before restarting.

The occurrence of an ST is latched into the ST pending bit. The GIE bit is not affected and the ST pending bit (**not accessible by the user**) is used to inhibit other interrupts and to direct the program to the ST service routine with the VIS instruction. The RPND instruction is used to clear the software interrupt pending bit. This bit is also cleared on reset.

The ST has the highest rank among all interrupts.

Nothing (except another ST) can interrupt an ST being serviced.



TL/DD/12522–17

FIGURE 17. Interrupt Block Diagram

CAN Block Description *

This device contains a CAN serial bus interface as described in the CAN Specification Rev. 2.0 part B.

*Patents Pending.

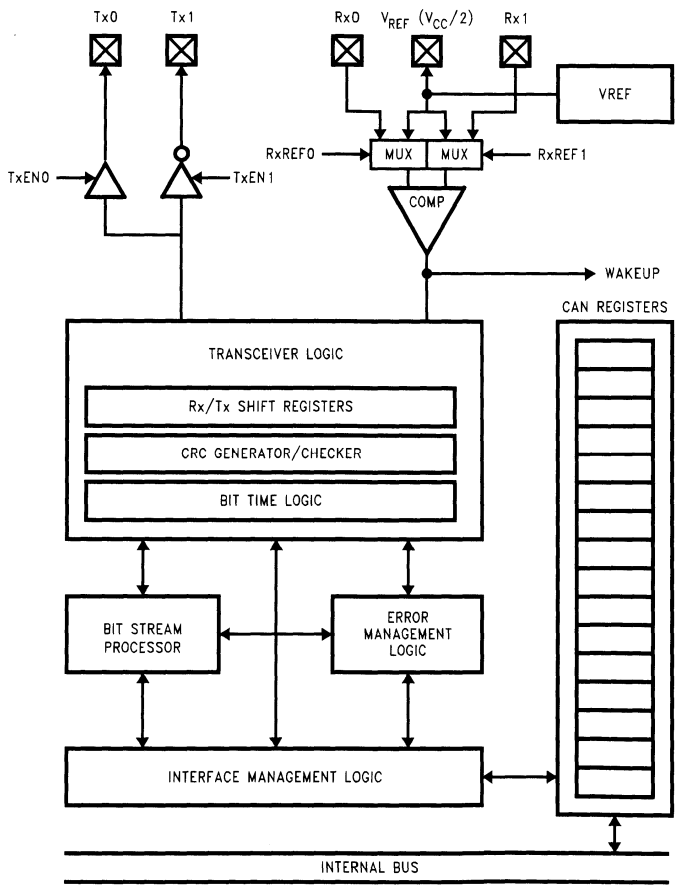
CAN Interface Block

This device supports applications which require a low speed CAN interface. It is designed to be programmed with two transmit and two receive registers. The user's program may check the status bytes in order to get information of the bus state and the received or transmitted messages. The device has the capability to generate an interrupt as soon as one byte has been transmitted or received. Care must be taken if more than two bytes in a message frame are to be transmitted/received. In this case the user's program must poll the transmit buffer empty (TBE)/receive buffer full (RBF) bits or enable their respective interrupts and perform a data exchange between the user data and the Tx/Rx registers.

Fully automatic transmission on error is supported for messages not longer than two bytes. Messages which are longer than two bytes have to be processed by software.

The interface is compatible with CAN Specification 2.0 part B, without the capability to receive/transmit extended frames. Extended frames on the bus are checked and acknowledged according to the CAN specification.

The maximum bus speed achievable with the CAN interface is a function of crystal frequency, message length and software overhead. The device can support a bus speed of up to 1 Mbit/S with a 10 MHz oscillator and 2 byte messages. The 1 Mbit/S bus speed refers to the rate at which protocol and data bits are transferred on the bus. Longer messages require slower bus speeds due to the time required for software intervention between data bytes. The device will support a maximum of 125k bit/s with eight byte messages and a 10 MHz oscillator.



TL/DD/12522–46

FIGURE 18. CAN Interface Block Diagram

Functional Block Description of the CAN Interface

Interface Management Logic (IML)

The IML executes the CPU's transmission and reception commands and controls the data transfer between CPU, Rx/Tx and CAN registers. It provides the CAN Interface with Rx/Tx data from the memory mapped Register Block. It also sets and resets the CAN status information and generates interrupts to the CPU.

Bit Stream Processor (BSP)

The BSP is a sequencer controlling the data stream between The Interface Management Logic (parallel data) and the bus line (serial data). It controls the transceive logic with regard to reception and arbitration, and creates error signals according to the bus specification

Transceive Logic (TCL)

The TCL is a state machine which incorporates the bit stuff logic and controls the output drivers, CRC logic and the Rx/Tx shift registers. It also controls the synchronization to the bus with the CAN clock signal generated by the BTL.

Error Management Logic (EML)

The EML is responsible for the fault confinement of the CAN protocol. It is also responsible for changing the error counters, setting the appropriate error flag bits and interrupts and changing the error status (passive, active and bus off).

Cyclic Redundancy Check (CRC) Generator and Register

The CRC Generator consists of a 15-bit shift register and the logic required to generate the checksum of the destuffed bit-stream. It informs the EML about the result of a receiver checksum.

The checksum is generated by the polynomial:

$$\chi^{15} + \chi^{14} + \chi^{10} + \chi^{8} + \chi^{7} + \chi^{4} + \chi^{3} + 1$$

Receive/Transmit (Rx/Tx) Registers

The Rx/Tx registers are 8-bit shift registers controlled by the TCL and the BSP. They are loaded or read by the Interface Management Logic, which holds the data to be transmitted or the data that was received.

Bit Time Logic (BTL)

The bit time logic divider divides the CKI input clock by the value defined in the CAN prescaler (CSCAL) and bus timing register (CTIM). The resultig bit time (tcan) can be computed by the formula:

$$t_{can} = \frac{CKI}{(1 + divider) \times (1 + 2 \times PS + PPS)}$$

Where *divider* is the value of the clock prescaler, *PS* is the programmable value of phase segment 1 and 2 (1..8) and *PPS* the programmed value of the propagation segment (1..8) (located in CTIM).

Bus Timing Considerations

The internal architecture of the CAN interface has been optimized to allow fast software response times within messages of more than two data bytes. The TBE (Transmit Buffer Empty) bit is set on the last bit of odd data bytes when CAN internal sample points are high.

It is the user's responsibility to ensure that the time between setting TBE and a reload of TxD2 is longer than the length of phase segment 2 as indicated in the following equation:

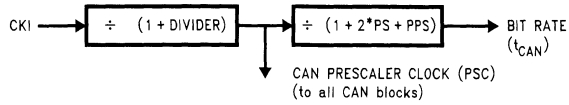
$$t_{LOAD} > \frac{(PS + 1) \times (CSCAL + 1)}{10} t_C = \text{absolute length of PS2}$$

Table II shows examples of the minimum required t_{LOAD} for different CSCAL settings based on a clock frequency of 10 MHz. Lower clock speeds require recalculation of the CAN bit rate and the mimimum t_{LOAD}.

TABLE II. CAN Timing (CKI = 10 MHz, t_c = 1 μs)

PS	CSCAL	CAN Bit Rate (kbit/s)	Minimum t_{LOAD} (μs)
4	3	250	2.0
4	9	100	5.0
4	15	62	8.0
4	24	40	12.5
4	39	25	20
4	99	10	50
4	199	5	100

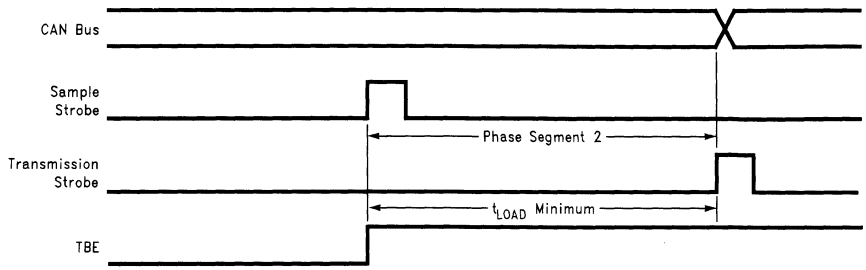
Functional Block Description of the CAN Interface (Continued)



TL/DD/12522–47

FIGURE 19. Bit Rate Generation

Figure 32 illustrates the minimum time required for t_{LOAD}.



TL/DD/12522–48

FIGURE 20. TBE Timing

In the case of an interrupt driven CAN interface, the calculation of the actual t_{LOAD} time would be done as follows:

```
INT:                                    ;Interrupt latency = 7tc = 7 µs
                PUSH        A           ;3tc = 3 µs
                LD          A,B         ;2tc = 2 µs
                PUSH        A           ;3tc = 3 µs
                VIS                     ;5tc = 5 µs
CANTX:                                  ;20tc = µs to this point
                •                       ;additional time for instructions which check
                •                       ;status prior to reloading the transmit data
                •                       ;registers with subsequent data bytes.
                LD          TXD2,DATA
                •
                •
                •
```

Functional Block Description of the CAN Interface (Continued)

Interrupt driven programs use more time than programs which poll the TBE flag, however programs which operate at lower baud rates (which are more likely to be sensitive to this issue) have more time for interrupt response.

Output Drivers/Input Comparators

The output drivers/input comparators are the physical interface to the bus. Control bits are provided to TRI-STATE the output drivers.

A dominant bit on the bus is represented as a "0" in the data registers and a recessive bit on the bus is represented as a "1" in the data registers.

TABLE III. Bus Level Definition

Bus Level	Pin Tx0	Pin Tx1	Data
"dominant"	drive low (GND)	dirve high (V_{CC})	0
"recessive"	TRI-STATE	TRI-STATE	1

Register Block

The register block consists of fifteen 8-bit registers which are described in more detail in the following paragraphs.

Note: The contents of the receiver related registers RxD1, RxD2, RDLC, RIDH and RTSTAT are only changed if a received frame passes the acceptance filter or the Receive Identifier Acceptance Filter bit (RIAF) is set to accept all received messages.

TRANSMIT DATA REGISTER 1 (TXD1) (Address X'00B0)

The Transmit Data Register 1 contians the first data byte to be transmitted within a frame and then the successive odd byte numbers (i.e., bytes number 1,3,..,7).

TRANSMIT DATA REGISTER 2 (TXD2) (Address X'00B1)

The Transit Data Register 2 contains the second data byte to be transmitted within a frame and then the successive even byte numbers (i.e., bytes number 2,4,..,8).

TRANSMIT DATA LENGTH CODE AND IDENTIFIER LOW REGISTER (TDLC) (Address X'00B2)

TID3	TID2	TID1	TID0	TDLC3	TDLC2	TCLC1	TDLC0
Bit 7							Bit 0

This register is read/write.

TID3..TIDO Transmit Identifier Bits 3..0 (lower 4 bits)

The transmit identifier is composed of eleven bits in total, bits 3 to 0 of the TID are stored in bits 7 to 4 of this register.

TDLC3..TDLC0 Transmit Data Length Code

These bits determine the number of data bytes to be transmitted within a frame. The CAN specification allows a maximum of eight data bytes in any message.

TRANSMIT IDENTIFIER HIGH (TID) (Address X'00B3)

TRTR	TID10	TID9	TID8	TID7	TID6	TID5	TID4
Bit 7							Bit 0

This register is read/write.

TRTR Transmit Remote Frame Request

This bit is set if the frame to be transmitted is a remote frame request.

TID10..TID4 Transmit Identifier Bits 10 .. 4 (higher 7 bits)

Bits TID10..TID4 are the upper 7 bits of the 11 bit transmit identifier.

RECEIVER DATA REGISTER 1 (RXD1) (Address X'00B4)

The Receive Data Register 1 (RXD1) contains the first data byte received in a frame and then successive odd byte numbers (i.e., bytes 1, 3,..7). This register is read-only.

RECEIVE DATA REGISTER 2 (RXD2) (Address X'00B5)

The Receive Data Register 2 (RXD2) contains the second data byte received in a frame and then successive even byte numbers (i.e., bytes 2,4,..,8). This register is read-only.

REGISTER DATA LENGTH CODE AND IDENTIFIER LOW REGISTER (RIDL) (Address X'00B6)

RID3	RID2	RID1	RID0	RDLC3	RDLC2	RDLC1	RDLC0
Bit 7							Bit 0

This register is read only.

RID3..RID0 Receive Identifier bits (lower four bits)

The RID3..RID0 bits are the lower four bits of the eleven bit long Receive Identifier. Any received message that matches the upper 7 bits of the Receive Identifier (RID10..RID4) is accepted if the Receive Identifier Acceptance Filter (RIAF) bit is set to zero.

RDLC3..RDLC0 Receive Data Length Code bits

The RDLC3..RDLC0 bits determine the number of data bytes within a received frame.

RECEIVE IDENTIFIER HIGH (RID) (Address X'00B7)

unused	RID10	RID9	RID8	RID7	RID6	RID5	RID4
Bit 7							Bit 0

This register is read/write.

RID10..RID4 Receive Identifier bits (upper bits)

The RID10...RID4 bits are the upper 7 bits of the eleven bit long Receive Identifier. If the Receive Identifier Acceptance Filter (RIAF) bit (see CBUS register) is set to zero, bits 4 to 10 of the received identifier are compared with the mask bits of RID4..RID10. If the corresponding bits match, the message is accepted. If the RIAF bit is set to a one, the filter function is disabled and all messages, independent of identifier, will be accepted.

Functional Block Description of the CAN Interface (Continued)

CAN PRESCALER REGISTER (CSCAL) (Address X'00B8)

CKS7	CKS6	CKS5	CKS4	CKS3	CKS2	CKS1	CKS0

Bit 7 Bit 0

This register is read/write.

CKS7..0 Prescaler divider select.

The resulting clock value is the CAN Prescaler clock.

CAN BUS TIMING REGISTER (CTIM) (00B9)

PPS2	PPS1	PPS0	PPS0	PS2	PS1	PS0	Reserved

Bit 7 Bit 0

This register is read/write.

PPS2..PPS0 Propagation Segment, bits 2..0

The PPS2..PPS0 bits determine the length of the propagation delay in Prescaler clock cycles (PSC) per bit time. (For a more detailed discussion of propagation delay and phase segments, see SYNCHRONIZATION on page 37.)

PS2..PS0 Phase Segment 1, bits 2..0

The PS2..PS0 bits fix the number of Prescaler clock cycles per bit time for phase segment 1 and phase segment 2. The PS2..PS0 bits also set the synchronization Jump Width to a value equal to the lesser of the 4 PSC or the length of PS1/2 (Min: 4 I length of PS1/2).

TABLE IV. Synchronization Jump Width

PS2	PS1	PS0	Length of Phase Segment ½	Synchronization Jump Width
0	0	0	1 t_{can}	1 t_{can}
0	0	1	2 t_{can}	2 t_{can}
0	1	0	3 t_{can}	3 t_{can}
0	1	1	4 t_{can}	4 t_{can}
1	0	0	5 t_{can}	4 t_{can}
1	0	1	6 t_{can}	4 t_{can}
1	1	0	7 t_{can}	4 t_{can}
1	1	1	8 t_{can}	4 t_{can}

LENGTH OF TIME SEGMENTS (See *Figure 32*)

- The Synchronization Segment is 1 CAN Prescaler clock (PSC)
- The Propagation Segment can be programmed (PPS) to be 1,2...,8 PSC in length.
- Phase Segment 1 and Phase Segment 2 are programmable (PS) to be 1,2,..,8 PSC long.

Note: (BTL settings at high speed; PSC = 0) Due to the on-chip delay from the rx-pins through the receive comparator (worst case assumption: 3 clocks delay * 2 (devices on the bus) + 1 tx delay) the user needs to set the sample point to > (2*3 + 1) i.e., > 7 CKI clocks to ensure correct communication on the bus under all circumstances. With prescaler settings of > 0 this is a given (i.e., no caution has to be applied).

Example: for 1 Mbit CTIM = b'10000100 (PSS = 5; PS1 = 2). Example for 500 kbit CTIM = b'01011100 (PPS = 3; PS1 = 8). – all at 10 MHz CKI and CSCAL = 0.

CAN BUS CONTROL REGISTER (CBUS) (00BA)

Reserved	RIAF	TxEN1	TxEN0	RxREF1	RxREF0	Reserved	FMOD

Bit 7 Bit 0

Reserved This bit is reserved and should be zero.

RIAF Receive identifier acceptance filter bit

If the RIAF bit is set to zero, bits 4 to 10 of the received identifier are compared with the mask bits of RID4..RID10 and if the corresponding bits match, the message is accepted. If the RIAF bit is set to a one, the filter function is disabled and all messages independent of the identifier will be accepted.

TxEN0, TxEN1 TxD Output Driver Enable

TABLE V. Output Drivers

TxEN1	TxEN0	Output
0	0	Tx0, Tx1 TRI-STATED, CAN input comparator disabled
0	1	Tx0 enabled
1	0	Tx1 enabled
1	1	Tx0 and Tx1 enabled

Bus synchronization of the device is done in the following way:

If the output was disabled (TxEN1, TxEN0 = "0") and either TxEN1 or TxEN0, or both are set to 1, the device will not start transmission or reception of a frame until eleven consecutive "recessive" bits have been received. Resetting the TxEN1 and TxEN0 bits will disable the output drivers and the CAN input comparator. All other CAN related registers and flags will be unaffected. It is recommended that the user reset the TxEN1 and TxEN0 bits before switching the device into the HALT mode (the CAN receive wakeup will still work) in order to reduce current consumption and to assure a proper resychronization to the bus after exiting the HALT mode.

Note: A "bus off" condition will also cause Tx0 and Tx1 to be at TRI-STATE (independent of the values of the TxEN1 and TxEN0 bits).

RXREF1 Reference voltage applied to Rx1 if bit is set

RXREF0 Reference voltage applied to Rx0 if bit is set

FMOD Fault Confinement Mode select

Setting the FMOD bit to "0" (default after power on reset) will select the Standard Fault Confinement mode. In this mode the device goes from "bus off" to "error active" after monitoring 128*11 recessive bits (including bus idle) on the bus. This mode has been implemented for compatibility with existing solutions. Setting the FMOD bit to "1" will select the Enhanced Fault Confinement mode. In this mode the device goes from "bus off" to "error active" after monitoring 128 "good" messages, as indicated by the reception of 11 consecutive "recessive" bits including the End of Frame, whereas the standard mode may time out after 128 x 11 recessive bits (e.g., bus idle).

Functional Block Description of the CAN Interface (Continued)

TRANSMIT CONTROL/STATUS (TCNTL) (00BB)

NS1	NS0	TERR	RERR	CEIE	TIE	RIE	TXSS

Bit 7 Bit 0

NS1..NS0 Node Status, i.e., Error Status.

TABLE VI. Node Status

NS1	NS0	Output
0	0	Error active
0	1	Error passive
1	0	Bus off
1	1	Bus off

The Node Status bits are read only.

TERR Transmit Error

This bit is automatically set when an error occurs during the transmission of a frame. TERR can be programmed to generate an interrupt by setting the Can Error Interrupt Enable bit (CEIE). This bit must be cleared by the user's software.

Note: This is used for messages for more than two bytes. If an error occurs during the transmission of a frame with more than 2 data bytes, the user's software has to handle the correct reloading of the data bytes to the TxD registers for retransmission of the frame. For frames with 2 or less data bytes the interface logic of this chip does an automatic retransmission. Regardless of the number of data bytes, the user's software must reset this bit if CEIE is enabled. Otherwise a new interrupt will be generated immediately after return from the interrupt service routine.

RERR Receiver Error

This bit is automatically set when an error occurred during the reception of a frame. RERR can be programmed to generate an interrupt by setting the Can Error Interrupt Enable bit (CEIE). This bit has to be cleared by the user's software.

CEIE CAN Error Interrupt Enable

If set by the user's software, this bit enables the tansmit and receive error interrupts. The interrupt pending flags are TERR and RERR. Resetting this bit with a pending error interrupt will inhibit the interrupt, but will not clear the cause of the interrupt (RERR or TERR). If the bit is then set without clearing the cause of the interrupt, the interrupt will reoccur.

TIE Transmit Interrupt Enable

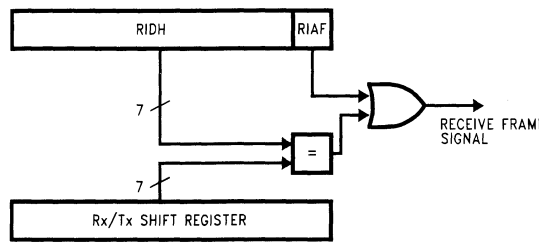
If set by the user's software, this bit enables the transmit interrupt. (See TBE and TXPND.) Resetting this bit with a pending transmit interrupt will inhibit the interrupt, but will not clear the cause of the interrupt. If the bit is then set without clearing the cause of the interrupt, the interrupt will reoccur.

RIE Receive Interrupt Enable

If set by the user's software, this bit enables the receive interrupt or a remote transmission request interrupt (see RBF, RFV and RRTR). Resetting this bit with a pending receive interrupt will inhibit the interrupt, but will not clear the cause of the interrupt. If the bit is then set without clearing the cause of the interrupt, the interrupt will reoccur.

TXSS Transmission Start/Stop

This bit is set by the user's software to initiate the transmission of a frame. Once this bit is set, a transmission is pending, as indicated by the TXPND flag being set. It can be reset by software to cancel a pending transmission. Resetting the TXSS bit will only cancel a transmission, if the transmission of a frame hasn't been started yet (bus idle), if arbitration has been lost (receiving) or if an error occurs during transmission. If the device has already started transmission (won arbitration) the TXPND and TXSS flags will stay set until the transmission is completed, even if the user's software has written zero to the TXSS bit. If one or more data bytes are to be transmitted, care must be taken by the user, that the Transmit Data Register(s) have been loaded before the TXSS bit is set. TXSS will be cleared on three conditions only: Successful completion of a transmitted message; successful cancellation of a pending transmision; Transition of the CAN interface to the bus-off state.



TL/DD/12522–49

FIGURE 21. Acceptance Filter Block-Diagram

Functional Block Description of the CAN Interface (Continued)

Writing a zero to the TXSS bit will request cancellation of a pending transmission but TXSS will not be cleared until completion of the operation. If an error occurs during transmission of a frame, the logic will check for cancellation requests prior to restarting transmission. If zero has been written to TXSS, retransmission will be canceled.

RECEIVE/TRANSMIT STATUS (RTSTAT) (Address X'00BC)

TBE	TXPND	RRTR	ROLD	RORN	RFV	RCV	RBF
1	0	0	0	0	0	0	0

Bit 7 Bit 0

This register is read only.

TBE Transmit Buffer Empty

This bit is set as soon as the TxD2 register is copied into the Rx/Tx shift register, i.e., the 1st data byte of each pair has been transmitted. The TBE bit is automatically reset if the TxD2 register is written (the user should write a dummy byte to the TxD2 register when transmitting an odd number of bytes of zero bytes). TBE can be programmed to generate an interrupt by setting the Transmit Interrupt Enable bit (TIE). When servicing the interrupt the user has to make sure that TBE gets cleared by executing a WRITE instruction on the TxD2 register, otherwise a new interrupt will be generated immediately after return from the interrupt service routine. The TBE bit is read only. It is set to 1 upon reset. TBE is also set upon completion of transmission of a valid message.

TXPND Transmission Pending

This bit is set as soon as the Transmit Start/Stop (TXSS) bit is set by the user. It will stay set until the frame was successfully transmitted, until the transmission was successfully canceled by writing zero to the Transmission Start/Stop bit (TXSS), or the device enters the bus-off state. Resetting the TXSS bit will only cancel a transmission if the transmission of a frame hasn't been started yet (bus idle) or if arbitration has been lost (receiving). If the device has already started transmission (won arbitration) the TXPND flag will stay set until the transmission is completed, even if the user's software has requested cancellation of the message. If an error occurs during transmission, a requested cancellation may occur prior to the begining of retransmission.

RRTR Received Remote Transmission Request

This bit is set when the remote transmission request (RTR) bit in a received frame was set. It is automatically reset through a read of the RXD1 register.

To detect RRTR the user can either poll this flag or enable the receive interrupt (the reception of a remote transmission request will also cause an interrupt if the receive interrupt is enabled). If the receive interrupt is enabled, the user should check the RRTR flag in the service routine in order to distinguish between a RRTR interrupt and a RBF interrupt. It is the responsibility of the user to clear this bit by reading the RXD1 register, before the next frame is received.

ROLD Received Overload Frame

This bit is automatically set when an Overload Frame was received on the bus. It is automatically reset through a read of the Receive/Transmit Status register. It is the responsibility of the user to clear this bit by reading the Receive/Transmit Status register, before the next frame is received.

RORN Receiver Overrun

This bit is automatically set on an overrun of the receive data register, i.e., if the user's program does not maintain the RxDn registers when receiving a frame. It it automatically reset through a read of the Receive/Transmit Status register. It is the responsibility of the user to clear this bit by reading the Receive/Transmit Status register before the next frame is received.

RFV Received Frame Valid

This bit is set if the received frame is valid, i.e., after the penultimate bit of the End of Frame is received. It is automatically reset through a read of the Receive/Transmit Status register. It is the responsibility of the user to clear this bit by reading the receive/transmit status register (RTSTAT), before the next frame is received. RFV will cause a Receive Interrupt if enabled by RIE. The user should be careful to read the last data byte (RxD1) of odd length messages (1, 3, 5 or 7 data bytes) on receipt of RFV. RFV is the only indication that the last byte of the message has been received.

RCV Receive Mode

This bit is set after the data length code of a message that passes the device's acceptance filter has been received. It is automatically reset after the CRC-delimiter of the same frame has been received. It indicates to the user's software that arbitration is lost and that data is coming in for that node.

RBF Receive Buffer Full

This bit is set if the second Rx data byte was received. It is reset automatically, after the RxD1-Register has been read by the software. RBF can be programmed to generate an interrupt by setting the Receive Interrupt Enable bit (RIE). When servicing the interrupt, the user has to make sure that RBF gets cleared by executing a LD instruction from the RxD1 register, otherwise a new interrupt will be generated immediately after return from the interrupt service routine. The RBF bit is read only.

TRANSMIT ERROR COUNTER (TEC) (Address X'00BD)

TEC7	TEC6	TEC5	TEC4	TEC3	TEC2	TEC1	TEC0

Bit 7 Bit 0

This register is read/write.

Functional Block Description of the CAN Interface (Continued)

For test purposes and to identify the node status, the transmit error counter, an 8-bit error counter, is mapped into the data memory. If the lower seven bits of the counter overflow, i.e., TEC7 is set, the device is error passive.

CAUTION

To prevent interference with the CAN fault confinement, the user must not write to the REC/TEC registers. Both counters are automatically updated following the CAN specification.

RECEIVE ERROR COUNTER (REC) (00BE)

ROVL	REC6	REC5	REC4	REC3	REC2	REC1	REC0
Bit 7							Bit 0

This register is read/write.

ROVL receive error counter overflow

For test purposes and to identify the node status the receive error counter, a 7-bit error counter, is mapped into the data memory. If the counter overflows the ROVL bit is set to indicate that the device is error passive and won't transmit any active error frames. If ROVL is set then the counter is frozen.

MESSAGE IDENTIFICATION

a. Transmitted Message

The user can select all 11 Transmit Identifier Bits to transmit any message whigh fulfills the CAN2.0, part B spec without an extended identifier (see note below). Fully automatic retransmission is supported for messages no longer than 2 bytes.

b. Received Messages

The lower four bits of the Receive Identifier are don't care, i.e., the controller will receive all messages that fit in that window (16 messages). The upper 7 bits can be defined by the user in the Receive Identifier High Register to mask out groups of messages. If the RIAF bit is set, all messages will be received.

Note: The CAN interface tolerates the extended CAN frame format of 29 identifier bits and gives an acknowledgment. If an error occurs the receive error counter will be increased, and decreased if the frame is valid.

BUS SYNCHRONIZATION DURING OPERATION

Resetting the TxEN1 and TxEN0 bits in Bus Control Register will disable the output drivers and do a resynchronization to the bus. All other CAN related registers and flags will be unaffected.

Bus synchronization of the device is this case is done in the following way:

If the output was disabled (TxEN1, TxEN0 = "0") and either TxEN1 or TxEN0, or both are set to 1, the device will not start transmission or reception of a frame until eleven consecutive "recessive" bits have been received.

A "bus off" condition will also cause the output drivers Tx1 and Tx0 to be at TRI-STATE (independent of the status of TxEN1 and TxEN0). The device will switch from "bus off" to "error active" mode as described under the FMOD-bit description (see Can Bus Control register). This will ensure that the device is synchronized to the bus, before starting to transmit or receive.

For information on bus synchronization and status of the CAN related registers after external reset refer to the RESET section.

ON-CHIP VOLTAGE REFERENCE

The on-chip voltage reference is a ratiometric reference. For electrical characteristics of the voltage reference refer to the electrical specifications section.

ANALOG SWITCHES

Analog switches are used for selecting between Rx0 and V_{REF} and between Rx1 and V_{REF}.

Basic CAN Concepts

The following paragraphs provide a generic overview of the basic concepts of the Controller Area Network (CAN) as described in *Chapter 4 of ISO/DIS11519-1.* Implementation related issues of the National Semiconductor device will be discussed as well.

This device will process standard frame format only. Extended frame formats will be acknowledged, however the data will be discarded. For this reason the description of frame formats in the following section will cover only the standard frame format.

The following section provides some more detail on how the device will handle received extended frames:

If the device's remote identifier acceptance filter bit (RIAF) is set to "1", extended frame messages will be acknowledged. However, the data will be discarded and the device will not reply to a remote transmission request received in extended frame format. If the device's RIAF bit is set to "0", the upper 7 received ID bits of an extended frame that match the device's receive identifier (RID) acceptance filtler bits, are stroed in the device's RID register. However, the device does not reply to an RTR and any data is discarded. The device will only acknowledge the message.

MULTI-MASTER PRIORITY BASED BUS ACCESS

The CAN protocol is message based protocol that allows a total of 2032 ($= 2^{11} - 16$) different messages in the standard format and 512 million ($= 2^{29} - 16$) different messages in the extended frame format.

MULTICAST FRAME TRANSFER BY ACCEPTANCE FILTERING

Every CAN Frame is put on the common bus. Each module receives every frame and filters out the frames which are not required for the module's task.

REMOTE DATA REQUEST

A CAN master module has the ability to set a specific bit called the "remote transmission request bit" (RTR) in a frame. This causes another module, either another master or a slave, to transmit a data frame after the current frame has been completed.

Basic CAN Concepts (Continued)

SYSTEM FLEXIBILITY

Additional modules can be added to an existing network without a configuration change. These modules can either perform completely new functions requiring new data or process existing data to perform a new function.

SYSTEM WIDE DATA CONSISTENCY

As the CAN network is message oriented, a message can be used like a variable which is automatically updated by the controlling processor. If any module cannot process information it can send an overload frame. The device is incapable of initiating an overload frame, but will join a overload frame initiated by another device as required by CAN specifications.

NON-DESTRUCTIVE CONTENTION-BASED ARBITRATION

The CAN protocol allows several transmitting modules to start a transmission at the same time as soon as they monitor the bus to be idle. During the start of transmission every node monitors the bus line to detect whether its message is overwritten by a message with a higher priority. As soon as a transmitting module detects another module with a higher priority accessing the bus, it stops transmitting its own frame and switches to receive mode. For illustration see *Figure 22*.

AUTOMATIC RETRANSMISSION OF FRAMES

If a data or remote frame is overwritten by either a higher-prioritized data frame, remote frame or an error frame, the transmitting module will automatically retransmit it. This device will handle the automatic retransmission of up to two data bytes automatically. Messages with more than 2 data bytes require the user's software to update the transmit registers.

ERROR DETECTION AND ERROR SIGNALING

All messages on the bus are checked by each CAN node and acknowledge if they are correct. If any node detects an error it starts the transmission of an error frame.

Switching Off Defective Nodes

There are two error counters, one for transmitted data and one for received data, which are incremented, depending on the error type, as soon as an error occurs. If either counter goes beyond a specific value the node goes to an error state. A valid frame causes the error counters to decrease.

The device can be in one of three states with respect to error handling:

- Error active
  An error active unit can participate in bus communication and sends an active ("dominant") error flag.
- Error passive
  An error passive unit can participate in bus communication. However, if the unit detects an error it is not allowed to send an active error flag. The unit sends only a passive ("recessive") error flag.
- Bus off

A unit that is "bus off" has the output drivers disabled, i.e., it does not participate in any bus activity.

(See ERROR MANAGEMENT AND DETECTION for more detailed information.)

Frame Formats

INTRODUCTION

There are basically two different types of frames used in the CAN protocol.

The data transmission frames are: data/remote frame

The control frames are: error/overload frame

Note: This device cannot send an overload frame as a result of not being able to process all information. However, the device is able to recognize an overload condition and join overload frames initiated by other devices.

If no message is being transmitted, i.e., the bus is idle, the bus is kept at the "recessive" level. *Figure 23* and *Figure 24* give an overview of the various CAN frame formats.

DATA AND REMOTE FRAME

Data frames consist of seven bit fields and remote frames consist of six different bit fields:

1. Start of Frame (SOF)
2. Arbitration field
3. Control field (IDE bit, R0 bit, and DLC field)
4. Data field (not in remote frame)
5. CRC field
6. ACK field
7. End of Frame (EOF)

A remote frame has no data field and is used for requesting data from other (remote) CAN nodes. *Figure 25* shows the format of a CAN data frame.

FRAME CODING

Remote and Data Frames are NRZ codes with bit-stuffing in every bit field which holds computable information for the interface, i.e., Start of Frame arbitration field, control field, data field (if present) and CRC field.

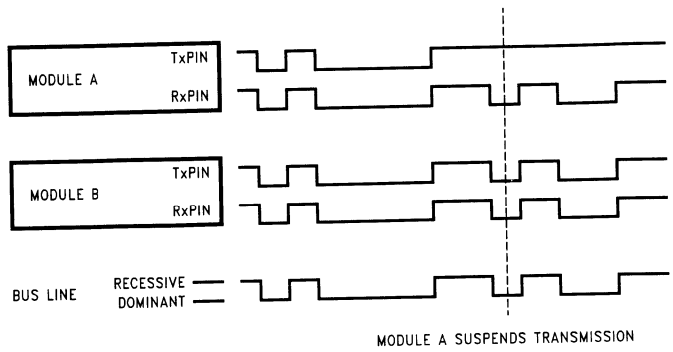
Error and overload frames are NRZ coded without bit stuffing.

BIT STUFFING

After five consecutive bits of the same value, a stuff bit of the inverted value is inserted by the transmitter and deleted by the receiver.

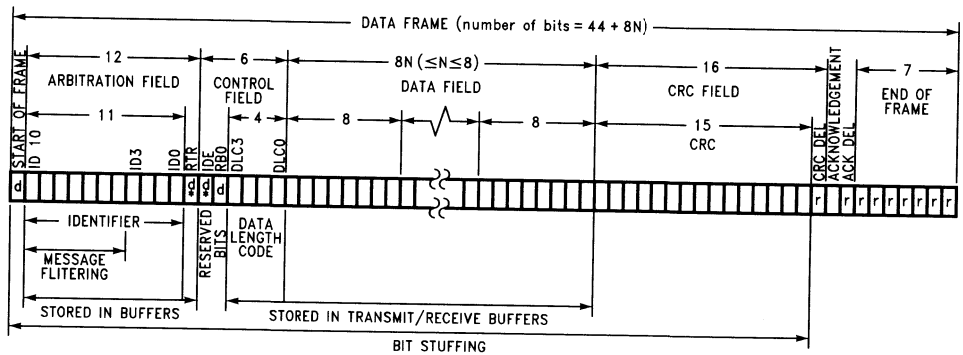
Destuffed Bit Stream	100000x	011111x
Stuffed Bit Stream	1000001x	0111110x
		x = {0,1}

Basic CAN Concepts (Continued)

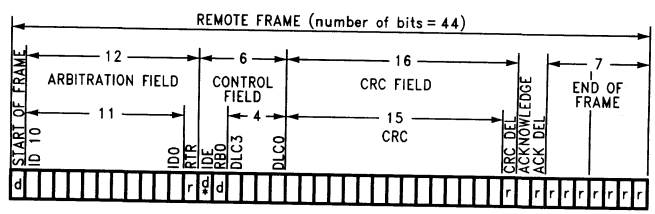


TL/DD/12522–50

FIGURE 22. CAN Message Arbitration



TL/DD/12522–51



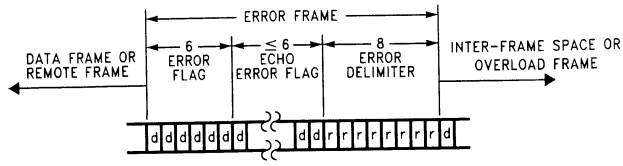
TL/DD/12522–52

A remote frame is identical to a data frame, except that the RTR bit is "recessive", and there is no data field.

IDE = Identifier Extension Bit
The IDE bit in the standard format is transmitted "dominant", whereas in the extended format the IDE bit is "recessive" and the id is expanded to 29 bits.
r = recessive
d = dominant

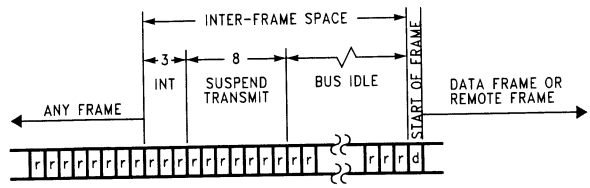
FIGURE 23. CAN Data Transmission Frames

Basic CAN Concepts (Continued)



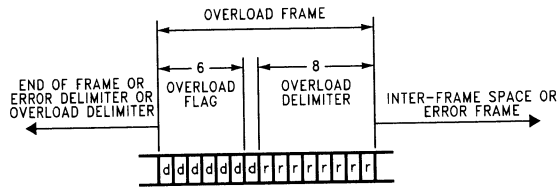
TL/DD/12522–53

An error frame can start anywhere in the middle of a frame.



TL/DD/12522–54

INT = Intermission
Suspend Transmission is only for error passive nodes.

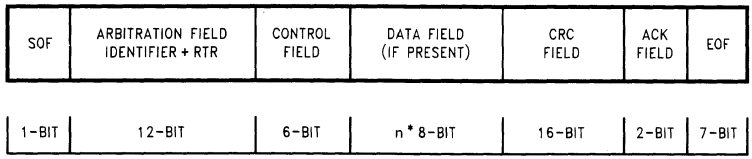


TL/DD/12522–55

An overload frame can only start at the end of a frame.

FIGURE 24. CAN Control Frames

Basic CAN Concepts (Continued)



TL/DD/12522–56

FIGURE 25. CAN Frame Format

START OF FRAME (SOF)

The Start of Frame indicates the beginning of data and remote frames. It consists of a single "dominant" bit. A node is only allowed to start transmission when the bus is idle. All nodes have to synchronize to the leading edge (first edge after the bus was idle) caused by SOF of the node which starts transmission first.

ARBITRATION FIELD

The arbitration field is composed of the identifier field and the RTR (Remote Transmission Request) bit. The value of the RTR bit is "dominant" in a data frame and "recessive" in a remote frame.

CONTROL FIELD

The control field consists of six bits. It starts with two bits reserved for future expansion followed by the four-bit Data Length Code. Receivers must accept all possible combinations of the two reserved bits. Until the function of these reserved bits is defined, the transmitter only sends "0" (dominant) bits. The first reserved bit (IDE) is actually defined to indicate an extended frame with 29 Identifier bits if set to "1". CAN chips must tolerate extended frames, even if they can only understand standard frames, to prevent the destruction of an extended frames on an existing network.

The Data Length Code indicates the number of bytes in the data field. This Data Length Code consists of four bits. The data field can be of length zero. The permissible number of data bytes for a data frame ranges from 0 to 8.

DATA FIELD

The Data field consists of the data to be transferred within a data frame. It can contain 0 to 8 bytes and each byte contains 8 bits. A remote frame has no data field.

CRC FIELD

The CRC field consists of the CRC sequence followed by the CRC delimiter. The CRC sequence is derived by the transmitter from the modulo 2 division of the preceding bit fields, starting with the SOF up to the end of the data field, excluding stuff-bits, by the generator polynomial:

$$\chi^{15} + \chi^{14} + \chi^{10} + \chi^{8} + \chi^{7} + \chi^{4} + \chi^{3} + 1$$

The remainder of this division is the CRC sequence transmitted over the bus. On the receiver side the module divides all bit fields up to the CRC delimiter, excluding stuff-bits, and checks if the result is zero. This will then be interpreted as a valid CRC. After the CRC sequence a single "recessive" bit is transmitted as the CRC delimiter.

ACK FIELD

The ACK field is two bits long and contains the ACK slot and the ACK delimiter. The ACK slot is filled with a "recessive" bit by the transmitter. This bit is overwritten with a "dominant" bit by every receiver that has received a correct CRC sequence. The second bit of the ACK field is a "recessive" bit called the acknowledge delimiter. As a consequence the acknowledge flag of a valid frame is surrounded by two "recessive" bits, the CRC-delimiter and the ACK delimiter.

EOF FIELD

The End of Frame Field closes a data and a remote frame. It consists of seven "recessive" bits.

INTERFRAME SPACE

Data and remote frames are separate from every preceding frame (data, remote, error and overload frames) by the interframe space see *Figure 26* and *Figure 27* for details. Error and overload frames are not preceded by an interframe space. They can be transmitted as soon as the condition occurs. The interframe space consists of a minimum of three bit fields depending on the error state of the node.

These bit fields are coded as follows:

The intermission has the fixed form of three "recessive" bits. While this bit field is active, no node is allowed to start a transmission of a data or a remote frame. The only action to be taken is signaling an overload condition. This means that an error in this bit field would be interpreted as an overload condition. Suspend transmission has to be inserted by error-passive nodes that were transmitter for the last message. This bit field has the form of eight "recessive" bits. However, it may be overwritten by a "dominant" start-bit from another non error passive node which starts transmission. The bus idle field consists of "recessive" bits. Its length is not specified and depends on the bus load.

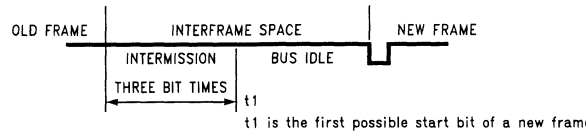
Basic CAN Concepts (Continued)

ERROR FRAME

The Error Frame consists of two bit fields: the error flag and the error delimiter. The error field is built up from the various error flags of the different nodes. Therefore, its length may vary from a minimum of six bits up to a maximum of twelve bits depending on when a module detects the error. Whenever a bit error, stuff error, form error, or acknowledgment error is detected by a node, this node starts transmission of the error flag at the next bit. If a CRC error is detected, transmission of the error flag starts at the bit following the acknowledge delimiter, unless an error flag for a previous error condition has already been started. *Figure 28* shows how a local fault at one module (module 2) leads to a 12-bit error frame on the bus.

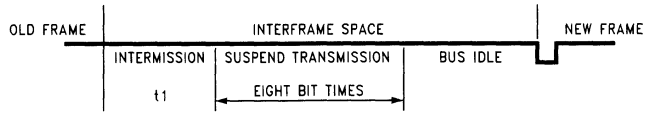
The bus level may either be "dominant" for an error-active node or "recessive" for an error-passive node. An error active node detecting an error, starts transmitting an active error flag consisting of six "dominant" bits. This causes the destruction of the actual frame on the bus. The other nodes detect the error flag as either a violation of the rule of bit-stuffing or the value of a fixed bit field is destroyed. As a consequence all other nodes start transmission of their own error flag. This means, that the error sequence which can be monitored on the bus as a maximum length of twelve bits. If an error passive node detects an error it transmits six "recessive" bits on the bus. This sequence does not destroy a message sent by another node and is not detected by other nodes. However, if the node detecting an error was the transmitter of the frame the other modules will get an error condition by a violation of the fixed bit or stuff rule. *Figure 27* shows how an error passive transmitter transmits a passive error frame and when it is detected by the receivers.

After any module has transmitted its active or passive error flag it waits for the error delimiter which consists of eight "recessive" bits before continuing.



TL/DD/12522–57

FIGURE 26. Interframe Space for Nodes Which Are Not Error Passive or Have Been Receiver for the Last Frame



TL/DD/12522–58

FIGURE 27. Interframe Space for Nodes Which Are Error Passive and Have Been Transmitter for the Last Frame

Basic CAN Concepts (Continued)

OVERLOAD FRAME

Like an error frame, an overload frame consists of two bit fields: the overload flag and the overload delimiter. The bit fields have the same length as the error frame field: six bits for the overload flag and eight bits for the delimiter. The overload frame can only be sent after the end of frame (EOF) field and in they way destroys the fixed form of the intermission field.

ORDER OF BIT TRANSMISSION

A frame is transmitted starting with the Start of Frame, sequentially followed by the remaining bit fields. In every bit field the MSB is transmitted first.

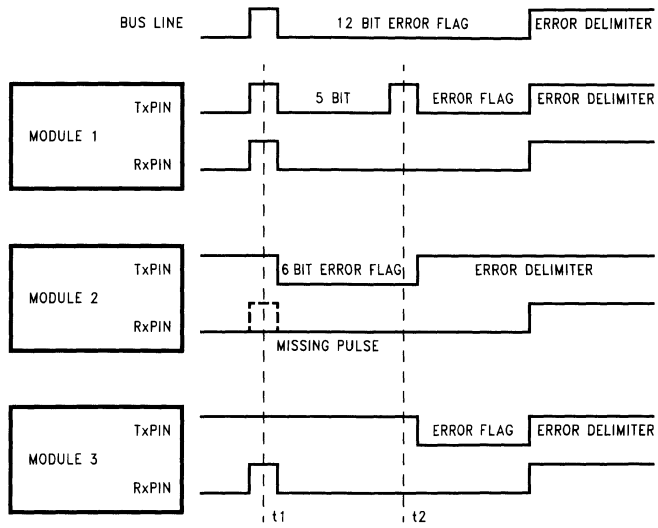
FRAME VALIDATION

Frames have a different validation point for transmitters and receivers. A frame is valid for the transmitter of a message, if there is no error until the end of the last bit of the End of Frame field. A frame is valid for a receiver, if there is no error until and including the end of the penultimate bit of the End of Frame.

FRAME ARBITRATION AND PRIORITY

Except for an error passive node which transmitted the last frame, all nodes are allowed to start transmission of a frame after the intermission, which can lead to two or more nodes starting transmission at the same time. To prevent a node from destroying another node's frame, it monitors the bus during transmission of the identifier field and the RTR-bit. As soon as it detects a "dominant" bit while transmitting a "recessive" bit it releases the bus, immediately stops transmission and starts receiving the frame. This causes no data or remote frame to be destroyed by another. Therefore the highest priority message with the identifier 0x000 out of 0x7EF (including the remote data request (RTR) bit) always gets the bus. This is only valid for standard CAN frame format. Note that while the CAN specification allows valid standard identifiers only in the range 0x000 to 0x7EF, the device will allow identifiers to 0x7FF.

There are three more items that should be taken into consideration to avoid unrecoverable collisions on the bus:



TL/DD/12522-59

module 1 = error active transmitter detects bit error at t2
module 2 = error active receiver with a local fault at t1
module 3 = error active receiver detects stuff error at t2

FIGURE 28. Error Frame—Error Active Transmitter

Basic CAN Concepts (Continued)

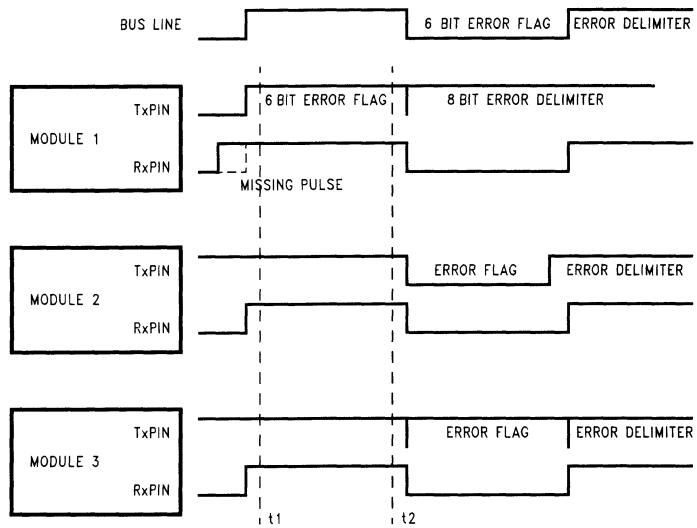
- Within one system each message must be assigned a unique identifier. This is to prevent bit errors, as one module may transmit a "dominant" data bit while the other is transmitting a "recessive" data bit. This could happen if two or more modules start transmission of a frame at the same time and all win arbitration.
- Data frames with a given identifier and a non-zero data length code may be initiated by one node only. Otherwise, in worst case, two nodes would count up to the bus-off state, due to bit errors, if they always start transmitting the same ID with different data.
- Every remote frame should have a system-wide data length code (DLC). Otherwise two modules starting transmission of a remote frame at the same time will overwrite each other's DLC which result in bit errors.

ACCEPTANCE FILTERING

Every node may perform acceptance filtering on the identifier of a data or a remote frame to filter out the messages which are not required by the node. In they way only the data of frames which match the acceptance filter is stored in the corresponding data buffers. However, every node which is not in the bus-off state and has received a correct CRC-sequence acknowledges each frame.

ERROR MANAGEMENT AND DETECTION

There are multiple mechanisms in the CAN protocol, to detect errors and to inhibit erroneous modules from disabling all bus activities.

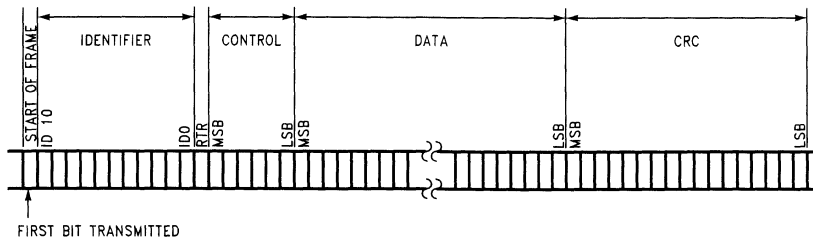


TL/DD/12522-60

module 1 = error active receiver with a local fault at t1
module 2 = error passive transmitter detects bit error at t2
module 3 = error passive receiver detects stuff error at t2

FIGURE 29. Error Frame—Error Passive Transmitter

Basic CAN Concepts (Continued)



TL/DD/12522–61

FIGURE 30. Order of Bit Transmission within a CAN Frame

The following errors can be detected:

- Bit Error
  A CAN device that is sending also monitors the bus. If the monitored bit value is different from the bit value that is sent, a bit error is detected. The reception of a "dominant" bit instead of a "recessive" bit during the transmission of a passive error flag, during the stuffed bit stream of the arbitration field or during the acknowledge slot, is not interpreted as a bit error.
- Stuff error
  A stuff error is detected, if the bit level after 6 consecutive bit times has not changed in a message field that has to be coded according to the bit stuffing method.
- Form Error
  A form error is detected, if a fixed frame bit (e.g., CRC delimiter, ACK delimiter) does not have the specified value. For a receiver a "dominant" bit during the last bit of End of Frame does NOT constitute a form error.
- Bit CRC Error
  A CRC error is detected if the remainder of the CRC calculation of a received CRC polynomial is non-zero.
- Acknowledgment Error
  An acknowledgment error is detected whenever a transmitting node does not get an acknowledgment from any other node (i.e., when the transmitter does not receive a "dominant" bit during the ACK frame).

The device can be in one of three states with respect to error handling:

- Error active
  An error active unit can participate in bus communication and sends an active ("dominant") error flag.
- Error passive
  An error passive unit can participate in bus communication. However, if the unit detects an error it is not allowed to send an active error flag. The unit sends only a passive ("recessive") error flag. A device is error passive when the transmit error counter is greater than 127 or when the receive error counter is greater than 127. A device becoming error passive sends an active error flag. An error passive device becomes error active again when both transmit and receive error counter are less than 128.
- Bus off
  A unit that is "bus off" has the output drivers disabled, i.e., it does not participate in any bus activity. A device is bus off when the transmit error counter is greater than 255. A bus off device will become error active again in one of two ways depending on which mode is selected by the user through the Fault Confinement Mode select bit (FMOD) in the CAN Bus Control Register (CBUS). Setting the FMOD bit to "0" (default after power on reset) will select the Standard Fault Confinement mode. In this mode the device goes from "bus off" to "error active" after monitoring 128*11 recessive bits (including bus idle) on the bus. This mode has been implemented for compatibility reasons with existing solutions. Setting the FMOD bit to "1" will select the Enhanced Fault Confinement mode. In this mode the device goes from "bus off" to "error active" after monitoring 128 "good" messages, as indicated by the reception of 11 consecutive "recessive" bits including the End of Frame. The enhanced mode offers the advantage that a "bus off" device (i.e., a device with a serious fault) is not allowed to destroy any messages on the bus until other devices can transmit at least 128 messages. This is not guaranteed in the standard mode, where a defective device could seriously impact bus communication. When the device goes from "bus off" to "error active", both error counters will have the value "0".

In each CAN module there are two error counters to perform a sophisticated error management. The receive error counter (REC) is 7 bits wide and switches the device to the error passive state if it overflows. The transmit error counter (TEC) is 8 bits wide. If it is greater than 127, the device is switched to the error passive state. As soon as the TEC overflows, the device is switched bus-off, i.e., it does not participate in any bus activity.

Basic CAN Concepts (Continued)

The counters are modified by the device's hardware according to the following rules:

TABLE VII. Receive Error Counter Handling

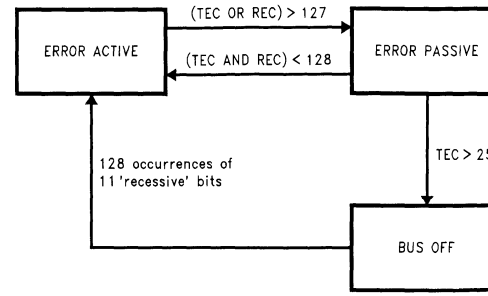
Condition	Receive Error Counter
A receiver detects a Bit Error during sending an active error flag.	Increment by 8
A receiver detects a "dominant" bit as the first bit after sending an error flag.	Increment by 8
After detecting the 14th consecutive "dominant" bit following an active error flag or overload flag or after detecting the 8th consecutive "dominant" bit following a passive error flag. After each sequence of additional 8 consecutive "dominant" bits.	Increment by 8
Any other error condition (stuff, frame, CRC, ACK).	Increment by 1
A valid reception or transmission.	Decrement by 1 if Counter is not 0

TABLE VIII. Transmit Error Counter Handling

Condition	Transmit Error Counter
A transmitter detects a Bit Error during sending an active error flag.	Increment by 8
After detecting the 14th consecutive "dominant" bit following an active error flag or overload flag or after detecting the 8th consecutive "dominant" bit following a passive error flag. After each sequence of additional 8 consecutive "dominant" bits.	Increment by 8
Any other error condition (stuff, frame, CRC, ACK).	Increment by 8
A valid reception or transmission.	Decrement by 1 if Counter is not 0

Special error handling for the TEC counter is performed in the following situations:

- A stuff error occurs during arbitration, when a transmitted "recessive" stuff bit is received as a "dorminant" bit. This does not lead to an incrementation of the TEC.
- An ACK-error occurs in an error passive device and no "dominant" bits are detected while sending the passive error flag. This does not lead to an incrementation of the TEC.
- If only one device is on the bus and this device transmits a message, it will get no acknowledgment. This will be detected as an error and message will be repeated. When the device goes "error passive" and detects an acknowledge error, the TEC counter is not incremented. Therefore the device will not go from "error passive" to the "bus off" state due to such a condition.



TL/DD/12522–62

FIGURE 31. CAN Bus States

Figure 31 shows the connection of different bus states according to the error counters.

SYNCHRONIZATION

Every receiver starts with a "hard synchronization" on the falling edge of the SOF bit. One bit time consists of four bit segments: Synchronization segment, propagation segment, phase segment 1 and phase segment 2.

A falling edge of the data signal should be in the synchronization segment. This segment has the fixed length of one time quanta. To compensate for the various delays within a network, the propagation segment is used. Its length is programmable from 1 to 8 time quanta. Phase segment 1 and phase segment 2 are used to resynchronize during an active frame. The length of these segments is from 1 to 8 time quanta long.

Two types of synchronization are supported:

Hard synchronization is done with the falling edge on the bus while the bus is idle, which is then interpreted as the SOF. It restarts the internal logic.

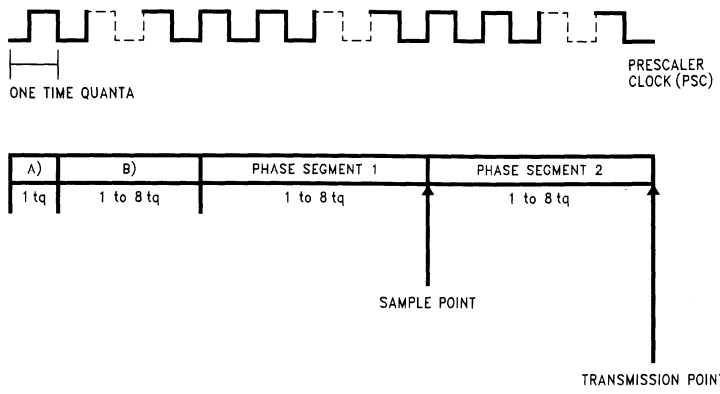
Soft synchronization is used to lengthen or shorten the bit time while a data or remote frame is received. Whenever a falling edge is detected in the propagation segment or in phase segment 1, the segment is lengthened by a specific value, the resynchronization jump width (see *Figure 33*).

If a falling edge lies in the phase segment 2 (as shown in *Figure 33*) it is shortened by the resynchronization jump width. Only one resynchronization is allowed during one bit time. The sample point lies between the two phase segments and is the point where the received data is supposed to be valid. The transmission point lies at the end of phase segment 2 to start a new bit time with the synchronization segment.

Note 1: The resynchronization jump width (RJW) is automatically determined from the programmed value of PS. If a soft resynchronization is done during phase segment 1 or the propagation segment, then RJW will either be equal to 4 internal CAN clocks (CKI/(1 + *divider*)) or the programmed value of PS, whichever is less. PS2 will never be shorter than 1 internal CAN clock.

Note 2: (PS1—BTL settings any PSC setting) The PS1 of the BTL should always be programmed to values greater than 1. To allow device resynchronization for positive and negative phase errors on the bus. (if PS1 is programmed to one, a bit time could only be lengthened and never shortened which basically disables half of the synchronization).

Basic CAN Concepts (Continued)



A) Synchronization segment
B) Propagation segment

FIGURE 32. Bit Timing

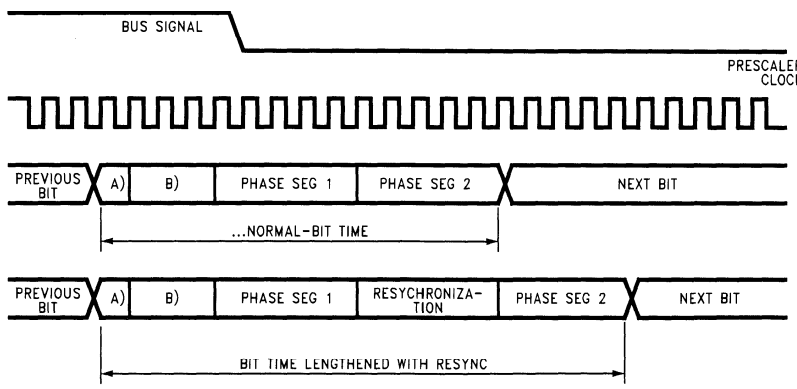


FIGURE 33. Resynchronization 1

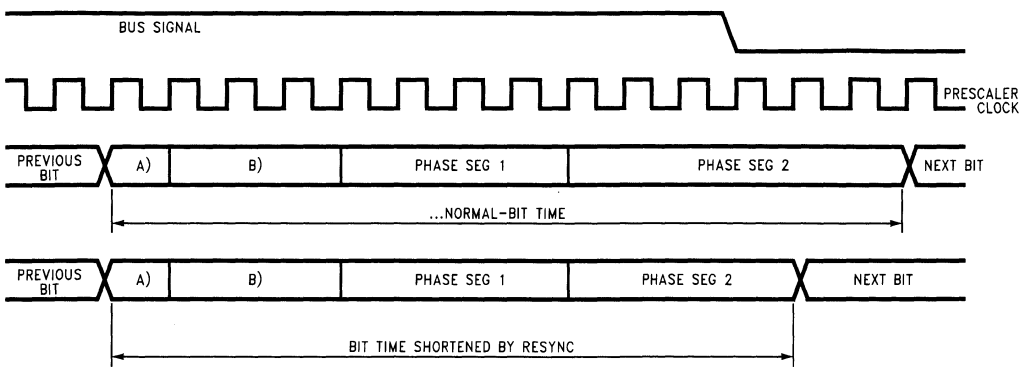


FIGURE 34. Resynchronization 2

Comparators

The device has two differential comparators. Port L is used for the comparators. The output of the comparators is multiplexed out to two pins. The following are the Port L assignments:

L0 Comparator 1 positive input
L1 Comparator 1 negative input
L2 Comparator 1 output
L3 Comparator 2 negative input
L4 Comparator 2 positive input
L5 Comparator 2 negative input
L6 comparator 2 output

Additionally the comparator output can be connected internally to the L-Port pin of the respective positive input and thereby generate an interrupt using the L-Port interrupt structure (neg/pos. edge, enable/disable).

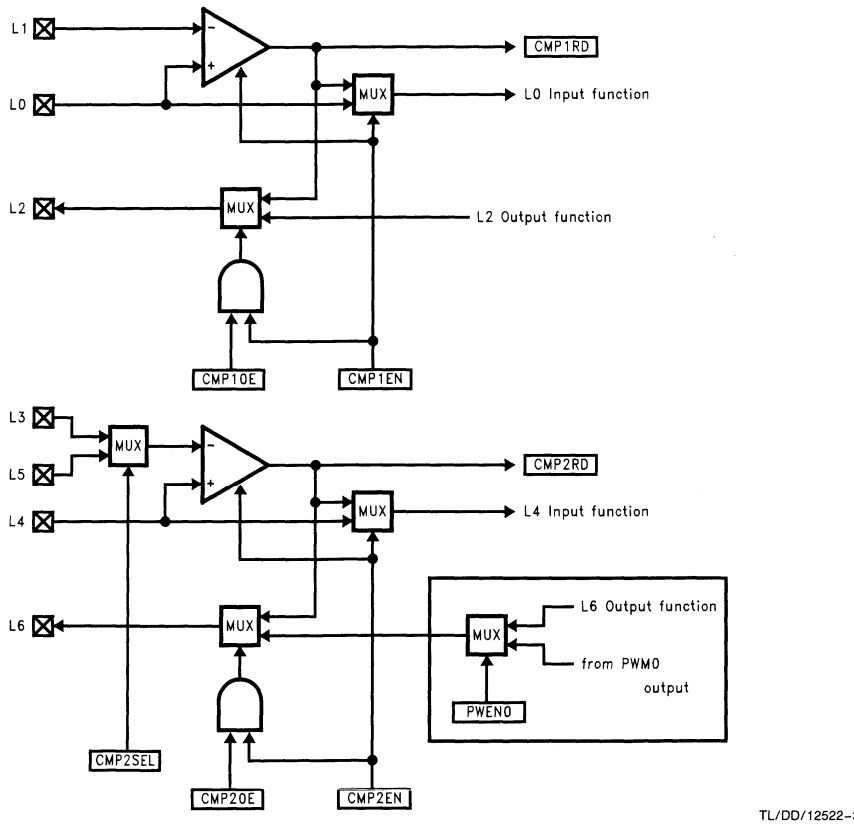
Note that in *Figure 32*, pin L6 has a second alternate function of supporting the PWM0 output. The comparator 2 output MUST be disabled in order to use PWM0 output on L6.

Figure 32 shows the Comparator Block Diagram.

COMPARATOR CONTROL REGISTER (CMPLS) (00D3)

These bits reside in the Comparator Register

CMP2 SEL	CMP2 OE	CMP2 RD	CMP2 EN	CMP1 OE	CMP1 RD	CMP1 EN	un- used
Bit 7							Bit 0



TL/DD/12522–36

Note: The SHADED area shows logic from PWM Timer. Comparator 2 output (CMP2OE) must be disabled in order to use PWM0 output.

FIGURE 35. Comparator Block

Comparators (Continued)

The register contains the following bits:

CMP1EN Enables comparator 1 ("1" = enable). If comparator 1 is disabled the associated L-pins can be used as standard I/O.

CMP1RD Reads comparator 1 output internally (CMP1EN = 1) Read-only, reads as a "0" if comparator not enabled.

CMP1OE Enables comparator 1 output ("1" = enable), CMP1EN bit must be set to enable this function.

CMP2EN Enables comparator 2 ("1" = enable). If comparator 2 is disabled the associated L-pins can be used as standard 1/O.

CMP2RD Reads comparator 2 output internally (CMP2EN = 1) Read-only, reads as a "0" if comparator not enabled.

CMP2OE Enables comparator 2 output ("1" = enable), CMP2EN bit must be set to enable this function.

CMP2SEL Selects which L port pin to use for comparator2 negative input. (CMP2SEL = 0 selects L5; CMP2SEL = 1 selects pin L3).

The Comparator Select/Control bits are cleared on RESET (the comparator is disabled). To save power, the program should also disable the comparator before the device enters the HALT mode.

The Comparator rise and fall times are symmetrical. The user program must set up the Configuration and Data registers of the L port correctly for comparator Inputs/Output.

Detection of Illegal Conditions

The device can detect various illegal conditions resulting from coding errors, transient noise, power supply voltage drops, runaway programs, etc.

Reading of undefined ROM gets zeroes. The opcode for software interrupt is zero. If the program fetches instructions from undefined ROM, this will force a software interrupt, thus signaling that an illegal condition has occurred.

The subroutine stack grows down for each call (jump to subroutine), interrupt, or PUSH, and grows up for each return or POP. The stack pointer is initialized to RAM location 02F Hex during reset. Consequently, if there are more returns than calls, the stack pointer will point to addresses 030 and 031 Hex (which are undefined RAM). Undefined RAM from addresses 030 to 03F Hex is read as all 1's, which in turn will cause the program to return to address 7FFF Hex. This is an undefined ROM location and the instruction fetched (all 0's) from this location will generate a software interrupt signaling an illegal condition.

Thus, the chip can detect the following illegal conditions:

1. Executing from undefined ROM.
2. Over "POP"ing the stack by having more returns than calls.

When the software interrupt occurs, the user can re-initialize the stack pointer and do a recovery procedure before restarting (this recovery program is probably similar to that following reset, but might not contain the same program initialization procedures).

MICROWIRE/PLUS

MICROWIRE/PLUS is a serial synchronous communications interface. The MICROWIRE/PLUS capability enables the device to interface with any of National Semiconductor's MICROWIRE peripherals (i.e., A/D converters, display drivers, E2PROMs etc.) and with other microcontrollers which support the MICROWIRE interface. It consists of an 8-bit serial shift register (SIO) with serial data input (SI), serial data output (SO) and serial shift clock (SK). *Figure 36* shows a block diagram of the MICROWIRE/PLUS logic.

The shift clock can be selected from either an internal source or an external source. Operating the MICROWIRE/PLUS arrangement with the internal clock source is called the Master mode of operation. Similarly, operating the MICROWIRE arrangement with an external shift clock is called the Slave mode of operation.

The CNTRL register is used to configure and control the MICROWIRE/PLUS mode. To use the MICROWIRE/PLUS, the MSEL bit in the CNTRL register is set to one. In the master mode the SK clock rate is selected by the two bits, SL0 and SL1, in the CNTRL register. Table VIII details the different clock rates that may be selected.

MICROWIRE/PLUS OPERATION

Setting the BUSY bit in the PSW register causes the MICROWIRE/PLUS to start shifting the data. It gets reset when eight data bits have been shifted. The user may reset the BUSY bit by software to allow less than 8 bits to shift. If enabled, an interrupt is generated when eight data bits have been shifted. The device may enter the MICROWIRE/PLUS mode either as a Master or as a Slave. *Figure 37* shows how two COP888 family microcontrollers and several peripherals may be interconnected using the MICROWIRE/PLUS arrangements.

Warning:

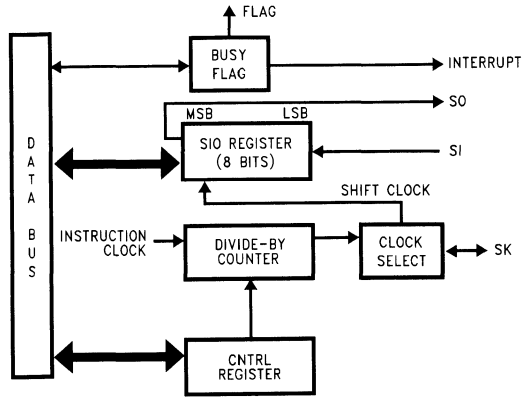
The SIO register should only be loaded when the SK clock is low. Loading the SIO register while the SK clock is high will result in undefined data in the SIO register. SK clock is normally low when not shifting.

Setting the BUSY flag when the input SK clock is high in the MICROWIRE/PLUS slave mode may cause the current SK clock for the SIO shift register to be narrow. For safety, the BUSY flag should only be set when the input SK clock is low.

MICROWIRE/PLUS Master Mode Operation

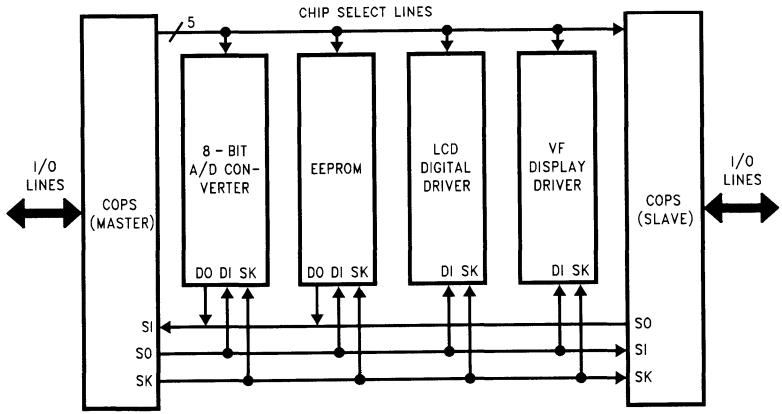
In the MICROWIRE/PLUS Master mode of operation the shift clock (SK) is generated internally. The MICROWIRE Master always initiates all data exchanges. The MSEL bit in the CNTRL register must be set to enable the SO and SK functions onto the G Port. The SO and SK pins must also be selected as outputs by setting appropriate bits in the Port G configuration register. Table IX summarizes the bit settings required for Master or Slave mode of operation.

MICROWIRE/PLUS (Continued)



TL/DD/12522–37

FIGURE 36. MICROWIRE/PLUS Block Diagram



TL/DD/12522–38

FIGURE 37. MICROWIRE/PLUS Application

MICROWIRE/PLUS (Continued)

TABLE IX. MICROWIRE/PLUS Master Mode Clock Selection

SL1	SL0	SK
0	0	$2 \times t_C$
0	1	$4 \times t_C$
1	x	$8 \times t_C$

Where t_C is the instruction cycle clock

MICROWIRE/PLUS Slave Mode Operation

In the MICROWIRE/PLUS Slave mode of operation the SK clock is generated by an external source. Setting the MSEL bit in the CNTRL register enables the SO and SK functions onto the G Port. The SK pin must be selected as an input and the SO pin is selected as an output pin by setting and resetting the appropriate bit in the Port G configuration register. Table VI summarizes the settings required to enter the Slave mode of operation.

The user must set the BUSY flag immediately upon entering the Slave mode. This will ensure that all data bits sent by the Master will be shifted properly. After eight clock pulses the BUSY flag will be cleared and the sequence may be repeated.

Alternate SK Phase Operation

The device allows either the normal SK clock or an alternate phase SK clock to shift data in and out of the SIO register. In both the modes the SK is normally low. In the normal mode data is shifted in on the rising edge of the SK clock and the data is shifted out on the falling edge of the SK clock. The SIO register is shifted on each falling edge of the SK clock in the normal mode. In the alternate SK phase mode the SIO register is shifted on the rising edge of the SK clock.

A control flag, SKSEL, allows either the normal SK clock or the alternate SK clock to be selected. Resetting SKSEL causes the MICROWIRE/PLUS logic to be clocked from the normal SK signal. Setting the SKSEL flag selects the alternate SK clock. The SKSEL is mapped into the G6 configuration bit. The SKSEL flag will power up in the reset condition, selecting the normal SK signal.

TABLE X. MICROWIRE/PLUS Mode Selection

G4 (SO) Config. Bit	G5 (SK) Config. Bit	G4 Fun.	G5 Fun.	Operation
1	1	SO	Int. SK	MICROWIRE/PLUS Master
0	1	TRI-STATE	Int. SK	MICROWIRE/PLUS Master
1	0	SO	Ext. SK	MICROWIRE/PLUS Slave
0	0	TRI-STATE	Ext. SK	MICROWIRE/PLUS Slave

This table assumes that the control flag MSEL is set.

Memory Map

All RAM, ports and registers (except A and PC) are mapped into data memory address space.

Address	Contents
00 to 2F	On-Chip RAM bytes (48 bytes)
30 to 7F	Unused RAM Address Space (Reads As All Ones)
80 to 9F	Unused RAM Address Space (Reads Undefined Data)
A0	PSCAL, PWM timer Prescaler Register
A1	RLON, PWM timer On-Time Register
A2	PWMCON, PWM Control Register
B0	TXD1, Transmit Data
B1	TXD2, Transmit 2 Data
B2	TDLC, Transmit Data Length Code and Identifier Low
B3	TID, Transmit Identifier High
B4	RXD1, Receive Data 1
B5	RXD2, Receive Data 2
B6	RIDL, Receive Data Length Code
B7	RID, Receive Identify High
B8	CSCAL, CAN Prescaler
B9	CTIM, Bus Timing Register
BA	CBUS, Bus Control Register
BB	TCNTL, Transmit/Receive Control Register
BC	RTSTAT Receive/Transmit Status Register
BD	TEC, Transmit Error Count Register
BE	REC, Receive Error Count Register
BF	Reserved
C0 to C7	Reserved
C8	WKEDG, MIWU Edge Select Register
C9	WKEN, MIWU Enable Register
CA	WKPND, MIWU Pending Register
CB	Reserved
CC	Reserved
CD to CF	Reserved

Memory Map (Continued)

Address	Contents
D0	PORTLD, Port L Data Register
D1	PORTLC, Port L Configuration Register
D2	PORTLP, Port L Input Pins (Read Only)
D3	CMPSL, Comparator Control Register
D4	PORTGD, Port G Data Register
D5	PORTGC, Port G Configuration Register
D6	PORTGP, Port G Input Pins (Read Only)
D7 to DB	Reserved
DC	PORTD, Port D Output Register
DD to DF	Reserved for Port D
E0–E5	Reserved
E6	T1RBLO, Timer T1 Autoload Register Lower Byte
E7	T1RBHI, Timer T1 Autoload Register Upper Byte
E8	ICNTRL, Interrupt Control Register
E9	SIOR, MICROWIRE/PLUS Shift Register
EA	TMR1LO, Timer T1 Lower Byte
EB	TMR1HI, Timer T1 Upper Byte
EC	T1RALO, Timer T1 Autoload Register Lower Byte
ED	T1RAHI, Timer T1 Autoload Register T1RA Upper Byte
EE	CNTRL, Control Register
EF	PSW, Processor Status Word Register
F0 to FB	On-Chip RAM Mapped as Registers
FC	X Register
FD	SP Register
FE	B Register
FF	Reserved (Note 1)

Note: Reading memory locations 30–7F Hex will return all ones. Reading other unused memory locations will return undefined data.

Note 1: In devices with more than 128 bytes of RAM, location 0FF is used as the Segment register to switch between different Segments of RAM memory. In this device location 0FF can be used as a general purpose, on-chip RAM mapped register. However, the user is advised that caution should be taken in porting software utilizing this memory location to a chip with more than 128 bytes of RAM.

Addressing Modes

There are ten addressing modes, six for operand addressing and four for transfer of control.

OPERAND ADDRESSING MODES

Register Indirect

This is the "normal" addressing mode. The operand is the data memory addressed by the B pointer or X pointer.

Register Indirect (with auto post Increment or decrement of pointer)

This addressing mode is used with the LD and X instructions. The operand is the data memory addressed by the B pointer or X pointer. This is a register indirect mode that automatically post increments or decrements the B or X register after executing the instruction.

Direct

The instruction contains an 8-bit address field that directly points to the data memory for the operand.

Immediate

The instruction contains an 8-bit immediate field as the operand.

Short Immediate

This addressing mode is used with the Load B Immediate instruction. The instruction contains a 4-bit immediate field as the operand.

Indirect

This addressing mode is used with the LAID instruction. The contents of the accumulator are used as a partial address (lower 8 bits of PC) for accessing a data operand from the program memory.

TRANSFER OF CONTROL ADDRESSING MODES

Relative

This mode is used for the JP instruction, with the instruction field being added to the program counter to get the new program location. JP has a range from −31 to +32 to allow a 1-byte relative jump (JP + 1 is implemented by a NOP instruction). There are no "pages" when using JP, since all 15 bits of PC are used.

Absolute

This mode is used with the JMP and JSR instructions, with the instruction field of 12 bits replacing the lower 12 bits of the program counter (PC). This allows jumping to any location in the current 4k program memory segment.

Absolute Long

This mode is used with the JMPL and JSRL instructions, with the instruction field of 15 bits replacing the entire 15 bits of the program counter (PC). This allows jumping to any location up to 32k in the program memory space.

Indirect

This mode is used with the JID instruction. The contents of the accumulator are used as a partial address (lower 8 bits of PC) for accessing a location in the program memory. The contents of this program memory location serve as a partial address (lower 8 bits of PC) for the jump to the next instruction.

Note: The VIS is a special case of the Indirect Transfer of Control addressing mode, where the double byte vector associated with the interrupt is transferred from adjacent addresses in the program memory into the program counter (PC) in order to jump to the associated interrupt service routine.

Instruction Set

Register and Symbol Definition

Registers	
A	8-Bit Accumulator Register
B	8-Bit Address Register
X	8-Bit Address Register
SP	8-Bit Stack Pointer Register
PC	15-Bit Program Counter Register
PU	Upper 7 Bits of PC
PL	Lower 8 Bits of PC
C	1-Bit of PSW Register for Carry
HC	1-Bit of PSW Register for Half Carry
GIE	1-Bit of PSW Register for Global Interrupt Enable
VU	Interrupt Vector Upper Byte
VL	Interrupt Vector Lower Byte

Symbols	
[B]	Memory Indirectly Addressed by B Register
[X]	Memory Indirectly Addressed by X Register
MD	Direct Addressed Memory
Mem	Direct Addressed Memory or [B]
Meml	Direct Addressed Memory or [B] or Immediate Data
Imm	8-Bit Immediate Data
Reg	Register Memory: Addresses F0 to FF (Includes B, X and SP)
Bit	Bit Number (0 to 7)
←	Loaded with
↔	Exchanged with

Instruction Set (Continued)

INSTRUCTION SET

ADD	A,Meml	ADD	A ← A + Meml
ADC	A,Meml	ADD with Carry	A ← A + Meml + C, C ← Carry, HC ← Half Carry
SUBC	A,Meml	Subtract with Carry	A ← A − Meml + C, C ← Carry, HC ← Half Carry
AND	A,Meml	Logical AND	A ← A and Meml
ANDSZ	A,Imm	Logical AND Immed., Skip if Zero	Skip next if (A and Imm) = 0
OR	A,Meml	Logical OR	A ← A or Meml
XOR	A,Meml	Logical EXclusive OR	A ← A xor Meml
IFEQ	MD,Imm	IF EQual	Compare MD and Imm, Do next if MD = Imm
IFEQ	A,Meml	IF EQual	Compare A and Meml, Do next if A = Meml
IFNE	A,Meml	IF Not Equal	Compare A and Meml, Do next if A ≠ Meml
IFGT	A,Meml	IF Greater Than	Compare A and Meml, Do next if A > Meml
IFBNE	#	If B Not Equal	Do next if lower 4 bits of B ≠ Imm
DRSZ	Reg	Decrement Reg., Skip if Zero	Reg ← Reg − 1, Skip if Reg = 0
SBIT	#,Mem	Set BIT	1 to bit, Mem (bit = 0 to 7 immediate)
RBIT	#,Mem	Reset BIT	0 to bit, Mem
IFBIT	#,Mem	IF BIT	If bit in A or Mem is true do next instruction
RPND		Reset PeNDing Flag	Reset Software Interrupt Pending Flag
X	A,Mem	EXchange A with Memory	A ⟷ Mem
X	A,[X]	EXchange A with Memory [X]	A ⟷ [X]
LD	A,Meml	LoaD A with Memory	A ← Meml
LD	A,[X]	LoaD A with Memory [X]	A ← [X]
LD	B,Imm	LoaD B with Immed.	B ← Imm
LD	Mem,Imm	LoaD Memory Immed.	Mem ← Imm
LD	Reg,Imm	LoaD Register Memory Immed.	Reg ← Imm
X	A, [B±]	EXchange A with Memory [B]	A ⟷ [B], (B ← B ± 1)
X	A, [X±]	EXchange A with Memory [X]	A ⟷ [X], (X ← ± 1)
LD	A, [B±]	LoaD A with Memory [B]	A ← [B], (B ← B ± 1)
LD	A, [X±]	LoaD A with Memory [X]	A ← [X], (X ← X ± 1)
LD	[B±],Imm	LoaD Memory [B] Immed.	[B] ← Imm, (B ← B ± 1)
CLR	A	CLeaR A	A ← 0
INC	A	INCrement A	A ← A + 1
DEC	A	DECrementA	A ← A − 1
LAID		Load A InDirect from ROM	A ← ROM (PU,A)
DCOR	A	Decimal CORrect A	A ← BCD correction of A (follows ADC, SUBC)
RRC	A	Rotate A Right thru C	C → A7 → . . . → A0 → C
RLC	A	Rotate A Left thru C	C ← A7 ← . . . ← A0 ← C
SWAP	A	SWAP nibbles of A	A7 . . . A4 ⟷ A3 . . . A0
SC		Set C	C ← 1, HC ← 1
RC		Reset C	C ← 0, HC ← 0
IFC		IF C	IF C is true, do next instruction
IFNC		IF Not C	If C is not true, do next instruction
POP	A	POP the stack into A	SP ← SP + 1, A ← [SP]
PUSH	A	PUSH A onto the stack	[SP] ← A, SP ← SP − 1
VIS		Vector to Interrupt Service Routine	PU ← [VU], PL ← [VL]
JMPL	Addr.	Jump absolute Long	PC ← ii (ii = 15 bits, 0k to 32k)
JMP	Addr.	Jump absolute	PC9 . . . 0 ← i (i = 12 bits)
JP	Disp.	Jump relative short	PC ← PC + r (r is −31 to +32, except 1)
JSRL	Addr.	Jump SubRoutine Long	[SP] ← PL, [SP−1] ← PU,SP−2, PC ← ii
JSR	Addr.	Jump SubRoutine	[SP] ← PL, [SP−1] ← PU,SP−2, PC9 . . . 0 ← i
JID		Jump InDirect	PL ← ROM (PU,A)
RET		RETurn from subroutine	SP + 2, PL ← [SP], PU ← [SP−1]
RETSK		RETurn and SKip	SP + 2, PL ← [SP],PU ← [SP−1]
RETI		RETurn from Interrupt	SP + 2, PL ← [SP],PU ← [SP−1],GIE ← 1
INTR		Generate an Interrupt	[SP] ← PL, [SP−1] ← PU, SP−2, PC ← 0FF
NOP		No OPeration	PC ← PC + 1

Instruction Execution Time

Most instructions are single byte (with immediate addressing mode instructions taking two bytes).

Most single byte instructions take one cycle time to execute.

Skipped instructions require x number of cycles to be skipped, where x equals the number of bytes in the skipped instruction opcode.

See the BYTES and CYCLES per INSTRUCTION table for details.

Bytes and Cycles per Instruction

The following table shows the number of bytes and cycles for each instruction in the format of byte/cycle.

Arithmetic and Logic Instructions

	[B]	Direct	Immed.
ADD	1/1	3/4	2/2
ADC	1/1	3/4	2/2
SUBC	1/1	3/4	2/2
AND	1/1	3/4	2/2
OR	1/1	3/4	2/2
XOR	1/1	3/4	2/2
IFEQ	1/1	3/4	2/2
IFGT	1/1	3/4	2/2
IFBNE	1/1		
DRSZ		1/3	
SBIT	1/1	3/4	
RBIT	1/1	3/4	
IFBIT	1/1	3/4	

RPND	1/1

Instructions Using A and C

CLRA	1/1
INCA	1/1
DECA	1/1
LAID	1/3
DCORA	1/1
RRCA	1/1
RLCA	1/1
SWAPA	1/1
SC	1/1
RC	1/1
IFC	1/1
IFNC	1/1
PUSHA	1/3
POPA	1/3
ANDSZ	2/2

Transfer of Control Instructions

JMPL	3/4
JMP	2/3
JP	1/3
JSRL	3/5
JSR	2/5
JID	1/3
VIS	1/5
RET	1/5
RETSK	1/5
RETI	1/5
INTR	1/7
NOP	1/1

Memory Transfer Instructions

	Register Indirect		Direct	Immed.	Register Indirect Auto Incr. and Decr.		
	[B]	[X]			[B+, B−]	[X+, X−]	
X A,*	1/1	1/3	2/3		1/2	1/3	
LD A,*	1/1	1/3	2/3	2/2	1/2	1/3	
LD B, Imm				1/1			(IF B < 16)
LD B, Imm				2/3			(IF B > 15)
LD Mem, Imm	2/2		3/3		2/2		
LD Reg, Imm			2/3				
IFEQ MD, Imm			3/3				

* = > Memory location addressed by B or X or directly.

COP888 Family Opcode Table

UPPER NIBBLE																LOWER NIBBLE
F	E	D	C	B	A	9	8	7	6	5	4	3	2	1	0	
JP −15	JP −31	LD 0F0, #i	DRSZ 0F0	RRCA	RC	ADC A, #i	ADC A,[B]	IFBIT 0,[B]	ANDSZ A, #i	LD B,#0F	IFBNE 0	JSR x000–x0FF	JMP x000–x0FF	JP 17	INTR	0
JP −14	JP −30	LD 0F1, #i	DRSZ 0F1	*	SC	SUBC A, #i	SUB A,[B]	IFBIT 1,[B]	*	LD B,#0E	IFBNE 1	JSR x100–x1FF	JMP x100–x1FF	JP 18	JP + 2	1
JP −13	JP −29	LD 0F2, #i	DRSZ 0F2	X A, [X]	X A, [B]	IFEQ A, #i	IFEQ A,[B]	IFBIT 2,[B]	*	LD B,#0D	IFBNE 2	JSR x200–x2FF	JMP x200–x2FF	JP 19	JP + 3	2
JP −12	JP −28	LD 0F3, #i	DRSZ 0F3	X A, [X−]	X A, [B−]	IFGT A, #i	IFGT A,[B]	IFBIT 3,[B]	*	LD B,#0C	IFBNE 3	JSR x300–x3FF	JMP x300–x3FF	JP 20	JP + 4	3
JP −11	JP −27	LD 0F4, #i	DRSZ 0F4	VIS	LAID	ADD A, #i	ADD A,[B]	IFBIT 4,[B]	CLRA	LD B,#0B	IFBNE 4	JSR x400–x4FF	JMP x400–x4FF	JP 21	JP + 5	4
JP −10	JP −26	LD 0F5, #i	DRSZ 0F5	RPND	JID	AND A, #i	AND A,[B]	IFBIT 5,[B]	SWAPA	LD B,#0A	IFBNE 5	JSR x500–x5FF	JMP x500–x5FF	JP 22	JP + 6	5
JP −9	JP −25	LD 0F6, #i	DRSZ 0F6	X A,[X]	X A,[B]	XOR A, #i	XOR A,[B]	IFBIT 6,[B]	DCORA	LD B,#09	IFBNE 6	JSR x600–x6FF	JMP x600–x6FF	JP 23	JP + 7	6
JP −8	JP −24	LD 0F7, #i	DRSZ 0F7	*	*	OR A,#i	OR A,[B]	IFBIT 7,[B]	PUSHA	LD B,#08	IFBNE 7	JSR x700–x7FF	JMP x700–x7FF	JP 24	JP + 8	7
JP −7	JP −23	LD 0F8, #i	DRSZ 0F8	NOP	RLCA	LD A,#i	IFC	SBIT 0,[B]	RBIT 0,[B]	LD B,#07	IFBNE 8	JSR x800–x8FF	JMP x800–x8FF	JP 25	JP + 9	8
JP −6	JP −22	LD 0F9, #i	DRSZ 0F9	IFNE A,[B]	IFEQ Md,#i	IFNE A,#i	IFNC	SBIT 1,[B]	RBIT 1,[B]	LD B,#06	IFBNE 9	JSR x900–x9FF	JMP x900–x9FF	JP 26	JP + 10	9
JP −5	JP −21	LD 0FA, #i	DRSZ 0FA	LD A, [X]	LD A, [B]	LD [B], #i	INCA	SBIT 2,[B]	RBIT 2,[B]	LD B,#05	IFBNE 0A	JSR xA00–xAFF	JMP xA00–xAFF	JP 27	JP + 11	A
JP −4	JP −20	LD 0FB, #i	DRSZ 0FB	LD A, [X−]	LD A, [B−]	LD [B−], #i	DECA	SBIT 3,[B]	RBIT 3,[B]	LD B,#04	IFBNE 0B	JSR xB00–xBFF	JMP xB00–xBFF	JP 28	JP + 12	B
JP −3	JP −19	LD 0FC, #i	DRSZ 0FC	LD Md,#i	JMPL	X A,Md	POPA	SBIT 4,[B]	RBIT 4,[B]	LD B,#03	IFBNE 0C	JSR xC00–xCFF	JMP xC00–xCFF	JP 29	JP + 13	C
JP −2	JP −18	LD 0FD, #i	DRSZ 0FD	DIR	JSRL	LD A,Md	RETSK	SBIT 5,[B]	RBIT 5,[B]	LD B,#02	IFBNE 0D	JSR xD00–xDFF	JMP xD00–xDFF	JP 30	JP + 14	D
JP −1	JP −17	LD 0FE, #i	DRSZ 0FE	LD A,[X]	LD A,[B]	LD [B],#i	RET	SBIT 6,[B]	RBIT 6,[B]	LD B,#01	IFBNE 0E	JSR xE00–xEFF	JMP xE00–xEFF	JP 31	JP + 15	E
JP −0	JP −16	LD 0FF, #i	DRSZ 0FF	*	*	LD B,#i	RETI	SBIT 7,[B]	RBIT 7,[B]	LD B,#00	IFBNE 0F	JSR xF00–xFFF	JMP xF00–xFFF	JP 32	JP + 16	F

where,

i is the immediate data

Md is a directly addressed memory location

* is an unused opcode

Note: The opcode 60 Hex is also the opcode for IFBIT #i,A

Development Support

SUMMARY

- iceMASTER™: IM-COP8/400—Full feature in-circuit emulation for all COP8 products. A full set of COP8 Basic and Feature Family device and package specific probes are available.
- COP8 Debug Module: Moderate cost in-circuit emulation and development programming unit.
- COP8 Evaluation and Programming Unit: EPU-COP888GG—low cost In-circuit simulation and development programming unit.
- Assembler: COP8-DEV-IBMA. A DOS installable cross development Assembler, Linker, Librarian and Utility Software Development Tool Kit.
- C Compiler: COP8C. A DOS installable cross development Software Tool Kit.
- OTP/EPROM Programmer Support: Covering needs from engineering prototype, pilot production to full production environments.

iceMASTER (IM) IN-CIRCUIT EMULATION

The iceMASTER IM-COP8/400 is a full feature, PC based, in-circuit emulation tool developed and marketed by MetaLink Corporation to support the whole COP8 family of products. National is a resale vendor for these products.

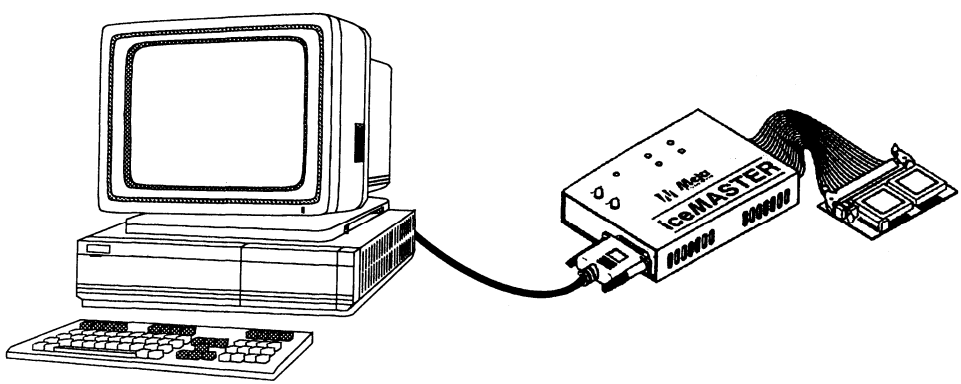
See *Figure 38* for configuration.

The iceMASTER IM-COP8/400 with its device specific COP8 Probe provides a rich feature set for developing, testing and maintaining product:

- Real-time in-circuit emulation; full 2.4V–5.5V operation range, full DC-10 MHz clock. Chip options are programmable or jumper selectable.
- Direct connection to application board by package compatible socket or surface mount assembly.
- Full 32 kbytes of loadable programming space that overlays (replaces) the on-chip ROM or EPROM. On-chip RAM and I/O blocks are used directly or recreated on the probe as necessary.
- Full 4k frame synchronous trace memory. Address, instruction, and 8 unspecified, circuit connectable trace lines. Display can be HLL source (e.g., C source), assembly or mixed.
- A full 64k hardware configurable break, trace on, trace off control, and pass count increment events.
- Tool set integrated interactive symbolic debugger—supports both assembler (COFF) and C Compiler (.COD) linked object formats.
- Real time performance profiling analysis; selectable bucket definition.
- Watch windows, content updated automatically at each execution break.
- Instruction by instruction memory/register changes displayed on source window when in single step operation.
- Single base unit and debugger software reconfigurable to support the entire COP8 family; only the probe personality needs to change. Debugger software is processor customized, and reconfigured from a master model file.
- Processor specific symbolic display of registers and bit level assignments, configured from master model file.
- Halt/Idle mode notification.
- On-line HELP customized to specific processor using master model file.
- Includes a copy of COP8-DEV-IBMA assembler and linker SDK.

IM Order Information

Base Unit	
IM-COP8/400-1	iceMASTER base unit, 110V power supply
IM-COP8/400-2	iceMASTER base unit, 220V power supply
iceMASTER Probe	
MHW-884BC28DWPC	28 DIP
28 DIP to 28 SO Adapter	
MHW-SOIC 28	28 SO



TL/DD/12522–43

FIGURE 38. COP8 iceMASTER Environment

Development Support (Continued)

iceMASTER DEBUG MODULE (DM)

The iceMASTER Debug Module is a PC based, combination in-circuit emulation tool and COP8 based OTP/EPROM programming tool developed and marketed by MetaLink Corporation to support the whole COP8 family of products. National is a resale vendor for these products.

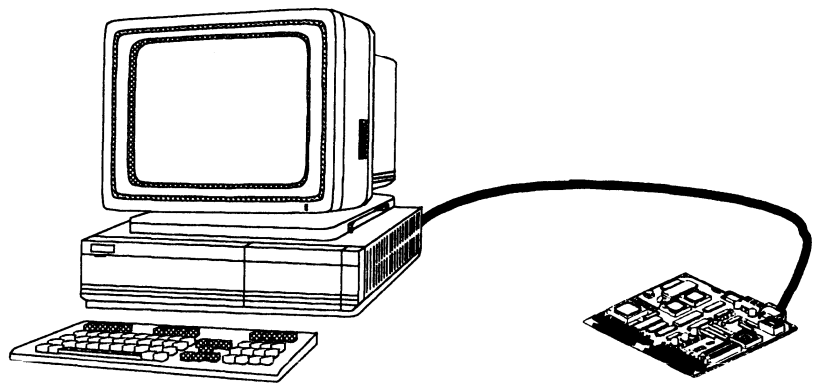
See *Figure 39* for configuration.

The iceMASTER Debug Module is a moderate cost development tool. It has the capability of in-circuit emulation for a specific COP8 microcontroller and in addition serves as a programming tool for COP8 OTP and EPROM product families. Summary of features is as follows:

- Real-time in-circuit emulation; full operating voltage range operation, full DC-10 MHz clock.
- All processor I/O pins can be cabled to an application development board with package compatible cable to socket and surface mount assembly.
- Full 32 kbytes of loadable programming space that overlays (replaces) the on-chip ROM or EPROM. On-chip RAM and I/O blocks are used directly or recreated as necessary.
- 100 frames of synchronous trace memory. The display can be HLL source (C source), assembly or mixed. The most recent history prior to a break is available in the trace memory.
- Configured break points; uses INTR instruction which is modestly intrusive.
- Software—only supported features are selectable.
- Tool set integrated interactive symbolic debugger—supports both assembler (COFF) and C Compiler (.COD) SDK linked object formats.
- Instruction by instruction memory/register changes displayed when in single step operation.
- Debugger software is processor customized, and reconfigured from a master model file.
- Processor specific symbolic display of registers and bit level assignments, configured from master model file.
- Halt/Idle mode notification.
- Programming menu supports full product line of programmable OTP and EPROM COP8 products. Program data is taken directly from the overlay RAM.
- Programming of 44 PLCC and 68 PLCC parts requires external programming. adapters.
- Includes wallmount power supply.
- On-board V_{PP} generator from 5V input or connection to external supply supported. Requires V_{PP} level adjustment per the family programming specification (correct level is provided on an on-screen pop-down display).
- On-line HELP customized to specific processor using master model file.
- Includes a copy of COP8-DEV-IBMA assembler and linker SDK.

DM Order Information

Debug Model Unit	
COP8-DM/888BC	
Cable Adapters	
DM-COP8/28D	28 DIP
28 DIP to 28 SO Adapter	
DM-COP8/28D-SO	28 SO



TL/DD/12522-44

FIGURE 39. COP8-DM Environment

Development Support (Continued)

COP8 ASSEMBLER/LINKER SOFTWARE DEVELOPMENT TOOL KIT

National Semiconductor offers a relocateable COP8 macro cross assembler, linker, librarian and utility software development tool kit. Features are summarized as follows:

- Basic and Feature Family instruction set by "device" type.
- Nested macro capability.
- Extensive set of assembler directives.
- Supported on PC/DOS platform.
- Generates National standard COFF output files.
- Integrated Linker and Librarian.
- Integrated utilities to generate ROM code file outputs.
- DUMPCOFF utility.

This product is integrated as a part of MetaLink tools as a development kit, fully supported by the MetaLink debugger. It may be ordered separately or it is bundled with the MetaLink products at no additional cost.

Order Information

Assembler SDK	
COP8-DEV-IBMA	Assembler SDK on installable 3.5" PC/DOS Floppy Disk Drive format. Periodic upgrades and most recent version is available on National's BBS and Internet.

COP8 C COMPILER

A C Compiler is developed and marketed by Byte Craft Limited. The COP8C compiler is a fully integrated development tool specifically designed to support the compact embedded configuration of the COP8 family of products.

Features are summarized as follows:

- ANSI C with some restrictions and extensions that optimize development for the COP8 embedded application.
- BITS data type extension. Register declaration #pragma with direct bit level definitions.
- C language support for interrupt routines.
- Expert system, rule based code geration and optimization.
- Performs consistency checks against the architectural definitions of the target COP8 device.
- Generates program memory code.
- Supports linking of compiled object or COP8 assembled object formats.
- Global optimization of linked code.
- Symbolic debug load format fully sourced level supported by the MetaLink debugger.

INDUSTRY WIDE OTP/EPROM PROGRAMMING SUPPORT

Programming support, in addition to the MetaLink development tools, is provided by a full range of independent approved vendors to meet the needs from the engineering laboratory to full production.

Approved List

Manufacturer	North America	Europe	Asia
BP Microsystems	(800) 225-2102 (713) 688-4600 Fax: 688-0920	+49-8152-4183 +49-8856-932616	+852-234-16611 +852-2710-8121
Data I/O	(800) 426-1045 (206) 881-6444 Fax: (206) 882-1043	+44-0734-440011	Call North America
HI-LO	(510) 623-8860	Call Asia	+886-2-764-0215 Fax: +886-2-756-6403
ICE Technology	(800) 624-8949 (919) 430-7915	+44-1226-767404 Fax: 0-1226-370-434	
MetaLink	(800) 638-2423 (602) 926-0797 Fax: (602) 693-0681	+49-80 9156 96-0 Fax: +49-80 9123 86	+852-737-1800
Systems General	(408) 263-6667	+41-1-9450300	+886-2-917-3005 Fax: +886-2-911-1283
Needhams	(916) 924-8037 Fax: (916) 924-8065		

Development Support (Continued)

AVAILABLE LITERATURE

For more information, please see the COP8 Basic Family User's Manual, Literature Number 620895, COP8 Feature Family User's Manual, Literature Number 620897 and National's Family of 8-bit Microcontrollers COP8 Selection Guide, Literature Number 630009.

DIAL-A-HELPER SERVICE

Dial-A-Helper is a service provided by the Microcontroller Applications group. The Dial-A-Helper is an Electronic Information System that may be accessed as a Bulletin Board System (BBS) via data modem, as an FTP site on the Internet via standard FTP client application or as an FTP site on the Internet using a standard Internet browser such as Netscape or Mosaic.

The Dial-A-Helper system provides access to an automated information storage and retrieval system . The system capabilities include a MESSAGE SECTION (electronic mail, when accessed as a BBS) for communications to and from the Microcontroller Applications Group and a FILE SECTION which consists of several file areas where valuable application software and utilities could be found.

DIAL-A-HELPER BBS via a Standard Modem

Modem:	CANADA/U.S.:	(800) NSC-MICRO
		(800) 672-6427
	EUROPE:	(+49) 0-8141-351332
	Baud:	14.4k
	Set-Up:	Length: 8-Bit
		Parity: None
		Stop Bit: 1
	Operation:	24 Hours, 7 Days

DIAL-A-HELPER via FTP

ftp nscmicro.nsc.com
user: anonymous
password: username@yourhost.site.domain

DIAL-A-HELPER via WorldWide Web Browser

ftp://nscmicro.nsc.com

National Semiconductor on the WorldWide Web

See us on the WorldWide Web at: http://www.national.com

CUSTOMER RESPONSE CENTER

Complete product information and technical support is available from National's customer response centers.

CANADA/U.S.:	Tel:	(800) 272-9959
	email:	support@tevm2.nsc.com
EUROPE:	email:	europe.support@nsc.com
	Deutsch Tel:	+49 (0) 180-530 85 85
	English Tel:	+49 (0) 180-532 78 32
	Français Tel:	+49 (0) 180-532 93 58
	Italiano Tel:	+49 (0) 180-534 16 80
JAPAN:	Tel:	+81-043-299-2309
S.E. ASIA:	Beijing Tel:	(+86) 10-6856-8601
	Shanghai Tel:	(+86) 21-6415-4092
	Hong Kong Tel:	(+852) 2737-1600
	Korea Tel:	(+82) 2-3771-6909
	Malaysia Tel:	(+60-4) 644-9061
	Singapore Tel:	(+65) 255-2226
	Taiwan Tel:	+886-2-521-3288
AUSTRALIA:	Tel:	(+61) 3-9558-9999
INDIA:	Tel:	(+91) 80-559-9467

National Semiconductor

COP87L88EB/COP87L89EB
8-Bit One Time Programmable (OTP) Microcontroller with CAN Interface, A/D and UART

General Description

The COP87L88EB/COP87L89EB are members of the COP8™ microcontroller feature family, which uses an 8-bit core architecture. They are pin and software compatible to the mask ROM COP888EB product family. (Continued)

Key Features

- CAN bus interface, with Software Power save mode
- 8-bit A/D Converter with 8 channels
- Fully buffered UART
- Multi-input wake up (MIWU) on both Port L and M
- SPI Compatible Master/Slave Interface
- 8096 bytes of on-board OTP EPROM with security feature
- 192 bytes of on-board RAM

Additional Peripheral Features

- Idle timer (programmable)
- Two 16-bit timer, with two 16-bit registers supporting
  - — Processor independent PWM mode
  - — External Event counter mode
  - — Input capture mode
- WATCHDOG™ and Clock Monitor
- MICROWIRE/PLUS™ serial I/O

I/O Features

- Memory mapped I/O
- Software selectable I/O options (TRI-STATE® outputs, Push pull outputs, Weak pull up input, High impedance input)
- Schmitt trigger inputs on Port G, L and M
- Packages: 44 PLCC with 31 I/O pins
  68 PLCC with 58 I/O pins

CPU/Instruction Set Features

- 1 μs instruction cycle time
- Fourteen multi-sourced vectored interrupts servicing
  - — External interrupt
  - — Idle Timer T0
  - — Timers (T1 and T2) (4 Interrupts)
  - — MICROWIRE/PLUS and SPI
  - — Multi-input Wake up
  - — Software Trap
  - — CAN interface (3 interrupts)
  - — UART (2 Inputs)
- Versatile easy to use instruction set
- 8-bit stacker pointer (SP) (Stack in RAM)
- Two 8-bit RegisterR Indirect Memory Pointers (B, X)

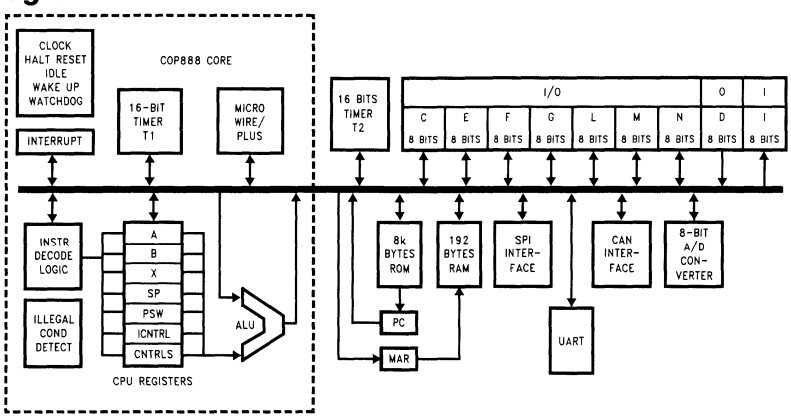
Fully Static CMOS

- Low current drain (typically < 1 μA)
- Two power saving modes: HALT, IDLE
- Single supply operation: 4.5V to 5.5V
- Temperature range: −40°C to +85°C

Development Support

- Emulation device for COP888EB
- Real time emulation and full program debug offered by MetaLink Development System

Block Diagram



TL/DD/12871-1

FIGURE 1. Block Diagram

General Description (Continued)

The devices are designed to perform complex embedded control applications such as those found in Automotive Control Applications, while providing control/diagnostic communications via the CAN bus interface. The devices comply with the basic CAN bus specification 2.0B (Passive). They are fully static devices fabricated using National's double metal silicon gate microCMOS technology. Efficient throughput is achieved through a regular efficient instruction set operating at a maximum of 1 μs instruction rate.

Basic Functional Description

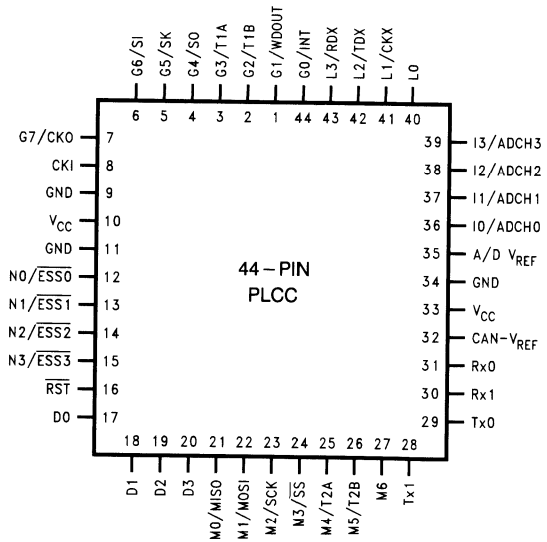
- CAN I/F—CAN serial bus interface block as described in the CAN specification part 2.0B (Passive)
  - — Interface rates up to 250k bit/s are supported utilizing standard message identifiers
- Programmable double buffered UART
- A/D—8-bit, 8 channel, 1-LSB Resolution, with improved Source Impedance and improved channel to channel cross talk immunity
- Multi-Input-Wake-Up (MIWU)—edge selectable wake-up and interrupt capability via input port and CAN interface (Port L, Port M and CAN I/F); supports Wake-Up capability on SPI, UART, and T2
- Port C—8-bit bi-directional I/O port
- Port D—8-bit Output port with high current drive capability (10 mA)
- Port E—8-bit bidirectional I/O
- Port F—8-bit bidirectional I/O
- Port G—8-bit bidirectional I/O port, including alternate functions for:
  - — MICROWIRE Input and Output
  - — Timer 1 Input or Output (Depending on mode selected)
  - — External Interrupt input
  - — WATCHDOG Output
- Port I—8-bit input port combining either digital input, or up to eight A/D input channels
- Port L—8-bit bidirectional I/O port, including alternate functions for:
  - — UART Transmit/Receive I/O
  - — Multi-input-wake up (MIWU on all pins)
- Port M—8-bit I/O port, with the following alternate function
  - — SPI Interface
  - — MIWU
  - — CAN Interface Wake-up (MSB)
  - — Timer 2 Input or Output (Depending on mode selected)
- Port N—8-bit bidirectional I/O
  - — SPI Slave Select Expander
- Two 16-bit multi-function Timer counters (T1 and T2) plus supporting registers
  - — (I/P Capture, PWM and Event Counting)
- Idle timer—Provides a basic time-base counter, (with interrupt) and automatic wake up from IDLE mode programmable
- MICROWIRE/PLUS—MICROWIRE serial peripheral interface, supporting both Master and Slave operation
- HALT and IDLE—Software programmable low current modes
  - — HALT—Processor stopped, Minimum current
  - — IDLE—Processor semi-active more than 60% power saving
- 8 kbytes ROM and 192 bytes of on board static RAM
- SPI Master/Slave interface includes 12 bytes Transmit and 12 bytes Receive FIFO Buffers. Operates up to 1M Bit/S
- On board programmable WATCHDOG and CLOCK Monitor

Applications

- Automobile Body Control and Comfort System
- Integrated Driver Informaiton Systems
- Steering Wheel Control
- Car Radio Control Panel
- Sensor/Actuator Applications in Automotive and Industrial Control

Connection Diagrams

Plastic Chip Carrier

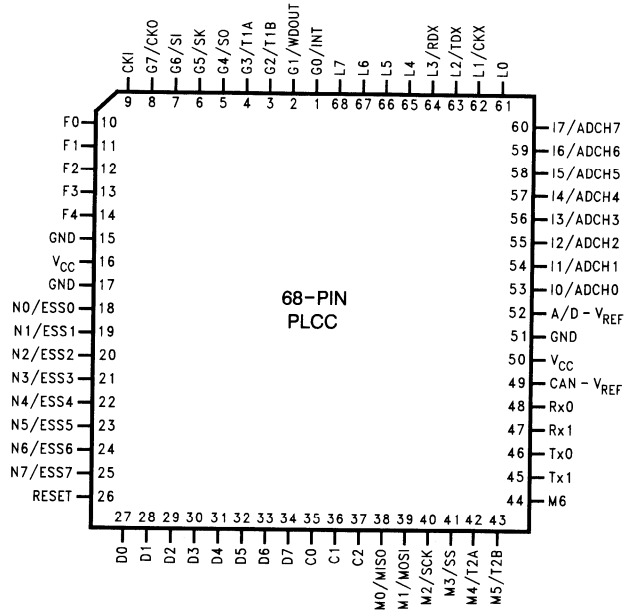


TL/DD/12871-2

Top View

Order Number COP87L88EB-XE
See NS Plastic Chip Package Number V44A

Plastic Leaded Chip Carrier



TL/DD/12871-3

Note:
-X Crystal Oscillator
-E Halt Enable

Top View

Order Number COP87L89EB-XE
See NS Plastic Chip Package Number V68A

FIGURE 2. Connection Diagrams

Connection Diagrams (Continued)

TABLE I. Pinouts for 44-Pin and 68-Pin Packages

Port Pin	Type	ALT Function	44-Pin PLCC	68-Pin PLCC
G0	I/O	INT	44	1
G1	I/O	WDOUT	1	2
G2	I/O	T1B	2	3
G3	I/O	T1A	3	4
G4	I/O	SO	4	5
G5	I/O	SK	5	6
G6	I	SI	6	7
G7	I	CKO	7	8
D0	O		17	27
D1	O		18	28
D2	O		19	29
D3	O		20	30
D4	O			31
D5	O			32
D6	O			33
D7	O			34
I0	I	ADCH0	36	53
I1	I	ADCH1	37	54
I2	I	ADCH2	38	55
I3	I	ADCH3	39	56
I4	I	ADCH4		57
I5	I	ADCH5		58
I6	I	ADCH6		59
I7	I	ADCH7		60
L0	I/O	MIWU	40	61
L1	I/O	MIWU;CKX	41	62
L2	I/O	MIWU;TDX	42	63
L3	I/O	MIWU;RDX	43	64
L4	I/O	MIWU		65
L5	I/O	MIWU		66
L6	I/O	MIWU		67
L7	I/O	MIWU		68
E4	I/O			
E5	I/O			
E6	I/O			
E7	I/O			
M0	I/O	MIWU;MISO	21	38
M1	I/O	MIWU;MOSI	22	39
M2	I/O	MIWU;SCK	23	40

Port Pin	Type	ALT Function	44-Pin PLCC	68-Pin PLCC
M3	I/O	MIWU;$\overline{SS}$	24	41
M4	I/O	MIWU;T2A	25	42
M5	I/O	MIWU;T2B	26	43
M6	I/O	MIWU	27	44
M7	I/O			
N0	I/O	ESS0	12	18
N1	I/O	ESS1	13	19
N2	I/O	ESS2	14	20
N3	I/O	ESS3	15	21
N4	I/O	ESS4		22
N5	I/O	ESS5		23
N6	I/O	ESS6		24
N7	I/O	ESS7		25
F0	I/O			10
F1	I/O			11
F2	I/O			12
F3	I/O			13
F4	I/O			14
F5	I/O			
F6	I/O			
F7	I/O			
C0	I/O			35
C1	I/O			36
C2	I/O			37
C3	I/O			
C4	I/O			
C5	I/O			
C6	I/O			
RX0	I		31	48
RX1	I		30	47
TX0	O		29	46
TX1	O		28	45
$CANV_{REF}$			32	49
CKI			8	9
RESET			16	26
DV_{CC}			10, 33	16, 50
GND			9, 11, 34	15, 17, 51
A/D V_{REF}			35	52

Absolute Maximum Ratings

If Military/Aerospace specified devices are required, please contact the National Semiconductor Sales Office/Distributors for availability and specifications.

Supply Voltage (V_{CC})	6V
Voltage at Any Pin	$-0.3V$ to $V_{CC}+0.3V$
Total Current into V_{CC} Pins (Source)	90 mA
Total Currnet out of GND Pins (Sink)	100 mA
Storage Temperature Range	$-65°C$ to $+150°C$

Note: *Absolute maximum ratings indicate limits beyond which damage to the device may occur. DC and AC electrical specifications are not ensured when operating the device at absolute maximum ratings.*

DC Electrical Characteristics $-40°C \le T_A \le +85°C$

Parameter	Conditions	Min	Typ	Max	Units
Operating Voltage		4.5		5.5	V
Power Supply Ripple (Note 1)	Peak-to-Peak			0.1 V_{CC}	V
Supply Current CKI = 10 MHz (Note 2)	V_{CC} = 5.5V, t_c = 1 μs			16	mA
HALT Current (Notes 3, 4)	V_{CC} = 5.5V, CKI = 0 MHz		< 1		μA
IDLE Current (Note 4) CKI = 10 MHz	V_{CC} = 5.5V, t_c = 1 μs			5.5	mA
Input Levels (V_{IH}, V_{IL})					
Reset, CKI					
Logic High		0.8V_{CC}			V
Logic Low				0.2V_{CC}	V
All Other Inputs					
Logic High		0.7V_{CC}			V
Logic Low				0.2V_{CC}	V
Hi-Z Input Leakage	V_{CC} = 5.5V			±2	μA
Input Pull-Up Current	V_{CC} = 5.5V, V_{IN} = 0V	−40		−250	μA
Port G, L and M Input Hysteresis	(Note 7)		0.05V_{CC}		V
Output Current Levels					
D Outputs					
Source	V_{CC} = 4.5V, V_{OH} = 3.3V	−0.4			mA
Sink	V_{CC} = 4.5V, V_{OL} = 1.0V	10			mA
CAN Transmitter Outputs					
Source (Tx1)	V_{CC} = 4.5V, V_{OH} = V_{CC} −0.1V	−1.5			mA
	V_{CC} = 4.5V, V_{OH} = V_{CC} − 0.6V	−10		+5.0	mA
Sink (Tx0)	V_{CC} = 4.5V, V_{OL} = 0.1V	1.5			mA
	V_{CC} = 4.5V, V_{OL} = 0.6V	10			mA
All Others					
Source (Weak Pull-Up)	V_{CC} = 4.5V, V_{OH} = 2.7V	−10		−110	μA
Source (Push-Pull)	V_{CC} = 4.5V, V_{OH} = 3.3V	−0.4			mA
Sink (Push-Pull)	V_{CC} = 4.5V, V_{OL} = 0.4V	1.6			mA
TRI-STATE Leakage	V_{CC} = 5.5V			±2.0	μA
Allowable Sink/Source Current per Pin					
D Outputs (sink)				15	mA
Tx0 (Sink) (Note 7)				30	mA
Tx1 (Source) (Note 7)				30	mA
All Other				3	mA
Maximum Input Current without Latchup (Notes 5, 7)	Room Temp			±200	mA
RAM Retention Voltage, V_r (Note 6)	500 ns Rise and Fall Time	2.0			V
Input Capacitance	(Note 7)			7	pF
Load Capacitance on D2				1000	pF

Note 1: Maxiumum rate of voltage change must be < 0.5V/ms

Note 2: Supply current is measured after running 2000 cycles with a square wave CKI input, CKO open, inputs at V_{CC} or GND, and outputs open.

Note 3: The HALT mode will stop CKI from oscillating in the Crystal configurations. Halt test conditions: All inputs tied to V_{CC}; Port C, G, E, F, L, M and N I/Os configured as outputs and programmed low; D outputs programmed high. Parameter refers to HALT mode entered via setting bit 7 of the Port G data register. Part will pull up CKI during HALT in crystal clock mode. Both CAN main comparator and the CAN Wakeup comparator need to be disabled.

Note 4:. HALT and IDLE current specifications assume CAN block comparators are disabled.

Note 5: Pins G6 and $\overline{\text{RESET}}$ are designed with a high voltage input network. These pins allow input voltages greater than V_{CC} and the pins will have sink current to V_{CC} when biased at voltages greater than V_{CC} (the pins do not have source current when biased at a voltage below V_{CC}). The effective resistance to V_{CC} is 750Ω (typical). These two pins will not latch up. The voltage at the pins must be limited to less than 14V.

Note 6: Condition and parameter valid only for part in HALT mode.

Note 7: Parameter characterized but not tested.

AC Electrical Characteristics $-40°C \le T_A \le +85°C$

Parameter	Conditions	Min	Typ	Max	Units
Instruction Cycle Time (t_c)					
Crystal/Resonator	$V_{CC} \ge 4.5V$	1.0		DC	μs
Inputs					
t_{SETUP}	$V_{CC} \ge 4.5V$	200			ns
t_{HOLD}	$V_{CC} \ge 4.5V$	60			ns
Output Propagation Delay (t_{PD1}, t_{PD0}) (Note 8)	$C_L = 100$ pF, $R_L = 2.2$ kΩ				
SK, SO	$V_{CC} \ge 4.5V$			0.7	μs
All others	$V_{CC} \ge 4.5V$			1	μs
MICROWIRE					
Setup Time (tUWS) (Note 9)		20			ns
Hold Time (tUWH) (Note 9)		56			ns
Output Pop Delay (tUPD)				220	ns
Input Pulse Width					
Interrupt High Time		1			t_c
Interrupt Low Time		1			t_c
Timer 1, 2 High Time		1			t_c
Timer 1, 2 Low Time		1			t_c
Reset Pulse Width (Note 9)		1.0			μs

t_c = Instruction Cycle Time

The maximum bus speed achievable with the CAN interface is a function of crystal frequency, message length and software overhead. The device can support a bus speed of up to 1 Mbit/S with a 10 MHz oscillator and 2 byte messages. The 1M bus speed refers to the rate at which protocol and data bits are transferred on the bus. Longer messages require slower bus speeds due to the time required for software intervention between data bytes. The device will support a maximum of 125k bits/s with eight byte messages and a 10 MHz oscillator.

Note: For device testing purpose of all AC parameters, V_{OH} will be tested at $0.5*V_{CC}$.

Note 8: The output propagation delay is referenced to the end of the instruction cycle where the output change occurs.

Note 9: Parameter not tested.

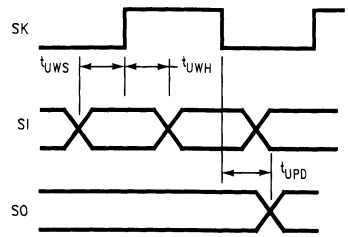
On-Chip Voltage Reference $-40°C \le T_A \le +85°C$

Parameter	Conditions	Min	Max	Units
Reference Voltage V_{REF}	$I_{OUT} < 80\ \mu A$, $V_{CC} = 5V$	$0.5V_{CC} - 0.12$	$0.5V_{CC} + 0.12$	V
Reference Supply Current, I_{DD}	$I_{OUT} = 0A$, (No Load) $V_{CC} = 5V$ (Note 1)		120	μA

Note 1: Reference supply I_{DD} is supplied for information purposes only, it is not tested.

CAN Comparator DC and AC Characteristics $4.8V \le V_{CC} \le 5.2V$, $-40°C \le T_A \le +85°C$

Parameter	Conditions	Min	Typ	Max	Units
Differential Input Voltage				± 25	mV
Input Offset Voltage	$1.5V < V_{IN} < V_{CC} - 1.5V$			± 10	mV
Input Common Mode Voltage Range		1.5		$V_{CC} - 1.5$	V
Input Hysteresis		8			mV



TL/DD/12871–4

FIGURE 3. MICROWIRE/PLUS Timing Diagram

A/D Converter Specifications ($4.5V \leq V_{CC} \leq 5.5V$) ($V_{SS} - 0.050V$) $\leq$ Any Input $\leq$ ($V_{CC} + 0.050V$)

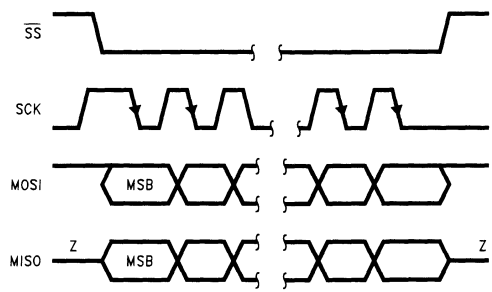
Parameter	Conditions	Min	Typ	Max	Units
Resolution				8	Bits
Absolute Accuracy	$V_{REF} = V_{CC}$			±2	LSB
Non-Linearity Deviation from the Best Straight Line				±1	LSB
Differential Non-Linearity				±1	LSB
Common Mode Input Range (Note 3)		GND		V_{CC}	V
DC Common Mode Error				±0.5	LSB
Off Channel Leakage Current			1	2.0	μA
On Channel Leakage Current			1	2.0	μA
A/D Clock Frequency (Note 2)		0.1		1.67	MHz
Conversion Time (Note 1)			17		A/D Clock Cycles
Internal Reference Resistance Turn-On Time (Note 4)				1	μs

Note 1: Conversion Time includes sample and hold time.

Note 2: See Prescaler description.

Note 3: For $V_{IN}(-) >= V_{IN}(+)$ the digital output code will be 0000 0000. Two on-chip doides are ties to each analog input. The diodes will forward conduct for analog input voltages below ground or above the V_{CC} supply. Be careful, during testing at low V_{CC} levels (4.5V), as high level analog inputs (5V) can cause this input diode to conduct—especially at elevated temperatures, and cause errors for analog inputs near full-scale. The spec allows 50 mV forward bias of either diode. This means that as long as the analog V_{IN} does not exceed the supply voltage by more than 50 mV, the output code will be correct. To achieve an absolute 0 V_{DC} to 5 V_{DC} input voltage range will therefore require a minimum supply voltage of 4.950 V_{DC} over temperature variations, initial tolerance and loading.

Note 4: Time for internal reference resistance to turn on after coming out of Halt or Idle Mode.



TL/DD/12871–5

FIGURE 4. SPI Timing Diagram

Pin Description

V_{CC} and GND are the power supply pins.

CKI is the clock input. The clock can come from a crystal oscillator (in conjunction with CKO). See Oscillator Description section.

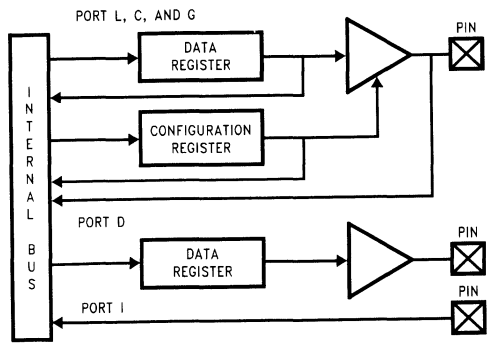
$\overline{\text{RESET}}$ is the master reset input. See Reset Description section.

The device contains seven bidirectional 8-bit I/O ports (C, E, F, G, L, M, N) where each individual bit may be independently configured as an input (Schmitt trigger inputs on all ports), output or TRI-STATE under program control. Three data memory address locations are allocated for each of these I/O ports. Each I/O port has two associated 8-bit memory mapped registers, the CONFIGURATION register and the output DATA register. A memory mapped address is also reserved for the input pins of each I/O port. (See the memory map for the various addresses associated with the I/O ports.) *Figure 5* shows the I/O port configurations for the device. The DATA and CONFIGURATION registers allow for each port bit to be individually configured under software control as shown below:

Configuration Register	Data Register	Port Set-Up
0	0	Hi-Z Input (TRI-STATE Output)
0	1	Input with Weak Pull-Up
1	0	Push-Pull Zero Output
1	1	Push-Pull One Output

Port L and M are 8-bit I/O ports, they support Multi-Input Wake-up (MIWU) on all eight pins. All L-pins and M-pins have Schmitt triggers on the inputs.

Port L and M only have one (1) interrupt vector.



TL/DD/12871–6

FIGURE 5. I/O Port Configurations

Port L has the following alternate features:

- L0 MIWU
- L1 MIWU or CKX
- L2 MIWU or TDX
- L3 MIWU or RDX
- L4 MIWU
- L5 MIWU
- L6 MIWU
- L7 MIWU

Port G is an 8-bit port with 5 I/O pins (G0–G5), an input pin (G6), and one dedicated output pin (G7). Pins G0–G6 all have Schmitt Triggers on their inputs. G7 serves as the dedicated output pin for the CKO clock output. There are two registers associated with the G Port, a data register and a configuration register. Therefore, each of the 6 I/O bits (G0–G5) can be individually configured under software control.

Since G6 is an input only pin and G7 is the dedicated CKO clock output pin the associated bits in the data and configuration registers for G6 and G7 are used for special purpose functions as outlined below. Reading the G6 and G7 data bits will return zeroes.

Note that the chip will be placed in the HALT mode by wirting a "1" to bit 7 of the Port G Data Register. Similarly the chip will be placed in the IDLE mode by writing a "1" to bit 6 of the Port G Data Register.

Writing a "1" to bit 6 of the Port G Configuration Register enables the MICROWIRE/PLUS to operate with the alternate phase of the SK clock

	Config. Reg.	Data Reg.
G7	CLKDLY	HALT
G6	Alternate SK	IDLE

Port G has the following alternate features:

- G0 INTR (External Interrupt Input)
- G1 Dedicated WATCHDOG output
- G2 (Timer T1 Capture Input)
- G3 T1A (Timer I/O)
- G4 SO (MICROWIRE Serial Data Output)
- G5 SK (MICROWIRE Serial Clock)
- G6 SI (MICROWIRE Serial Data Input)

Port G has the following dedicated function:

- G7 CKO Oscillator dedicated output

Pin Description (Continued)

Port M is a bidirectional I/O, it may be configured in software as Hi-Z input, weak pull-up, or push-pull output. These pins may be used as general purpose input/output pins or for selected altlernate functions.

Port M pins have optional alternate functions. Each pin (M0–M5) has been assigned an alternate data, configuration, or wakeup source. If the respective alternate function is selected the content of the associated bits in the configuration and/or data register are ignored. If an alternate wakeup source is selected the input level at the respective pin will be ignored for the purpose of triggering a wakeup event, however it will still be possible to read that pin by accessing the input register. The SPI (Serial Peripheral Interface) block, for example, uses four of the Port M pins to automatically re-configure its MISO (Master Input, Slave Output), MOSI (Master Output, Slave Input), SCK (Serial Clock) and Slave-Select pins as inputs or outputs, depending on whether the interface has been configured as a Master or Slave. When the SPI interface is disabled those pins are available as general purpose I/O pins configurable by user software writing to the associated data and configuration bits. The CAN interface on the device makes use of one of the Port M's alternate wake-ups, to trigger a wakeup if such a condition has been detected on the CAN's dedicated receive pins.

Port M has the following alternate pin functions:

- M0 Multi-input Wakeup or MISO
- M1 Multi-input Wakeup or MOSI
- M2 Multi-input Wakeup or SCK
- M3 Multi-input Wakeup or $\overline{SS}$
- M4 Multi-input Wakeup or T2A
- M5 Multi-input Wakeup or T2B
- M6 Multi-input Wakeup
- M7 Multi-input Wakeup or CAN

Ports C, E, F and N are general-purpose, bidirectional I/O ports.

Any device package that has Port C, E, F, M, N but has fewer than eight pins, contains unbonded, floating pads internally on the chip. For these types of devices, the software should write a 1 to the configuration register bits corresponding to the non-existent port pins. This configures the port bits as outputs, thereby reducing leakage current of the device.

Port N is an 8-bit wide port with alternate function capability used for extending the slave select ($\overline{SS}$) lines of the on SPI interface. The SPI expander block provides mutually exclusive slave select extension signals ($\overline{ESS0}$ to $\overline{ESS7}$) according to the state of the $\overline{SS}$ line and specific contents of the SPI shift register. These slave select extension lines can be routed to the Port N I/O pins by enabling the alternate function of the port in the PORTNX register. If enabled, the internal signal on the $\overline{ESSx}$ line causes the ports state to change exactly like a change to the PORTND register. It is the user's responsibility to switch the port to an output when enabling the alternate function.

Port N has the following alternate pin functions:

- N0 $\overline{ESS0}$
- N1 $\overline{ESS1}$
- N2 $\overline{ESS2}$
- N3 $\overline{ESS3}$
- N4 $\overline{ESS4}$
- N5 $\overline{ESS5}$
- N6 $\overline{ESS6}$
- N7 $\overline{ESS7}$

CAN pins: For the on-chip CAN interface this device has five dedicated pins with the following features:

V_{REF} On-chip reference voltage with the value of $V_{CC}/2$

Rx0 CAN receive data input pin.

RX1 CAN receive data input pin.

Tx0 CAN transmit data output pin. This pin may be put in the TRI-STATE mode with the TXEN0 bit in the CAN Bus control register.

Tx1 CAN transmit data output pin. This pin may be put in the TRI-STATE mode with the TXEN1 bit in the CAN Bus control register.

ALTERNATE PORT FUNCTIONS

Many general-purpose pins have alternate functions. The software can program each pin to be used either for a general-purpose or for a specific function. The chip hardware determines which of the pins have alternate functions, and what those functions are. This section lists the alternate functions available on each of the pins.

Port D is an 8-bit output port that is preset high when RESET goes low. The user can tie two or more port D outputs (except D2) together in order to get a higher drive.

Note: Care must be exercised with D2 pin operation. At RESET, the external loads on this pin must ensure that the output voltages stay above 0.8 V_{CC} to prevent the chip from entering special modes. Also keep the external loading on D2 to < 1000 pF.

Port 1 is an 8-bit Hi-Z input port, and also provides the analog inputs to the A/D converter. If unterminated, Port 1 pins will draw power only when addressed.

Functional Description

The architecture of the device utilizes a modified Harvard architecture. With the Harvard architecture, the control store program memory (ROM) is separated from the data store memory (RAM). Both ROM and RAM have their own separate addressing space with separate address buses. The architecture, though based on Harvard architecture, permits transfer of data from ROM to RAM.

CPU REGISTERS

The CPU can do an 8-bit addition, subtraction, logical or shift operation in one instruction (t_C) cycle time.

There are five CPU registers:

A is the 8-bit Accumulator Register

PC is the 15-bit Program Counter Register

PU is the upper 7 bits of the program counter (PC)

PL is the lower 8 bits of the program counter (PC)

B is an 8-bit RAM address pointer, which can be optionally post auto incremented or decremented.

X is an 8-bit alternate RAM address pointer, which can be optionally post auto incremented or decremented.

SP is the 8-bit stack pointer, which points to the subroutine/interrupt stack (in RAM). The SP is initialized to RAM address 02F with reset.

All the CPU registers are memory mapped with the exception of the Accumulator (A) and the Program Counter (PC).

PROGRAM MEMORY

Program memory for the device consists of 8 kbytes of OTP EPROM. These bytes may hold program instructions or constant data (data tables for the LAID instruction, jump vectors for the JID instruction and interrupt vectors for the VIS instruction). The program memory is addressed by the 15-bit program counter (PC). All interrupts in the device vector to program memory location 0FF Hex.

The device can be configured to inhibit external reads of the program memory. This is done by programming the Security Byte.

SECURITY FEATURE

The program memory array has an associate Security Byte that is located outside of the program address range. This byte can be addressed only from programming mode by a programmer tool.

Security is an optional feature and can only be asserted after the memory array has been programmed and verified. A secured part will read all 00(hex) by a programmer. The part will fail Blank Check and will fail Verify operations. A Read operation will fill the programmer's memory with 00(hex). The Security byte itself is always readable with value of 00(hex) if unsecure and FF(hex) if secure.

DATA MEMORY

The data memory address space includes the on-chip RAM and data registers, the I/O registers (Configuration, Data and Pin), the control registers, the MICROWIRE/PLUS SIO shift register, and the various registers, and counters associated with the timers (with the exception of the IDLE timer). Data memory is addressed directly by the instruction or indirectly by the B, X and SP pointers.

The device has 192 bytes of RAM. Sixteen bytes of RAM are mapped as "registers" at addresses 0F0 to 0FF Hex. These registers can be loaded immediately, and also decremented and tested with the DRSZ (decrement register and skip if zero) instruction. The memory pointer registers X, SP, and B are memory mapped into this space at address locations 0FC to 0FE Hex respectively, with the other registers (other than reserved register 0FF) being available for general usage.

The instruction set permits any bit in memory to be set, reset or tested. All I/O and registers (except A and PC) are memory mapped; therefore I/O bits and register bits can be directly and individually set, reset and tested. The accumulator (A) bits can also be directly and individually tested.

Note: RAM contents are undefined upon power-up.

RESET

The $\overline{\text{RESET}}$ input when pulled low initializes the microcontroller. Initialization will occur whenever the $\overline{\text{RESET}}$ input is pulled low. Upon initialization, the data and configuration registers for Ports L and G, are cleared, resulting in these Ports being initialized to the TRI-STATE mode. Port D is initialized high with $\overline{\text{RESET}}$. The PC, CNTRL, and INCTRL control registers are cleared. The Multi-Input Wakeup registers WKEN, WKEDG, and WKPND are cleared. The Stack Pointer, SP, is initialized to 06F Hex.

The following initializations occur with $\overline{\text{RESET}}$:

SPI:
- SPICNTRL: Cleared
- SPISTAT: Cleared
- STBE Bit: Set

T1CNTRL & T2CNTRL: Cleared

ITMR: Cleared and IDLE timer period is reset to 4k Instr. CLK

ENAD: Cleared

ADDSLT: Random

SIOR: Unaffected after RESET with power already applied.
Random after RESET at power on.

Port L: TRI-STATE

Port G: TRI-STATE

Port D: HIGH

PC: CLEARED

PSW, CNTRL and ICNTRL registers: CLEARED

Accumulator and Timer 1:

RANDOM after RESET with power already applied
RANDOM after RESET at power-on

SP (Stack Pointer): Loaded with 6F Hex

B and X Pointers:

UNAFFECTED after RESET with power already applied
RANDOM after RESET at power-up

RAM:

UNAFFECTED after RESET with power already applied
RANDOM after RESET at power-up

CAN: The CAN Interface comes out of external reset in the "error active" state and waits until the user's software sets either one or both of the TXEN0, TXEN1 bits to "1". After that, the device will not start transmission or reception of a frame util eleven consecutive "recessive" (undriven) bits have been received. This is done to ensure that the output drivers are not enamble during an active message on the bus.

CSCAL, CTIM, TCNTL, TEC, REC: CLEARED

RTSTAT: CLEARED with the exception of the TBE bit which is set to 1

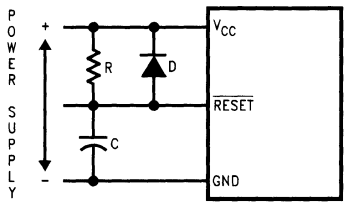
RID, RIDL, TID, TDLC: RANDOM

Functional Description (Continued)

WATCHDOG: The device comes out of reset with both the WATCHDOG logic and the Clock Monitor detector armed, with the WATCHDOG service window bits set and the Clock Monitor bit set. The WATCHDOG and Clock Monitor circuits are inhibited during reset. The WATCHDOG service window bits being initialized high default to the maximum WATCHDOG service window of 64k t_C clock cycles. The Clock Monitor bit being initialized high will cause a Clock Monitor bit being initialized high will cause a Clock Monitor error following reset if the clock has not reached the minimum specified frequency at the termination of reset. A Clock Monitor error will cause an active low error output on pin G1. This error output will continue until 16 t_C–32 t_C clock cycles following the clock frequency reaching the minimum specified value, at which time the G1 output will enter the TRI-STATE mode.

The $\overline{\text{RESET}}$ signal goes directly to the HALT latch to restart a halted chip.

When using external reset, the external RC network shown in *Figure 6* should be used to ensure that the $\overline{\text{RESET}}$ pin is held low until the power supply to the chip stabilizes. Under no circumstances should the $\overline{\text{RESET}}$ pin be allowed to float.



TL/DD/12871–7

RC > 5 x Power Supply Rise Time

FIGURE 6. Recommended Reset Circuit

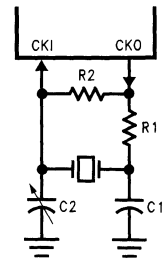
Oscillator Circuits

The chip can be driven by a clock input on the CKI input pin which can be between DC and 10 MHz. The CKO output clock is on pin G7. The CKI input frequency is divided by 10 to produce the instruction cycle clock ($1/t_C$).

Figure 7 shows the Crystal diagram.

CRYSTAL OSCILLATOR

CKI and CKO can be connected to make a closed loop crystal (or resonator) controlled oscillator.



TL/DD/12871–8

FIGURE 7. Crystal Oscillator Diagram

Table II shows the component values required for various standard crystal values.

TABLE II. Crystal Oscillator Configuration, T_A = 25°C

R1 (kΩ)	R2 (MΩ)	C1 (pF)	C2 (pF)	CKI Freq. (MHz)	Conditions
0	1	30	30–36	10	V_{CC} = 5V
0	1	30	30–36	4	V_{CC} = 5V
0	1	200	100–150	0.455	V_{CC} = 5V

2

Control Registers

CNTRL Register (Address X'00EE)

The Timer1 (T1 and MICROWIRE/PLUS control register contains the following bits:

SL1 & SL0 Select the MICROWIRE/PLUS clock divide by (00 = 2, 01 = 4, 1x = 8)

IEDG External interrupt edge polarity select (0 = Rising edge, 1 = Falling edge)

MSEL Selects G5 and G4 as MICROWIRE/PLUS signals SK and SO respectively

T1C0 Timer T1 Start/Stop control in timer Timer T1 Underflow Interrupt Pending Flag in timer mode 3

T1C1 Timer T1 mode control bit

T1C2 Timer T1 mode control bit

T1C3 Timer T1 mode control bit

T1C3	T1C2	T1C1	T1C0	MSEL	IEDG	SL1	SL0

Bit 7 Bit 0

PSW Register (Address X'00EF)

The PSW register contains the following select bits:

GIE Global interrupt enable (enables interrupts)

EXEN Enable external interrupt

BUSY MICROWIRE/PLUS busy shifting flag

EXPND External interrupt pending

T1ENA Timer T1 Interrupt Enable for Timer Underflow or T1A Input capture edge

T1PNDA Timer T1 Interrupt Pending Flag (Autoreload RA in mode 1, T1 Underflow in Mode 2, T1A capture edge in mode 3)

C Carry Flag

HC Half Carry Flag

HC	C	T1PNDA	T1ENA	EXPND	BUSY	EXEN	GIE

Bit 7 Bit 0

The Half-Carry bit is also affected by all the instructions that affect the Carry flag. The SC (Set Carry) and RC (Reset Carry) instructions will respectively set or clear both the carry flags. In addition to the SC and RC instructions, ADC, SUBC, RRC and RLC instructions affect the carry and Half Carry flags.

ICNTRL Register (Address X'00E8)

The ICNTRL register contains the following bits:

T1ENB Timer T1 Interrupt Enable for T1B Input capture edge

T1PNDB Timer T1 Interrupt Pending Flag for T1B capture edge

WEN Enable MICROWIRE/PLUS interrupt

WPND MICROWIRE/PLUS interrupt pending

T0EN Timer T0 Interrupt Enable (Bit 12 toggle)

T0PND Timer T0 Interrupt pending

LPEN Port L Interrupt Enable (Multi-Input Wakeup/ Interrupt)

Bit 7 could be used as a flag

Unused	LPEN	T0PND	T0EN	WPND	WEN	T1PNDB	T1ENB

Bit 7 Bit 0

T2CNTRL Register (Address X'00C6)

The T2CNTRL register contains the following bits:

T2ENB Timer T2 Interrupt Enable for T2B Input capture edge

T2PNDB Timer T2 Interrupt Pending Flag for T2B capture edge

T2ENA Timer T2 Interrupt Enable for Timer Underflow or T2 Input capture edge

T2PNDA Timer T2 Interrupt Pending Flag (Autoreload RA in mode 1, T2 Underflow in mode 2, T2A capture edge in mode 3)

T2C0 Timer T2 Start/Stop control in timer modes 1 and 2 Timer T2 Underflow Interrupt Pending Flag in timer mode 3

T2C1 Timer T2 mode control bit

T2C2 Timer T2 mode control bit

T2C3 Timer T2 mode control bit

T2C3	T2C2	T2C1	T2C0	T2PNDA	T2ENA	T2PNDB	T2ENB

Bit 7 Bit 0

Timers

The device contains a very versatile set of timers (T0, T1 and T2). All timers and associated autoreload/capture registers power up containing random data.

TIMER T0 (IDLE TIMER)

The device supports applications that require maintaining real time and low power with the IDLE mode. This IDLE mode support is furnished by the IDLE timer T0, which is a 16-bit timer. The Timer T0 runs continuously at the fixed rate of the instruction cycle clock, t_C. The user cannot read or write to the IDLE Timer T0, which is a count down timer.

The Timer T0 supports the following functions:

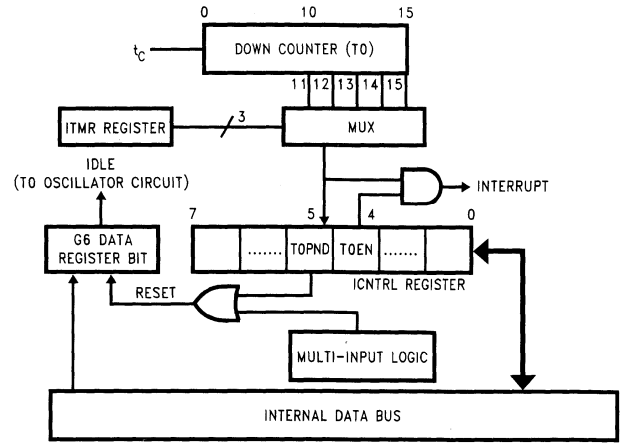
- Exit out of the Idle Mode (See Idle Mode description)
- WATCHDOG logic (See WATCHDOG description)
- Start up delay out of the HALT mode

Figure 8 is a functional block diagram showing the structure of the IDLE Timer and its associated interrupt logic.

Bits 11 through 15 of the ITMR register can be selected for triggering the IDLE Timer interrupt. Each time the selected bit underflows (every 4k, 8k, 16k, 32k or 64k instruction cycles), the IDLE Timer interrupt pending bit T0PND is set, thus generating an interrupt (if enabled), and bit 6 of the Port G data register is reset, thus causing an exit from the IDLE mode if the device is in that mode.

In order for an interrupt to be generated, the IDLE Timer interrupt enable bit T0EN must be set, and the GIE (Global Interrupt Enable) bit must also be set. The T0PND flag and T0EN bit are bits 5 and 4 of the ICNTRL register, respectively. The interrupt can be used for any purpose. Typically, it is used to perform a task upon exit from the IDLE mode. For more information on the IDLE mode, refer to the Power Save Modes section.

Timers (Continued)



TL/DD/12871–9

FIGURE 8. Functional Block Diagram for Idle Timer T0

The Idle Timer period is selected by bits 0–2 of the ITMR register Bits 3–7 of the ITMR Register are reserved and should not be used as software flags.

TABLE III. Idle Timer Window Length

ITSEL2	ITSEL1	ITSEL0	Idle Timer Period (Instruction Cycles)
0	0	0	4,096
0	0	1	8,192
0	1	0	16,384
0	1	1	32,768
1	X	X	65,536

The ITMR register is cleared on Reset and the Idle Timer period is reset to 4,096 instruction cycles.

ITMR Register (Address X'0xCF)

Reserved	ITSEL2	ITSEL1	ITSLE0

Bit 7 Bit 0

Any time the IDLE Timer period is changed there is the possibility of generating a spurious IDLE Timer interrupt by setting the T0PND bit. The user is advised to disable IDLE Timer interrupts prior to changing the value of the ITSEL bits of the ITMR Register and then clear the T0PND bit before attempting to synchronize operation to the IDLE Timer.

TIMER T1 and TIMER T2

The device has a set of three powerful timer/counter blocks, T1 and T2. The associated features and functioning of a timer block are described by referring to the timer block Tx. Since the three timer blocks, T1 and T2 are identical, all comments are equally applicable to either of the three timer blocks.

Each timer block consists of a 16-bit timer, Tx, and two supporting 16-bit autoreload/capture registers, RxA and RxB. Each timer block has two pins associated with it, TxA and TxB. The pin TxA supports I/O required by the timer block, while the pin TxB is an input to the timer block. The powerful and flexible timer block allows the device to easily perform all timer functions with minimal software overhead. The timer block has three operating modes: Processor Independent PWM mode, External Event Counter mode, and Input Capture mode.

The control bits TxC3, TxC2, and TxC1 allow selection of the different modes of operation.

Timers (Continued)

Mode 1. Processor Independent PWM Mode

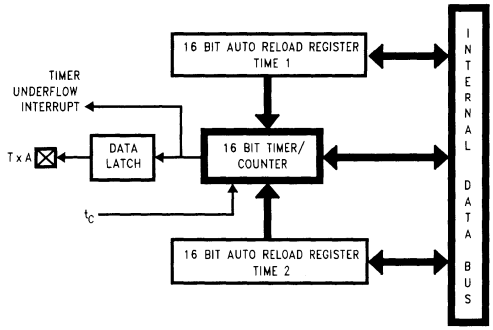
As the name suggests, this mode allows the device to generate a PWM signal with very minimal user intervention.

The user only has to define the parameters of the PWM signal (ON time and OFF time). Once begun, the timer block will continuously generate the PWM signal completely independent of the microcontroller. The user software services the timer block only when the PWM parameters require updating.

In this mode the timer Tx counts down at a fixed rate of t_C. Upon every underflow the timer is alternately reloaded with the contents of supporting registers, RxA and RxB. The very first underflow of the timer causes the timer to reload from the register RxA. Subsequent underflows cause the timer to be reloaded from the registers alternately beginning with the register RxB.

The Tx Timer control bits, TxC3, TxC2 and TxC1 set up the timer for PWM mode operation.

Figure 9 shows a block diagram of the timer in PWM mode.



TL/DD/12871–10

FIGURE 9. Timer in PWM Mode

The underflows can be programmed to toggle the TxA output pin. The underflows can also be programmed to generate interrupts.

Underflows from the timer are alternately latched into two pending flags, TxPNDA and TxPNDB. The user must reset these pending flags under software control. Two control enable flags, TxENA and TxENB, allow the interrupts from the timer underflow to be enabled or disabled. Setting the timer enable flag TxENA will cause an interrupt when a timer underflow causes the RxA register to be reloaded into the timer. Setting the timer enable flag TxENB will cause an interrupt when a timer underflow causes the RxB register to be reloaded into the timer. Resetting the timer enable flags will disable the associated interrupts.

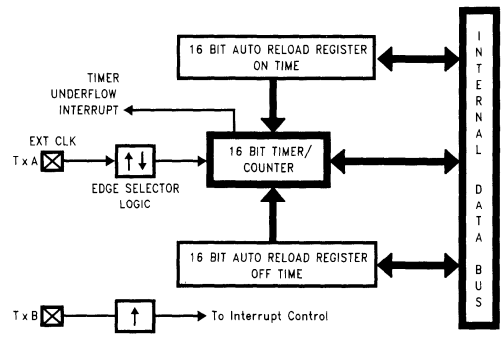
Either or both of the timer underflow interrupts may be enabled. This gives the user the flexibility of interrupting once per PWM period on either the rising or falling edge of the PWM output. Alternatively, the user may choose to interrupt on both edges of the PWM output.

Mode 2. External Event Counter Mode

This mode is quite similar to the processor independent PWM mode described above. The main difference is that the timer, Tx, is clocked by the input signal from the TxA pin. The Tx timer control bits, TxC3, TxC2 and TxC1 allow the timer to be clocked either on a positive or negative edge from the TxA pin. Underflows from the timer are latched into the TxPNDA pending flag. Setting the TxENA control flag will cause an interrupt when the timer underflows.

In this mode the input pin TxB can be used as an independent positive edge sensitive interrupt input if the TxENB control flag is set. The occurrence of the positive edge on the TxB input pin is latched to the TxPNDB flag.

Figure 10 shows a block diagram of the timer in External Event Counter mode.



TL/DD/12871–11

FIGURE 10. Timer in External Event Counter Mode

Note: The PWM output is not available in this mode since the TxA pin is being used as the counter input clock.

Mode 3. Input Capture Mode

The device can precisely measure external frequencies or time external events by placing the timer block, Tx, in the input capture mode.

In this mode, the timer Tx is constantly running at the fixed t_C rate. The two registers, RxA and RxB, act as capture registers. Each register acts in conjunction with a pin. The register RxA acts in conjunction with the TxA pin and the register RxB acts in conjunction with the TxB pin.

The timer value gets copied over into the register when a trigger event occurs on its corresponding pin. Control bits, TxC3, TxC2 and TxC1, allow the trigger events to be specified either as a positive or a negative edge. The trigger condition for each input pin can be specified independently.

Timers (Continued)

The trigger conditions can also be programmed to generate interrupts. The occurrence of the specified trigger condition on the TxA and TxB pins will be respectively latched into the pending flags, TxPNDA and TxPNDB. The control flag TxENA allows the interrupt on TxA to be either enabled or disabled. Setting the TxENA flag enables interrupts to be generated when the selected trigger condition occurs on the TxA pin. Similarly, the flag TxENB controls the interrupts from the TxB pin.

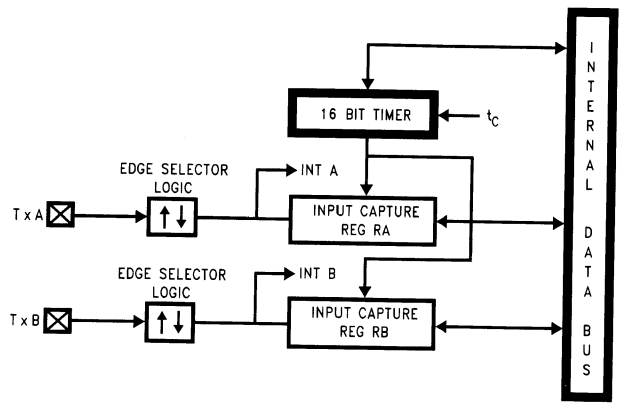
Underflows from the timer can also be programmed to generate interrupts. Underflows are latched into the timer TxC0 pending flag (the TxC0 control bit serves as the timer underflow interrupt pending flag in the Input Capture mode). Consequently, the TxC0 control bit should be reset when entering the Input Capture mode. The timer underflow interrupt is enabled with the TxENA control flag. When a TxA interrupt occurs in the Input Capture mode, the user must check both whether a TxA input capture or a timer underflow (or both) caused the interrupt.

Figure 11 shows a block diagram of the timer in Input Capture mode.

TIMER CONTROL FLAGS

The timers T1 and T2 have identical control structures. The control bits and their functions are summarized below.

TxC0	Timer Start/Stop control in Modes 1 and 2 (Processor Independent PWM and Extenral Event Counter), where 1 = State, 0 = Stop
	Timer Underflow Interrupt Pending Flag in Mode 3 (Input Capture)
TxPNDA	Timer Interrupt Pending Flag
TxPNDB	Timer Interrupt Pending Flag
TxENA	Timer Interrupt Enable Flag
TxENB	Timer Interrupt Enable Flag
	1 = Timer Interrupt Enabled
	0 = Timer Interrupt Disabled
TxC3	Timer mode control
TxC2	Timer mode control
TxC1	Timer mode control



TL/DD/12871–12

FIGURE 11. Timer in Input Capture Mode

Timers (Continued)

The timer mode control bits (TxC3, TxC2 and TxC1) are detailed below:

TxC3	TxC2	TxC1	Timer Mode	Interrupt A Source	Interrupt B Source	Timer Counts On
0	0	0	MODE 2 (External Event Counter)	Time Underflow	Pos. TxB Edge	TxA Pos. Edge
0	0	1	MODE 2 (External Event Counter)	Timer Underflow	Pos. TxB Edge	TxA Neg. Edge
1	0	1	MODE 1 (PWM) TxA Toggle	Autoreload RA	Autoreload RB	t_C
1	0	0	MODE 1 (PWM) No TxA Toggle	Autoreload RA	Autoreload RB	t_C
0	1	0	MODE 3 (Capture) Captures: TxA Pos. Edge TxB Pos. Edge	Pos. TxA Edge or Timer Underflow	Pos. TxB Edge	t_C
1	1	0	MODE 3 (Capture) Captures: TxA Pos. Edge TxB Neg. Edge	Pos. TxA Edge or Timer Underflow	Neg. TxB Edge	t_C
0	1	1	MODE 3 (Capture) Captures: TxA Neg. Edge TxB Pos. Edge	Neg. TxA Edge or Timer Underflow	Pos. TxB Edge	t_C
1	1	1	MODE 3 (Capture) Captures: TxA Neg. Edge TxB Neg. Edge	Neg. TxA Edge or Timer Underflow	Neg. TxB Edge	t_C

Power Save Modes

The device offer the user two power save modes of operation: HALT and IDLE. In the HALT mode, all microcontroller activities are stopped. In the IDLE mode, the on-board oscillator circuitry and timer T0 are active but all other microcontroller activities are stopped. In either mode, all on-board RAM, registers, I/O states, and timers (with the exception of T0) are unaltered.

HALT MODE

The device is placed in the HALT mode by writing a "1" to the HALT flag (G7 data bit). All microcontroller activities, including the clock, and timers, are stopped. In the HALT mode, the power requirements of the device are minimal and the applied voltage (V_{CC}) may be decreased to Vr (Vr = 2.0V) without altering the state of the machine.

CAN HALT/IDLE mode:

In order to reduce the device overall current consumption in HALT/IDLE mode a two step power save mechanism is implemented on the device:

Step 1: Disable main receive comparator. This is done by resetting both the TxEN0 and TxEN1 bits in the CBUS register. Note: These bits should always be reset before entering HALT/IDLE mode to allow proper resynchronization to the CAN bus after exiting HALT/IDLE mode.

Step 2: Disable the CAN wake-up comparators, this is done by resetting bit 7 in the port-m wakeup enable register (MWKEN) a transition on the CAN bus will then not wake the device up.

Note: If both the main receive comparator and the wake-up comparator are disabled the on chip CAN voltage reference is also disabled. The CAN-V_{REF} output is then High-Z

The following table shows the two CAN power save modes and the active CAN transceiver blocks:

Step 1	Step 2	Main-Comp	Wake-Up-Comp	CAN-V_{REF}	V_{REF} Pin
0	0	on	on	on	$V_{CC}/2$
0	1	on	off	on	$V_{CC}/2$
1	0	off	on	on	$V_{CC}/2$
1	1	off	off	off	High-Z

Power Save Modes (Continued)

The device supports two different ways of exiting the HALT mode. The first method of exiting the HALT mode is with the Multi-Input Wakeup feature on the L & M port. The second method of exiting the HALT mode is by pulling the RESET pin low.

Since a crystal or ceramic resonator may be selected as the oscillator, the Wakeup signal is not allowed to start the chip running immediately since crystal oscillators and ceramic resonators have a delayed start up time to reach full amplitude and frequency stability. The IDLE timer is used to generate a fixed delay to ensure that the oscillator has indeed stabilized before allowing instruction execution. In this case, upon detecting a valid Wakeup signal, only the oscillator circuitry is enabled. The IDLE timer is loaded with a value of 256 and is clocked with the t_C instruction cycle clock. The t_C clock is derived by dividing the oscillator clock down by a factor of 10. The Schmitt trigger following the CKI inverter on the chip ensures that the IDLE timer is clocked only when the oscillator has a sufficiently large amplitude to meet the Schmitt trigger specifications. This Schmitt trigger is not part of the oscillator closed loop. The start-up time-out from the IDLE timer enables the clock signals to be routed to the rest of the chip.

The device has two mask options associated with the HALT mode. The first mask option enables the HALT mode feature, while the second mask option disables the HALT mode. With the HALT mode enable mask option, the device will enter and exit the HALT mode as described above. With the HALT disable mask option, the device cannot be placed in the HALT mode (writing a "1" to the HALT flag will have to effect).

IDLE MODE

The device is placed in the IDLE mode by wirting a "1" to the IDLE flag (G6 data bit). In this mode, all activity, except the associated on-board oscillator circuitry, ad the IDLE Timer T0, is stopped. The power supply requirements of the microcontroller in this mode of operation are typically around 30% of normal power requirement of the microcontroller.

As with the HALT mode, the device can be returned to normal operation with a reset, or with a Multi-Input Wakeup from the Port L or CAN Interface. Alternately, the microcontroller resumes normal operation from the IDLE mode when the thirteenth bit (representing 4.096 ms at internal clock frequency of 1 MHz, $t_C = 1\ \mu s$) of the IDLE Timer toggles.

This toggle condition of the thirteenth bit of the IDLE Timer T0 is latched into the T0PND pending flag.

The user has the option of being interrupted with a transition on the thirteenth bit of the IDLE Timer T0. The interrupt can be enabled or disabled via the T0EN control bit. Setting the T0EN flag enables the interrupt and vice versa.

The user can enter the IDLE mode with the Timer T0 interrupt enabled. In this case, when the T0PND bit gets set, the device will first execute the Timer T0 interrupt service routine and then return to the instruciton following the "Enter Idle Mode" instruction.

Alternatively, the user can enter the IDLE mode with the IDLE Timer T0 interrupt disabled. In this case, the device will resume normal operation with the instruction immediately following the "Enter IDLE Mode" instruction.

Note: It is necessary to program two NOP instructions following both the set HALT mode and set IDLE mode instructions. These NOP instructions are necessary to allow clock resynchronization following the HALT or IDLE modes.

Multi-Input Wakeup

The Multi-Input Wakeup feature is used to return (wakeup) the device from either the HALT or IDLE modes. Alternately, the Multi-Input Wakeup/Interrupt feature may also be used to generate up to 7 edge selectable external interrupts.

Note: The following description is for both the Port L and the M port. When the document refers to the registers WKEGD, WKEN or WKPND, the user will have to put either M (for M port) or L (for port) in front of the register, i.e., LWKEN (Port L WKEN), MWKEN (Port M WKEN).

Figures 12 and *13* shows the Multi-Input Wakeup logic for the microcontroller. The Multi-Input Wakeup feature utilizes the L Port. The user selects which particular Port L bit (or combination of Port L bits) will cause the device to exit the HALT or IDLE modes. The selection is done through the Reg: WKEN. The Reg: WKEN is an 8-bit read/write register, which contains a control bit for every Port L bit. Setting a particular WKEN bit enables a Wakeup from the associated Port L pin.

The user can select whether the trigger condition on the selected Port L pin is going to be either a positive edge (low to high transition) or a negative edge (high to low transition). This selection is made via the Reg: WKEDG, which is an 8-bit control register with a bit assigned to each Port L pin. Setting the control bit will select the trigger condition to be a negative edge on that particular Port L pin. Resetting the bit selects the trigger condition to be a positive edge. Changing an edge select entails several steps in order to avoid a pseudo Wakeup condition as a result of the edge change. First, the associated WKEN bit should be reset, followed by the edge select change in WKEDG. Next, the associated WKPND bit should be cleared, followed by the associated WKEN bit being re-enabled.

An example may serve to clarify this procedure. Suppose we wish to change the edge select from positive (low going high) to negative (high going low) for Port L bit 5, where bit 5 has previously been enabled for an input interrupt. The program would be as follows:

```
RBIT 5, WKEN   ;Disable Port bit 5 for
                    wake-up
SBIT 5, WKEDG  ;Select neg-rising edge
RBIT 5, WKPND  ;Clear pending bit
SBIT 5, WKEN   ;Re-enable the bit
```

Multi-Input Wakeup (Continued)

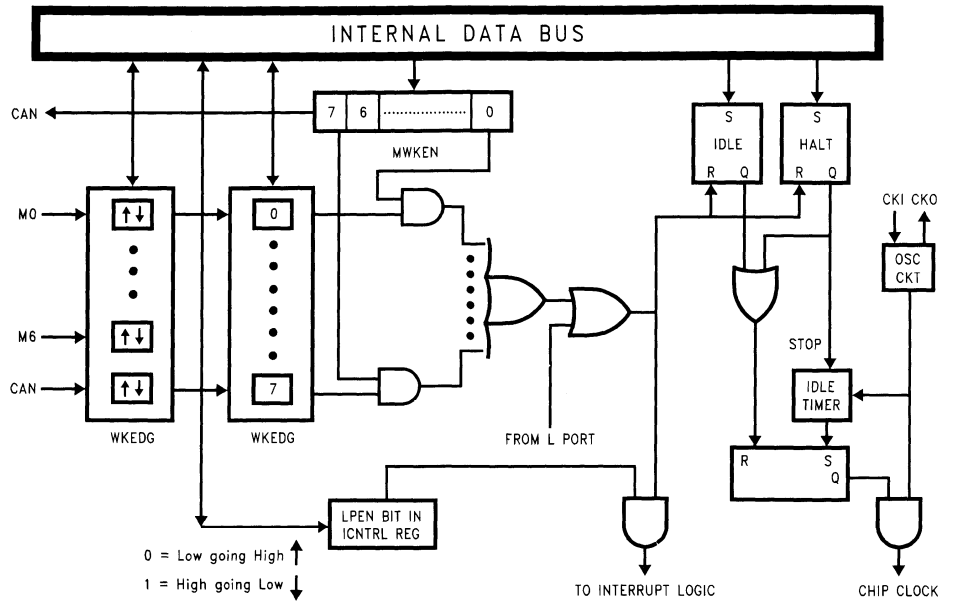


FIGURE 12. Port M Multi-Input Wake-up Logic

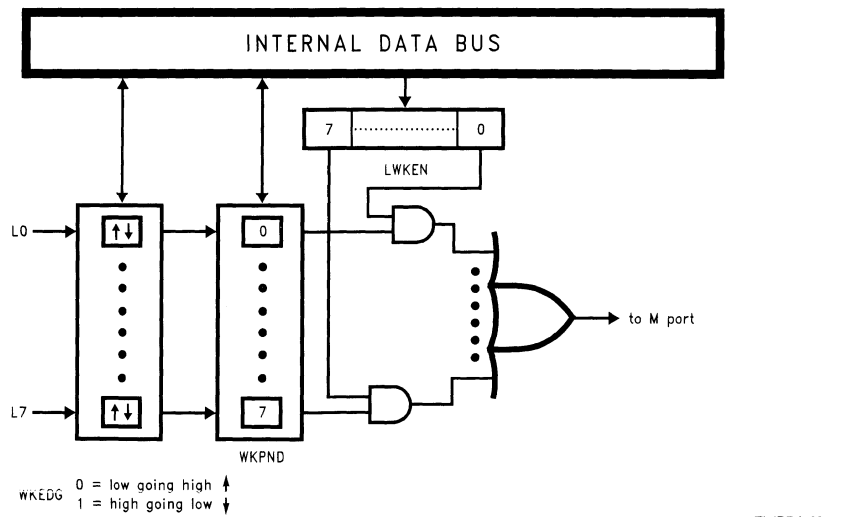


FIGURE 13. Port L Multi-Input Wake-Up Logic

Multi-Input Wakeup (Continued)

If the Port L bits have been used as outputs and then changed to inputs with Multi-Input Wakeup/Interrupt, a safety procedure should also be followed to avoid inherited pseudo wakeup conditions. After the selected Port L bits have been changed from output to input but before the associated WKEN bits are enabled, the associated edge select bits in WKEDG should be set or reset for the desired edge selects, followed by the assoicated WKPND bits being cleared.

This same procedure should be used following reset, since the Port L inputs are left floating as a result of reset. The occurrence of the selected trigger condition for Multi-Input Wakeup is latched to a pending register called WKPND. The respective bits of the WKPND register will be set on the occurrence of the selected trigger edge on the corresponding Port L and Port M pin. The user has the responsibility of clearing these pending flags. Since WKPND is a pending register for the occurrence of selected wakeup conditions, the device will not enter the HALT mode if any wakeup bit is both enabled and pending. Consequently, the user has the responsibility of clearing the pending flags before attempting to enter the HALT mode.

The WKEN, WKPND and WKEDG are all read/write registers, and are cleared at reset.

PORT L INTERRUPTS

Port L provides the user with additional eight fully selectable, edge sensitive interrupts which are all vectored into the same service subroutine.

The interrupt from Port L shares logic with the wake up circuitry. The register WKEN allows interrupts from Port L to be individually enabled or disabled. The register WKEDG specifies the trigger condition to be either a positive or a negative edge. Finally, the register WKPND latches in the pending trigger conditions.

The GIE (global interrupt enable) bit enables the interrupt function. A control flag, LPEN, functions as a global interrupt enable for Port L interrupts. Setting the LPEN flag will enable interrupts and vice versa. A separate global pending flag is not needed since the register WKPND is adequate.

Since Port L is also used for waking the device out of the HALT or IDLE modes, the user can elect to exit the HALT or IDLE modes either with or without the interrupt enabled. If he elects to disable the interrupt, then the device will restart execution from the instruction immediately following the instruction that placed the microcontroller in the HALT or IDLE modes. In the other case, the device will first execute the interrupt service routine and then revert to normal operation.

The Wakeup signal will not start the chip running immediately since crystal oscillators or ceramic resonators have a finite start up time. The IDLE Timer (T0) generates a fixed delay to ensure that the oscillator has indeed stabilized before allowing the device to execute instructions. In this case, upon detecting a valid Wakeup signal, only the oscillator circuitry and the IDLE Timer T0 are enabled. The IDLE Timer is loaded with a value of 256 and is clocked from the t_C instruction cycle clock. The t_C clock is derived by dividing down the oscillator clock by a factor of 10. A Schmitt trigger following the CKI on-chip inverter ensures that the IDLE timer is clocked only when the oscillator has a sufficiently large amplitude to meet the Schmitt trigger specifications. This Schmitt trigger is not part of the oscillator closed loop. The start-up time-out from the IDLE timer enables the clock signals to be routed to the rest of the chip.

PORT M INTERRUPTS

Port M provides the user with seven fully selectable, edge sensitive interrupts which are all vectored into the same service subroutine.

The interrupt from Port M shares logic with the wake up circuitry. The MWKEN register allows interrupts from Port M to be individually enabled or disabled. The MWKEDG register specifies the trigger condition to be either a positive or a negative edge. The MWKPND register latches in the pending trigger conditions.

The LPEN control flag in the ICNTRL register functions as a global interrupt enable for Port M interrupts. Setting the LPEN flag enables interrupts. Note that the GIE bit in the PSW register must also be set to enable these Port L interrupts. A global pending flag is not needed since each pin has a corresponding pending flag in the MWKPND register.

Multi-Input Wakeup (Continued)

Since Port M is also used for exiting the device from the HALT or IDLE mode, the user can elect to exit the HALT or IDLE mode either with or without the interrupt enabled. If the user elects to disable the interrupt, then the device restarts execution from the point at which it was stopped (first instruction cycle of the instruction following the enter HALT or IDLE mode instruction). In the other case, the device finishes the instruction which was being executed when the part was stopped (the NOP(1) instruction following the enter HALT or IDLE mode instruction), and then branches to the interrupt service routine. The device then reverts to normal operation.

Note 1: The user must plsce two NOPs after an enter HALT or IDLE mode instruction.

To prevent erroneous clearing of the SPI receive FIFO when entering HALT/IDLE mode, the user needs to enable the MIWU on port M3. ($\overline{SS}$) by setting bit 3 in the MWKEN register.

CAN RECEIVE WAKEUP

The CAN Receive Wakeup source can be enabled or disabled. There is no specific enable bit for the CAN Wakeup feature. Although the wakeup feature on pins L0..17 and M0..M7 can be programmed to generate an interrupt (Port L or Port M interrupt), no interrupt is generated upon a CAN receive wakeup condition. The CAN block has it's own, dedicated receiver interrupt upon receive buffer full (see CAN Section).

CAN Wake-Up:

The CAN interface can be programmed to wake the device from HALT/IDLE mode. This is done by setting bit 7 in the Port M wake-up enable register (MWKEN). A transition on the bus will cause the bit 7 of the Port M wake-up pending (MWKPND) to be set and thereby waking up the device. The frame on the CAN bus will be lost. The MWEDG (m port wake-up edge) register bit 7 can be programmed high or low (high will wake-up on the first falling edge on Rx0).

Resetting bit 7 in the MWKEN will disable the CAN wake-up.

The following sequence should be executed before entering HALT/IDLE mode:

```
RBIT 7, MWKPND; ;clear CAN wake-up pending
LD A, CBUS
AND A, #0CF    ; ;resetTxEN0 and TxEN1
X  A, CBUS     ; ;disable main receive
                  comparator
```

After the device woke-up the CBUS bits TxEN0 and/or TxEN1 need be set to allow synchronization on the bus and to enable transmission/reception of CAN frames.

Interrupts

The device supports a vectored interrupt scheme. It supports a total of fourteen interrupt sources. The following table lists all the possible device interrupt sources, their arbitration ranking and the memory locations reserved for the interrupt vector for each source.

Two bytes of program memory space are reserved for each interrupt source. All interrupt sources except the software interrupt are maskable. Each of the maskable interrupts have an Enable bit and a Pending bit. A maskable interrupt is active if its associated enable and pending bits are set. If GIE = 1 and an interrupt is active, then the processor will be interrupted as soon as it is ready to start executing an instruction except if the above conditions happen during the Software Trap service routine. This exception is described in the Software Trap sub-section.

The interruption process is accomplished with the INTR instruction (opcode 00), which is jammed inside the instruction Register and replaces the opcode about to be executed. The following steps are performed for every interrupt:

1. The GIE (Global Interrupt Enable) bit is reset.
2. The address of the instruction about to be executed is pushed into the stack.
3. The PC (Program Counter) branches to address 00FF. This procedure takes 7 t_C cycles to execute.

At this time, since GIE = 0, other maskable interrupts are disabled. The user is now free to do whatever context switching is required by saving the context of the machine in the stack with PUSH instructions. The user would then program a VIS (Vector Interrupt Select) instruction in order to branch to the interrupt service routine of the highest priority interrupt enabled and pending at the time of the VIS. Note that this is not necessarily the interrupt that caused the branch to address location 00FF Hex prior to the context switching.

Thus, if an interrupt with a higher rank than the one which caused the interruption becomes active before the decision of which interrupt to service is made by the VIS, then the interrupt with the higher rank will override any lower ones and will be acknowledged. The lower priority interrupt(s) are still pending, however, and will cause another interrupt immediately following the completion of the interrupt service routine associated with the higher priority interrupt just serviced. This lower priority interrupt will occur immediately following the RETI (Return from Interrupt) instruction at the end of the interrupt service routine just completed.

Inside the interrupt service routine, the associated pending bit has to be cleared by software. The RETI (Return from Interrupt) instruction at the end of the interrupt service routine will set the GIE (Global Interrupt Enable) bit, allowing the processor to be interrupted again if another interrupt is active and pending.

Interrupts (Continued)

The VIS instruction looks at all the active interrupts at the time it is executed and performs an indirect jump to the beginning of the service routine of the one with the highest rank.

The addresses of the different interrupt service routines, called vectors, are chosen by the user and stored in ROM in a table starting at 01E0 (assuming that VIS is located between 00FF and 01DF). The vectors are 15-bit wide and therefore occupy 2 ROM locations.

VIS and the vector table must be located in the same 256-byte block (0y00 to 0yFF) except if VIS is located at the last address of a block. In this case, the table must be in the next block. The vector table cannot be inserted in the first 256-byte block.

The vector of the maskable interrupt with the lowest rank is located at 0yE0 (Hi-Order byte) and 0yE1 (Lo-Order byte) and so forth in increasing rank number. The vector of the maskable interrupt with the highest rank is located at 0yFA (Hi-Order byte) and 0yFB (Lo-Order byte).

The Software Trap has the highest rank and its vector is located at 0yFE and 0yFF.

If, by accident, a VIS gets executed and no interrupt is active, then the PC (Program Counter) will branch to a vector located at 0yE0–0yE1.

WARNING:

A Default VIS interrupt handle routine must be present. As a minimum, this handler should confirm that the GIE bit is cleared (this indicates that the interrupt sequence has been taken), take care of any required housekeeping, restore context and return. Some sort of Warm Restart procedure should be implemented. These events can occur without any error on the part of the system designer or programmer.

Note: There is always the possibility of an interrupt occurring during an instruction which is attempting to reset the GIE bit or any other interrupt enable bit. If tis occurs when a single cycle instruction is being used to reset the interrupt enable bit, the interrupt enable bit will be reset but an interrupt may still occur. This is because interrupt processing is started at the same time as the interrupt bit is being reset. To avoid this scenario, the user should always use a two, three, or four cycle instruction to reset interrupt enable bits.

Figure 14 shows the Interrupt Block diagram.

TABLE IV. Interrupt Vector Table

Arbitration Rank	Interrupt Source	Description	Vector Address[a]
1	Software Trap	INTR Instruction	0yFE–0yFF
2	reserved	NMI	0yFC–0yFD
3	CAN Receive	RBF, RFV set	0yFA–0yFB
4	CAN Error (transmit/receive)	TERR, RERR set	0yF8–0yF9
5	CAN Transmit	TBE set	0yF6–0yF7
6	Pin G0 Edge	External	0yF4–0yF5
7	MICROWIRE/PLUS SPI Interface	BUSY Goes Low SRBF or STBE set	0yF2–0yF3
8	Timer T0	Idle Timer Underflow	0yF0–0yF1
9	UART	receive buffer full	0yEE–0yEF
10	UART	transmit buffer empty	0yEC–0yED
11	Timer T2	T2A/Underflow	0yEA–0yEB
12	Timer T2	T2B	0yE8–0yE9
13	Timer T1	T1A/Underflow	0yE6–0yE7
14	Timer T1	T1B	0yE4–0yE5
15	Port L, Port M; MIWU	Port L Edge or Port M Edge	0yE2–0yE3
16	Default VIS Interrupt	VIS Interrupt	0yE0–0yE1

a. y = 1 to 7F, depending on the location of the VIS instruction.

Interrupts (Continued)

SOFTWARE TRAP

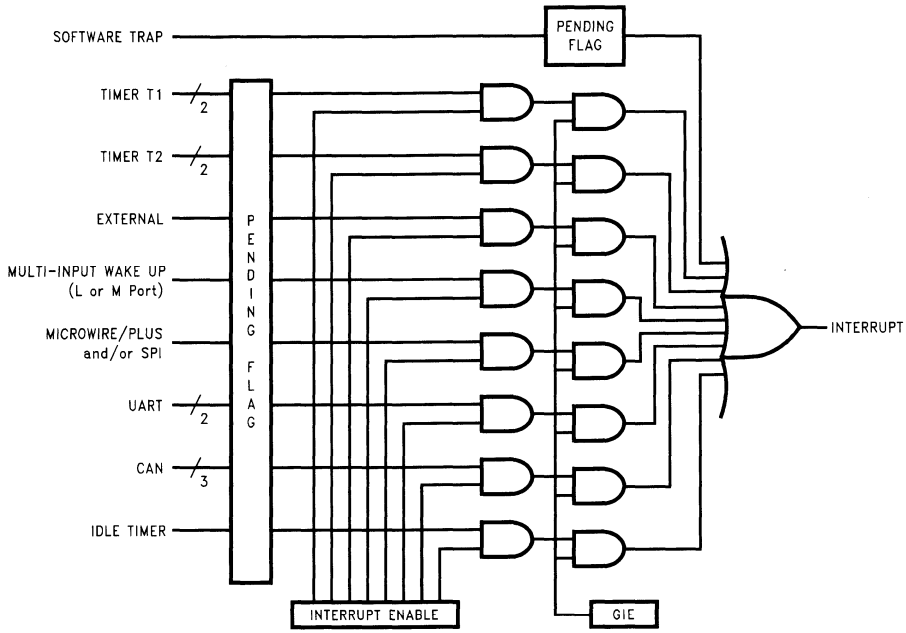
The Software Trap (ST) is a special kind of non-maskable interrupt which occurs when the INTR instruction (used to acknowledge interrupts) is fetched from ROM and placed inside the instruction register. This may happen when the PC is pointing beyond the available ROM address space or when the stack is over-popped.

When an ST occurs, the user can re-initialize the stack pointer and do a recovery procedure (similar to RESET, but not necessarily containing all of the same initialization procedures) before restarting.

The occurrence of an ST is latched into the ST pending bit. The GIE bit is not affected and the ST pending bit (**not accessible by the user**) is used to inhibit other interrupts and to direct the program to the ST service routine with the VIS instruciton. The RPND instruction is used to clear the software interrupt pending bit. This bit is also cleared on reset.

The ST as the highest rank among all interrupts.

Nothing (except another ST) can interrupt an ST being serviced.



TL/DD/12871-15

FIGURE 14. Interrupt Block Diagram

CAN Block Description *

This device contains a CAN serial bus interface as described in the CAN Specification Rev. 2.0 part B.

*Patents Pending.

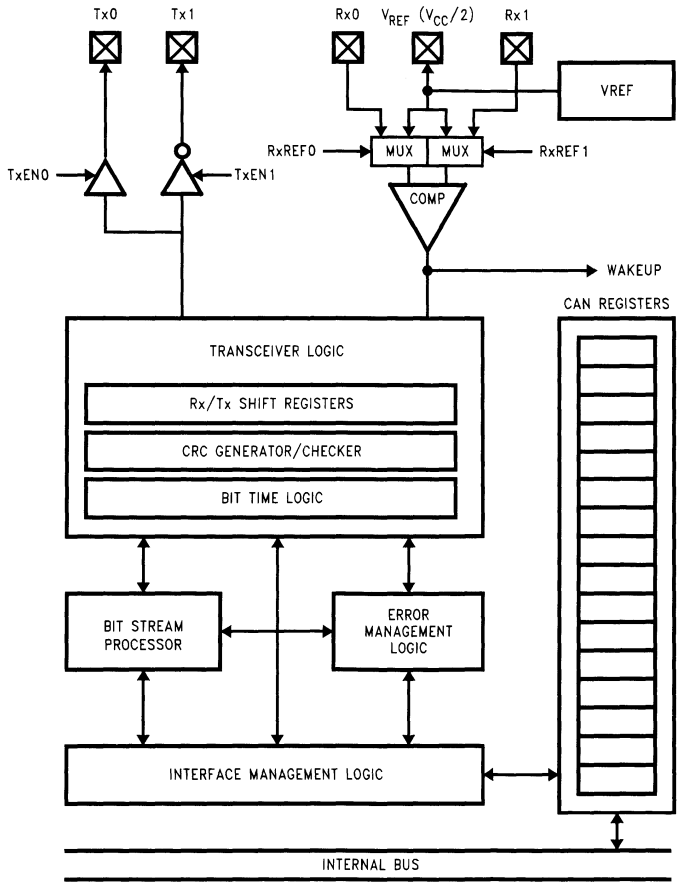
CAN Interface Block

This device supports applications which require a low speed CAN interface. It is designed to be programmed with two transmit and two receive registers. The user's program may check the status bytes in order to get information of the bus state and the received or transmitted messages. The device has the capability to generate an interrupt as soon as one byte has been transmitted or received. Care must be taken if more than two bytes in a message frame are to be transmitted/received. In this case the user's program must poll the transmit buffer empty (TBE)/receive buffer full (RBF) bits or enable their respective interrupts and perform a data exchange between the user data and the Tx/Rx registers.

Fully automatic transmission on error is supported for messages not longer than two bytes. Messages which are longer than two bytes have to be processed by software.

The interface is compatible with CAN Specification 2.0 part B, without the capability to receive/transmit extended frames. Extended frames on the bus are checked and acknowledged according to the CAN specification.

The maximum bus speed achievable with the CAN interface is a function of crystal frequency, message length and software overhead. The device can support a bus speed of up to 1 Mbit/s with a 10 MHz oscillator and 2 byte messages. The 1 Mbit/s bus speed refers to the rate at which protocol and data bits are transferred on the bus. Longer messages require slower bus speeds due to the time required for software intervention between data bytes. The device will support a maximum of 125k bit/s with eight byte messages and a 10 MHz oscillator.



TL/DD/12871–16

FIGURE 15. CAN Interface Block Diagram

Functional Block Description of the CAN Interface

Interface Management Logic (IML)

The IML executes the CPU's transmission and reception commands and controls the data transfer between CPU, Rx/Tx and CAN registers. It provides the CAN Interface with Rx/Tx data from the memory mapped Register Block. It also sets and resets the CAN status information and generates interrupts to the CPU.

Bit Stream Processor (BSP)

The BSP is a sequencer controlling the data stream between The Interface Management Logic (parallel data) and the bus line (serial data). It controls the transceive logic with regard to reception and arbitration, and creates error signals according to the bus specification

Transceive Logic (TCL)

The TCL is a state machine which incorporates the bit stuff logic and controls the output drivers, CRC logic and the Rx/Tx shift registers. It also controls the synchronization to the bus with the CAN clock signal generated by the BTL.

Error Management Logic (EML)

The EML is responsible for the fault confinement of the CAN protocol. It is also responsible for changing the error counters, setting the appropriate error flag bits and interrupts and changing the error status (passive, active and bus off).

Cyclic Redundancy Check (CRC) Generator and Register

The CRC Generator consists of a 15-bit shift register and the logic required to generate the checksum of the destuffed bit-stream. It informs the EML about the result of a receiver checksum.

The checksum is generated by the polynomial:

$$\chi^{15} + \chi^{14} + \chi^{10} + \chi^{8} + \chi^{7} + \chi^{4} + \chi^{3} + 1$$

Receive/Transmit (Rx/Tx) Registers

The Rx/Tx registers are 8-bit shift registers controlled by the TCL and the BSP. They are loaded or read by the Interface Management Logic, which holds the data to be transmitted or the data that was received.

Bit Time Logic (BTL)

The bit time logic divider divides the CKI input clock by the value defined in the CAN prescaler (CSCAL) and bus timing register (CTIM). The resultig bit time (tcan) can be computed by the formula:

$$t_{can} = \frac{CKI}{(1 + divider) \times (1 + 2 \times PS + PPS)}$$

Where *divider* is the value of the clock prescaler, *PS* is the programmable value of phase segment 1 and 2 (1..8) and *PPS* the programmed value of the propagation segment (1..8) (located in CTIM).

Bus Timing Considerations

The internal architecture of the CAN interface has been optimized to allow fast software response times within messages of more than two data bytes. The TBE (Transmit Buffer Empty) bit is set on the last bit of odd data bytes when CAN internal sample points are high.

It is the user's responsibility to ensure that the time between setting TBE and a reload of TxD2 is longer than the length of phase segment 2 as indicated in the following equation:

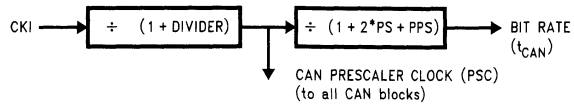
$$t_{LOAD} > \frac{(PS + 1) \times (CSCAL + 1)}{10} t_C = \text{absolute length of PS2}$$

Table V shows examples of the minimum required t_{LOAD} for different CSCAL settings based on a clock frequency of 10 MHz. Lower clock speeds require recalculation of the CAN bit rate and the mimimum t_{LOAD}.

TABLE V. CAN Timing (CKI = 10 MHz t_c = 1 μs)

PS	CSCAL	CAN Bit Rate (kbit/s)	Minimum t_{LOAD} (μs)
4	3	250	2.0
4	9	100	5.0
4	15	62	8.0
4	24	40	12.5
4	39	25	20
4	99	10	50
4	199	5	100

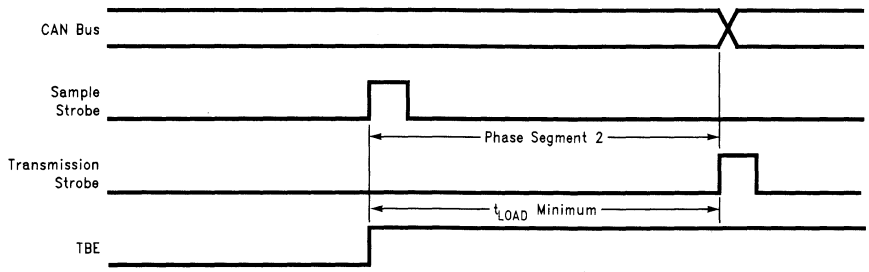
Functional Block Description of the CAN Interface (Continued)



TL/DD/12871–17

FIGURE 16. Bit Rate Generation

Figure 17 illustrates the minimum time required for t_{LOAD}.



TL/DD/12871–18

FIGURE 17. TBE Timing

In the case of an interrupt driven CAN interface, the calculation of the actual t_{LOAD} time would be done as follows:

```
INT:                                ;Interrupt latency = 7tc = 7 µs
                PUSH        A       ;3tc = 3 µs
                LD          A,B     ;2tc = 2 µs
                PUSH        A       ;3tc = 3 µs
                VIS                 ;5tc = 5 µs
CANTX:                              ;20tc = µs to this point
                •                   ;additional time for instructions which check
                •                   ;status prior to reloading the transmit data
                •                   ;registers with subsequent data bytes.
                LD          TXD2,DATA
                •
                •
                •
```

Functional Block Description of the CAN Interface (Continued)

Interrupt driven programs use more time than programs which poll the TBE flag, however programs which operate at lower baud rates (which are more likely to be sensitive to this issue) have more time for interrupt response.

Output Drivers/Input Comparators

The output drivers/input comparators are the physical interface to the bus. Control bits are provided to TRI-STATE the output drivers.

A dominant bit on the bus is represented as a "0" in the data registers and a recessive bit on the bus is represented as a "1" in the data registers.

TABLE VI. Bus Level Definition

Bus Level	Pin Tx0	Pin Tx1	Data
"dominant"	drive low (GND)	dirve high (V_{CC})	0
"recessive"	TRI-STATE	TRI-STATE	1

Register Block

The register block consists of fifteen 8-bit registers which are described in more detail in the following paragraphs.

Note: The contents of the receiver related registers RxD1, RxD2, RDLC, RIDH and RTSTAT are only changed if a received frame passes the acceptance filter or the Receive Identifier Acceptance Filter bit (RIAF) is set to accept all received messages.

TRANSMIT DATA REGISTER 1 (TXD1) (Address X'00A0)

The Transmit Data Register 1 contians the first data byte to be transmitted within a frame and then the successive odd byte numbers (i.e., bytes number 1,3,..,7).

TRANSMIT DATA REGISTER 2 (TXD2) (Address X'00A1)

The Transit Data Register 2 contains the second data byte to be transmitted within a frame and then the successive even byte numbers (i.e., bytes number 2,4,..,8).

TRANSMIT DATA LENGTH CODE AND IDENTIFIER LOW REGISTER (TDLC) (Address X'00A2)

TID3	TID2	TID1	TID0	TDLC3	TDLC2	TCLC1	TDLC0

Bit 7 Bit 0

This register is read/write.

TID3..TIDO Transmit Identifier Bits 3..0 (lower 4 bits)

The transmit identifier is composed of eleven bits in total, bits 3 to 0 of the TID are stored in bits 7 to 4 of this register.

TDLC3..TDLC0 Transmit Data Length Code

These bits determine the number of data bytes to be transmitted within a frame. The CAN specification allows a maximum of eight data bytes in any message.

TRANSMIT IDENTIFIER HIGH (TID) (Address X'00A3)

TRTR	TID10	TID9	TID8	TID7	TID6	TID5	TID4

Bit 7 Bit 0

This register is read/write.

TRTR Transmit Remote Frame Request

This bit is set if the frame to be transmitted is a remote frame request.

TID10..TID4 Transmit Identifier Bits 10 .. 4 (higher 7 bits)

Bits TID10..TID4 are the upper 7 bits of the 11 bit transmit identifier.

RECEIVER DATA REGISTER 1 (RXD1) (Address X'00A4)

The Receive Data Register 1 (RXD1) contains the first data byte received in a frame and then successive odd byte numbers (i.e., bytes 1, 3,..7). This register is read-only.

RECEIVE DATA REGISTER 2 (RXD2) (Address X'00A5)

The Receive Data Register 2 (RXD2) contains the second data byte received in a frame and then successive even byte numbers (i.e., bytes 2,4,..,8). This register is read-only.

REGISTER DATA LENGTH CODE AND IDENTIFIER LOW REGISTER (RIDL) (Address X'00A6)

RID3	RID2	RID1	RID0	RDLC3	RDLC2	RDLC1	RDLC0

Bit 7 Bit 0

This register is read only.

RID3..RID0 Receive Identifier bits (lower four bits)

The RID3..RID0 bits are the lower four bits of the eleven bit long Receive Identifier. Any received message that matches the upper 7 bits of the Receive Identifier (RID10..RID4) is accepted if the Receive Identifier Acceptance Filter (RIAF) bit is set to zero.

RDLC3..RDLC0 Receive Data Length Code bits

The RDLC3..RDLC0 bits determine the number of data bytes within a received frame.

RECEIVE IDENTIFIER HIGH (RID) (Address X'00A7)

unused	RID10	RID9	RID8	RID7	RID6	RID5	RID4

Bit 7 Bit 0

This register is read/write.

RID10..RID4 Receive Identifier bits (upper bits)

The RID10...RID4 bits are the upper 7 bits of the eleven bit long Receive Identifier. If the Receive Identifier Acceptance Filter (RIAF) bit (see CBUS register) is set to zero, bits 4 to 10 of the received identifier are compared with the mask bits of RID4..RID10. If the corresponding bits match, the message is accepted. If the RIAF bit is set to a one, the filter function is disabled and all messages, independent of identifier, will be accepted.

Functional Block Description of the CAN Interface (Continued)

CAN PRESCALER REGISTER (CSCAL) (Address X'00A8)

CKS7	CKS6	CKS5	CKS4	CKS3	CKS2	CKS1	CKS0

Bit 7 Bit 0

This register is read/write.

CKS7..0 Prescaler divider select.

The resulting clock value is the CAN Prescaler clock.

CAN BUS TIMING REGISTER (CTIM) (00A9)

PPS2	PPS1	PPS0	PPS0	PS2	PS1	PS0	Reserved

Bit 7 Bit 0

This register is read/write.

PPS2..PPS0 Propagation Segment, bits 2..0

The PPS2..PPS0 bits determine the length of the propagation delay in Prescaler clock cycles (PSC) per bit time. (For a more detailed discussion of propagation delay and phase segments, see SYNCHRONIZATION on page 41.)

PS2..PS0 Phase Segment 1, bits 2..0

The PS2..PS0 bits fix the number of Prescaler clock cycles per bit time for phase segment 1 and phase segment 2. The PS2..PS0 bits also set the synchronization Jump Width to a value equal to the lesser of the 4 PSC or the length of PS1/2 (Min: 4 I length of PS1/2).

TABLE VII. Synchronization Jump Width

PS2	PS1	PS0	Length of Phase Segment ½	Synchronization Jump Width
0	0	0	1 t_{can}	1 t_{can}
0	0	1	2 t_{can}	2 t_{can}
0	1	0	3 t_{can}	3 t_{can}
0	1	1	4 t_{can}	4 t_{can}
1	0	0	5 t_{can}	4 t_{can}
1	0	1	6 t_{can}	4 t_{can}
1	1	0	7 t_{can}	4 t_{can}
1	1	1	8 t_{can}	4 t_{can}

LENGTH OF TIME SEGMENTS (See *Figure 29*)

- The Synchronization Segment is 1 CAN Prescaler clock (PSC)
- The Propagation Segment can be programmed (PPS) to be 1,2...,8 PSC in length.
- Phase Segment 1 and Phase Segment 2 are programmable (PS) to be 1,2,..,8 PSC long.

Note: (BTL settings at high speed; PSC = 0) Due to the on-chip delay from the rx-pins through the receive comparator (worst case assumption: 3 clocks delay * 2 (devices on the bus) + 1 tx delay) the user needs to set the sample point to > (2*3 + 1) i.e., > 7 CKI clocks to ensure correct communication on the bus under all circumstances. With prescaler settings of > 0 this is a given (i.e., no caution has to be applied).

Example: for 1 Mbit CTIM = b'10000100 (PSS = 5; PS1 = 2). Example for 500 kbit CTIM = b'01011100 (PPS = 3; PS1 = 8). – all at 10 MHz CKI and CSCAL = 0.

CAN BUS CONTROL REGISTER (CBUS) (00AA)

Reserved	RIAF	TxEN1	TxEN0	RxREF1	RxREF0	Reserved	FMOD

Bit 7 Bit 0

Reserved This bit is reserved and should be zero.

RIAF Receive identifier acceptance filter bit

If the RIAF bit is set to zero, bits 4 to 10 of the received identifier are compared with the mask bits of RID4..RID10 and if the corresponding bits match, the message is accepted. If the RIAF bit is set to a one, the filter function is disabled and all messages independent of the identifier will be accepted.

TxEN0, TxEN1 TxD Output Driver Enable

TABLE VIII. Output Drivers

TxEN1	TxEN0	Output
0	0	Tx0, Tx1 TRI-STATED, CAN input comparator disabled
0	1	Tx0 enabled
1	0	Tx1 enabled
1	1	Tx0 and Tx1 enabled

Bus synchronization of the device is done in the following way:

If the output was disabled (TxEN1, TxEN0 = "0") and either TxEN1 or TxEN0, or both are set to 1, the device will not start transmission or reception of a frame until eleven consecutive "recessive" bits have been received. Resetting the TxEN1 and TxEN0 bits will disable the output drivers and the CAN input comparator. All other CAN related registers and flags will be unaffected. It is recommended that the user reset the TxEN1 and TxEN0 bits before switching the device into the HALT mode (the CAN receive wakeup will still work) in order to reduce current consumption and to assure a proper resychronization to the bus after exiting the HALT mode.

Note: A "bus off" condition will also cause Tx0 and Tx1 to be at TRI-STATE (independent of the values of the TxEN1 and TxEN0 bits).

RXREF1 Reference voltage applied to Rx1 if bit is set

RXREF0 Reference voltage applied to Rx0 if bit is set

FMOD Fault Confinement Mode select

Setting the FMOD bit to "0" (default after power on reset) will select the Standard Fault Confinement mode. In this mode the device goes from "bus off" to "error active" after monitoring 128*11 recessive bits (including bus idle) on the bus. This mode has been implemented for compatibility with existing solutions. Setting the FMOD bit to "1" will select the Enhanced Fault Confinement mode. In this mode the device goes from "bus off" to "error active" after monitoring 128 "good" messages, as indicated by the reception of 11 consecutive "recessive" bits including the End of Frame, whereas the standard mode may time out after 128 x 11 recessive bits (e.g., bus idle).

Functional Block Description of the CAN Interface (Continued)

TRANSMIT CONTROL/STATUS (TCNTL) (00AB)

NS1	NS0	TERR	RERR	CEIE	TIE	RIE	TXSS

Bit 7 Bit 0

NS1..NS0 Node Status, i.e., Error Status.

TABLE IX. Node Status

NS1	NS0	Output
0	0	Error active
0	1	Error passive
1	0	Bus off
1	1	Bus off

The Node Status bits are read only.

TERR Transmit Error

This bit is automatically set when an error occurs during the transmission of a frame. TERR can be programmed to generate an interrupt by setting the Can Error Interrupt Enable bit (CEIE). This bit must be cleared by the user's software.

Note: This is used for messages for more than two bytes. If an error occurs during the transmission of a frame with more than 2 data bytes, the user's software has to handle the correct reloading of the data bytes to the TxD registers for retransmission of the frame. For frames with 2 or less data bytes the interface logic of this chip does an automatic retransmission. Regardless of the number of data bytes, the user's software must reset this bit if CEIE is enabled. Otherwise a new interrupt will be generated immediately after return from the interrupt service routine.

RERR Receiver Error

This bit is automatically set when an error occurred during the reception of a frame. RERR can be programmed to generate an interrupt by setting the Can Error Interrupt Enable bit (CEIE). This bit has to be cleared by the user's software.

CEIE CAN Error Interrupt Enable

If set by the user's software, this bit enables the tansmit and receive error interrupts. The interrupt pending flags are TERR and RERR. Resetting this bit with a pending error interrupt will inhibit the interrupt, but will not clear the cause of the interrupt (RERR or TERR). If the bit is then set without clearing the cause of the interrupt, the interrupt will reoccur.

TIE Transmit Interrupt Enable

If set by the user's software, this bit enables the transmit interrupt. (See TBE and TXPND.) Resetting this bit with a pending transmit interrupt will inhibit the interrupt, but will not clear the cause of the interrupt. If the bit is then set without clearing the cause of the interrupt, the interrupt will reoccur.

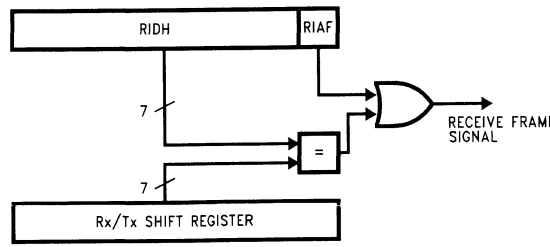
RIE Receive Interrupt Enable

If set by the user's software, this bit enables the receive interrupt or a remote transmission request interrupt (see RBF, RFV and RRTR). Resetting this bit with a pending receive interrupt will inhibit the interrupt, but will not clear the cause of the interrupt. If the bit is then set without clearing the cause of the interrupt, the interrupt will reoccur.

TXSS Transmission Start/Stop

This bit is set by the user's software to initiate the transmission of a frame. Once this bit is set, a transmission is pending, as indicated by the TXPND flag being set. It can be reset by software to cancel a pending transmission. Resetting the TXSS bit will only cancel a transmission, if the transmission of a frame hasn't been started yet (bus idle), if arbitration has been lost (receiving) or if an error occurs during transmission. If the device has already started transmission (won arbitration) the TXPND and TXSS flags will stay set until the transmission is completed, even if the user's software has written zero to the TXSS bit. If one or more data bytes are to be transmitted, care must be taken by the user, that the Transmit Data Register(s) have been loaded before the TXSS bit is set. TXSS will be cleared on three conditions only: Successful completion of a transmitted message; successful cancellation of a pending transmision; Transition of the CAN interface to the bus-off state.

Writing a zero to the TXSS bit will request cancellation of a pending transmission but TXSS will not be cleared until completion of the operation. If an error occurs during transmission of a frame, the logic will check for cancellation requests prior to restarting transmission. If zero has been written to TXSS, retransmission will be canceled.



TL/DD/12871-19

FIGURE 18. Acceptance Filter Block-Diagram

Functional Block Description of the CAN Interface (Continued)

RECEIVE/TRANSMIT STATUS (RTSTAT) (Address X'00AC)

TBE	TXPND	RRTR	ROLD	RORN	RFV	RCV	RBF
1	0	0	0	0	0	0	0

Bit 7 Bit 0

This register is read only.

TBE Transmit Buffer Empty

This bit is set as soon as the TxD2 register is copied into the Rx/Tx shift register, i.e., the 1st data byte of each pair has been transmitted. The TBE bit is automatically reset if the TxD2 register is written (the user should write a dummy byte to the TxD2 register when transmitting an odd number of bytes of zero bytes). TBE can be programmed to generate an interrupt by setting the Transmit Interrupt Enable bit (TIE). When servicing the interrupt the user has to make sure that TBE gets cleared by executing a WRITE instruction on the TxD2 register, otherwise a new interrupt will be generated immediately after return from the interrupt service routine. The TBE bit is read only. It is set to 1 upon reset. TBE is also set upon completion of transmission of a valid message.

TXPND Transmission Pending

This bit is set as soon as the Transmit Start/Stop (TXSS) bit is set by the user. It will stay set until the frame was successfully transmitted, until the transmission was successfully canceled by writing zero to the Transmission Start/Stop bit (TXSS), or the device enters the bus-off state. Resetting the TXSS bit will only cancel a transmission if the transmission of a frame hasn't been started yet (bus idle) or if arbitration has been lost (receiving). If the device has already started transmission (won arbitration) the TXPND flag will stay set until the transmission is completed, even if the user's software has requested cancellation of the message. If an error occurs during transmission, a requested cancellation may occur prior to the begining of retransmission.

RRTR Received Remote Transmission Request

This bit is set when the remote transmission request (RTR) bit in a received frame was set. It is automatically reset through a read of the RXD1 register.

To detect RRTR the user can either poll this flag or enable the receive interrupt (the reception of a remote transmission request will also cause an interrupt if the receive interrupt is enabled). If the receive interrupt is enabled, the user should check the RRTR flag in the service routine in order to distinguish between a RRTR interrupt and a RBF interrupt. It is the responsibility of the user to clear this bit by reading the RXD1 register, before the next frame is received.

ROLD Received Overload Frame

This bit is automatically set when an Overload Frame was received on the bus. It is automatically reset through a read of the Receive/Transmit Status register. It is the responsibility of the user to clear this bit by reading the Receive/Transmit Status register, before the next frame is received.

RORN Receiver Overrun

This bit is automatically set on an overrun of the receive data register, i.e., if the user's program does not maintain the RxDn registers when receiving a frame. It it automatically reset through a read of the Receive/Transmit Status register. It is the responsibility of the user to clear this bit by reading the Receive/Transmit Status register before the next frame is received.

RFV Received Frame Valid

This bit is set if the received frame is valid, i.e., after the penultimate bit of the End of Frame is received. It is automatically reset through a read of the Receive/Transmit Status register. It is the responsibility of the user to clear this bit by reading the receive/transmit status register (RTSTAT), before the next frame is received. RFV will cause a Receive Interrupt if enabled by RIE. The user should be careful to read the last data byte (RxD1) of odd length messages (1, 3, 5 or 7 data bytes) on receipt of RFV. RFV is the only indication that the last byte of the message has been received.

RCV Receive Mode

This bit is set after the data length code of a message that passes the device's acceptance filter has been received. It is automatically reset after the CRC-delimiter of the same frame has been received. It indicates to the user's software that arbitration is lost and that data is coming in for that node.

RBF Receive Buffer Full

This bit is set if the second Rx data byte was received. It is reset automatically, after the RxD1-Register has been read by the software. RBF can be programmed to generate an interrupt by setting the Receive Interrupt Enable bit (RIE). When servicing the interrupt, the user has to make sure that RBF gets cleared by executing a LD instruction from the RxD1 register, otherwise a new interrupt will be generated immediately after return from the interrupt service routine. The RBF bit is read only.

TRANSMIT ERROR COUNTER (TEC) (Address X'00AD)

TEC7	TEC6	TEC5	TEC4	TEC3	TEC2	TEC1	TEC0

Bit 7 Bit 0

This register is read/write.

For test purposes and to identify the node status, the transmit error counter, an 8-bit error counter, is mapped into the data memory. If the lower seven bits of the counter overflow, i.e., TEC7 is set, the device is error passive.

CAUTION

To prevent interference with the CAN fault confinement, the user must not write to the REC/TEC registers. Both counters are automatically updated following the CAN specification.

RECEIVE ERROR COUNTER (REC) (00AE)

ROVL	REC6	REC5	REC4	REC3	REC2	REC1	REC0

Bit 7 Bit 0

This register is read/write.

ROVL receive error counter overflow

For test purposes and to identify the node status the receive error counter, a 7-bit error counter, is mapped into the data memory. If the counter overflows the ROVL bit is set to indicate that the device is error passive and won't transmit any active error frames. If ROVL is set then the counter is frozen.

Functional Block Description of the CAN Interface (Continued)

MESSAGE IDENTIFICATION

a. Transmitted Message

The user can select all 11 Transmit Identifier Bits to transmit any message whigh fulfills the CAN2.0, part B spec without an extended identifier (see note below). Fully automatic retransmission is supported for messages no longer than 2 bytes.

b. Received Messages

The lower four bits of the Receive Identifier are don't care, i.e., the controller will receive all messages that fit in that window (16 messages). The upper 7 bits can be defined by the user in the Receive Identifier High Register to mask out groups of messages. If the RIAF bit is set, all messages will be received.

Note: The CAN interface tolerates the extended CAN frame format of 29 identifier bits and gives an acknowledgment. If an error occurs the receive error counter will be increased, and decreased if the frame is valid.

BUS SYNCHRONIZATION DURING OPERATION

Resetting the TxEN1 and TxEN0 bits in Bus Control Register will disable the output drivers and do a resynchronization to the bus. All other CAN related registers and flags will be unaffected.

Bus synchronization of the device is this case is done in the following way:

If the output was disabled (TxEN1, TxEN0 = "0") and either TxEN1 or TxEN0, or both are set to 1, the device will not start transmission or reception of a frame until eleven consecutive "recessive" bits have been received.

A "bus off" condition will also cause the output drivers Tx1 and Tx0 to be at TRI-STATE (independent of the status of TxEN1 and TxEN0). The device will switch from "bus off" to "error active" mode as described under the FMOD-bit description (see Can Bus Control register). This will ensure that the device is synchronized to the bus, before starting to transmit or receive.

For information on bus synchronization and status of the CAN related registers after external reset refer to the RESET section.

ON-CHIP VOLTAGE REFERENCE

The on-chip voltage reference is a ratiometric reference. For electrical characteristics of the voltage reference refer to the electrical specifications section.

ANALOG SWITCHES

Analog switches are used for selecting between Rx0 and V_{REF} and between Rx1 and V_{REF}.

Basic CAN Concepts

The following paragraphs provide a generic overview of the basic concepts of the Controller Area Network (CAN) as described in *Chapter 4 of ISO/DIS11519-1*. Implementation related issues of the National Semiconductor device will be discussed as well.

This device will process standard frame format only. Extended frame formats will be acknowledged, however the data will be discarded. For this reason the description of frame formats in the following section will cover only the standard frame format.

The following section provides some more detail on how the device will handle received extended frames:

If the device's remote identifier acceptance filter bit (RIAF) is set to "1", extended frame messages will be acknowledged. However, the data will be discarded and the device will not reply to a remote transmission request received in extended frame format. If the device's RIAF bit is set to "0", the upper 7 received ID bits of an extended frame that match the device's receive identifier (RID) acceptance filtler bits, are stroed in the device's RID register. However, the device does not reply to an RTR and any data is discarded. The device will only acknowledge the message.

MULTI-MASTER PRIORITY BASED BUS ACCESS

The CAN protocol is message based protocol that allows a total of 2032 ($= 2^{11} - 16$) different messages in the standard format and 512 million ($= 2^{29} - 16$) different messages in the extended frame format.

MULTICAST FRAME TRANSFER BY ACCEPTANCE FILTERING

Every CAN Frame is put on the common bus. Each module receives every frame and filters out the frames which are not required for the module's task.

REMOTE DATA REQUEST

A CAN master module has the ability to set a specific bit called the "remote transmission request bit" (RTR) in a frame. This causes another module, either another master or a slave, to transmit a data frame after the current frame has been completed.

SYSTEM FLEXIBILITY

Additional modules can be added to an existing network without a configuration change. These modules can either perform completely new functions requiring new data or process existing data to perform a new function.

SYSTEM WIDE DATA CONSISTENCY

As the CAN network is message oriented, a message can be used like a variable which is automatically updated by the controlling processor. If any module cannot process information it can send an overload frame. The device is incapable of initiating an overload frame, but will join a overload frame initiated by another device as required by CAN specifications.

NON-DESTRUCTIVE CONTENTION-BASED ARBITRATION

The CAN protocol allows several transmitting modules to start a transmission at the same time as soon as they monitor the bus to be idle. During the start of transmission every node monitors the bus line to detect whether its message is overwritten by a message with a higher priority. As soon as a transmitting module detects another module with a higher priority accessing the bus, it stops transmitting its own frame and switches to receive mode. For illustration see *Figure 19*.

AUTOMATIC RETRANSMISSION OF FRAMES

If a data or remote frame is overwritten by either a higher-prioritized data frame, remote frame or an error frame, the transmitting module will automatically retransmit it. This device will handle the automatic retransmission of up to two data bytes automatically. Messages with more than 2 data bytes require the user's software to update the transmit registers.

Basic CAN Concepts (Continued)

ERROR DETECTION AND ERROR SIGNALING

All messages on the bus are checked by each CAN node and acknowledge if they are correct. If any node detects an error it starts the transmission of an error frame.

Switching Off Defective Nodes

There are two error counters, one for transmitted data and one for received data, which are incremented, depending on the error type, as soon as an error occurs. If either counter goes beyond a specific value the node goes to an error state. A valid frame causes the error counters to decrease.

The device can be in one of three states with respect to error handling:

- Error active
  An error active unit can participate in bus communication and sends an active ("dominant") error flag.
- Error passive
  An error passive unit can participate in bus communication. However, if the unit detects an error it is not allowed to send an active error flag. The unit sends only a passive ("recessive") error flag.
- Bus off

A unit that is "bus off" has the output drivers disabled, i.e., it does not participate in any bus activity.

(See ERROR MANAGEMENT AND DETECTION for more detailed information.)

Frame Formats

INTRODUCTION

There are basically two different types of frames used in the CAN protocol.

The data transmission frames are: data/remote frame

The control frames are: error/overload frame

Note: This device cannot send an overload frame as a result of not being able to process all information. However, the device is able to recognize an overload condition and join overload frames initiated by other devices.

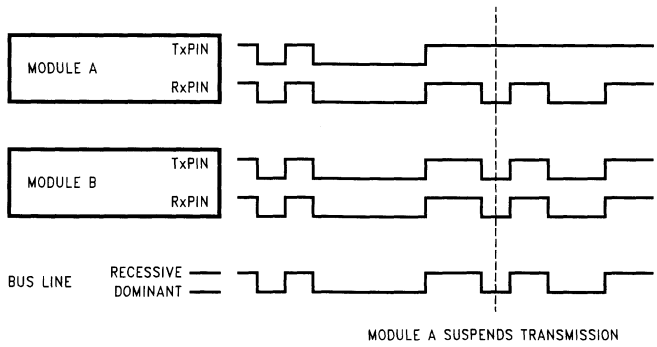
If no message is being transmitted, i.e., the bus is idle, the bus is kept at the "recessive" level. *Figure 20* and *Figure 21* give an overview of the various CAN frame formats.

DATA AND REMOTE FRAME

Data frames consist of seven bit fields and remote frames consist of six different bit fields:

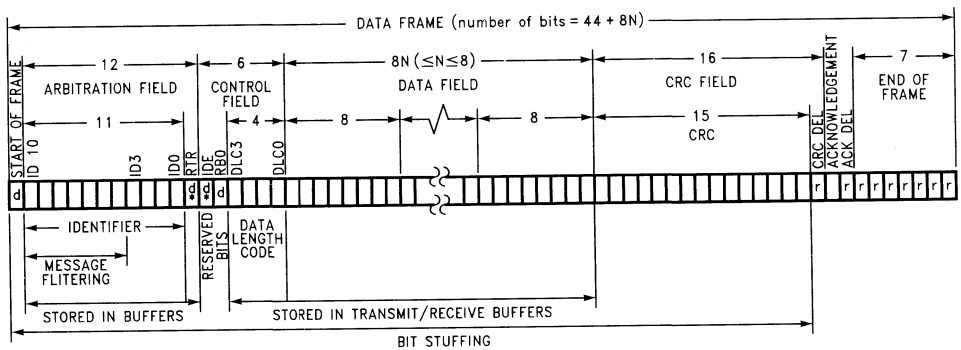
1. Start of Frame (SOF)
2. Arbitration field
3. Control field (IDE bit, R0 bit, and DLC field)
4. Data field (not in remote frame)
5. CRC field
6. ACK field
7. End of Frame (EOF)

A remote frame has no data field and is used for requesting data from other (remote) CAN nodes. *Figure 22* shows the format of a CAN data frame.

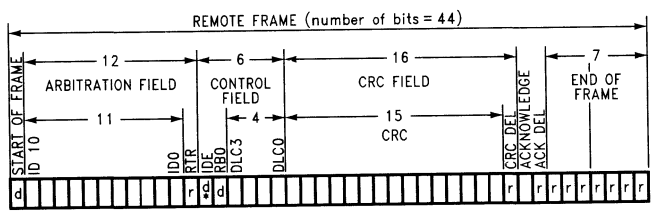


TL/DD/12871–20

FIGURE 19. CAN Message Arbitration



TL/DD/12871-21



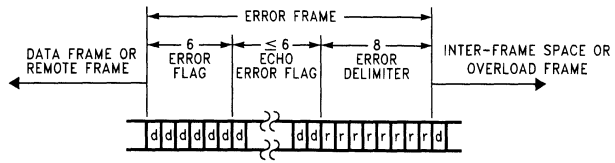
TL/DD/12871-22

A remote frame is identical to a data frame, except that the RTR bit is "recessive", and there is no data field.

IDE = Identifier Extension Bit
The IDE bit in the standard format is transmitted "dominant", whereas in the extended format the IDE bit is "recessive" and the id is expanded to 29 bits.
r = recessive
d = dominant

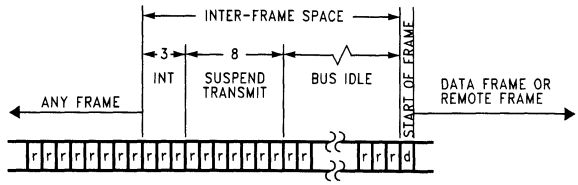
FIGURE 20. CAN Data Transmission Frames

Frame Formats (Continued)



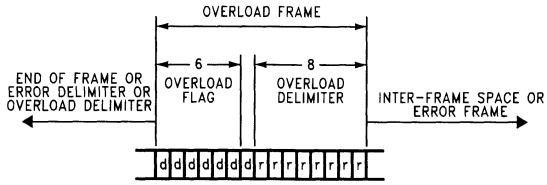
TL/DD/12871–23

An error frame can start anywhere in the middle of a frame.



TL/DD/12871–24

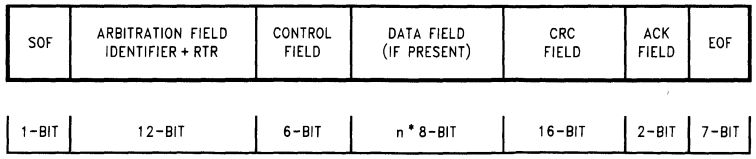
INT = Intermission
Suspend Transmission is only for error passive nodes.



TL/DD/12871–25

An overload frame can only start at the end of a frame.

FIGURE 21. CAN Control Frames



$n \in (0,8)$

TL/DD/12871–26

FIGURE 22. CAN Frame Format

Frame Formats (Continued)

FRAME CODING

Remote and Data Frames are NRZ codes with bit-stuffing in every bit field which holds computable information for the interface, i.e., Start of Frame arbitration field, control field, data field (if present) and CRC field.

Error and overload frames are NRZ coded without bit stuffing.

BIT STUFFING

After five consecutive bits of the same value, a stuff bit of the inverted value is inserted by the transmitter and deleted by the receiver.

Destuffed Bit Stream	100000x	011111x
Stuffed Bit Stream	1000001x	0111110x
		x = {0,1}

START OF FRAME (SOF)

The Start of Frame indicates the beginning of data and remote frames. It consists of a single "dominant" bit. A node is only allowed to start transmission when the bus is idle. All nodes have to synchronize to the leading edge (first edge after the bus was idle) caused by SOF of the node which starts transmission first.

ARBITRATION FIELD

The arbitration field is composed of the identifier field and the RTR (Remote Transmission Request) bit. The value of the RTR bit is "dominant" in a data frame and "recessive" in a remote frame.

CONTROL FIELD

The control field consists of six bits. It starts with two bits reserved for future expansion followed by the four-bit Data Length Code. Receivers must accept all possible combinations of the two reserved bits. Until the function of these reserved bits is defined, the transmitter only sends "0" (dominant) bits. The first reserved bit (IDE) is actually defined to indicate an extended frame with 29 Identifier bits if set to "1". CAN chips must tolerate extended frames, even if they can only understand standard frames, to prevent the destruction of an extended frames on an existing network.

The Data Length Code indicates the number of bytes in the data field. This Data Length Code consists of four bits. The data field can be of length zero. The permissible number of data bytes for a data frame ranges from 0 to 8.

DATA FIELD

The Data field consists of the data to be transferred within a data frame. It can contain 0 to 8 bytes and each byte contains 8 bits. A remote frame has no data field.

CRC FIELD

The CRC field consists of the CRC sequence followed by the CRC delimiter. The CRC sequence is derived by the transmitter from the modulo 2 division of the preceding bit fields, starting with the SOF up to the end of the data field, excluding stuff-bits, by the generator polynomial:

$$\chi^{15} + \chi^{14} + \chi^{10} + \chi^{8} + \chi^{7} + \chi^{4} + \chi^{3} + 1$$

The remainder of this division is the CRC sequence transmitted over the bus. On the receiver side the module divides all bit fields up to the CRC delimiter, excluding stuff-bits, and checks if the result is zero. This will then be interpreted as a valid CRC. After the CRC sequence a single "recessive" bit is transmitted as the CRC delimiter.

ACK FIELD

The ACK field is two bits long and contains the ACK slot and the ACK delimiter. The ACK slot is filled with a "recessive" bit by the transmitter. This bit is overwritten with a "dominant" bit by every receiver that has received a correct CRC sequence. The second bit of the ACK field is a "recessive" bit called the acknowledge delimiter. As a consequence the acknowledge flag of a valid frame is surrounded by two "recessive" bits, the CRC-delimiter and the ACK delimiter.

EOF FIELD

The End of Frame Field closes a data and a remote frame. It consists of seven "recessive" bits.

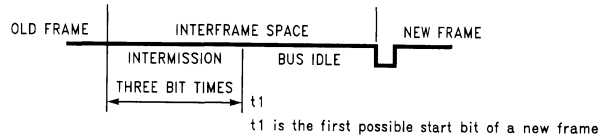
INTERFRAME SPACE

Data and remote frames are separate from every preceding frame (data, remote, error and overload frames) by the interframe space see *Figure 23* and *Figure 24* for details. Error and overload frames are not preceded by an interframe space. They can be transmitted as soon as the condition occurs. The interframe space consists of a minimum of three bit fields depending on the error state of the node.

These bit fields are coded as follows:

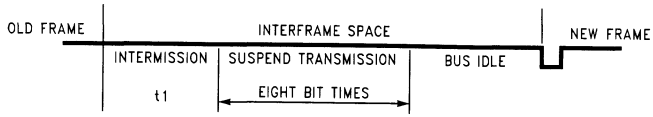
The intermission has the fixed form of three "recessive" bits. While this bit field is active, no node is allowed to start a transmission of a data or a remote frame. The only action to be taken is signaling an overload condition. This means that an error in this bit field would be interpreted as an overload condition. Suspend transmission has to be inserted by error-passive nodes that were transmitter for the last message. This bit field has the form of eight "recessive" bits. However, it may be overwritten by a "dominant" start-bit from another non error passive node which starts transmission. The bus idle field consists of "recessive" bits. Its length is not specified and depends on the bus load.

Frame Formats (Continued)



TL/DD/12871-27

FIGURE 23. Interframe Space for Nodes Which Are Not Error Passive or Have Been Receiver for the Last Frame



TL/DD/12871-28

FIGURE 24. Interframe Space for Nodes Which Are Error Passive and Have Been Transmitter for the Last Frame

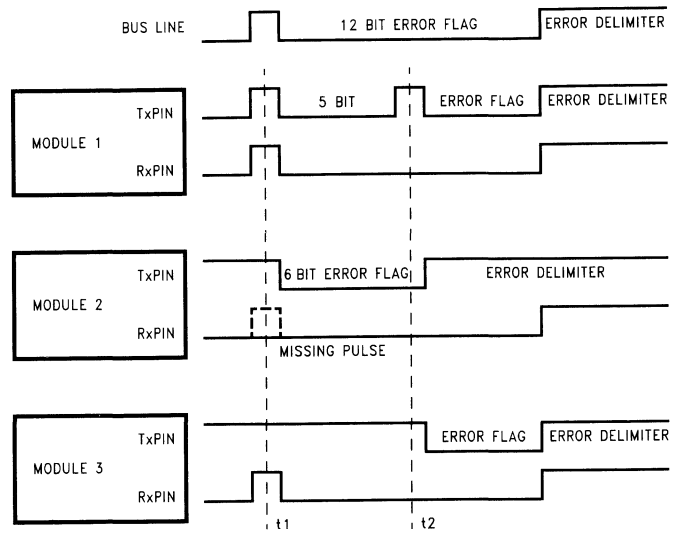
ERROR FRAME

The Error Frame consists of two bit fields: the error flag and the error delimiter. The error field is built up from the various error flags of the different nodes. Therefore, its length may vary from a minimum of six bits up to a maximum of twelve bits depending on when a module detects the error. Whenever a bit error, stuff error, form error, or acknowledgment error is detected by a node, this node starts transmission of the error flag at the next bit. If a CRC error is detected, transmission of the error flag starts at the bit following the acknowledge delimiter, unless an error flag for a previous error condition has already been started. *Figure 25* shows how a local fault at one module (module 2) leads to a 12-bit error frame on the bus.

The bus level may either be "dominant" for an error-active node or "recessive" for an error-passive node. An error active node detecting an error, starts transmitting an active error flag consisting of six "dominant" bits. This causes the destruction of the actual frame on the bus. The other nodes detect the error flag as either a violation of the rule of bit-stuffing or the value of a fixed bit field is destroyed. As a consequence all other nodes start transmission of their own error flag. This means, that the error sequence which can be monitored on the bus as a maximum length of twelve bits. If an error passive node detects an error it transmits six "recessive" bits on the bus. This sequence does not destroy a message sent by another node and is not detected by other nodes. However, if the node detecting an error was the transmitter of the frame the other modules will get an error condition by a violation of the fixed bit or stuff rule. *Figure 26* shows how an error passive transmitter transmits a passive error frame and when it is detected by the receivers.

After any module has transmitted its active or passive error flag it waits for the error delimiter which consists of eight "recessive" bits before continuing.

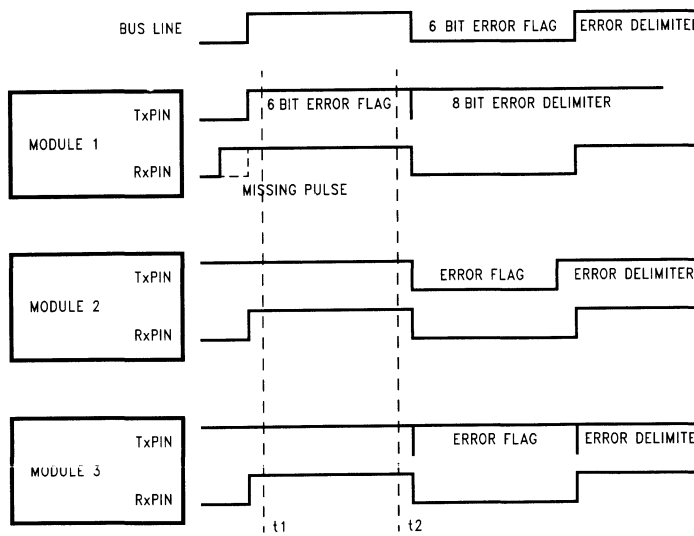
Frame Formats (Continued)



TL/DD/12871–29

module 1 = error active transmitter detects bit error at t2
module 2 = error active receiver with a local fault at t1
module 3 = error active receiver detects stuff error at t2

FIGURE 25. Error Frame—Error Active Transmitter



TL/DD/12871–30

module 1 = error active receiver with a local fault at t1
module 2 = error passive transmitter detects bit error at t2
module 3 = error passive receiver detects stuff error at t2

FIGURE 26. Error Frame—Error Passive Transmitter

Frame Formats (Continued)

OVERLOAD FRAME

Like an error frame, an overload frame consists of two bit fields: the overload flag and the overload delimiter. The bit fields have the same length as the error frame field: six bits for the overload flag and eight bits for the delimiter. The overload frame can only be sent after the end of frame (EOF) field and in they way destroys the fixed form of the intermission field.

ORDER OF BIT TRANSMISSION

A frame is transmitted starting with the Start of Frame, sequentially followed by the remaining bit fields. In every bit field the MSB is transmitted first.

FRAME VALIDATION

Frames have a different validation point for transmitters and receivers. A frame is valid for the transmitter of a message, if there is no error until the end of the last bit of the End of Frame field. A frame is valid for a receiver, if there is no error until and including the end of the penultimate bit of the End of Frame.

FRAME ARBITRATION AND PRIORITY

Except for an error passive node which transmitted the last frame, all nodes are allowed to start transmission of a frame after the intermission, which can lead to two or more nodes starting transmission at the same time. To prevent a node from destroying another node's frame, it monitors the bus during transmission of the identifier field and the RTR-bit. As soon as it detects a "dominant" bit while transmitting a "recessive" bit it releases the bus, immediately stops transmission and starts receiving the frame. This causes no data or remote frame to be destroyed by another. Therefore the highest priority message with the identifier 0x000 out of 0x7EF (including the remote data request (RTR) bit) always gets the bus. This is only valid for standard CAN frame format. Note that while the CAN specification allows valid standard identifiers only in the range 0x000 to 0x7EF, the device will allow identifiers to 0x7FF.

There are three more items that should be taken into consideration to avoid unrecoverable collisions on the bus:

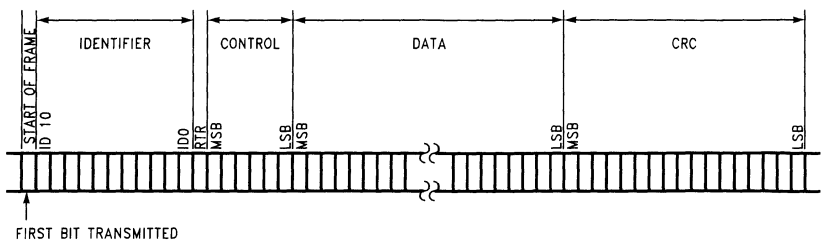
- Within one system each message must be assigned a unique identifier. This is to prevent bit errors, as one module may transmit a "dominant" data bit while the other is transmitting a "recessive" data bit. This could happen if two or more modules start transmission of a frame at the same time and all win arbitration.
- Data frames with a given identifier and a non-zero data length code may be initiated by one node only. Otherwise, in worst case, two nodes would count up to the bus-off state, due to bit errors, if they always start transmitting the same ID with different data.
- Every remote frame should have a system-wide data length code (DLC). Otherwise two modules starting transmission of a remote frame at the same time will overwrite each other's DLC which result in bit errors.

ACCEPTANCE FILTERING

Every node may perform acceptance filtering on the identifier of a data or a remote frame to filter out the messages which are not required by the node. In they way only the data of frames which match the acceptance filter is stored in the corresponding data buffers. However, every node which is not in the bus-off state and has received a correct CRC-sequence acknowledges each frame.

ERROR MANAGEMENT AND DETECTION

There are multiple mechanisms in the CAN protocol, to detect errors and to inhibit erroneous modules from disabling all bus activities.



TL/DD/12871-31

FIGURE 27. Order of Bit Transmission within a CAN Frame

Frame Formats (Continued)

The following errors can be detected:

- Bit Error
  A CAN device that is sending also monitors the bus. If the monitored bit value is different from the bit value that is sent, a bit error is detected. The reception of a "dominant" bit instead of a "recessive" bit during the transmission of a passive error flag, during the stuffed bit stream of the arbitration field or during the acknowledge slot, is not interpreted as a bit error.
- Stuff error
  A stuff error is detected, if the bit level after 6 consecutive bit times has not changed in a message field that has to be coded according to the bit stuffing method.
- Form Error
  A form error is detected, if a fixed frame bit (e.g., CRC delimiter, ACK delimiter) does not have the specified value. For a receiver a "dominant" bit during the last bit of End of Frame does NOT constitute a form error.
- Bit CRC Error
  A CRC error is detected if the remainder of the CRC calculation of a received CRC polynomial is non-zero.
- Acknowledgment Error
  An acknowledgment error is detected whenever a transmitting node does not get an acknowledgment from any other node (i.e., when the transmitter does not receive a "dominant" bit during the ACK frame).

The device can be in one of three states with respect to error handling:

- Error active
  An error active unit can participate in bus communication and sends an active ("dominant") error flag.
- Error passive
  An error passive unit can participate in bus communication. However, if the unit detects an error it is not allowed to send an active error flag. The unit sends only a passive ("recessive") error flag. A device is error passive when the transmit error counter is greater than 127 or when the receive error counter is greater than 127. A device becoming error passive sends an active error flag. An error passive device becomes error active again when both transmit and receive error counter are less than 128.
- Bus off
  A unit that is "bus off" has the output drivers disabled, i.e., it does not participate in any bus activity. A device is bus off when the transmit error counter is greater than 255. A bus off device will become error active again in one of two ways depending on which mode is selected by the user through the Fault Confinement Mode select bit (FMOD) in the CAN Bus Control Register (CBUS). Setting the FMOD bit to "0" (default after power on reset) will select the Standard Fault Confinement mode. In this mode the device goes from "bus off" to "error active" after monitoring 128*11 recessive bits (including bus idle) on the bus. This mode has been implemented for compatibility reasons with existing solutions. Setting the FMOD bit to "1" will select the Enhanced Fault Confinement mode. In this mode the device goes from "bus off" to "error active" after monitoring 128 "good" messages, as indicated by the reception of 11 consecutive "recessive" bits including the End of Frame. The enhanced mode offers the advantage that a "bus off" device (i.e., a device with a serious fault) is not allowed to destroy any messages on the bus until other devices can transmit at least 128 messages. This is not guaranteed in the standard mode, where a defective device could seriously impact bus communication. When the device goes from "bus off" to "error active", both error counters will have the value "0".

In each CAN module there are two error counters to perform a sophisticated error management. The receive error counter (REC) is 7 bits wide and switches the device to the error passive state if it overflows. The transmit error counter (TEC) is 8 bits wide. If it is greater than 127, the device is switched to the error passive state. As soon as the TEC overflows, the device is switched bus-off, i.e., it does not participate in any bus activity.

Frame Formats (Continued)

The counters are modified by the device's hardware according to the following rules:

TABLE X. Receive Error Counter Handling

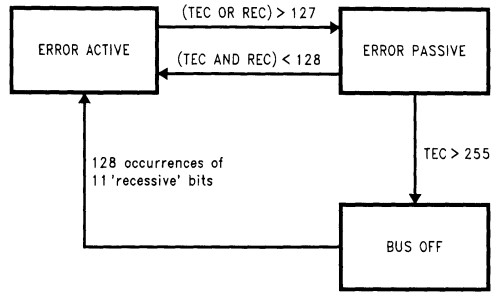
Condition	Receive Error Counter
A receiver detects a Bit Error during sending an active error flag.	Increment by 8
A receiver detects a "dominant" bit as the first bit after sending an error flag.	Increment by 8
After detecting the 14th consecutive "dominant" bit following an active error flag or overload flag or after detecting the 8th consecutive "dominant" bit following a passive error flag. After each sequence of additional 8 consecutive "dominant" bits.	Increment by 8
Any other error condition (stuff, frame, CRC, ACK).	Increment by 1
A valid reception or transmission.	Decrement by 1 if Counter is not 0

TABLE XI. Transmit Error Counter Handling

Condition	Transmit Error Counter
A transmitter detects a Bit Error during sending an active error flag.	Increment by 8
After detecting the 14th consecutive "dominant" bit following an active error flag or overload flag or after detecting the 8th consecutive "dominant" bit following a passive error flag. After each sequence of additional 8 consecutive "dominant" bits.	Increment by 8
Any other error condition (stuff, frame, CRC, ACK).	Increment by 8
A valid reception or transmission.	Decrement by 1 if Counter is not 0

Special error handling for the TEC counter is performed in the following situations:

- A stuff error occurs during arbitration, when a transmitted "recessive" stuff bit is received as a "dorminant" bit. This does not lead to an incrementation of the TEC.
- An ACK-error occurs in an error passive device and no "dominant" bits are detected while sending the passive error flag. This does not lead to an incrementation of the TEC.
- If only one device is on the bus and this device transmits a message, it will get no acknowledgment. This will be detected as an error and message will be repeated. When the device goes "error passive" and detects an acknowledge error, the TEC counter is not incremented. Therefore the device will not go from "error passive" to the "bus off" state due to such a condition.



TL/DD/12871–32

FIGURE 28. CAN Bus States

Figure 28 shows the connection of different bus states according to the error counters.

SYNCHRONIZATION

Every receiver starts with a "hard synchronization" on the falling edge of the SOF bit. One bit time consists of four bit segments: Synchronization segment, propagation segment, phase segment 1 and phase segment 2.

A falling edge of the data signal should be in the synchronization segment. This segment has the fixed length of one time quanta. To compensate for the various delays within a network, the propagation segment is used. Its length is programmable from 1 to 8 time quanta. Phase segment 1 and phase segment 2 are used to resynchronize during an active frame. The length of these segments is from 1 to 8 time quanta long.

Two types of synchronization are supported:

Hard synchronization is done with the falling edge on the bus while the bus is idle, which is then interpreted as the SOF. It restarts the internal logic.

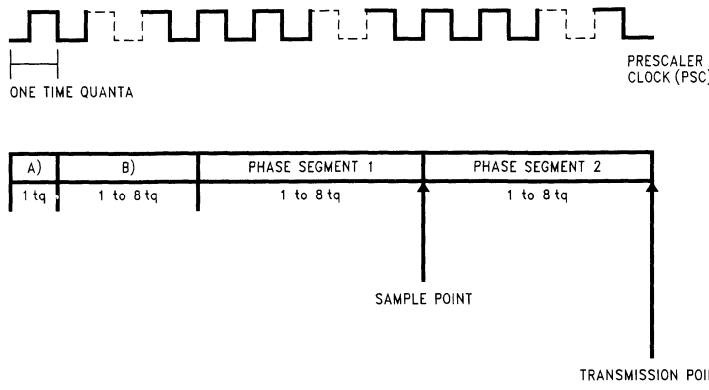
Soft synchronization is used to lengthen or shorten the bit time while a data or remote frame is received. Whenever a falling edge is detected in the propagation segment or in phase segment 1, the segment is lengthened by a specific value, the resynchronization jump width (see *Figure 30*).

If a falling edge lies in the phase segment 2 (as shown in *Figure 30*) it is shortened by the resynchronization jump width. Only one resynchronization is allowed during one bit time. The sample point lies between the two phase segments and is the point where the received data is supposed to be valid. The transmission point lies at the end of phase segment 2 to start a new bit time with the synchronization segment.

Note 1: The resynchronization jump width (RJW) is automatically determined from the programmed value of PS. If a soft resynchronization is done during phase segment 1 or the propagation segment, then RJW will either be equal to 4 internal CAN clocks (CKI/(1 + *divider)*) or the programmed value of PS, whichever is less. PS2 will never be shorter than 1 internal CAN clock.

Note 2: (PS1—BTL settings any PSC setting) The PS1 of the BTL should always be programmed to values greater than 1. To allow device resynchronization for positive and negative phase errors on the bus. (if PS1 is programmed to one, a bit time could only be lengthened and never shortened which basically disables half of the synchronization).

Frame Formats (Continued)



A) Synchronization segment
B) Propagation segment

FIGURE 29. Bit Timing

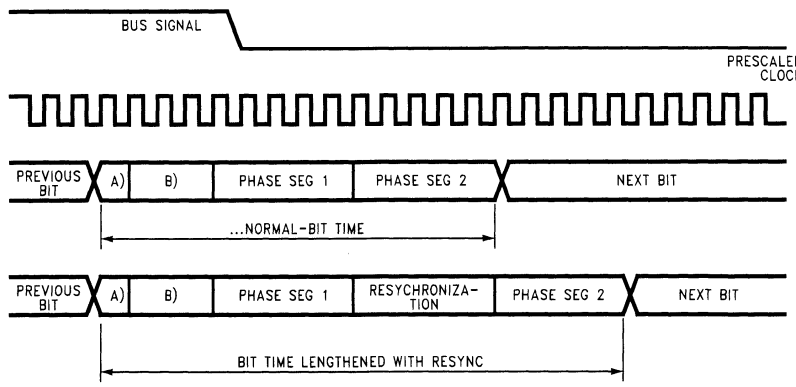


FIGURE 30. Resynchronization 1

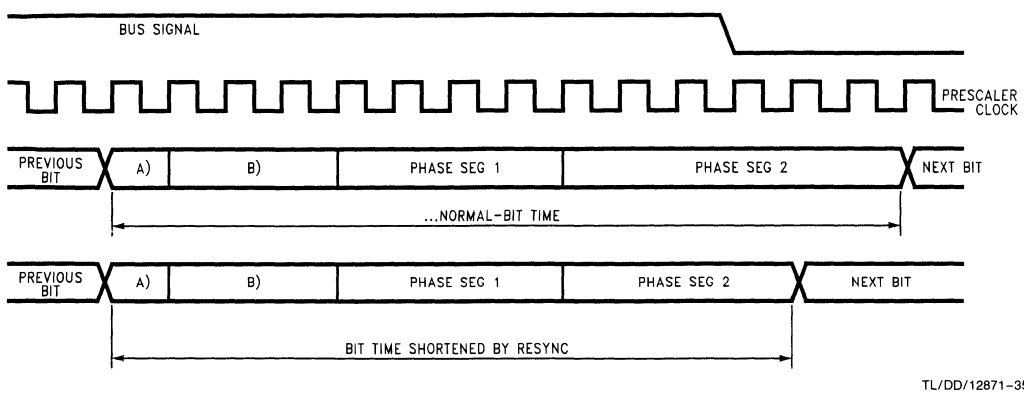


FIGURE 31. Resynchronization 2

Detection of Illegal Conditions

The device can detect various illegal conditions resulting from coding errors, transient noise, power supply voltage drops, runaway programs, etc.

Reading of underfined ROM gets zeros. The opcode for software interrupt is zero. If the program fetches instructions from undefined ROM, this will force a software interrupt, thus signaling that an illegal condition has occurred.

The subroutine stack grows down for each call (jump to subroutine), interrupt, or PUSH, and grows up for each return or POP. The stack pointer is initialized to RAM location 02F Hex during reset. Consequently, if there are more returns than calls, the stack pointer will point to addresses 030 and 031 Hex (which are undefined RAM). Undefined RAM from address 030 to 03F Hex is read as all 1's, which in turn will cause the program to return to address 7FFF Hex. This is an undefined ROM location and the instruction fetched (all 0's) from this location will generate a software interrupt signaling an illegal condition.

Thus, the chip can detect the following illegal conditions:

1. Executing from undefined ROM.
2. Over "POP"ing the stack by having more returns than calls.

When the software interrupt occurs, the user can re-initialize the stack pointer and do a recovery procedure before restarting (this recovery program is probably similar to that following reset, but might not contain the same program initialization procedures).

MICROWIRE/PLUS

MICROWIRE/PLUS is a serial synchronous communications interface. The MICROWIRE/PLUS capability enables the device to interface with any of National Semiconductor's MICROWIRE peripherals (i.e., A/D converters, display drivers, E2PROMs etc.) and with other microcontrollers which support the MICROWIRE interface. It consists of an 8-bit serial shift register (SIO) with serial data input (SI), serial data output (SO) and serial shift clock (SK). *Figure 32* shows a block diagram of the MICROWIRE/PLUS logic.

The shift clock can be selected from either an internal source or an external source. Operating the MICROWIRE/PLUS arrangement with the internal clock source is called the Master mode of operation. Similarly, operating the MICROWIRE/PLUS arrangement with an external shift clock is called the Slave mode of operation.

The CNTRL register is used to configure and control the MICROWIRE/PLUS mode. To use the MICROWIRE/PLUS, the MSEL bit in the CNTRL register is set to one. In the master mode the SK clock rate is selected by the two bits, SL0 and SL1, in the CNTRL register. Table XII details the different clock rates that may be selected.

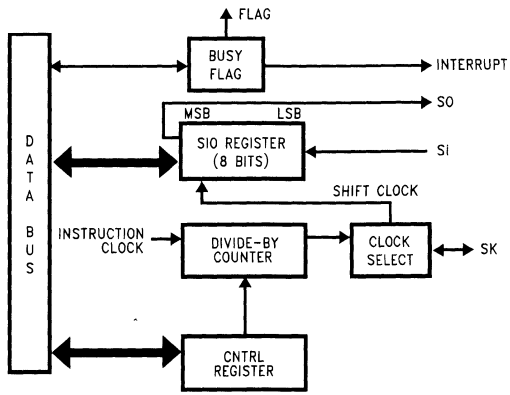
MICROWIRE/PLUS OPERATION

Setting the BUSY bit in the PSW register causes the MICROWIRE/PLUS to start shifting the data. It gets reset when eight data bits have been shifted. The user may reset the BUSY bit by software to allow less than 8 bits to shift. If enabled, an interrupt is generated when eight data bits have been shifted. The device may enter the MICROWIRE/PLUS mode either as a Master or as a Slave. *Figure 33* shows how two COP888 family microcontrollers and several peripherals may be interconnected using the MICROWIRE/PLUS arrangements.

WARNING:

The SIO register should only be loaded when the SK clock is low. Loading the SIO register while the SK clock is high will result in undefined data in the SIO register. SK clock is normally low when not shifting.

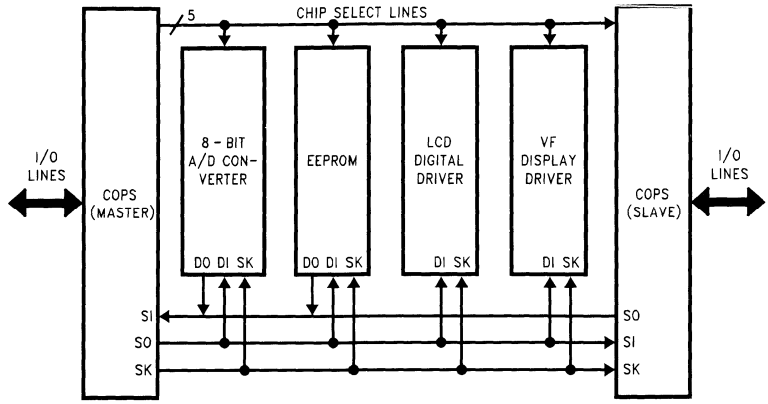
Setting the BUSY flag when the input SK clock is high in the MICROWIRE/PLUS slave mode may cause the current SK clock for the SIO shift register to be narrow. For safety, the BUSY flag should only be set when the input SK clock is low.



TL/DD/12871-36

FIGURE 32. MICROWIRE/PLUS Block Diagram

MICROWIRE/PLUS (Continued)



TL/DD/12871–37

FIGURE 33. MICROWIRE/PLUS Application

MICROWIRE/PLUS Master Mode Operation

In the MICROWIRE/PLUS Master mode of operation the shift clock (SK) is generated internally. The MICROWIRE Master always initiates all data exchanges. The MSEL bit in the CNTRL register must be set to enable the SO and SK functions onto the G Port. The SO and SK pins must also be selected as outputs by setting appropriate bits in the Port G configuraiton register. Table XIII summarizes the bit settings required for Master or Slave mode of operation.

TABLE XII. MICROWIRE/PLUS Master Mode Clock Selection

SL1	SL0	SK
0	0	$2 X t_c$
0	1	$4 X t_c$
1	x	$8 X t_c$

Where t_c is the instruction cycle clock

MICROWIRE/PLUS Slave Mode Operation

In the MICROWIRE/PLUS Slave mode of operation the SK clock is generated by an external source. Setting the MSEL bit in the CNTRL register enables the SO and SK functions onto the G Port. The SK pin must be selected as an input and the SO pin is selected as an output pin by setting and resetting the appropriate bit in the Port G configuration register. Table V summarizes the settings required to enter the Slave mode of operation.

The user must set the BUSY flag immediately upon entering the Slave mode. This will ensure that all data bits sent by the Master will be shifted properly. After eight clock pulses the BUSY flag will be cleared and the sequence may be repeated.

Alternate SK Phase Operation

The device allows either the normal SK clock or an alternate phase SK clock to shift data in and out of the SIO register. In both the modes the SK is normally low. In the normal mode data is shifted in on the rising edge of the SK clock and the data is shifted out on the falling edge of the SK clock. The SIO register is shifted on each falling edge of the SK clock in the normal mode. In the alternate SK phase mode the SIO register is shifted on the rising edge of the SK clock.

A control flag, SKSEL, allows either the normal SK clock or the alternate SK clock to be selected. Resetting SKSEL causes the MICROWIRE/PLUS logic to be clocked from the normal SK signal. Setting the SKSEL flag selects the alternate SK clock. The SKSEL is mapped into the G6 configuration bit. The SKSEL flag will power up in the reset condition, selecting the normal SK signal.

TABLE XIII. MICROWIRE/PLUS Mode Selection

G4 (SO) Config. Bit	G5 (SK) Config. Bit	G4 Fun.	G5 Fun.	Operation
1	1	SO	Int. SK	MICROWIRE/ PLUS Master
0	1	TRI-STATE	Int. SK	MICROWIRE/ PLUS Master
1	0	SO	Ext. SK	MICROWIRE/ PLUS Slave
0	0	TRI-STATE	Ext. SK	MICROWIRE/ PLUS Slave

This table assumes that the control flag MSEL is set.

Serial Peripheral Interface

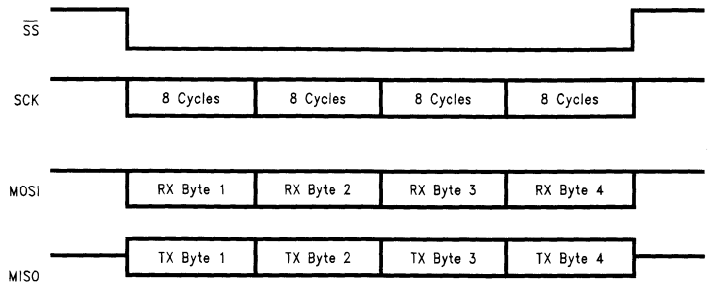


FIGURE 34. SPI Transmission Example

The Serial Peripheral Interface (SPI) is used in master-slave bus systems. It is a synchronous bidirectional serial communication interface with two data lines MISO and MOSI (**M**aster **I**n **S**lave **O**ut, **M**aster **O**ut **S**lave **I**n). A serial clock and a slave select ($\overline{SS}$) signal are always generated by the SPI Master. The interface receives/transmits protocol frames with up to 12 bytes length within a frame, where a frame is defined as the time between a falling edge and a rising edge of $\overline{SS}$.

THEORY OF OPERATION

Figure 36 shows a block diagram illustrating the basic operation of the SPI circuit. In the SPI interface, data is transmitted/received in packets of 8 bits length which are shifted into/out of a shift register with the active edge of the shift clock SCK. Two 12 byte FIFOs, which serve as a receive and a transmit buffer, allow a maximum message length of 12 x 8 bits in both transmit and receive direction without CPU intervention. With CPU intervention, many more bytes can be received. Two registers, the SPI Control Register (SPICNTL) and the SPI Status Register (SPISTAT), are used to control the SPI interface via the internal COP bus. Several different operation modes, such as master or slave operation, are possible.

An $\overline{SS}$-Expander allows the generation of up to 8 signals on the N-port, which can be used as additional $\overline{SS}$-signals ($\overline{ESS}$[7:0]) or as host programmable general purpose signals. The $\overline{SS}$-Expander is programmed with the content of the first MOSI-byte (i.e., the content of the 1st byte [7:0] appears at $\overline{ESS}$[7:0]) (N-port[7:0]), respectively), if the $\overline{ESS}$ programming mode is selected. The $\overline{ESS}$ programming mode is selected by the condition MOSI = L at the falling edge of $\overline{SS}$.

Use of the $\overline{ESS}$ expander requires the setup of four conditions by the user.

1. Set the SESSEN bit of SPICNTL.
2. Set PORTNX to select which bits are used for $\overline{SS}$ expansion.
3. Configure the PORTNC register to enable the desired $\overline{SS}$ expansion bits as outputs.
4. Have an $\overline{ESS}$ condition (MOSI = low at the falling edge of $\overline{SS}$).

Loop Back Mode

Setting the SLOOP bit enables the Loop Back mode, which can be used for test purposes. If the Loop Back mode is selected, TX FIFO data are communicated to the RX FIFO via the SPI Register. In the slave mode, MISO output is internally connected to the MOSI input. In the master mode, the MOSI output is internally connected to the MISO input.

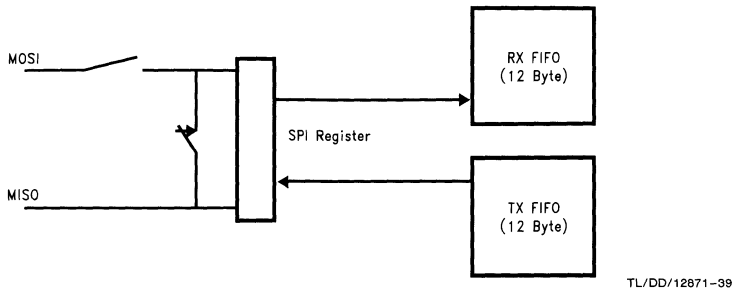
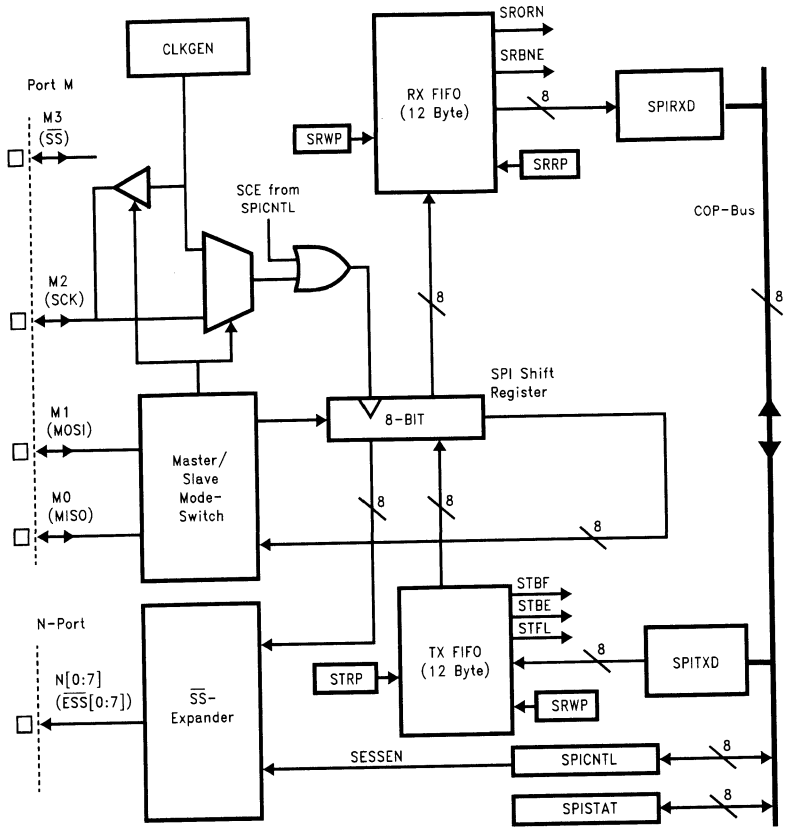


FIGURE 35. Loop Back Mode Block Diagram

Serial Peripheral Interface (Continued)



TL/DD/12871–40

FIGURE 36. SPI Block Diagram

The SPIU Control Register

TABLE XIV. SPI Control (SPICNTL) (0098)

Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
SRIE	STIE	SESSEN	SPIMOD[1:0]		SCE	SPIEN	SLOOP
0	0	0	0	0	0	0	0

Serial Peripheral Interface (Continued)

B7	SRIE	**S**PI **R**eceive **I**nterrupt **E**nable 0—disable receive interrupt 0—enable receive interrupt
B6	STIE	**S**PI **T**ransmit buffer **I**nterrupt **E**nable 0—disable transmit buffer interrupt 0—enable transmit buffer interrupt
B5	SESSEN	**S**PI $\overline{\textbf{SS}}$ Expander ($\overline{\text{ESS}}$) enable 0—The detection of the $\overline{\text{ESS}}$ programming mode is disabled, i.e., the value of MOSI at the falling edge of $\overline{\text{SS}}$ is "don't care". 1—$\overline{\text{ESS}}$ programming mode detection is enabled, i.e., if the condition "MOSI = 0 at the falling edge of $\overline{\text{SS}}$" occurs, the $\overline{\text{SS}}$-Expander is selected and bits [7:0] of the first transmitted byte determine the state of the N-port ($\overline{\text{ESS}}$[7:0]). $\overline{\text{ESS}}$[7:0] will go 1 at the positive edge of $\overline{\text{SS}}$.
B[4:3]	SPIMOD[1:0]	**SPI** operation **mode** select SPIMOD[1:0] 0 0: Slave mode, —SCK is SPI clock input —MISO is SPI data output —MOSI is SPI data input —$\overline{\text{SS}}$ is slave select input 1 0: Standard Master mode, —SCK is SPI clock output (CKI/40) —MISO is SPI data input —MOSI is SPI data output —$\overline{\text{SS}}$ is slave select output In the Master mode, 3 different SPI clock frequencies are available: 0 1: $f_{SCK} = 1/(t_c) = CKI/10$ 1 0: $f_{SCK} = 1/(4\ t_c) = CKI/40$ 1 1: $f_{SCK} = 1/(16\ t_c) = CKI/160$
B2	SCE	**S**PI **a**ctive clock **e**dge select 0: data are shifted out on the falling edge of SCK and are shifted in on the rising edge of SCK 1: data are shifted out on the rising edge of SCK and are shifted in on the falling edge of SCK
B1	SPIEN	**SPI en**able Enables the SPI interface and the alternate functions of the MISO, MOSI, SCK and $\overline{\text{SS}}$ pins. 0: disable SPI 1: enable SPI, all Port M $\overline{\text{ESS}}$ signals are set to 1
B0	SLOOP	**S**PI **loop** back mode 0: disable loop back mode 1: enable loop back mode, MISO and MOSI are internally connected (see *Figure 37*)

Serial Peripheral Interface (Continued)

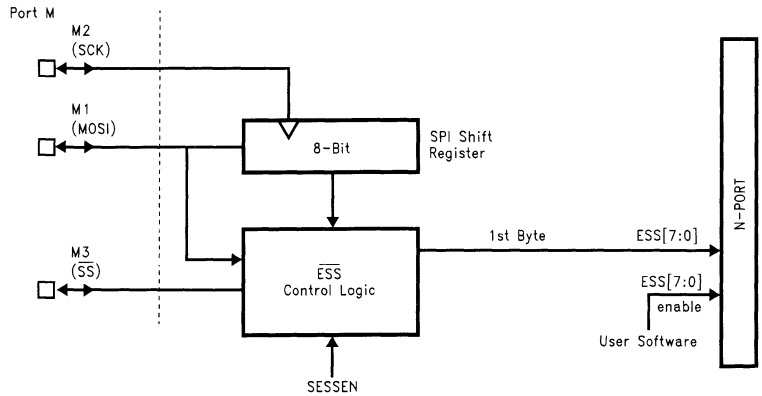
PROGRAMMING THE SPI EXPANDER

If the $\overline{SS}$ Expander is enabled by setting SESSEN = 1 in the SPI Control Register (SPICNTL), the N-port will be programmed with the content of the first MOSI-byte (i.e., the content of the 1st byte [7:0] appears at N-port[7:0] after complete reception of the first byte), if the $\overline{ESS}$ programming mode is detected. If any bytes follow after the 1st MOSI byte, all data will be ignored by the SPI.

The $\overline{ESS}$ programming mode is detected by the $\overline{ESS}$ control logic, which decodes the condition "MOSI = L at the falling edge of $\overline{SS}$. For further details, see *Figure 37*.

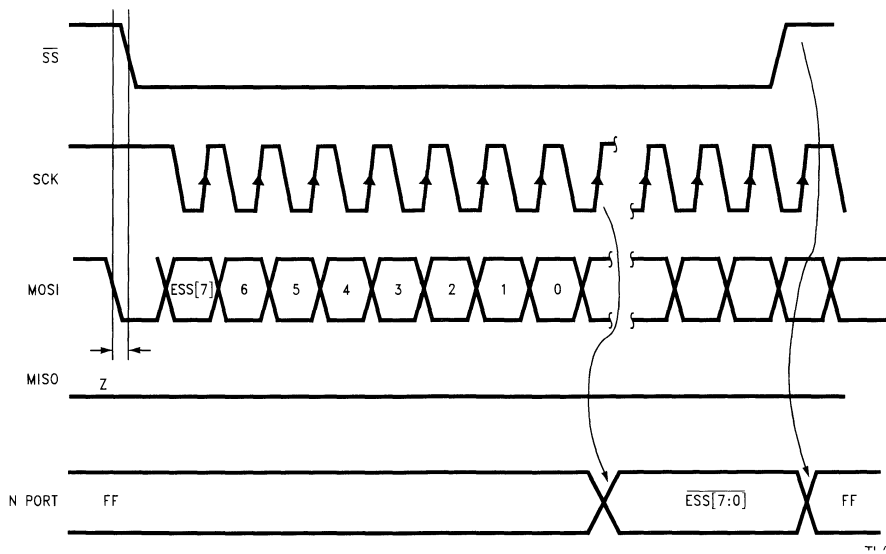
The selected N-port bits will be set to 1 after the positive edge of $\overline{SS}$.

Single N-port bits may be enabled for use as $\overline{SS}$ expansion, or disabled to allow for general purpose I/O, by the respective bits in the PORTNX register.



TL/DD/12871-41

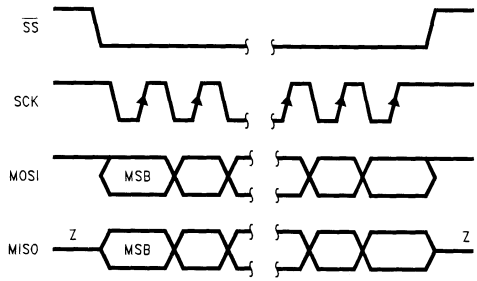
FIGURE 37. Programming the SPI Expander



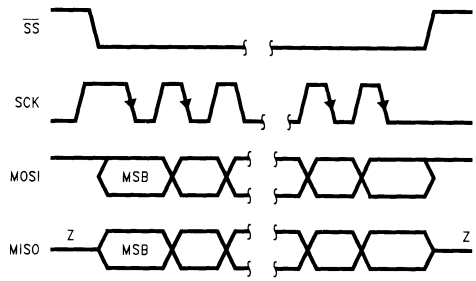
SESSEN = 1, SCE = 0. If MOSI = 0 at the falling edge of $\overline{SS}$, the $\overline{ESS}$ programming mode is detected and all N-port alternate functions are enabled.

FIGURE 38. Programming the $\overline{SS}$ Expander

SPI Status Register

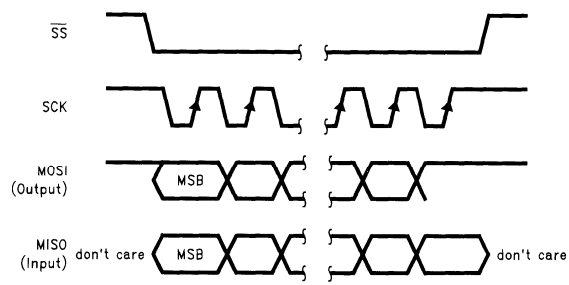


a) Slave mode; rising SCK edge is active edge. (SPIMOD[1,0] = [0,0], SCE = 0)

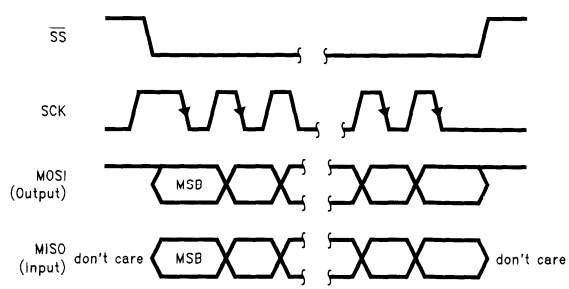


b) Slave mode; falling SCK edge is active edge. (SPIMOD[1,0] = [0,0], SCE = 1)

FIGURE 39. Slave Mode Communication



a) Master mode; rising SCK edge is active edge. (SPIMOD[1,0] = [1,0], SCE = 0)



b) Master mode; falling SCK edge is active edge. (SPIMOD[1,0] = [1,0], SCE = 1)

FIGURE 40. Master Mode Communication

SPI Status Register (Continued)

TABLE XV. SPI Status Register (SPISTAT) (0099)

Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
SRORN	SRBNE	STBF	STBE	STFL	SESSDET	x	x
0	0	1	1	0	0		0

The SPI Status Register is a read only register.

B7	SRORN	**S**PI **r**eceiver **o**verrun. This bit is set on the attempt to overwrite valid data in the RX FIFO by the SPI interface. (The condition to detect this is: SRWP = SRRP & COP has not read the data at SRRP and attempting to write to the RX FIFO by the SPI interface.) This bit can generate a receive interrupt if the receive interrupt is enabled (SRIE = 1). **Note 0:** At this condition the write operation will not be executed and all data get lost. **Note 1:** The SRORN bit stays set until the reset condition. This bit is reset with a dummy write to the SPISTAT register. (As the register is read only a dummy write does not have any effect on any other bits in this register.) As a result of the SRORN condition, the SRWP becomes frozen (i.e., does not change until the SRORN bit is reset) and the SPI will not store any new data in the RX FIFO. **Note 2:** With the SRRP being still available, the user can read the data in the RX FIFO before resetting the SRORN bit.
B6	SRBNE	**S**PI **R**eceive **b**uffer **n**ot **e**mpty This bit is set with a write to the SPI RX FIFO resulting in SRWP ! = SRRP (caution at rollover!). This bit is reset with the read of the SPIRXD register resulting in SRWP to be equal to SRRP.
B5	STBF	This bit can generate a receive interrupt if enabled with the RIE bit. **S**PI **T**ransmit **b**uffer **f**ull This bit is set after a write operation to the SPITXD register (from the COP side), which results in STRP = STWP. It gets reset as soon as the STRP gets incremented - by the SPI if reading data out of the TX FIFO.
B4	STBE	**S**PI **t**ransmit **b**uffer **e**mpty This bit is set after the last bit of the a read from the SPITXD register, which results in STRP = STWP. It gets reset as soon as the STWP gets incremented - by the COP if writing data into the TX FIFO. It is set on reset.
B3	STFL	**S**PI **T**ransmit **b**uffer **f**lush This bit indicates that the contents of the transmit buffer got discharged by the $\overline{SS}$ signal becoming high before all data in the transmit buffer could be transmitted. This bit gets set if the $\overline{SS}$ signal gets high and 1. STRP != STWP **or** 2. STRP = STWP and the current byte has not been completely transmitted from the SPI shift register These conditions will reset STRP and STWP to 0. These are virtual pointers and cannot be viewed. **Note:** STRP = STWP & STBE = 1 will generate an interrupt. This bit gets reset with a write to the SPITXD register.
B2	SESSDET	**S**PI $\overline{SS}$ **E**xpander **d**etection This bit indicates the detection of a $\overline{SS}$ expand condition (MOSI = 0 at the falling edge of $\overline{SS}$) immediately after the N-port has been programmed (8th SCK bit, 8 μs at SCK = 1 MHz). This bit is reset at the rising edge of $\overline{SS}$. 1: $\overline{SS}$ expand condition detected. 0: normal communication. **Note:** The SPI master must hold $\overline{SS}$ = 0 long enough to allow the device to read SESSDET. Otherwise the SESSDET information will get lost.
B1		unused
B0		unused

SPI Status Register (Continued)

SPI SYNCHRONIZATION

After the SPI is enabled (SPIEN = 1), the SPI internal receive and transmit shift clock is kept disabled until $\overline{SS}$ becomes inactive. This includes $\overline{SS}$ being active at the time SPIEN is set, i.e., no receive/transmit is possible until $\overline{SS}$ becomes inactive after enabling the SPI.

HALT/IDLE MODE

If the device enters the HALT/IDLE mode, both RX and TX FIFOs get reset (Flushed). If the device is exiting HALT/IDLE mode, and SPI synchronization takes place as described above. SPIRXD and SPITXD have the same state as after Reset, SPISTAT bits after HALT/IDLE mode are:

```
SRORN:    unchanged
SRBNE:    0
STBF:     0
STBE:     1
STFL:     1
SESSDET:  x (depending on SS
          and MOSI line)
```

TRANSMISSION START IN MASTER MODE

The transmission of data in the Master mode is started if the user controlled $\overline{SS}$ signal is switched active. No SCK will be generated in Master mode and thus no data is transmitted if the $\overline{SS}$ signal is kept high, i.e., $\overline{SS}$ must be switched low to generate SCK. Resetting the $\overline{SS}$ signal in the Master mode will immediately stop the transmission and flush the transmit FIFO. Thus, the user must only reset the $\overline{SS}$ if:

a) TBE is set or

b) SCK is high (SCE = 0) or low (SCE = 1)

TX AND RX FIFO

If the SPI is disabled (SPIEN = 0), all SPI FIFO related pointers are reset and kept at zero until the SPI is enabled again. Also, the Read/Write operation to both SPITXD and SPIRXD will not cause the pointers to change, if SPIEN is set, Read operations from the RXFIFO and Write operation to TXFIFO will increment the respective Read/Write pointers.

SPIRXD SPI Receive Data Register

SPIRXD is at address location "009A". It is a read/write register.

This register holds the receive data at the current SRRP location: a COP read operation from this register to the accumulator will read the RX FIFO at the SRRP location and increment SRRP afterwards. A write to this register (by the controllers SW) will write to the RX FIFO at the current SRRP location. The SRRP is not changed.

Note: During breakpoint the SRRP is not incremented.

A write to this register from the SPI interface side will write to the current SRWP location and increment SRWP afterwards.

SPITXD SPI Transmit Data Register

SPITXD is at address location "009B". It is a read/write register.

This register holds the transmit data at the current STWP location: a write from the controller to this register will write to the STWP location and increment the STWP afterwards. A read from the controller to this register will read the TX FIFO at the current STWP location. The pointer is not changed.

Writing data into this register will start a transmission of data in the master mode.

Note: No read modify write instructions should be used on this register.

Reading this register from the SPI side will read the byte at the current STRP location and afterwards increment STRP.

SPI RX FIFO

The SPI RX FIFO is a 12 byte first in first out buffer. SPI RX FIFO data are read from the controller by reading the SPIRXD register. A pointer (SRRP) controls the controller read location. Data is written to this register by the SPI interface. The write location is controlled by the SRWP. SRWP is incremented after data is stored to the FIFO SRWP is never decremented SRWP has a roll-over 10 → 11 → 0 → 1 → 2 → etc. It is a circularly linked list.

SRRP is incremented after data is read from the FIFO SRRP is never decremented SRRP has a roll-over 10 → 11 → 0 → 1 → 2 → etc. It is a circularly linked list.

Both pointers are cleared at reset.

The following bits indicate the status of the RX FIFO: SRBNE = (SRWP != SRRP) and !SRORN .SRORN is set at (SRWP = SRRP) and after a write from the SPI side, reset at write to SPISTAT.

Special conditions: if .SRORN is set, no writes to the RX FIFO are allowed from the SPI side. SRWP is frozen. Resetting .SRORN (after it was set) clears both SRWP and SRRP. To prevent erroneous clearing of the Receive FIFO when entering HALT/IDLE mode, the user needs to enable the MIWU or port M3 ($\overline{SS}$) by setting bit 3 in MWKEN register.

SPI TX FIFO

The SPI TX FIFO is a 12 byte first in first out buffer. Data is written to the FIFO by the controller executing a write instruction to the SPITXD register. A pointer (STWP) controls the controller write location. Data is read from this register by the SPI interface. The read location is controlled by the STRP. STRP is incremented after data is read from the FIFO STRP is never decremented STRP has a roll-over 10 → 11 → 0 → 1 → 2 → etc. It is a circularly linked list.

STWP is incremented after data is written to the FIFO STWP is never decremented STWP has a roll-over 10 → 11 → 0 → 1 → 2 → etc. It is a circularly linked list.

Both pointers are cleared at reset.

The following bits indicate the status of the TX FIFO: STBF = set at (STRP = STWP) after a write from the controller reset at ((STRP != STWP) I STBE) after a read from the SPI STBE = (STRP = STWP) after a read from the SPI.

Special conditions: If the $\overline{SS}$ signal becomes high before data the last bit of the last byte in the TX FIFO is transmitted both STRP and STWP will be set to 0. The STFL bit will be set. (STBE will be set as well.)

Note: The SRRP, SRWP, STRP and STWP registers are not available to the user. Their operation description is included for clarity and to enhance the user's understanding.

A/D Converter

The device contains an 8-channel, multiplexed input, successive approximation, Analog-to-Digital convertor. The device contains AGND/AV_{CC} and ADV_{REF} for voltage reference.

OPERATING MODES

The A/D convertor supports ratiometric measurements. It supports both Single Ended and Differential modes of operation.

Four specific analog channel selection modes are supported. These are as follows:

Allow any specific channel to be selected at one time. The A/D convertor performs the specific conversion requested and stops.

Allow any specific channel to be scanned continuously. In other words, the user specifies the channel and the A/D convertor scans it continuously. At any arbitrary time the user can immediately read the result of the last conversion. The user must wait for only the first conversion to complete.

Allow any differential channel pair to be selected at one time. The A/D convertor performs the specific differential conversion requested and stops.

Allow any differential channel pair to be scanned continuously. In other words, the user specifies the differential channel pair and the A/D convertor scans it continuously. At any arbitrary time the user can immediately read the result of the last differential conversion. The user must wait for only the first conversion to complete.

The A/D convertor is supported by two memory mapped registers, the result register and the mode control register. When the device is reset, the mode control register (ENAD) is cleared, the A/D is powered down and the A/D result register has unknown data.

A/D Control Register

The ENAD control register contains 3 bits for channel selection, 2 bits for prescaler selection, 2 bits for mode selection and a Busy bit. An A/D conversion is initiated by setting the ADBSY bit and the ENAD control register. The result of the conversion is available to the user in the A/D result register, ADRSLT, when ADBSY is cleared by the hardware on completion of the conversion.

ENAD (address (0xCB)

CHANNEL SELECT			MODE SELECT		PRESCALER SELECT		BUSY
ADCH2	ADCH1	ADCH0	ADMOD1	ADMOD0	PSC1	PSC0	ADBSY

Bit 7 Bit 0

CHANNEL SELECT

This 3-bit field selects one of eight channels to be the V_{IN+}. The mode selection determines the V_{IN-} input.

Single Ended mode:

Bit 7	Bit 6	Bit 5	Channel No.
0	0	0	0
0	0	1	1
0	1	0	2
0	1	1	3
1	0	0	4
1	0	1	5
1	1	0	6
1	1	1	7

Differential mode:

Bit 7	Bit 6	Bit 5	Channel Pairs (+, −)
0	0	0	0, 1
0	0	1	1, 0
0	1	0	2, 3
0	1	1	3, 2
1	0	0	4, 5
1	0	1	5, 4
1	1	0	6, 7
1	1	1	7, 6

MODE SELECT

This 2-bit field is used to select the mode of operation (single conversion, continuous conversions, differential, single ended) as shown in the following table.

Bit 4	Bit 3	Mode
0	0	Single Ended mode, single conversion
0	1	Single Ended mode, continuous scan of a single channel into the result register
1	0	Differential mode, single conversion
1	1	Differential mode, continuous scan of a channel pair into the result register

A/D Converter (Continued)

PRESCALER SELECT

This 2-bit field is used to select one of the four prescaler clocks for the A/D converter. The following table shows the various prescaler options.

A/D Convertor Clock Prescaler

Bit 2	Bit 1	Clock Select
0	0	Divide by 2
0	1	Divide by 4
1	0	Divide by 6
1	1	Divide by 12

BUSY BIT

The ADBSY bit of the ENAD register is used to control starting and stopping of the A/D conversion. When ADBSY is cleared, the prescale logic is disabled and the A/D clock is turned off. Setting the ADBSY bit starts the A/D clock and initiates a conversion based on the mode select value currently in the ENAD register. Normal completion of an A/D conversion clears the ADBSY bit and turns off the A/D convertor.

The ADBSY bit remains a one during continuous conversion. The user can stop continuous conversion by writing a zero to the ADBSY bit.

If the user wishes to restart a conversion which is already in progress, this can be accomplished only by writing a zero to the ADBSY bit to stop the current conversion and then by writing a one to ADBSY to start a new conversion. This can be done in two consecutive instructions.

A/D Operation

The A/D convertor interface works as follows. Setting the ADBSY bit in the A/D control register ENAD initiates an A/D conversion. The conversion sequence starts at the beginning of the write to ENAD operation which sets ADBSY, thus powering up the A/D. At the first falling edge of the convertor clock following the write operation, the sample signal turns on for seven clock cycles. If the A/D is in single conversion mode, the conversion complete signal from the A/D will generate a power down for the A/D convertor and will clear the ADBSY bit in the ENAD register at the next instruction cycle boundary. If the A/D is in continuous mode, the conversion complete signal will restart the conversion sequence by deselecting the A/D for one convertor clock cycle before starting the next sample. The A/D 8-bit result is immediately loaded into the A/D result register (ADRSLT) upon completion. Internal logic prevents transient data (resulting from the A/D writing a new result over an old one) being read from ADRSLT.

Inadvertent changes to the ENAD register during conversion are prevented by the control logic of the A/D. Any attempt to write any bit of the ENAD Register except ADBSY, while ADBSY is a one, is ignored. ADBSY must be cleared either by completion or an A/D conversion or by the user before the prescaler, conversion mode or channel select values can be changed. After stopping the current conversion, the user can load different values for the prescaler, conversion mode or channel select and start a new conversion in one instruction.

It is important for the user to realize that, when used in differential mode, only the positive input to the A/D converter is sampled and held. The negative input is constantly connected and should be held stable for the duration of the conversion. Failure to maintain a stable negative input will result in incorrect conversion.

PRESCALER

The A/D Convertor (A/D) contains a prescaler option that allows four different clock selections. The A/D clock frequency is equal to CKI divided by the prescaler value. Note that the prescaler value must be chosen such that the A/D clock falls within the specified range. The maximum A/D frequency is 1.67 MHz. This equates to a 600 ns A/D clock cycle.

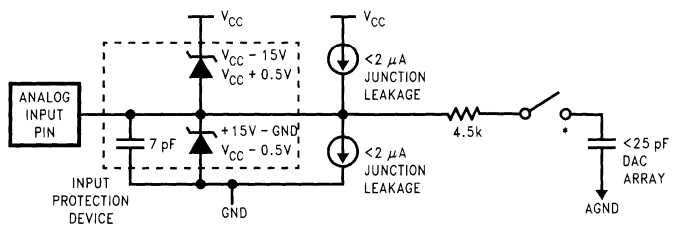
The A/D convertor takes 17 A/D clock cycles to complete a conversion. Thus the minimum A/D conversion time for the device is 10.2 μs when a prescaler of 6 has been selected. The 17 A/D clock cycles needed for conversion consist of 1 cycle at the beginning for reset, 7 cycles for sampling, 8 cycles for converting, and 1 cycle for loading the result into the A/D result register (ADRSLT). This A/D result register is a read-only register. The user cannot write into ADRSLT.

The ADBSY flag provides an A/D clock inhibit function, which saves power by powering down the A/D when it is not in use.

Note: The A/D convertor is also powered down when the device is in either the HALT or IDLE modes. If the A/D is running when the device enters the HALT or IDLE modes, the A/D powers down and then restarts the conversion with a corrupted sampled voltage (and thus an invalid result) when the device comes out of the HALT or IDLE modes.

Analog Input and Source Resistance Considerations

Figure 41 shows the A/D pin model in single ended mode. The differential mode has a similar A/D pin model. The leads to the analog inputs should be kept as short as possible. Both noise and digital clock coupling to an A/D input can cause conversion errors. The clock lead should be kept away from the analog input line to reduce coupling. The A/D channel input pins do not have any internal output driver circuitry connected to them because this circuitry would load the analog input signals due to output buffer leakage current.



* The analog switch is closed only during the sample time.

FIGURE 41. A/D Pin Model (Single Ended Mode)

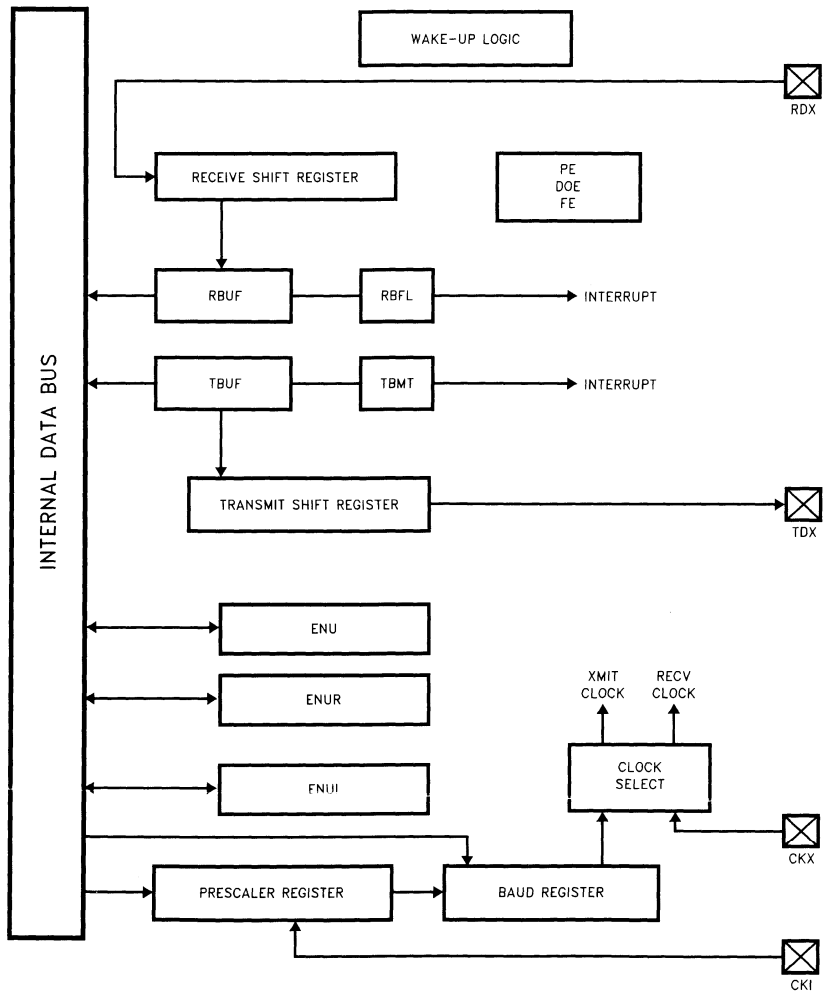
A/D Converter (Continued)

Source impedances greater than 3 kΩ on the analog input lines will adversely affect the internal RC charging time during input sampling. As shown in *Figure 41*, the analog switch to the DAC array is closed only during the 7 A/D cycle sample time. Large source impedances on the analog inputs may result in the DAC array not being charged to the correct voltage levels, causing scale errors.

If large source resistance is necessary, the recommended solution is to slow down the A/D clock speed in proportion to the source resistance. The A/D convertor may be operated at the maximum speed for $R_S < 3$ kΩ. For $R_S > 3$ kΩ, A/D clock speed needs to be reduced. For example, with $R_S = 6$ kΩ, the A/D convertor may be operated at half the maximum speed. A/D convertor clock speed may be slowed down by either increasing the A/D prescaler divide-by or decreasing the CKI clock frequency. The A/D minimum clock speed is 100 kHz.

UART

The device contains a full-duplex software programmable UART. The UART *(Figure 42)* consists of a transmit shift register, a receiver shift register and seven addressable registers, as follows: a transmit buffer register (TBUF), a receiver buffer register (RBUF), a UART control and status register (ENU), a UART receive control and status register (ENUR), a UART interrupt and clock source register (ENUI), a prescaler select register (PSR) and baud (BAUD) register. The ENU register contains flags for transmit and receive functions; this register also determines the length of the data frame (7, 8 or 9 bits), the value of the ninth bit in transmission, and parity selection bits. The ENUR register flags framing, data overrun and parity errors while the UART is receiving.



TL/DD/12871–48

FIGURE 42. UART Block Diagram

UART (Continued)

Other functions of the ENUR register include saving the ninth bit received in the data frame, enabling or disabling the UART's attention mode of operation and providing additional receiver/transmitter status information via RCVG and XMTG bits. The determination of an internal or external clock source is done by the ENUI register, as well as selecting the number of stop bits and enabling or disabling transmit and receive interrupts. A control flag in this register can also select the UART mode of operation: asynchronous or synchronous.

UART CONTROL AND STATUS REGISTERS

The operation of the UART is programmed through three registers: ENU, ENUR and ENUI. The function of the individual bits in these registers is as follows:

ENU-UART Control and Status Register (Address at 0BA)

PEN	PSEL1	XBIT9/ PSEL0	CHL1	CHL0	ERR	RBFL	TBMT
0RW	0RW	0RW	0RW	0RW	0R	0R	1R

Bit 7 Bit 0

ENU-UART Receive Control and Status Register (Address at 0BB)

DOE	FE	PE	SPARE	RBIT9	ATTN	XMTG	RCVG
0RD	0RD	0RD	0RW*	0R	0RW	0R	0R

Bit 7 Bit 0

ENUI-UART Interrupt and Clock Source Register (Address at 0BC)

STP2	STP78	ETDX	SSEL	XRCLK	XTCLK	ERI	ETI
0RW	0RW	0RW	0RW	0RW	0RW	0RW	0RW

Bit 7 Bit 0

* Bit is not used.

0 Bit is cleared on reset.

1 Bit is set to one on reset.

R Bit is read-only; it cannot be written by software.

R/W Bit is read/write.

D Bit is cleared on read; when read by software as a one, it is cleared automatically. Writing to the bit does not affect its state.

DESCRIPTION OF UART REGISTER BITS

ENU—UART CONTROL AND STATUS REGISTER

TBMT: This bit is set when the UART transfers a byte of data from the TBUF register into the TSFT register for transmission. It is automatically reset when software writes into the TBUF register.

RBFL: This bit is set when the UART has received a complete character and has copied it into the RBUF register. It is automatically reset when software reads the character from RBUF.

ERR: This bit is a global UART error flag which gets set if any or a combination of the errors (DOE, FE, PE) occur.

CHL1, CHL0: These bits select the character frame format. Parity is not included and is generated/verified by hardware.

CHL1 = 0, CHL0 = 0 The frame contains eight data bits.

CHL1 = 0, CHL0 = 1 The frame continues seven data bits.

CHL1 = 1, CHL0 = 0 The frame continues nine data bits.

CHL1 = 1, CHL0 = 1 Loopback Mode selected. Transmitter output internally looped back to receiver input. Nine bit framing format is used.

XBIT9/PSEL0: Programs the ninth bit for transmission when the UART is operating with nine data bits per frame. For seven or eight data bits per frame, this bit in conjunction with PSEL1 selects parity.

PSEL1, PSEL0: Parity select bits.

PSEL1 = 0, PSEL0 = 0 Odd Parity (if Parity enabled)

PSEL1 = 0, PSEL1 = 1 Odd Parity (if Parity enabled)

PSEL1 = 1, PSEL0 = 0 Mark(1) (if Parity enabled)

PSEL1 = 1, PSEL1 = 1 Space(0) (if Parity enabled)

PEN: This bit enables/disabled Parity (7- and 8-bit modes only).

PEN = 0 Parity disabled.

PEN = 1 Parity enabled.

ENUR—UART RECEIVE CONTROL AND STATUS REGISTER

RCVG: This bit is set high whenever a framing error occurs and goes low when RDX goes high.

XMTG: This bit is set to indicate that the UART is transmitting. It gets reset at the end of the last frame (end of last Stop bit).

ATTN: ATTENTION Mode is enabled while this bit is set. This bit is cleared automatically on receiving a character with data bit nine set.

RBIT9: Contains the ninth data bit received when the UART is operating with nine data bits per frame.

SPARE: Reserved for future use.

PE: Flags a Parity Error.

PE = 0 Indicates no Parity Error has been detected since the last time the ENUR register was read.

PE = 1 Indicates the occurrence of a Parity Error.

FE: Flags a Framing Error.

FE = 0 Indicates no Framing Error has been detected since the last time the ENUR register was read.

FE = 1 Indicates the occurrence of a Framing Error.

DOE: Flags a Data Overrun Error.

DOE = 0 Indicates no Data Overrun Error has been detected since the last time the ENUR register was read.

DOE = 1 Indicates the occurrence of a Data Overrun Error.

ENUE—UART INTERRUPT AND CLOCK SOURCE REGISTER

ETI: This bit enables/disables interrupt from the transmitter section.

ETI = 0 Interrupt from the transmitter is disabled.

ETI = 1 Interrupt from the transmitter is enabled.

ERI: This bit enables/disables interrupt from the receiver section.

ERI = 0 Interrupt from the receiver is disabled.

ERI = 1 Interrupt from the receiver is enabled.

UART (Continued)

XTCLK: This bit selects the clock source for the transmitter section.

XTCLK = 0 The clock source is selected through the PSR and BAUD registers.

XTCLK = 1 Signal on CKX (L1) pin is used as the clock.

XRCLK: This bit selects the clock source for the receiver section.

XRCLK = 0 The clock source is selected through the PSR and BAUD registers.

XRCLK = 1 Signal on CKX (L1) pin is used as the clock.

SSEL: UART mode select.

SSEL = 0 Asynchronous Mode.

SSEL = 1 Synchronous Mode.

ETDX: TDX (UART Transmit Pin) is the alternate function assigned to Port L pin L2; it is selected by setting EDTX bit. To simulate line break generation, software should reset ETDX bit and output logic zero to TDX pin through Port L data and configuration registers.

STP78: This bit is set to program the last Stop bit to be 7/8th of a bit in length.

STP2: This bit programs the number of Stop bits to be transmitted.

STP2 = 0 One Stop bit transmitted.

STP2 = 1 Two Stop bits transmitted.

Associated I/O Pins

Data is transmitted on the TDX pin and received on the RDX pin. TDX is the alternate function assigned to Port L pin L2; it is selected by setting ETDX (in the ENUI register) to one. RDX is an inherent function of Port L pin L3, requiring no setup.

The baud rate clock for the UART can be generated on-chip, or can be taken from an external source. Port L pin L1 (CKX) is the external clock I/O pin. The CKX pin can be either an input or an output, as determined by Port L Configuration and Data registers (Bit 1). As an input, it accepts a clock signal which may be selected to drive the transmitter and/or receiver. As an output, it presents the internal Baud Rate Generator output.

UART Operation

The UART has two modes of operation; asynchronous mode and synchronous mode.

ASYNCHRONOUS MODE

This mode is selected by resetting the SSEL (in the ENUI register) bit to zero. The input frequency to the UART is 16 times the baud rate.

The TSFT and TBUF registers double-buffer data for transmission. While TSFT is shifting out the current character on the TDX pin, the TBUF register may be loaded by software with the next byte to be transmitted. When TSFT finishes transmitting the current character the contents of TBUF are transferred to the TSFT register and the Transmit Buffer Empty Flag (TBMT in the ENU register) is set. The TBMT flag is automatically reset by the UART when software loads a new character into the TBUF register. There is also the XMTG bit which is set to indicate that the UART is transmitting. This bit gets reset at the end of the last frame (end of last Stop bit). TBUF is a read/write register.

The RSFT and RBUF registers double-buffer data being received. The UART receiver continually monitors the signal on the RDX pin for a low level to detect the beginning of a Start bit. Upon sensing this low level, it waits for half a bit time and samples again. If the RDX pin is still low, the receiver considers this to be a valid Start bit, and the remaining bits in the character frame are each sampled a single time, at the mid-bit position. Serial data input on the RDX pin is shifted into the RSFT register. Upon receiving the complete character, the contents of the RSFT register are copied into the RBUF register and the Received Buffer Full Flag (RBFL) is set. RBFL is automatically reset when software reads the character from the RBUF register. RBUF is a read only register. There is also the RCVG bit which is set high when a framing error occurs and goes low once RDX goes high. TBMT, XMTG, RBFL and RCVG are read only bits.

SYNCHRONOUS MODE

In this mode data is transferred synchronously with the clock. Data is transmitted on the rising edge and received on the falling edge of the synchronous clock.

This mode is selected by setting SSEL bit in the ENUI register. The input frequency to the UART is the same as the baud rate.

When an external clock input is selected at the CKX pin, data transmit and receive are performed synchronously with this clock through TDX/RDX pins.

If data transmit and receive are selected with the CKX pin as clock output, the device generates the synchronous clock output at the CKX pin. The internal baud rate generator is used to produce the synchronous clock. Data transmit and receive are performed synchronously with this clock.

FRAMING FORMATS

The UART supports several serial framing formats *(Figure 43)*. The format is selected using control bits in the ENU, ENUR and ENUI registers.

The first format (1, 1a, 1b, 1c) for data transmission (CHL0 = 1, CHL1 = 0) consists of Start bit, seven Data bits (excluding parity) and 7/8, one or two Stop bits. In applications using parity, the parity bit is generated and verified by hardware.

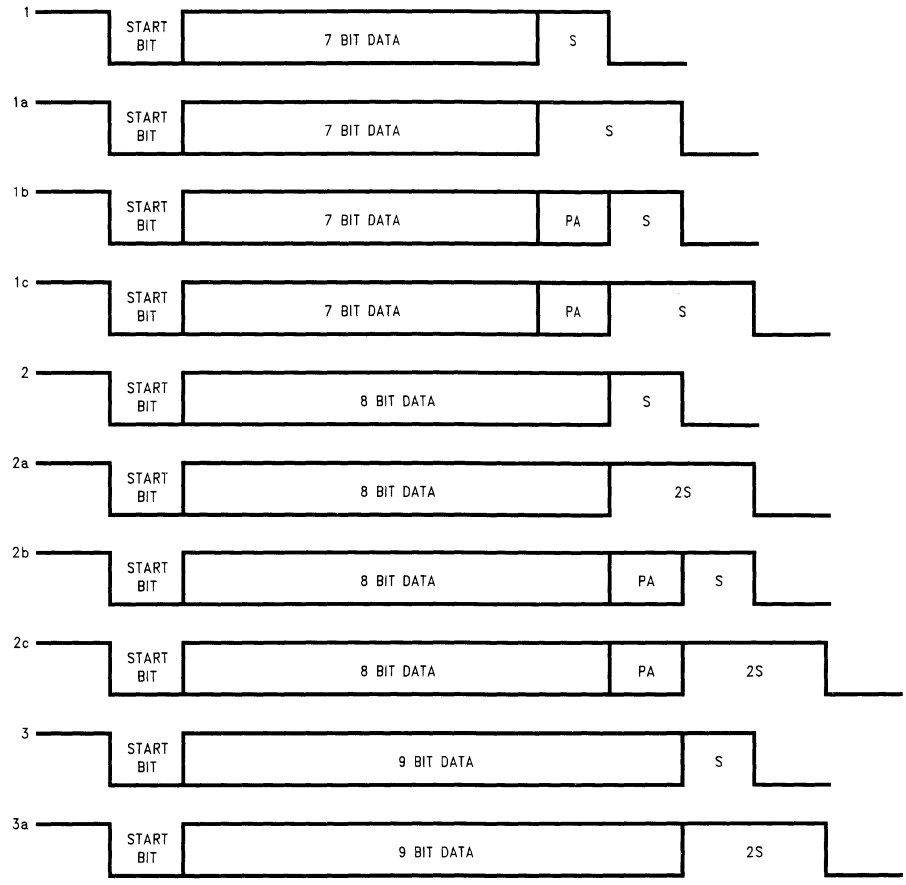
The second format (CHL0 = 0, CHL1 = 0) consists of one Start bit, eight Data bits (excluding parity) and 7/8, one or two Stop bits. Parity bit is generated and verified by hardware.

The third format for transmission (CHL0 = 0, CHL1 = 1) consists of one Start bit, nine Data bits and 7/8, one or two Stop bits. This format also supports the UART "ATTENTION" feature. When operating in this format, all eight bits of TBUF and RBUF are used for data. The ninth data bit is transmitted and received using two bits in the ENU and ENUR registers, called XBIT9 and RBIT9. RBIT9 is a read only bit. Parity is not generated or verified in this mode.

For any of the above framing formats, the last Stop bit can be programmed to be 7/8th of a bit in length. If two Stop bits are selected and the 7/8th bit is set (selected), the second Stop bit will be 7/8th of a bit in length.

The parity is enabled/disabled by PEN bit located in the ENU register. Parity is selected for 7-bit and 8-bit modes only. If parity is enabled (PEN = 1), the parity selection is then performed by PSEL0 and PSEL1 bits located in the ENU register.

UART Operation (Continued)



TL/DD/12871–49

FIGURE 43. Framing Formats

Note that the XBIT9/PSEL0 bit located in the ENU register serves two mutually exclusive functions. This bit programs the ninth bit for transmission when the UART is operating with nine data bits per frame. There is no parity selection in this framing format. For other framing formats XBIT9 is not needed and the bit is PSEL0 used in conjunction with PSEL1 to select parity.

The frame formats for the receiver differ form the transmitter in the number to Stop bits required. The receiver only requires one Stop bit in a frame, regardless of the setting of the Stop bit selection bits in the control register. Note that an implicit assumption is made for full duplex UART operatioin that the framing formats are the same for the transmitter and receiver.

UART INTERRUPTS

The UART is capable of generating interrupts. Interrupts are generated on Receive Buffer Full and Transmit Buffer Empty. Both interrupts have individual interrupt vectors. Two bytes of program memory space are reserved for each interrupt vector. The two vectors are located at addresses 0xEC to 0xEF Hex in the program memory space. The interrupts can be individually enabled or disabled using Enable Transmit Interrupt (ETI) and Enable Receive Interrupt (ERI) bits in the ENUI register.

The interrupt from the Transmitter is set pending, and remains pending, as long as both the TBMT and ETI bits are set. To remove this interrupt, software must either clear the ETI bit or write to the TBUF register (thus clearing the TBMT bit).

The interrupt from the receiver is set pending, and remains pending, as long as both the RBFL and ERI bits are set. To remove this interrupt, software must either clear the ERI bit or read from the RBUF register (thus clearing the RBFL bit).

Baud Clock Generation

The clock inputs to the transmitter and receiver sections of the UART can be individually selected to come either from an external source at the CKX pin (port L, pin L1) or from a source selected in the PSR and BAUD registers. Internally, the basic baud clock is created from the oscillator frequency through a two-stage divider chain consisting of a 1–16

Baud Clock Generation (Continued)

(increments of 0.5) prescaler and an 11-bit binary counter. *(Figure 44)*. The divide factors are specified through two read/write registers shown in *Figure 45*. Note that the 11-bit Baud Rate Divisor spills over into the Prescaler Select Register (PSR). PSR is cleared upon reset.

As shown in Table XVI, a Prescaler Factor of 0 corresponds to NO CLOCK. NO CLOCK condition is the UART power down mode where the UART clock is turned off for power saving purpose. The user must also turn the UART clock off when a different baud rate is chosen.

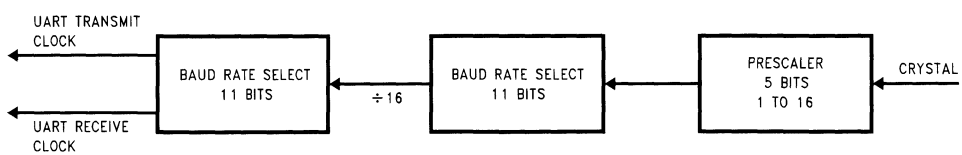
The correspondences between the 5-bit Prescaler Select and Prescaler factors are shown in Table XVI. There are many ways to calculate the two divisor factors, but one particularly effective method would be to achieve a 1.8432 MHz frequency coming out of the first stage. The 1.8432 MHz prescaler output is then used to drive the software programmable baud rate counter to create a x16 clock for the following baud rates: 110, 134.5, 150, 300, 600, 1200, 1800, 2400, 3600, 4800, 7200, 9600, 19200 and 38400 (Table XVII). Other baud rates may be created by using appropriate divisors. The x16 clock is then divided by 16 to provide the rate for the serial shift registers of the transmitter and receiver.

TABLE XVI. Prescaler Factors

Prescaler Select	Prescaler Factor
00000	NO CLOCK
00001	1
00010	1.5
00011	2
00100	2.5
00101	3
00110	3.5
00111	4
01000	4.5
01001	5

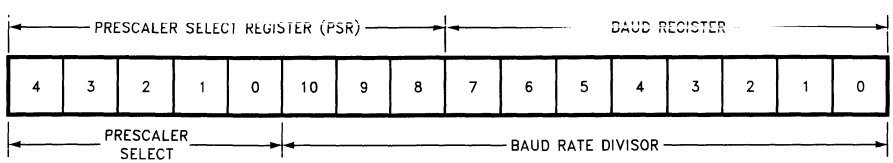
TABLE XVI. Prescaler Factors (Continued)

Prescaler Select	Prescaler Factor
01010	5.5
01011	6
01100	6.5
01101	7
01110	7.5
01111	8
100000	8.5
10001	9
10010	9.5
10011	10
10100	10.5
10101	11
10110	11.5
10111	12
11000	12.5
11001	13
11010	13.5
11011	14
11100	14.5
11101	15
11110	15.5
11111	16



TL/DD/12871-50

FIGURE 44. UART BAUD Clock Generation



TL/DD/12871-51

FIGURE 45. UART BAUD Clock Divisor Registers

Baud Clock Generation (Continued)

TABLE XVII. Baud Rate Divisors (1.8432 MHz Prescaler Output)

Baud Rate	Baud Rate Divisor − 1 (N-1)
110 (110.03)	1046
134.5 (134.58)	855
150	767
300	383
600	191
1200	95
1800	63
2400	47
3600	31
4800	23
7200	15
9600	11
19200	5
38400	2

Note: The entries in Table XVII assume a prescaler output of 1.8432 MHz. In the asynchronous mode the baud rate could be as high as 625k.

As an example, considering the Asynchronous Mode and a CKI clock of 4.608 MHz, the prescaler factor selected is:

$$4.608/1.8432 = 2.5$$

The 2.5 entry is available in Table XVI. The 1.8432 MHz prescaler output is then used with proper Baud Rate Divisor (Table II) to obtain different baud rates. For a baud rate of 19200 e.g., the entry in Table XVII is 5.

$$N - 1 = 5 \text{ (N} - 1 \text{ is the value from Table XVII)}$$

$$N = 6 \text{ (N is the Baud Rate Divisor)}$$

$$\text{Baud Rate} = 1.8432 \text{ MHz}/(16 \times 6) = 19200$$

The divide by 16 is performed because in the asynchronous mode, the input frequency to the UART is 16 times the baud rate. The equation to calculate baud rates is given below.

The actual Baud Rate may be found from:

$$BR = Fc/(16 \times N \times P)$$

Where:

BR is the Baud Rate

Fc is the CKI frequency

N is the Baud Rate Divisior (Table XVII).

P is the Prescaler Divide Factor selected by the value in the Prescaler Select Register (Table XVI)

Note: In the Synchronous Mode, the divisor 16 is replaced by two.

Example:

Asynchronous Mode:

Crystal Frequency = 5 MHz

Desired baud rate = 9600

Using the above equation N X P can be calculated first.

$$N \times P = (5 \times 106)/(16 \times 9600) = 32.552$$

Now 32.552 is divided by each Prescaler Factor (Table III) to obtain a value closet to an integer. This factor happens to be 6.5 (P = 6.5).

$$N = 32.552/6.5 = 5.008 \text{ (N} = 5)$$

The programmed value (from Table IV) should be 4 (N − 1).

Using the above values calculated for N and P:

$$BR = (5 \times 106)/(16 \times 5 \times 6.5) = 9615.384$$

$$\% \text{ error} = (9615.385 - 9600)/9600 = 0.16$$

Effect of HALT/IDLE

The UART logic is reinitialized when either the HALT or IDLE modes are entered. This reinitialization sets the TBMT flag and resets all read only bits in the UART control and status registers. Read/Write bits remain unchanged. The Transmit Buffer (TBUF) is not affected, but the Transmit Shift register (TSFT) bits are set to one. The receiver registers RBUF and RSFT are not affected.

The device will exit from the HALT/IDLE modes when the Start bit of a character is detected at the RDX (L3) pin. This feature is obtained by using the Multi-Input Wakeup scheme provided on the device.

Before entering the HALT or IDLE modes the user program must select the Wakeup source to be on the RXD pin. This selection is done by setting bit 3 of WKEN (Wakeup Enable) register. The Wakeup trigger condition is then selected to be high to low transition. This is done via the WKEDG register. (Bit 3 is one.)

If the device is halted and crystal oscillator is used, the Wakeup signal will not start the chip running immediately because of the finite start up time requirement of the crystal oscillator. The idle timer (T0) generates a fixed (256 t_c) delay to ensure that the oscillator has indeed stabilized before allowing the device to execute code. The user has to consider this delay when data transfer is expected immediately after exiting the HALT mode.

Diagnostic

Bits CHARL0 and CHARL1 in the ENU register provide a loopback feature for diagnostic testing of the UART. When these bits are set to one, the following occur: The receiver input pin (RDX) is internally connected to the transmitter output pin (TDX); the output of the Transmitter Shift Register is "looped back" into the Receive Shift Register input. In this mode, data that is transmitted is immediately received. This feature allows the processor to verify the transmit and receive data paths of the UART.

Note that the framing format for this mode is the nine bit format; one Start bit, nine data bits, and 7/8, one or two Stop bits. Parity is not generated or verified in this mode.

Attention Mode

The UART Receiver section supports an alternate mode of operation, referred to as ATTENTION Mode. This mode of operation is selected by the ATTN bit in the ENUR register. The data format for transmission must also be selected as having nine Data bits and either 7/8, one or two Stop bits.

The ATTENTION mode of operation is intended for use in networking the device with other processors, Typically in such environments the messages consists of device addresses, indicating which of several destinations should receive them, and the actual data. This Mode supports a scheme in which addresses are flagged by having the ninth bit of the data field set to a 1. If the ninth bit is reset to a zero the byte is a Data byte.

While in ATTENTION mode, the UART monitors the communication flow, but ignores all characters until an address character is received. Upon receiving an address character, the UART signals that the character is ready by setting the RBFL flag, which in turn interrupts the processor if UART Receiver interrupts are enabled. The ATTN bit is also cleared automatically at this point, so that data characters as well as address characters are recognized. Software examines the contents of the RBUF and responds by deciding either to accept the subsequent data stream (by leaving the ATTN bit reset) or to wait until the next address character is seen (by setting the ATTN bit again).

Operation of the UART Transmitter is not affected by selection of this Mode. The value of the ninth bit to be transmitted is programmed by setting XBIT9 appropriately. The value of the ninth bit received is obtained by reading RBIT9. Since this bit is located in ENUR register where the error flags reside, a bit operation on it will reset the error flags.

WATCHDOG

The device contains a WATCHDOG and clock monitor. The WATCHDOG is designed to detect the user program getting stuck in infinite loops resulting in loss of program control or "runaway" programs. The Clock Monitor is used to detect the absence of a clock or a very slow clock below a specified rate on the CKI pin.

The WATCHDOG consists of two independent logic blocks: WD UPPER and WD LOWER. WD UPPER establishes the upper limit on the service window and WD LOWER defines the lower limit of the service window.

Servicing the WATCHDOG consists of writing a specific value to a WATCHDOG Service Register named WDSVR which is memory mapped in the RAM. This value is composed of three fields, consisting of a 2-bit Window Select, a 5-bit Key Data field, and the 1-bit Clock Monitor Select field. Table XVIII shows the WDSVR register.

TABLE XVIII. WATCHDOG Service Register

Window Select		Key Data					Clock Monitor
X	X	0	1	1	0	0	Y
Bit 7							Bit 0

The lower limit of the service window is fixed at 2048 instruction cycles. Bits 7 and 6 of the WDSVR register allow the user to pick an upper limit of the service window.

Table XIX shows the four possible combinations of lower and upper limits for the WATCHDOG service window. This flexibility in choosing the WATCHDOG service window prevents any undue burden on the user software.

TABLE XIX. WATCHDOG Service Window Select

WDSVR Bit 7	WDSVR Bit 6	Service Window (Lower-Upper Limits)
0	0	2k–8k t_C Cycles
0	1	2k–16k t_C Cycles
1	0	2k–32k t_C Cycles
1	1	2k–64k t_C Cycles

Bits 5, 4, 3, 2 and 1 of the WDSVR register represent the 5-bit Key Data field. The key data is fixed at 01100. Bit 0 of the WDSVR Register is the Clock Monitor Select bit.

Clock Monitor

The Clock Monitor aboard the device can be selected or deselected under program control. The Clock Monitor is guaranteed not to reject the clock if the instruction cycle clock ($1/t_C$) is greater or equal to 10 kHz. This equates to a clock input rate on CKI of greater or equal to 100 kHz.

WATCHDOG Operation

The WATCHDOG and Clock Monitor are disabled during reset. The device comes out of reset with the WATCHDOG armed, the WATCHDOG Window Select (bits 6, 7 of the WDSVR Register) set, and the Clock Monitor bit (bit 0 of the WDSVR Register) enabled. Thus, a Clock Monitor error will occur after coming out of reset, if the instruction cycle clock frequency has not reached a minimum specified value, including the case where the oscillator fails to start.

The WDSVR register can be written to only once after reset and the key data (bits 5 through 1 of the WDSVR Register) must match to be a valid write. This write to the WDSVR register involves two irrevocable choices: (i) the selection of the WATCHDOG service window (ii) enabling or disabling of the Clock Monitor. Hence, the first write to WDSVR Register involves selecting or deselecting the Clock Monitor, select the WATCHDOG service window and match the WATCHDOG key data. Subsequent writes to the WDSVR register will compare the value being written by the user to the WATCHDOG service window value and the key data (bits 7 through 1) in the WDSVR Register. Table XX shows the sequence of events that can occur.

WATCHDOG Operation (Continued)

The user must service the WATCHDOG at least once before the upper limit of the service window expires. The WATCHDOG may not be serviced more than once in every lower limit of the service window. The user may service the WATCHDOG as many times as wished in the time period between the lower and upper limits of the service window. The first write to the WDSVR Register is also counted as a WATCHDOG service.

The WATCHDOG has an output pin associated with it. This is the WDOUT pin, on pin 1 of the Port G. WDOUT is active low. The WDOUT pin is in the high impedance state in the inactive state. Upon triggering the WATCHDOG, the logic will pull the WDOUT (G1) pin low for an additonal 16 t_C–32 t_C cycle after the signal level on WDOUT pin goes below the lower Schmitt trigger threshold. After this delay, the device will stop forcing the WDOUT output low.

The WATCHDOG service window will restart when the WDOUT pin goes high. It is recommended that the user tie the WDOUT pin back to V_{CC} through a resistor in order to pull WDOUT high.

A WATCHDOG service while the WDOUT signal is active will be ignored. The state of the WDOUT pin is not guaranteed on reset, but if the powers up low then the WATCHDOG will time out and WDOUT will enter high impedance state.

The Clock Monitor forces the G1 pin low upon detecting a clock frequency error. The Clock Monitor error will continue until the clock frequency has reached the minimum specified value, after which the G1 output will enter the high impedance TRI-STATE mode following 16 t_C–32 t_C clock cycles. The Clock Monitor generates a continual Clock Monitor error if the oscillator fails to start, or fails to reach the minimum specified frequency. The specification for the Clock Monitor is as follows:

$1/t_C > 10$ kHz—No clock rejection.

$1/t_C < 10$ Hz—Guaranteed clock rejection.

WATCHDOG AND CLOCK MONITOR SUMMARY

The following salient points regarding the COP888 WATCHDOG and CLOCK MONITOR should be noted:

- Both the WATCHDOG and Clock Monitor detector circuits are inhibited during RESET.
- Following RESET, the WATCHDOG and CLOCK MONITOR are both enabled, with the WATCHDOG having the maximum service window selected.
- The WATCHDOG service window and Clock Monitor enable/disable option can only be changed once, during the initial WATCHDOG service following RESET.
- The initial WATCHDOG service must match the key data value in the WATCHDOG Service register WDSVR in order to avoid a WATCHDOG error.
- Subsequent WATCHDOG services must match all three data fields in WDSVR in order to avoid WATCHDOG errors.
- The correct key data value cannot be read from the WATCHDOG Service register WDSVR. Any attempt to read this key data value of 01100 from WDSVR will read as key data value of all 0's.
- The WATCHDOG detector circuit is inhibited during both the HALT and IDLE modes.
- The Clock Monitor detector circuit is active during both the HALT and IDLE modes. Consequently, the device inadvertently entering the HALT mode will be detected as a Clock Monitor error (provided that the Clock Monitor enable option has been selected by the program).
- With the single-pin R/C oscillator mask option selected and the CLKDLY bit reset, the WATCHDOG service window will resume following HALT mode from where it left off before entering the HALT mode.
- With the crystal oscillator mask option selected, or with the single-pin R/C oscillator mask option selected and the CLKDLY bit set, the WATCHDOG service window will be set to its selected value from WDSVR following HALT. Consequently, the WATCHDOG should not be serviced for at least 2048 instruction cycles following HALT, but must be serviced within the selected window to avoid a WATCHDOG error.
- The IDLE timer T0 is not initialized with RESET.
- The user can sync in to the IDLE counter cycle with an IDLE counter (T0) interrupt or by monitoring the T0PND flag. The T0PND flag is set whenever the thirteenth bit of the IDLE counter toggles (every 4096 instruction cycles). The user is responsible for resetting the T0PND flag.
- A hardware WATCHDOG service occurs just as the device exits the IDLE mode. Consequently, the WATCHDOG should not be serviced for at least 2048 instruction cycles following IDLE, but must be serviced within the selected window to avoid a WATCHDOG error.
- Following RESET, the initial WATCHDOG service (where the service window and the CLOCK MONITOR enable/disable must be selected) may be programmed anywhere within the maximum service window (65,536 instruction cycles) initialized by RESET. Note that this initial WATCHDOG service may be programmed within the intitial 2048 instruction cycles without causing a WATCHDOG error.

TABLE XX. WATCHDOG Service Actions

Key Data	Window Data	Clock Monitor	Action
Match	Match	Match	Valid Service: Restart Service Window
Don't Care	Mismatch	Don't Care	Error: Generate WATCHDOG Output
Mismatch	Don't Care	Don't Care	Error: Generate WATCHDOG Output
Don't Care	Don't Care	Mismatch	Error: Generate WATCHDOG Output

Memory Map

All RAM, ports and registers (except A and PC) are mapped into data memory address space.

Address	Contents
0000 to 006F	On-Chip RAM bytes (112 bytes)
0070 to 007F	Unused RAM Address Space (Reads as All Ones)
0080	PORTMD, Port M Data Register
0081	PORTMC, Port M Configuration Register
0082	PORTMP, Port M Input Pins (Read Only)
0083	reserved for Port M
0084	MMIWU Edge Select Register (MWKEDG)
0085	MMIWU Enable Register (MWKEN)
0086	MMIWU Pending Register (MWKPND)
0087	reserved for MMIWU
0088	PORTND, Port N Data Register
0089	PORTNC, Port N Configuration Register
008A	PORTNP, Port N Input Pins (Read Only)
008B	PORTNX, Port N Alternate Function Enable
008C to 008F	Unused RAM Address Space (Reads Undefined Data)
0090	PORTED, Port E Data Register
0091	PORTEC, Port E Configuration Register
0092	PORTEP, Port E Input Pins (Read Only)
0093	reserved for Port E
0094	PORTFD, Port F Data Register
0095	PORTFC, Port F Configuration Register
0096	PORTFP, Port F Input Pins (Read Only)
0097	reserved for Port F
0098	SPICNTL, SPI Control Register
0099	SPISTAT, SPI Status Register
009A	SPIRXD, SPI Current Receive Data (Read Only)
009B	SPITXD, SPI Transmit Data
009c to 009F	unused
00A0	TXD1, Transmit 1 Data
00A1	TXD2, Transmit 2 Data
00A2	TDLC, Transmit Data Length Code and Identifier Low
00A3	TID, Transmit Identifier High
00A4	RXD1, Receive Data 1
00A5	RXD2, Receive Data 2
00A6	RIDL, Receive Data Length Code
00A7	RID, Receive Identify HIgh
00A8	CSCAL, CAN Prescaler
00A9	CTIM, Bus Timing Register
00AA	CBUS, Bus Control Register
00AB	TCNTL, Transmit/Receive Control Register
00AC	RTSTAT Receive/Transmit Status Register
00AD	TEC, Transmit Error Count Register
00AE	REC, Receive Error Count Register
00AF	PLATST, CAN Bit Stream Processor Test Register

Address	Contents
00B8	UART Transmit Buffer (TBUF)
00B9	UART Receive Buffer (RBUF)
00BA	UART Control Status (ENU)
00BB	UART Receive Control Status (ENUR)
00BC	UART Interrupt and Clock (ENUI)
00BD	UART Baud Register (BAUD)
00BE	UART Prescaler Register (PSR)
00BF	reserved for UART
00C0	Timer T2 Lower Byte (TMR2LO)
00C1	Timer T2 Upper Byte (TMR2HI)
00C2	Timer T2 Autoload Register T2RA Lower Byte (T2RALO)
00C3	Timer T2 Autoload Register T2RA Upper Byte (T2RAHI)
00C4	Timer T2 Autoload Register T2RB Lower Byte (T2RBLO)
00C5	Timer T2 Autoload Register T2RB Upper Byte (T2RBHI)
00C6	Timer T2 Control Register (T2CNTRL)
00C7	WATCHDOG Service Register (Reg:WDSVR)
00C8	LMIWU Edge Select Register (LWKEDG)
00C9	LMIWU Enable Register (LWKEN)
00CA	LLMIWU Pending Register (LWKPND)
00CB	A/D Converter Control Register (Reg:ENAD)
00CC	A/D Converter Result Register (Reg:ADRSLT)
00CD to 00CE	Reserved
00CF	IDLE Timer Control Register (Reg:ITMR)
00D0	PORTLD, Port L Data Register
00D1	PORTLC, Port L Configuration Register
00D2	PORTLP, Port L Input Pins (Read Only)
00D3	Reserved for Port L
00D4	PORTGD, Port G Data Register
00D5	PORTGC, Port G Configuration Register
00D6	PORTGP, Port G Input Pins (Read Only)
00D7	Port I Input Pins (Read Only)
00D8	Port CD, Port C Data Register
00D9	Port CC, Port C Configuration Register
00DA	Port CP, Port C Input Pins (Read Only)
00DB	Reserved for Port C
00DC	Port D
00DD to 00DF	Reserved for Port D
00E0 to 00E5	Reserved for EE Control Registers
00E6	Timer T1 Autoload Register T1RB Lower Byte (T1BRLO)
00E7	Timer T1 Autoload Register T1RB Upper Byte (T1BRHI)
00E8	ICNTRL Register
00E9	MICROWIRE/PLUS Shift Register (SOIR)
00EA	Timer T1 Lower Byte (TMR1LO)
00EB	Timer T1 Upper Byte (TMR1HI)
00EC	Timer T1 Autoload Register T1RA Lower Byte (T1RALO)

Memory Map (Continued)

Address	Contents
00ED	Timer T1 Autoload Register T1RA Upper Byte (T1RAHI)
00EE	CNTRL, Control Register
00EF	PSW, Processor Status Word Register
00F0 to 00FB	On-Chip RAM Mapped as Registers
00FC	X Register
00FD	SP Register
00FE	B Register
00FF	S Register
0100 to 013F	On-Chip RAM Bytes (64 Bytes)
Reading memory locations 0070H–007FH will return all ones. Reading unused memory locations 00xxH–00xxH will return undefined data.	

Addressing Modes

There are ten addressing modes, six for operand addressing and four for transfer of control.

OPERAND ADDRESSING MODES

Register Indirect

This is the "normal" addressing mode. The operand is the data memory addressed by the B pointer or X pointer.

Register Indirect (with auto post increment or decrement of pointer)

This addressing mode is used with the LD and X instructions. The operand is the data memory addressed by the B pointer or X pointer. This is a register indirect mode that automatically post increments or decrements the B or X register after executing the instruction.

Direct

The instruction contains an 8-bit address field that directly points to the data memory for the operand.

Immediate

The instruction contains an 8-bit immediate field as the operand.

Short Immediate

This addressing mode is used with the Load B Immediate instruction. The instruction contains a 4-bit immediate field as the operand.

Indirect

This addressing mode is used with the LAID instruction. The contents of the accumulator are used as a partial address (lower 8 bits of PC) for accessing a data operand from the program memory.

TRANSFER OF CONTROL ADDRESSING MODES

Relative

This mode is used for the JP instruction, with the instruction field being added to the program counter to get the new program location. JP has a range from −31 to +32 to allow a 1-byte relatie jump (JP + 1 is implemented by a NOP instruction). There are no "pages" when using JP, since all 15 bits of PC are used.

Absolute

The mode is used with the JMP and JSR instructions, with the instruction field of 12 bits replacing the lower 12 bits of the program counter (PC). This allows jumping to any location in the current 4k program memory segment.

Absolute Long

This mode is used with the JMPL and JSRL instructions, with the instruction field of 15 bits replacing the entire 15 bits of the program counter (PC). This allows jumping to any location up to 32k in the program memory space.

Indirect

This mode is used with the JID instruction. The contents of the accumulator are used as a partial address (lower 8 bits of PC) for accessing a location in the program memory. The contents of this program memory location serve as a partial address (lower 8 bits of PC) for the jump to the next instruction.

Note: The VIS is a special case of the Indirect Transfer of Control addressing mode, where the double byte vector associated with the interrupt is transferred from adjacent addresses in the program memory into the program counter (PC) in order to jump to the associated interrupt service routine.

Instruction Set

Register and Symbol Definition

Registers	
A	8-Bit Accumulator Register
B	8-Bit Address Register
X	8-Bit Address Register
SP	8-Bit Stack Pointer Register
PC	15-Bit Program Counter Register
PU	Upper 7 Bits of PC
PL	Lower 8 Bits of PC
C	1 Bit of PSW Register for Carry
HC	1 Bit of PSW Register for Half Carry
GIE	1 Bit of PSW Register for Global Interrupt Enable
VU	Interrupt Vector Upper Byte
VL	Interrupt Vector Lower Byte

Symbols	
[B]	Memory Indirectly Addressed by B Register
[X]	Memory Indirectly Addressed by X Register
MD	Direct Addressed Memory
Mem	Direct Addressed Memory or [B]
Meml	Direct Addressed Memory or [B] or Immediate Data
Imm	8-Bit Immediate Data
Reg	Register Memory: Addresses F0 to FF (Includes B, X and SP)
Bit	Bit Number (0 to 7)
←	Loaded with
⟷	Exchanged with

Instruction Set (Continued)

ADD	A,Meml	ADD	A ← A + Meml
ADC	A,Meml	ADD with Carry	A ← A + Meml + C, C ← Carry, HC ← Half Carry,
SUBC	A,Meml	Subtract with Carry	A ← A − Meml + C, C ← Carry, HC ← Half Carry
AND	A,Meml	Logical AND	A ← A and $\overline{\text{Meml}}$
ANDSZ	A,Imm	Logical AND Immed., Skip if Zero	Skip next if (A and Imm) = 0
OR	A,Meml	Logical OR	A ← A or Meml
XOR	A,Meml	Logical EXclusive OR	A ← A xor Meml
IFEQ	MD,Imm	IF EQual	Compare MD and Imm, Do next if MD = Imm
IFEQ	A,Meml	IF EQual	Compare A and Meml, Do next if A = Meml
IFNE	A,Meml	IF Not Equal	Compare A and Meml, Do next if A ≠ Meml
IFGT	A,Meml	IF Greater Than	Compare A and Meml, Do next if A > Meml
IFBNE	#	IF B Not Equal	Do next if lower 4 bits of B ≠ Imm
DRSZ	Reg	Decrement Reg., Skip if Zero	Reg ← Reg − 1, Skip if Reg = 0
SBIT	#,Mem	Set BIT	1 to bit, Mem (bit = 0 to 7 immediate)
RBIT	#,Mem	Reset BIT	0 to bit, Mem
IFBIT	#,Mem	IF BIT	If bit in A or Mem is true do next instruction
RPND		Reset PeNDing Flag	Reset Software Interrupt Pending Flag
X	A,Mem	EXchange A with Memory	A ⟷ Mem
X	A,[X]	EXchange A with Memory [X]	A ⟷ [X]
LD	A,Meml	LoaD A with Memory	A ← Meml
LD	A,[X]	LoaD A with Memory [X]	A ← [X]
LD	B,Imm	LoaD B with Immed.	B ← Imm
LD	Mem,Imm	LoaD Memory Immed.	Mem ← Imm
LD	Reg,Imm	LoaD Register Memory Immed.	Reg ← Imm
X	A, [B]	EXchange A with Memory [B]	A ⟷ [B], (B ← B 1)
X	A, [X]	EXchange A with Memory [X]	A ⟷ [X], (X ← X 1)
LD	A, [B]	LoaD A with Memory [B]	A ← [B], (B ← B 1)
LD	A, [X]	LoaD A with Memory [X]	A ← [X], (X ← X 1)
LD	[B],Imm	LoaD Memory [B] Immed.	[B] ← Imm, (B ← B 1)
CLR	A	CLeaR A	A ← 0
INC	A	INCrement A	A ← A + 1
DEC	A	DECrement A	A ← A − 1
LAID		Load A InDirect from ROM	A ← ROM (PU,A)
DCOR	A	Decimal CORrect A	A ← BCD correction of A (follows ADC, SUBC)
RRC	A	Rotate A Right thru C	C → A7 → . . . → A0 → C
RLC	A	Rotate A Left thru C	C ← A7 ← . . . ← A0 ← C
SWAP	A	SWAP nibbles of A	A7 . . . A4 ⟷ A3 . . . A0
SC		Set C	C ← 1, HC ← 1
RC		Reset C	C ← 0, HC ← 0
IFC		IF C	IF C is true, do next instruction
IFNC		IF Not C	If C is not true, do next instruction
POP	A	POP the stack into A	SP ← SP + 1, A ← [SP]
PUSH	A	PUSH A onto the stack	[SP] ← A, SP ← SP − 1
VIS		Vector to Interrupt Service Routine	PU ← [VU], PL ← [VL]
JMPL	Addr.	Jump absolute Long	PC ← ii (ii = 15 bits, 0 to 32k)
JMP	Addr.	Jump absolute	PC9 . . . 0 ← i (i = 12 bits)
JP	Disp.	Jump relative short	PC ← PC + r (r is −31 to +32, except 1)
JSRL	Addr.	Jump SubRoutine Long	[SP] ← PL, [SP−1] ← PU,SP−2, PC ← ii
JSR	Addr.	Jump SubRoutine	[SP] ← PL, [SP−1] ← PU,SP−2, PC9 . . . 0 ← i
JID		Jump InDirect	PL ← ROM (PU,A)
RET		RETurn from subroutine	SP + 2, PL ← [SP], PU ← [SP−1]
RETSK		RETurn and SKip	SP + 2, PL ← [SP],PU ← [SP−1]
RETI		RETurn from Interrupt	SP + 2, PL ← [SP],PU ← [SP−1],GIE ← 1
INTR		Generate an Interrupt	[SP] ← PL, [SP−1] ← PU, SP−2, PC ← 0FF
NOP		No OPeration	PC ← PC + 1

Instruction Set (Continued)

INSTRUCTION EXECUTION TIME

Most instructions are single byte (with immediate addressing mode instructions taking two bytes).

Most single byte instructions take one cycle time to execute.

Skipped instructions require x number of cycles to be skipped, where x equals the number of bytes in the skipped instruction opcode.

See the BYTES and CYCLES per INSTRUCTION table for details.

Bytes and Cycles per Instruction

The following table shows the number of bytes and cycles for each instruction in the format of byte/cycle.

Arithmetic and Logic Instructions

	[B]	Direct	Immed.
ADD	1/1	3/4	2/2
ADC	1/1	3/4	2/2
SUBC	1/1	3/4	2/2
AND	1/1	3/4	2/2
OR	1/1	3/4	2/2
XOR	1/1	3/4	2/2
IFEQ	1/1	3/4	2/2
IFGT	1/1	3/4	2/2
IFBNE	1/1		
DRSZ		1/3	
SBIT	1/1	3/4	
RBIT	1/1	3/4	
IFBIT	1/1	3/4	

RPND	1/1

Instructions Using A and C

CLRA	1/1
INCA	1/1
DECA	1/1
LAID	1/3
DCORA	1/1
RRCA	1/1
RLCA	1/1
SWAPA	1/1
SC	1/1
RC	1/1
IFC	1/1
IFNC	1/1
PUSHA	1/3
POPA	1/3
ANDSZ	2/2

Transfer of Control Instructions

JMPL	3/4
JMP	2/3
JP	1/3
JSRL	3/5
JSR	2/5
JID	1/3
VIS	1/5
RET	1/5
RETSK	1/5
RETI	1/5
INTR	1/7
NOP	1/1

Memory Transfer Instructions

	Register Indirect		Direct	Immed.	Register Indirect Auto Incr. and Decr.		
	[B]	[X]			[B+, B−]	[X+, X−]	
X A,*	1/1	1/3	2/3		1/2	1/3	
LD A,*	1/1	1/3	2/3	2/2	1/2	1/3	
LD B, Imm				1/1			(IF B < 16)
LD B, Imm				2/3			(IF B > 15)
LD Mem, Imm	2/2		3/3		2/2		
LD Reg, Imm			2/3				
IFEQ MD, Imm			3/3				

* = > Memory location addressed by B or X or directly.

COP888 Family Opcode Table

Lower Nibble	Upper Nibble															
	F	E	D	C	B	A	9	8	7	6	5	4	3	2	1	0
0	JP−15	JP−31	LD 0F0, #i	DRSZ 0F0	RRCA	RC	ADC A,#i	ADC A,[B]	IFBIT 0,[B]	ANDSZ A, #i	LD B,#0F	IFBNE 0	JSR x000−x0FF	JMP x000−x0FF	JP+17	INTR
1	JP−14	JP−30	LD 0F1, #i	DRSZ 0F1	*	SC	SUBC A, #i	SUBC A,[B]	IFBIT 1,[B]	*	LD B,#0E	IFBNE 1	JSR x100−x1FF	JMP x100−x1FF	JP+18	JP+2
2	JP−13	JP−29	LD 0F2, #i	DRSZ 0F2	X A,[X+]	X A,[B+]	IFEQ A,#i	IFEQ A,[B]	IFBIT 2,[B]	*	LD B,#0D	IFBNE 2	JSR x200−x2FF	JMP x200−x2FF	JP+19	JP+3
3	JP−12	JP−28	LD 0F3, #i	DRSZ 0F3	X A,[X−]	X A,[B−]	IFGT A,#i	IFGT A,[B]	IFBIT 3,[B]	*	LD B,#0C	IFBNE 3	JSR x300−x3FF	JMP x300−x3FF	JP+20	JP+4
4	JP−11	JP−27	LD 0F4, #i	DRSZ 0F4	VIS	LAID	ADD A,#i	ADD A,[B]	IFBIT 4,[B]	CLRA	LD B,#0B	IFBNE 4	JSR x400−x4FF	JMP x400−x4FF	JP+21	JP+5
5	JP−10	JP−26	LD 0F5, #i	DRSZ 0F5	RPND	JID	AND A,#i	AND A,[B]	IFBIT 5,[B]	SWAPA	LD B,#0A	IFBNE 5	JSR x500−x5FF	JMP x500−x5FF	JP+22	JP+6
6	JP−9	JP−25	LD 0F6, #i	DRSZ 0F6	X A,[X]	X A,[B]	XOR A,#i	XOR A,[B]	IFBIT 6,[B]	DCORA	LD B,#09	IFBNE 6	JSR x600−x6FF	JMP x600−x6FF	JP+23	JP+7
7	JP−8	JP−24	LD 0F7, #i	DRSZ 0F7	*	*	OR A,#i	OR A,[B]	IFBIT 7,[B]	PUSHA	LD B,#08	IFBNE 7	JSR x700−x7FF	JMP x700−x7FF	JP+24	JP+8
8	JP−7	JP−23	LD 0F8, #i	DRSZ 0F8	NOP	RLCA	LD A,#i	IFC	SBIT 0,[B]	RBIT 0,[B]	LD B,#07	IFBNE 8	JSR x800−x8FF	JMP x800−x8FF	JP+25	JP+9
9	JP−6	JP−22	LD 0F9, #i	DRSZ 0F9	IFNE A,[B]	IFEQ Md,#i	IFNE A,#i	IFNC	SBIT 1,[B]	RBIT 1,[B]	LD B,#06	IFBNE 9	JSR x900−x9FF	JMP x900−x9FF	JP+26	JP+10
A	JP−5	JP−21	LD 0FA, #i	DRSZ 0FA	LD A,[X+]	LD A,[B+]	LD [B+],#i	INCA	SBIT 2,[B]	RBIT 2,[B]	LD B,#05	IFBNE 0A	JSR xA00−xAFF	JMP xA00−xAFF	JP+27	JP+11
B	JP−4	JP−20	LD 0FB, #i	DRSZ 0FB	LD A,[X−]	LD A,[B−]	LD [B−],#i	DECA	SBIT 3,[B]	RBIT 3,[B]	LD B,#04	IFBNE 0B	JSR xB00−xBFF	JMP xB00−xBFF	JP+28	JP+12
C	JP−3	JP−19	LD 0FC, #i	DRSZ 0FC	LD Md,#i	JMPL	X A,Md	POPA	SBIT 4,[B]	RBIT 4,[B]	LD B,#03	IFBNE 0C	JSR xC00−xCFF	JMP xC00−xCFF	JP+29	JP+13
D	JP−2	JP−18	LD 0FD, #i	DRSZ 0FD	DIR	JSRL	LD A,Md	RETSK	SBIT 5,[B]	RBIT 5,[B]	LD B,#02	IFBNE 0D	JSR xD00−xDFF	JMP xD00−xDFF	JP+30	JP+14
E	JP−1	JP−17	LD 0FE, #i	DRSZ 0FE	LD A,[X]	LD A,[B]	LD [B],#i	RET	SBIT 6,[B]	RBIT 6,[B]	LD B,#01	IFBNE 0E	JSR xE00−xEFF	JMP xE00−xEFF	JP+31	JP+15
F	JP−0	JP−16	LD 0FF, #i	DRSZ 0FF	*	*	LD B,#i	RETI	SBIT 7,[B]	RBIT 7,[B]	LD B,#00	IFBNE 0F	JSR xF00−xFFF	JMP xF00−xFFF	JP+32	JP+16

Where,

i is the immediate data

Md is a directly addressed memory location

* is an unused opcode

Note: The opcode 60 Hex is also the opcode for IFBIT #i,A

Mask Options

The COP684E and COP884EB mask programmable options are shown below. The options are programmed at the same time as the ROM pattern submission.

OPTION 1: CLOCK CONFIGURATION

= 1 Crystal Oscillator (CKI/10)

G7 (CKO) is clock generator output to crystal/resonator

CKI is the clock input

OPTION 2: HALT

= 1 Enable HALT mode

OPTION 3: BONDING OPTIONS

= 1 68-Pin PLCC

= 2 44-Pin PLCC

The chip can be driven by a clock input on the CKI input pin which can be between DC and 10 MHz. The CKO output clock is on pin G7. The CKI input frequency is divided down by 10 to produce the instruction cycle clock ($1/t_C$).

Development Support

SUMMARY

- iceMASTER: IM-COP8/400—Full feature in-circuit emulation for all COP8 products. A full set of COP8 Basic and Feature Family device and package specific probes are available.
- COP8 Debug Module: Moderate cost in-circuit emulation and development programming unit.
- COP8 Evaluation and Programming Unit: EPU-COP888GG—low cost in-circuit simulation and development programming unit.
- Assembler: COP8-DEV-IBMA. A DOS installable cross development Assembler, Linker, Librarian and Utility Software Development Tool Kit.
- C Compiler: COP8C. A DOS installable cross development Software Tool Kit.
- OTP/EPROM Programmer Support: Covering needs from engineering prototype, pilot production to full production environments.

Development Support (Continued)

IceMASTER (IM) IN-CIRCUIT EMULATION

The iceMASTER IM-COP8/400 is a full feature, PC based, in-circuit emulation tool development and marketed by MetaLink Corporation to support the whole COP8 family of products. National is a resale vendor for these products.

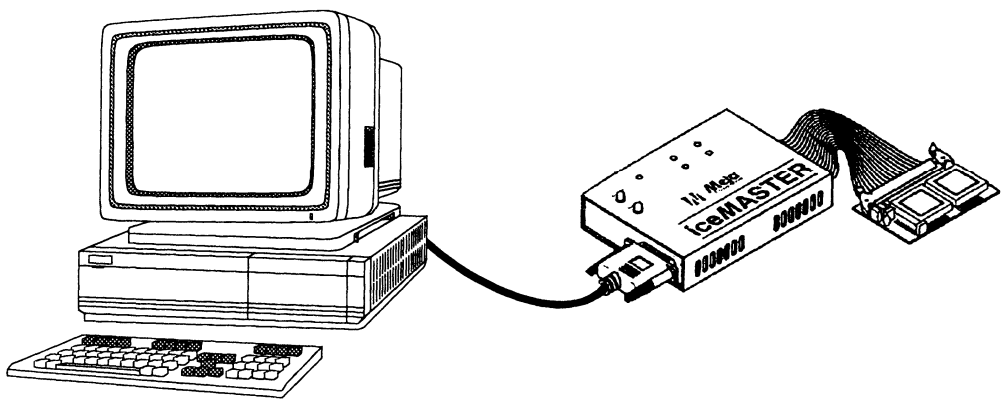
See *Figure 46* for configuration.

The iceMASTER IM-COP8/400 with its device specific COP8 Probe provides a rich feature set for developing, testing and maintaining product:

- Real-time in-circuit emulation; full 2.4V–5.5V operation range, full DC-10 MHz clock. Chip options are programmable or jumper selectable.
- Direct connection to application board by package compatible socket or surface mount assembly.
- Full 32-kbyte of loadable programming space that overlays (replaces) the on-chip ROM or EPROM. On-chip RAM and I/O blocks are used directly or recreated on the probe as necessary.
- Full 4k frame synchronous trace memory. Address, instruction, and eight unspecified, circuit connectable trace lines. Display can be HLL source (e.g., C source), assembly or mixed.
- A full 64k hardware configurable break, trace on, trace off control, and pass count increment events.
- Tool set integrated interactive symbolic debugger—supports both assembler (COFF) and C Compiler (.COD) linked object formats.
- Real time performance profiling analysis; selectable bucket definition.
- Watch windows, content updated automatically at each execution break.
- Instruction by instruction memory/register changes displayed on source window when in single step operation.
- Single base unit and debugger software reconfigurable to support the entire COP8 family; only the probe personality needs to change. Debugger software is processor customized, and reconfigured from a master model file.
- Processor specific symbolic display of registers and bit level assignments, configured from master model file.
- Halt/Idle mode notification.
- On-Line HELP customized to specific processor using master model file.
- Includes a copy of COP8-DEV-IBMA assembler and linker SDK.

IM Order-Information

Base Unit	
IM-COP8/400-1	iceMASTER Base Unit, 110V Power Supply
IM-COP8/400-2	iceMASTER Base Unit, 220V Power Supply
iceMASTER Probe	
MHW-888EB44PWPC MHW-888EB68PWPC	44 PLCC 68 PLCC



TL/DD/12871–52

FIGURE 46. COP8 iceMASTER Environment

Development Support (Continued)

iceMASTER DEBUG MODULE (DM)

The iceMASTER Debug Module is a PC based, combination in-circuit emulation tool and COP8 based OTP/EPROM programming tool developed and marketed by MetaLink Corporation to support the whole COP8 family of products. National is a resale vendor for these products.

See *Figure 47* for configuration.

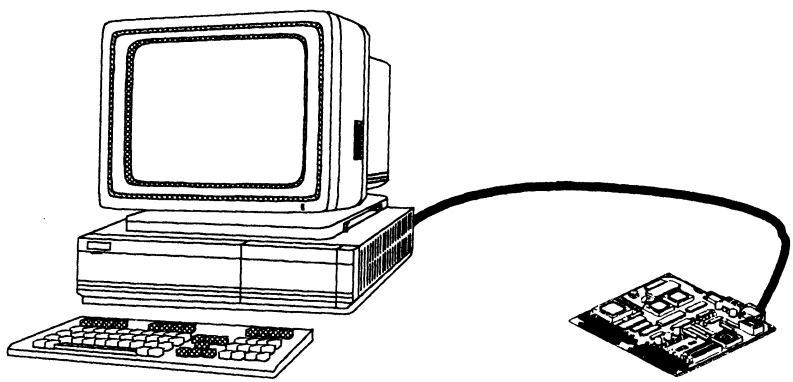
The iceMASTER Debug Module is a moderate cost development tool. It has the capability of in-circuit emulation for a specific COP8 microcontroller and in addition serves as a programming tool for COP8 OTP and EPROM product families. Summary of features is as follows:

- Real-time in-circuit emulation; full operating voltage range operation, full DC-10 MHz clock.
- All processor I/O pins can be cabled to an application development board with package compatible cable to socket and surface mount assembly.
- Full 32 kbyte of loadable programming space that overlays (replaces) the on-chip ROM or EPROM. On-chip RAM and I/O blocks are used directly or recreated as necessary.
- 100 frames of synchronous trace memory. The display can be HLL source (C source), assembly or mixed. The most recent history prior to a break is available in the trace memory.
- Configured break points; uses INTR instruction which is modestly intrusive.
- Software—only supported features are selectable.
- Tool set integrated interactive symbolic debugger—supports both assembler (COFF) and C Compiler (.COD) SDK linked object formats.
- Instruction by instruction memory/register changes displayed when in single step operation.
- Debugger software is processor customized, and reconfigured from a master model file.
- Processor specific symbolic display of registers and bit level assignments, configured from master model file.
- Halt/Idle mode notification.
- Programming menu supports full product line of programmable OTP and EPROM COP8 products. Program data is taken directly from the overlay RAM.
- Programming of 44 PLCC and 68 PLCC parts requires external programming adapters.
- Includes wallmount power supply.
- On-board VPP generator from 5V input or connection to external supply supported. Requires VPP level adjustment per the family programming specification (correct level is provided on an on-screen pop-down display).
- On-line HELP customized to specific processor using master model file.
- Includes a copy of COP8-DEV-IBMA assembler and linker SDK.

DM Order-Information

Debug Module Unit	
COP8-DM/888EB	
Cable Adapters	
DM-COP8/44P	44 PLCC
DM-COP8/68P	68 PLCC

Please contact local sales office for ordering information of programming adapter.



TL/DD/12871-53

FIGURE 47. COP8-DM Environment

Development Support (Continued)

COP8 ASSEMBLER/LINKER SOFTWARE DEVELOPMENT TOOL KIT

National Semiconductor offers a relocatable COP8 macro cross assembler, linker, librarian and utility software development tool kit. Features are summarized as follows:

- Basic and Feature Family instruction set by "device" type.
- Nested macro capability.
- Extensive set of assembler directives.
- Supported on PC/DOS platform.
- Generates National standard COFF output files.
- Integrated Linker and Librarian.
- Integrated utilities to generate ROM code file outputs.
- DUMPCOFF utility.

This product is integrated as a part of MetaLink tools as a development kit, fully supported by the MetaLink debugger. It may be ordered separately or it is bundled with the MetaLink products at no additional cost.

Order Information

Assembler SDK:	
COP8-DEV-IBMA	Assembler SDK on installable 3.5" PC/DOS Floppy Disk Drive format. Periodic upgrades and most recent version is available on National's BBS and Internet.

COP8 C COMPILER

A C Compiler is developed and marketed by Byte Craft Limited. The COP8C compiler is a fully integrated development tool specifically designed to support the compact embedded configuration of the COP8 family of products.

Features are summarized as follows:

- ANSI C with some restrictions and extensions that optimize development for the COP8 embedded application.
- BITS data type extension. Register declaration #pragma with direct bit level definitions.
- C language support for interrupt routines.
- Expert system, rule based code generation and optimization.
- Performs consistency checks against the architectural definitions of the target COP8 device.
- Generates program memory code.
- Supports linking of compiled object or COP8 assembled object formats.
- Global optiomization of linked code.
- Symbolic debug load format fully source level supported by the MetaLink debugger.

SINGLE CHIP OTP/EMULATOR SUPPORT

The COP8 family is supported by single chip OTP emulators. For detailed information refer to the emulator specific datasheet and the emulator selection table below:

OTP Emulator Ordering Information

Device Number	Clock Option	Package	Emulates
COP87L88EB-XE	Crystal	44 PLCC	COP888EB
COP87L89EB-XE	Crystal	68 PLCC	COP889EB

INDUSTRY WIDE OTP/EPROM PROGRAMMING SUPPORT

Programming support, in addition to the MetaLink development tools, is provided by a full range of independent approved vendors to meet the needs from the engineering laboratory to full production.

Approved List

Manufacturer	North America	Europe	Asia
BP Microsystems	(800) 225-2102 (713) 688-4600 Fax: (713) 688-0920	+49-8152-4183 +49-8856-932616	+852-234-16611 +852-2710-8121
Data I/O	(800) 426-1045 (206) 881-6444 Fax: (206) 882-1043	+44-0734-440011	Call North America
HI-LO	(510) 623-8860	Call Asia	+886-2-764-0215 Fax: +886-2-756-6403
ICE Technology	(800) 624-8949 (919) 430-7915	+44-1226-761404 Fax: 0-1226-370-434	
MetaLink	(800) 638-2423 (602) 926-0797 Fax: (602) 693-0681	+49-80 9156 96-0 Fax: +49-80 9123 86	+852-737-1800
Systems General	(408) 263-6667	+41-1-9450300	+886-2-917-3005 Fax: +886-2-911-1283
Needhams	(916) 924-8037 Fax: (916) 924-8065		

Development Support (Continued)

AVAILABLE LITERATURE

For more information, please see the COP8 Basic Family User's Manual, Literature Number 620895, COP8 Feature Family User's Manual, Literature Number 620897 and National's Family of 8-bit Microcontrollers COP8 Selection Guide, Literature Number 630009.

DIAL-A-HELPER SERVICE

Dial-A-Helper is a service provided by the Microcontroller Applications group. The Dial-A-Helper is an Electronic Information System that may be accessed as a Bulletin Board System (BBS) via data modem, as an FTP site on the Internet via standard FTP client application or as an FTP site on the Internet using a standard Internet browser such as Netscape or Mosaic.

The Dial-A-Helper system provides access to an automated information storage and retrieval system. The system capabilities include a MESSAGE SECTION (electronic mail, when accessed as a BBS) for communications to and from the Microcontroller Applications Group and a FILE SECTION which consists of several file areas where valuable application software and utilities could be found.

DIAL-A-HELPER BBS via a Standard Modem

Modem: CANADA/U.S.: (800) NSC-MICRO
(800) 672-6427
EUROPE: (+49) 0-814-135 13 32
Baud: 14.4k
Set-up: Length: 8-Bit
Parity: None
Stop Bit: 1
Operation: 24 Hrs., 7 Days

DIAL-A-HELPER via FTP

ftp nscmicro.nsc.com
user: anonymous
password: username@yourhost.site.domain

DIAL-A-HELPER via a WorldWide Web Browser

ftp://nscmicro.nsc.com

National Semiconductor on the WorldWide Web

See us on the WorldWide Web at: http://www.national.com

CUSTOMER RESPONSE CENTER

Complete product information and technical support is available from National's customer response centers.

CANADA/U.S.:	Tel:	(800) 272-9959
	email:	support @tevm2.nsc.com
EUROPE:	email:	europe.support@nsc.com
	Deutsch Tel:	+49 (0) 180-530 85 85
	English Tel:	+49 (0) 180-532 78 32
	Français Tel:	+49 (0) 180-532 93 58
	Italiano Tel:	+49 (0) 180-534 16 80
JAPAN:	Tel:	+81-043-299-2309
S.E. ASIA:	Beijing Tel:	(+86) 10-6856-8601
	Shanghai Tel:	(+86) 21-6415-4092
	Hong Kong Tel:	(+852) 2737-1600
	Korea Tel:	(+82) 2-3771-6909
	Malaysia Tel:	(+60-4) 644-9061
	Singapore Tel:	(+65) 255-2226
	Taiwan Tel:	+886-2-521-3288
AUSTRALIA:	Tel:	(+61) 3-9558-9999
INDIA:	Tel:	(+91) 80-559-9467

National Semiconductor

PRELIMINARY

COP87L88CL/COP87L84CL 8-Bit One-Time Programmable (OTP) Microcontroller

General Description

The COP87L88CL/COP87L84CL OTP microcontrollers are members of the COP8™ feature family using an 8-bit core architecture. It is pin and software compatible to the mask ROM COP888CL/COP884CL product family.

(Continued)

Key Features

- Two 16-bit timers, each with two 16-bit registers supporting:
  - — Processor independent PWM mode
  - — External event counter mode
  - — Input capture mode
- 4 kbytes on-board EPROM with security feature
- 128 bytes on-board RAM

Additional Peripheral Features

- Idle timer
- Multi-Input Wake-Up (MIWU) with optional interrupts (8)
- WATCHDOG™ and clock monitor logic
- MICROWIRE/PLUS™ serial I/O

I/O Features

- Memory mapped I/O
- Software selectable I/O options (TRI-STATE® output, push-pull output, weak pull-up input, high impedance input)
- Schmitt trigger inputs on ports G and L
- Packages:
  - — 44 PLCC with 39 I/O pins
  - — 40 DIP with 33 I/O pins
  - — 28 DIP with 24 I/O pins
  - — 28 SO with 24 I/O pins (contact local sales office for availability)

CPU/Instruction Set Features

- 1 μs instruction cycle time
- Ten multi-source vectored interrupts servicing
  - — External interrupt
  - — Idle timer T0
  - — Two timers (each with 2 Interrupts)
  - — MICROWIRE/PLUS
  - — Multi-Input Wake Up
  - — Software trap
  - — Default VIS (default interrupt)
- Versatile and easy to use instruction set
- 8-bit Stack Pointer SP—stack in RAM
- Two 8-bit register indirect data memory pointers (B and X)

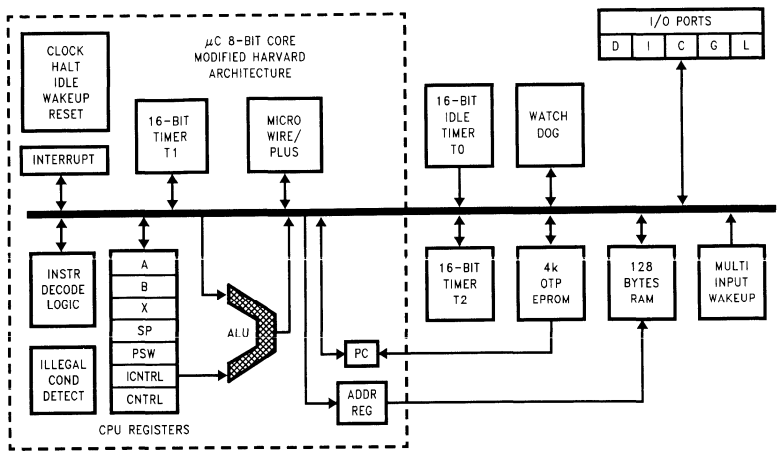
Fully Static CMOS

- Two power saving modes: HALT and IDLE
- Single supply operation: 2.7V–5.5V
- Temperature range: −40°C to +85°C

Development Support

- Emulation device for the COP888CL/COP884CL
- Real time emulation and full program debug offered by MetaLink Development System

Block Diagram



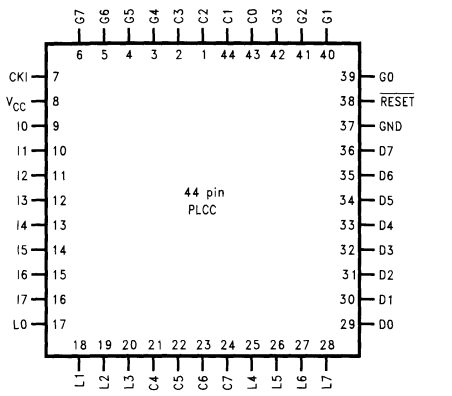
TL/DD/12524–16

General Description (Continued)

The device is a fully static part, fabricated using double-metal silicon gate microCMOS technology. Features include an 8-bit memory mapped architecture, MICROWIRE/PLUS™ serial I/O, two 16-bit timer/counters supporting three modes (Processor Independent PWM generation, External Event counter, and Input Capture mode capabilities). Each I/O pin has software selectable configurations. The devices operates over a voltage range of 2.7V to 5.5V. High throughput is achieved with an efficient, regular instruction set operating at a maximum of 1 μs per instruction rate.

Connection Diagrams

Plastic Chip Carrier



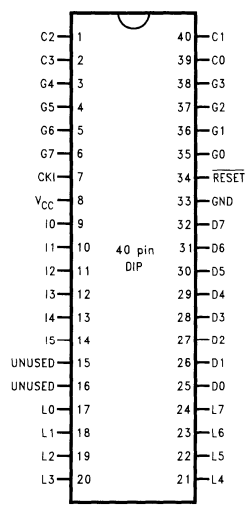
TL/DD/12524–1

Top View

Order Number COP87L88CLV-XE
See NS Package Number V44A

Note: -X Crystal Oscillator
-E Halt Enable

Dual-In-Line Package

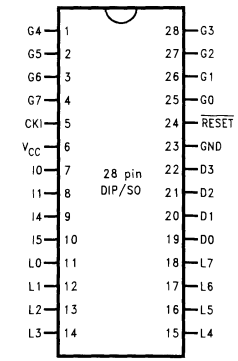


TL/DD/12524–2

Top View

Order Number COP87L84CLN-XE
See NS Package Number N40A

Dual-In-Line Package



TL/DD/12524–3

Top View

Order Number COP87L84CLN-XE
or COP87L84CLM-XE
See NS Package Number M28B or N28B

FIGURE 1. COP87L88CL/COP87L84CL Connection Diagrams

Connection Diagrams (Continued)

Pinouts for 28-, 40- and 44-Pin Packages

Port	Type	Alt. Fun	Alt. Fun	28-Pin Pkg.	40-Pin Pkg.	44-Pin Pkg.
L0	I/O	MIWU		11	17	17
L1	I/O	MIWU		12	18	18
L2	I/O	MIWU		13	19	19
L3	I/O	MIWU		14	20	20
L4	I/O	MIWU	T2A	15	21	25
L5	I/O	MIWU	T2B	16	22	26
L6	I/O	MIWU		17	23	27
L7	I/O	MIWU		18	24	28
G0	I/O	INT		25	35	39
G1	WDOUT			26	36	40
G2	I/O	T1B		27	37	41
G3	I/O	T1A		28	38	42
G4	I/O	SO		1	3	3
G5	I/O	SK		2	4	4
G6	I	SI		3	5	5
G7	I/CKO	Halt Restart		4	6	6
D0	O			19	25	29
D1	O			20	26	30
D2	O			21	27	31
D3	O			22	28	32
I0	I			7	9	9
I1	I			8	10	10
I2	I				11	11
I3	I				12	12
I4	I			9	13	13
I5	I			10	14	14
I6	I					15
I7	I					16
D4	O				29	33
D5	O				30	34
D6	O				31	35
D7	O				32	36
C0	I/O				39	43
C1	I/O				40	44
C2	I/O				1	1
C3	I/O				2	2
C4	I/O					21
C5	I/O					22
C6	I/O					23
C7	I/O					24
Unused*					16	
Unused*					15	
V_{CC}				6	8	8
GND				23	33	37
CKI				5	7	7
$\overline{\text{RESET}}$				24	34	38

* = On the 40-pin package, Pins 15 and 16 must be connected to GND.

Absolute Maximum Ratings (Note)

If Military/Aerospace specified devices are required, please contact the National Semiconductor Sales Office/Distributors for availability and specifications.

Supply Voltage (V_{CC})	7V
Voltage at Any Pin	−0.3V to V_{CC} + 0.3V
Total Current into V_{CC} Pin (Source)	100 mA
Total Current out of GND Pin (Sink)	110 mA
Storage Temperature Range	−65°C to +140°C

Note: *Absolute maximum ratings indicate limits beyond which damage to the device may occur. DC and AC electrical specifications are not ensured when operating the device at absolute maximum ratings.*

DC Electrical Characteristics −40°C ≤ T_A ≤ +85°C unless otherwise specified

Parameter	Conditions	Min	Typ	Max	Units
Operating Voltage		2.7		5.5	V
Power Supply Ripple (Note 1)	Peak-to-Peak			0.1 V_{CC}	V
Supply Current (Note 2)					
CKI = 10 MHz	V_{CC} = 5.5V, t_c = 1 μs			16.5	mA
CKI = 4 MHz	V_{CC} = 4.0V, t_c = 2.5 μs			6.5	mA
HALT Current (Note 3)	V_{CC} = 5.5V, CKI = 0 MHz			12	μA
IDLE Current, CKI = 10 MHz	V_{CC} = 5.5V, t_c = 1 μs			3.5	mA
CKI = 1 MHz	V_{CC} = 4.0V, t_c = 10 μs			0.7	mA
Input Levels					
RESET					
Logic High		0.8 V_{CC}			V
Logic Low				0.2 V_{CC}	
CKI (External and Crystal Osc. Modes)					
Logic High		0.7 V_{CC}			
Logic Low				0.2 V_{CC}	
All Other Inputs					
Logic High		0.7 V_{CC}			
Logic Low				0.2 V_{CC}	
Hi-Z Input Leakage	V_{CC} = 5.5V	−2		+2	μA
Input Pullup Current	V_{CC} = 5.5V	40		250	μA
G and L Port Input Hysteresis			0.05 V_{CC}	0.35 V_{CC}	V
Output Current Levels					
D Outputs					
Source	V_{CC} = 4.5V, V_{OH} = 3.3V	0.4			mA
Sink (Note 4)	V_{CC} = 4.5V, V_{OL} = 1V	10			mA
All Others					
Source (Weak Pull-Up Mode)	V_{CC} = 4.5V, V_{OH} = 2.7V	10		100	μA
Source (Push-Pull Mode)	V_{CC} = 4.5V, V_{OH} = 3.3V	0.4			mA
Sink (Push-Pull Mode)	V_{CC} = 4.5V, V_{OL} = 0.4V	1.6			mA
TRI-STATE Leakage	V_{CC} = 5.5V	−2		+2	μA
Allowable Sink/Source Current per Pin					mA
D Outputs (Sink)				15	
All others				3	
Maximum Input Current without Latchup (Note 5)	T_A = 25°C			±100	mA
RAM Retention Voltage, V_r	500 ns Rise and Fall Time (Min)	2			V
Input Capacitance				7	pF
Load Capacitance on D2				1000	pF

Note 1: Rate of voltage change must be less then 0.5V/ms.

Note 2: Supply current is measured after running 2000 cycles with a square wave CKI input, CKO open, inputs at rails and outputs open.

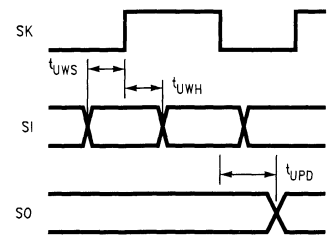
Note 3: The HALT mode will stop CKI from oscillating in the RC and the Crystal configurations by bringing CKI high. Test conditions: All inputs tied to V_{CC}, L and G ports in the TRI-STATE mode and tied to ground, all outputs low and tied to ground. The clock monitor is disabled.

Note 4: The user must guarantee that D2 pin does not source more than 10 mA during RESET. If D2 sources more than 10 mA during RESET, the device will go into programming mode.

Note 5: Pins G5 and RESET are designed with a high voltage input network for factory testing. These pins allow input voltages greater than V_{CC} and the pins will have sink current to V_{CC} when biased at voltages greater than V_{CC} (the pins do not have source current when biased at a voltage below V_{CC}). The effective resistance to V_{CC} is 750Ω (typical). These two pins will not latch up. The voltage at the pins must be limited to less than 14V.

AC Electrical Characteristics $-40°C \leq T_A \leq +85°C$ unless otherwise specified

Parameter	Conditions	Min	Typ	Max	Units
Instruction Cycle Time (t_c)					
Crystal or Resonator		1		DC	μs
R/C Oscillator		3		DC	
Inputs					
t_{SETUP}		200			ns
t_{HOLD}		60			
Output Propagation Delay	$R_L = 2.2k, C_L = 100$ pF				
t_{PD1}, t_{PD0}					μs
SO, SK	$4V \leq V_{CC} \leq 6V$			0.7	
All Others	$4V \leq V_{CC} \leq 6V$			1	
MICROWIRE™ Setup Time (t_{UWS})		20			
MICROWIRE Hold Time (t_{UWH})		56			ns
MICROWIRE Output Propagation Delay (t_{UPD})				220	
Input Pulse Width					
Interrupt Input High Time		1			
Interrupt Input Low Time		1			t_c
Timer Input High Time		1			
Timer Input Low Time		1			
Reset Pulse Width		1			μs



TL/DD/12524–4

FIGURE 2. MICROWIRE/PLUS Timing

Pin Descriptions

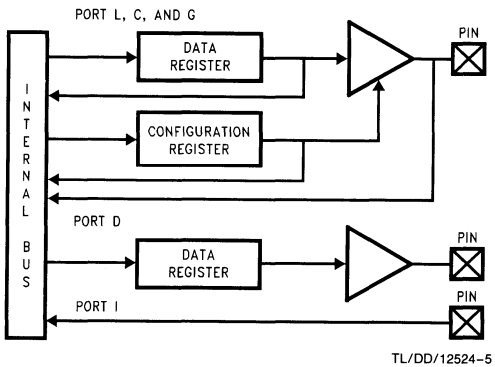
V_{CC} and GND are the power supply pins.

CKI is the clock input. This can come from an R/C generated oscillator, or a crystal oscillator (in conjunction with CKO). See Oscillator Description section.

$\overline{\text{RESET}}$ is the master reset input. See Reset Description section.

The device contains three bidirectional 8-bit I/O ports (C, G and L), where each individual bit may be independently configured as an input (Schmitt trigger inputs on ports G and L), output or TRI-STATE under program control. Three data memory address locations are allocated for each of these I/O ports. Each I/O port has two associated 8-bit memory mapped registers, the CONFIGURATION register and the output DATA register. A memory mapped address is also reserved for the input pins of each I/O port. (See the memory map for the various addresses associated with the I/O ports.) *Figure 3* shows the I/O port configurations. The DATA and CONFIGURATION registers allow for each port bit to be individually configured under software control as shown below:

CONFIGURATION Register	DATA Register	Port Set-Up
0	0	Hi-Z Input (TRI-STATE Output)
0	1	Input with Weak Pull-Up
1	0	Push-Pull Zero Output
1	1	Push-Pull One Output



TL/DD/12524–5

FIGURE 3. I/O Port Configurations

PORT L is an 8-bit I/O port. All L-pins have Schmitt triggers on the inputs.

Port L supports Multi-Input Wakeup (MIWU) on all eight pins. L4 and L5 are used for the timer input functions T2A and T2B.

Port L has the following alternate features:

- L0 MIWU
- L1 MIWU
- L2 MIWU
- L3 MIWU
- L4 MIWU or T2A
- L5 MIWU or T2B
- L6 MIWU
- L7 MIWU

Port G is an 8-bit port with 5 I/O pins (G0, G2–G5), an input pin (G6), and two dedicated output pins (G1 and G7). Pins G0 and G2–G6 all have Schmitt Triggers on their inputs. Pin G1 serves as the dedicated WDOUT WATCHDOG output, while pin G7 is either input or output depending on the oscillator mask option selected. With the crystal oscillator option selected, G7 serves as the dedicated output pin for the CKO clock output. With the single-pin R/C oscillator mask option selected, G7 serves as a general purpose input pin, but is also used to bring the device out of HALT mode with a low to high transition. There are two registers associated with the G Port, a data register and a configuration register. Therefore, each of the 5 I/O bits (G0, G2–G5) can be individually configured under software control.

Since G6 is an input only pin and G7 is the dedicated CKO clock output pin or general purpose input (R/C clock configuration), the associated bits in the data and configuration registers for G6 and G7 are used for special purpose functions as outlined below. Reading the G6 and G7 data bits will return zeros.

Note that the chip will be placed in the HALT mode by writing a "1" to bit 7 of the Port G Data Register. Similarly the chip will be placed in the IDLE mode by writing a "1" to bit 6 of the Port G Data Register.

Writing a "1" to bit 6 of the Port G Configuration Register enables the MICROWIRE/PLUS to operate with the alternate phase of the SK clock. The G7 configuration bit, if set high, enables the clock start up delay after HALT when the R/C clock configuration is used.

	Config Reg.	Data Reg.
G7	CLKDLY	HALT
G6	Alternate SK	IDLE

Port G has the following alternate features:

- G0 INTR (External Interrupt Input)
- G2 T1B (Timer T1 Capture Input)
- G3 T1A (Timer T1 I/O)
- G4 SO (MICROWIRE Serial Data Output)
- G5 SK (MICROWIRE Serial Clock)
- G6 SI (MICROWIRE Serial Data Input)

Pin Descriptions (Continued)

Port G has the following dedicated functions:

G1 WDOUT WATCHDOG and/or Clock Monitor dedicated output

G7 CKO Oscillator dedicated output or general purpose input

Port C is an 8-bit I/O port. The 28-pin device does not have a full complement of Port C pins. The unavailable pins are not terminated. A read operation for these unterminated pins will return unpredictable values.

Port I is an 8-bit Hi-Z input port. The 28-pin device does not have a full complement of Port I pins. The unavailable pins are not terminated (i.e. they are floating). A read operation from these unterminated pins will return unpredictable values. The user should ensure that the software takes this into account by either masking out these inputs, or else restricting the accesses to bit operations only. If unterminated, Port I pins will draw power only when addressed. The I port leakage current may be higher in 28-pin devices.

Port D is a recreated 8-bit output port that is preset high when RESET goes low. D port recreation is one clock cycle behind the normal port timing. The user can tie two or more D port outputs (except D2 pin) together in order to get a higher drive.

Functional Description

The architecture of the device is modified Harvard architecture. With the Harvard architecture, the control store program memory (ROM) is separated from the data store memory (RAM). Both ROM and RAM have their own separate addressing space with separate address buses. The architecture, though based on Harvard architecture, permits transfer of data from ROM to RAM.

CPU REGISTERS

The CPU can do an 8-bit addition, subtraction, logical or shift operation in one instruction (t_C) cycle time.

There are five CPU registers:

A is the 8-bit Accumulator Register

PC is the 15-bit Program Counter Register

PU is the upper 7 bits of the program counter (PC)
PL is the lower 8 bits of the program counter (PC)

B is an 8-bit RAM address pointer, which can be optionally post auto incremented or decremented.

X is an 8-bit alternate RAM address pointer, which can be optionally post auto incremented or decremented.

SP is the 8-bit stack pointer, which points to the subroutine/interrupt stack (in RAM). The SP is initialized to RAM address 06F with reset.

All the CPU registers are memory mapped with the exception of the Accumulator (A) and the Program Counter (PC).

PROGRAM MEMORY

Program memory consists of 4 kbytes of OTP EPROM. These bytes may hold program instructions or constant data (data tables for the LAID instruction, jump vectors for the JID instruction, and interrupt vectors for the VIS instruction). The program memory is addressed by the 15-bit program counter (PC). All interrupts vector to program memory location 0FF Hex.

The device can be configured to inhibit external reads of the program memory. This is done by programming the Security Byte.

SECURITY FEATURE

The program memory array has an associate Security Byte that is located outside of the program address range. This byte can be addressed only from programming mode by a programmer tool.

Security is an optional feature and can only be asserted after the memory array has been programmed and verified. A secured part will read all 00(hex) by a programmer. The part will fail Blank Check and will fail Verify operations. A Read operation will fill the programmer's memory with 00(hex). The Security Byte itself is always readable with a value of 00(hex) if unsecure and FF(hex) if secure.

DATA MEMORY

The data memory address space includes the on-chip RAM and data registers, the I/O registers (Configuration, Data and Pin), the control registers, the MICROWIRE/PLUS SIO shift register, and the various registers, and counters associated with the timers (with the exception of the IDLE timer). Data memory is addressed directly by the instruction or indirectly by the B, X and SP pointers.

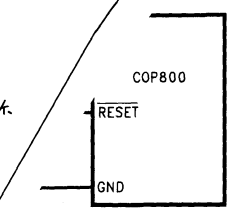
The device has 128 bytes of RAM. Sixteen bytes of RAM are mapped as "registers" at addresses 0F0 to 0FF Hex. These registers can be loaded immediately, and also decremented and tested with the DRSZ (decrement register and skip if zero) instruction. The memory pointer registers X, SP, and B are memory mapped into this space at address locations 0FC to 0FE Hex respectively, with the other registers (other than reserved register 0FF) being available for general usage.

The instruction set permits any bit in memory to be set, reset or tested. All I/O and registers on the device (except A and PC) are memory mapped; therefore, I/O bits and register bits can be directly and individually set, reset and tested. The accumulator (A) bits can also be directly and individually tested.

Reset

The $\overline{\text{RESET}}$ input when pulled low initializes the microcontroller. Initialization will occur whenever the $\overline{\text{RESET}}$ input is pulled low. Upon initialization, the data and configuration registers for Ports L, G, and C are cleared, resulting in these Ports being initialized to the TRI-STATE mode. Pin G1 of the G Port is an exception (as noted below) since pin G1 is dedicated as the WATCHDOG and/or Clock Monitor error output pin. Port D is initialized high with $\overline{\text{RESET}}$. The PC, PSW, CNTRL, ICNTRL, and T2CNTRL control registers are cleared. The Multi-Input Wakeup registers WKEN, WKEDG, and WKPND are cleared. The Stack Pointer, SP, is initialized to 06F Hex.

The device comes out of reset with both the WATCHDOG logic and the Clock Monitor detector armed, and with both the WATCHDOG service window bits set and the Clock Monitor bit set. The WATCHDOG and Clock Monitor detector circuits are inhibited during reset. The WATCHDOG service window bits are initialized to the maximum WATCHDOG service window of 64k t_C clock cycles. The Clock Monitor bit is initialized high, and will cause a Clock Monitor error following reset if the clock has not reached the minimum specified frequency at the termination of reset. A Clock Monitor



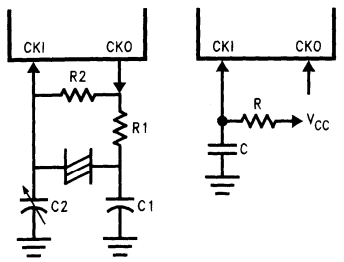
TL/DD/12524–6

ecommended Reset Circuit

ircuits

driven by a clock input on the CKI input pin between DC and 10 MHz. The CKO output in G7 (crystal configuration). The CKI input frequency divided down by 10 to produce the instruction ck ($1/t_C$).

5 shows the Crystal and R/C diagrams.



TL/DD/12524–7

FIGURE 5. Crystal and R/C Oscillator Diagrams

CRYSTAL OSCILLATOR

CKI and CKO can be connected to make a closed loop crystal (or resonator) controlled oscillator.

Table I shows the component values required for various standard crystal values.

TABLE I. Crystal Oscillator Configuration, T_A = 25°C

R1 (kΩ)	R2 (MΩ)	C1 (pF)	C2 (pF)	CKI Freq (MHz)	Conditions
0	1	30	30–36	10	V_{CC} = 5V
0	1	30	30–36	4	V_{CC} = 5V
0	1	200	100–150	0.455	V_{CC} = 5V

R/C OSCILLATOR

By selecting CKI as a single pin oscillator input, a single pin R/C oscillator circuit can be connected to it. CKO is available as a general purpose input, and/or HALT restart pin.

Table II shows the variation in the oscillator frequencies as functions of the component (R and C) values.

TABLE II. R/C Oscillator Configuration, T_A = 25°C

R (kΩ)	C (pF)	CKI Freq (MHz)	Instr. Cycle (μs)	Conditions
3.3	82	2.2–2.7	3.7–4.6	V_{CC} = 5V
5.6	100	1.1–1.3	7.4–9.0	V_{CC} = 5V
6.8	100	0.9–1.1	8.8–10.8	V_{CC} = 5V

Note: 3k ≤ R ≤ 200k, 50 pF ≤ C ≤ 200 pF

Control Registers

CNTRL Register (Address X'00EE)

The Timer1 (T1) and MICROWIRE/PLUS control register contains the following bits:

SL1 & SL0 Select the MICROWIRE/PLUS clock divide by (00 = 2, 01 = 4, 1x = 8)

IEDG External interrupt edge polarity select (0 = Rising edge, 1 = Falling edge)

MSEL Selects G5 and G4 as MICROWIRE/PLUS signals
SK and SO respectively

2

Control Registers (Continued)

T1C0 Timer T1 Start/Stop control in timer
Timer T1 Underflow Interrupt Pending Flag in timer mode 3

T1C1 Timer T1 mode control bit

T1C2 Timer T1 mode control bit

T1C3 Timer T1 mode control bit

T1C3	T1C2	T1C1	T1C0	MSEL	IEDG	SL1	SL0

Bit 7 Bit 0

PSW Register (Address X'00EF)

The PSW register contains the following select bits:

GIE Global interrupt enable (enables interrupts)

EXEN Enable external interrupt

BUSY MICROWIRE/PLUS busy shifting flag

EXPND External interrupt pending

T1ENA Timer T1 Interrupt Enable for Timer Underflow or T1A Input capture edge

T1PNDA Timer T1 Interrupt Pending Flag (Autoreload RA in mode 1, T1 Underflow in Mode 2, T1A capture edge in mode 3)

C Carry Flag

HC Half Carry Flag

HC	C	T1PNDA	T1ENA	EXPND	BUSY	EXEN	GIE

Bit 7 Bit 0

The Half-Carry bit is also affected by all the instructions that affect the Carry flag. The SC (Set Carry) and RC (Reset Carry) instructions will respectively set or clear both the carry flags. In addition to the SC and RC instructions, ADC, SUBC, RRC and RLC instructions affect the carry and Half Carry flags.

ICNTRL Register (Address X'00E8)

The ICNTRL register contains the follc

T1ENB Timer T1 Interrupt Enable f(
edge

T1PNDB Timer T1 Interrupt Pending
ture edge

WEN Enable MICROWIRE/PLUS in(

WPND MICROWIRE/PLUS interrupt p(

T0EN Timer T0 Interrupt Enable (Bit 1

T0PND Timer T0 Interrupt pending

LPENL Port Interrupt Enable (Multi-
eup/Interrupt)
Bit 7 could be used as a flag

T2CNTRL Register (Address X'00C6)

Unused	LPEN	T0PND	T0EN	WPND	WEN	T1PNDE

Bit 7

The T2CNTRL register contains the following bits:

T2ENB Timer T2 Interrupt Enable for T2B Input ca
edge

T2PNDB Timer T2 Interrupt Pending Flag for T2B
ture edge

T2ENA Timer T2 Interrupt Enable for Timer Underfl
or T2A Input capture edge

T2PNDA Timer T2 Interrupt Pending Flag (Autoreload R/
in mode 1, T2 Underflow in mode 2, T2A capture edge in mode 3)

T2C0 Timer T2 Start/Stop control in timer modes 1 and 2 Timer T2 Underflow Interrupt Pending Flag in timer mode 3

T2C1 Timer T2 mode control bit

T2C2 Timer T2 mode control bit

T2C3 Timer T2 mode control bit

T2C3	T2C2	T2C1	T2C0	T2PNDA	T2ENA	T2PNDB	T2ENB

Bit 7 Bit 0

Timers

The device contains a very versatile set of timers (T0, T1, T2). All timers and associated autoreload/capture registers power up containing random data.

TIMER T0 (IDLE TIMER)

The device supports applications that require maintaining real time and low power with the IDLE mode. This IDLE mode support is furnished by the IDLE timer T0, which is a 16-bit timer. The Timer T0 runs continuously at the fixed rate of the instruction cycle clock, t_C. The user cannot read or write to the IDLE Timer T0, which is a count down timer.

The Timer T0 supports the following functions:

Exit out of the Idle Mode (See Idle Mode description)
WATCHDOG logic (See WATCHDOG description)
Start up delay out of the HALT mode

The IDLE Timer T0 can generate an interrupt when the thirteenth bit toggles. This toggle is latched into the T0PND pending flag, and will occur every 4 ms at the maximum clock frequency (t_C = 1 μs). A control flag T0EN allows the interrupt from the thirteenth bit of Timer T0 to be enabled or disabled. Setting T0EN will enable the interrupt, while resetting it will disable the interrupt.

TIMER T1 AND TIMER T2

The device has a set of two powerful timer/counter blocks, T1 and T2. The associated features and functioning of a timer block are described by referring to the timer block Tx. Since the two timer blocks, T1 and T2, are identical, all comments are equally applicable to either timer block.

Each timer block consists of a 16-bit timer, Tx, and two supporting 16-bit autoreload/capture registers, RxA and RxB. Each timer block has two pins associated with it, TxA and TxB. The pin TxA supports I/O required by the timer block, while the pin TxB is an input to the timer block. The powerful and flexible timer block allows the device to easily perform all timer functions with minimal software overhead. The timer block has three operating modes: Processor Independent PWM mode, External Event Counter mode, and Input Capture mode.

The control bits TxC3, TxC2, and TxC1 allow selection of the different modes of operation.

Mode 1. Processor Independent PWM Mode

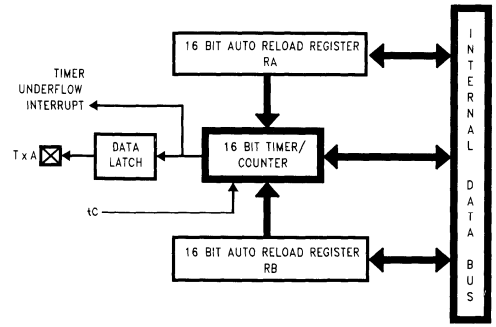
As the name suggests, this mode allows the device to generate a PWM signal with very minimal user intervention.

The user only has to define the parameters of the PWM signal (ON time and OFF time). Once begun, the timer block will continuously generate the PWM signal completely independent of the microcontroller. The user software services the timer block only when the PWM parameters require updating.

In this mode the timer Tx counts down at a fixed rate of t_C. Upon every underflow the timer is alternately reloaded with the contents of supporting registers, RxA and RxB. The very first underflow of the timer causes the timer to reload from the register RxA. Subsequent underflows cause the timer to be reloaded from the registers alternately beginning with the register RxB.

The Tx Timer control bits, TxC3, TxC2 and TxC1 set up the timer for PWM mode operation.

Figure 6 shows a block diagram of the timer in PWM mode.



TL/DD/12524–9

FIGURE 6. Timer in PWM Mode

The underflows can be programmed to toggle the TxA output pin. The underflows can also be programmed to generate interrupts.

Underflows from the timer are alternately latched into two pending flags, TxPNDA and TxPNDB. The user must reset these pending flags under software control. Two control enable flags, TxENA and TxENB, allow the interrupts from the timer underflow to be enabled or disabled. Setting the timer enable flag TxENA will cause an interrupt when a timer underflow causes the RxA register to be reloaded into the timer. Setting the timer enable flag TxENB will cause an interrupt when a timer underflow causes the RxB register to be reloaded into the timer. Resetting the timer enable flags will disable the associated interrupts.

Either or both of the timer underflow interrupts may be enabled. This gives the user the flexibility of interrupting once per PWM period on either the rising or falling edge of the PWM output. Alternatively, the user may choose to interrupt on both edges of the PWM output.

Mode 2. External Event Counter Mode

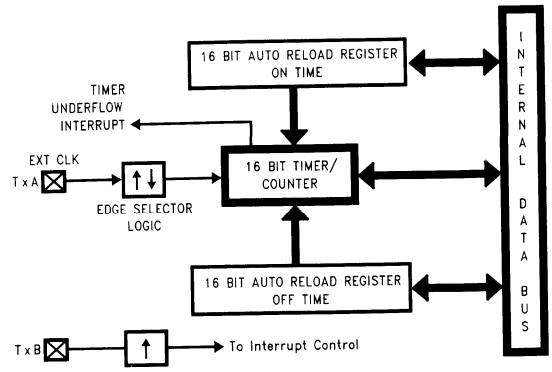
This mode is quite similar to the processor independent PWM mode described above. The main difference is that the timer, Tx, is clocked by the input signal from the TxA pin. The Tx timer control bits, TxC3, TxC2 and TxC1 allow the timer to be clocked either on a positive or negative edge from the TxA pin. Underflows from the timer are latched into the TxPNDA pending flag. Setting the TxENA control flag will cause an interrupt when the timer underflows.

In this mode the input pin TxB can be used as an independent positive edge sensitive interrupt input if the TxENB control flag is set. The occurrence of a positive edge on the TxB input pin is latched into the TxPNDB flag.

Figure 7 shows a block diagram of the timer in External Event Counter mode.

Note: The PWM output is not available in this mode since the TxA pin is being used as the counter input clock.

Timers (Continued)



TL/DD/12524–10

FIGURE 7. Timer in External Event Counter Mode

Mode 3. Input Capture Mode

The device can precisely measure external frequencies or time external events by placing the timer block, Tx, in the input capture mode.

In this mode, the timer Tx is constantly running at the fixed t_C rate. The two registers, RxA and RxB, act as capture registers. Each register acts in conjunction with a pin. The register RxA acts in conjunction with the TxA pin and the register RxB acts in conjunction with the TxB pin.

The timer value gets copied over into the register when a trigger event occurs on its corresponding pin. Control bits, TxC3, TxC2 and TxC1, allow the trigger events to be specified either as a positive or a negative edge. The trigger condition for each input pin can be specified independently.

The trigger conditions can also be programmed to generate interrupts. The occurrence of the specified trigger condition on the TxA and TxB pins will be respectively latched into the pending flags, TxPNDA and TxPNDB. The control flag TxENA allows the interrupt on TxA to be either enabled or disabled. Setting the TxENA flag enables interrupts to be generated when the selected trigger condition occurs on the TxA pin. Similarly, the flag TxENB controls the interrupts from the TxB pin.

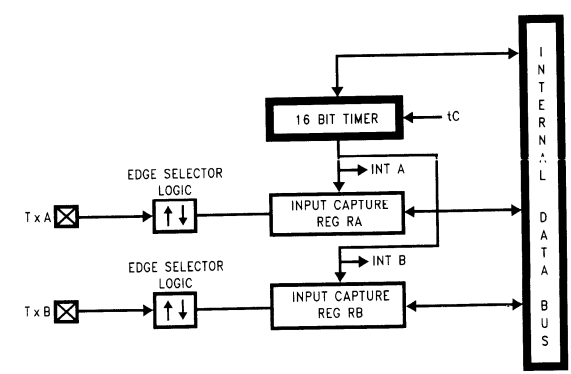
Underflows from the timer can also be programmed to generate interrupts. Underflows are latched into the timer TxC0 pending flag (the TxC0 control bit serves as the timer underflow interrupt pending flag in the Input Capture mode). Consequently, the TxC0 control bit should be reset when entering the Input Capture mode. The timer underflow interrupt is enabled with the TxENA control flag. When a TxA interrupt occurs in the Input Capture mode, the user must check both whether a TxA input capture or a timer underflow (or both) caused the interrupt.

Figure 8 shows a block diagram of the timer in Input Capture mode.

TIMER CONTROL FLAGS

The timers T1 and T2 have indentical control structures. The control bits and their functions are summarized below.

TxC0	Timer Start/Stop control in Modes 1 and 2 (Processor Independent PWM and External Event Counter), where 1 = Start, 0 = Stop Timer Underflow Interrupt Pending Flag in Mode 3 (Input Capture)
TxPNDA	Timer Interrupt Pending Flag
TxPNDB	Timer Interrupt Pending Flag
TxENA	Timer Interrupt Enable Flag
TxENB	Timer Interrupt Enable Flag 1 = Timer Interrupt Enabled 0 = Timer Interrupt Disabled
TxC3	Timer mode control
TxC2	Timer mode control
TxC1	Timer mode control



TL/DD/12524–11

FIGURE 8. Timer in Input Capture Mode

Timers (Continued)

The timer mode control bits (TxC3, TxC2 and TxC1) are detailed below:

TxC3	TxC2	TxC1	Timer Mode	Interrupt A Source	Interrupt B Source	Timer Counts On
0	0	0	MODE 2 (External Event Counter)	Timer Underflow	Pos. TxB Edge	TxA Pos. Edge
0	0	1	MODE 2 (External Event Counter)	Timer Underflow	Pos. TxB Edge	TxA Neg. Edge
1	0	1	MODE 1 (PWM) TxA Toggle	Autoreload RA	Autoreload RB	t_C
1	0	0	MODE 1 (PWM) No TxA Toggle	Autoreload RA	Autoreload RB	t_C
0	1	0	MODE 3 (Capture) Captures: TxA Pos. Edge TxB Pos. Edge	Pos. TxA Edge or Timer Underflow	Pos. TxB Edge	t_C
1	1	0	MODE 3 (Capture) Captures: TxA Pos. Edge TxB Neg. Edge	Pos. TxA Edge or Timer Underflow	Neg. TxB Edge	t_C
0	1	1	MODE 3 (Capture) Captures: TxA Neg. Edge TxB Pos. Edge	Neg. TxA Edge or Timer Underflow	Pos. TxB Edge	t_C
1	1	1	MODE 3 (Capture) Captures: TxA Neg. Edge TxB Neg. Edge	Neg. TxA Edge or Timer Underflow	Neg. TxB Edge	t_C

Power Save Modes

The device offers the user two power save modes of operation: HALT and IDLE. In the HALT mode, all microcontroller activities are stopped. In the IDLE mode, the on-board oscillator circuitry and timer T0 are active but all other microcontroller activities are stopped. In either mode, all on-board RAM, registers, I/O states, and timers (with the exception of T0) are unaltered.

HALT MODE

The device is placed in the HALT mode by writing a "1" to the HALT flag (G7 data bit). All microcontroller activities, including the clock, timers, are stopped. The WATCHDOG logic is disabled during the HALT mode. However, the clock monitor circuitry, if enabled, remains active and will cause the WATCHDOG output pin (WDOUT) to go low. If the HALT mode is used and the user does not want to activate the WDOUT pin, the Clock Monitor should be disabled after the device comes out of reset (resetting the Clock Monitor control bit with the first write to the WDSVR register). In the HALT mode, the power requirements are minimal and the applied voltage (V_{CC}) may be decreased to Vr ($Vr = 2.0V$) without altering the state of the machine.

The device supports three different ways of exiting the HALT mode. The first method of exiting the HALT mode is with the Multi-Input Wakeup feature on the L port. The second method is with a low to high transition on the CKO (G7) pin. This method precludes the use of the crystal clock configuration (since CKO becomes a dedicated output), and so may be used with an RC clock configuration. The third method of exiting the HALT mode is by pulling the $\overline{\text{RESET}}$ pin low.

Since a crystal or ceramic resonator may be selected as the oscillator, the Wakeup signal is not allowed to start the chip running immediately since crystal oscillators and ceramic resonators have a delayed start up time to reach full amplitude and frequency stability. The IDLE timer is used to generate a fixed delay to ensure that the oscillator has indeed stabilized before allowing instruction execution. In this case, upon detecting a valid Wakeup signal, only the oscillator circuitry is enabled. The IDLE timer is loaded with a value of 256 and is clocked with the t_C instruction cycle clock. The t_C clock is derived by dividing the oscillator clock down by a factor of 10. The Schmitt trigger following the CKI inverter on the chip ensures that the IDLE timer is clocked only when the oscillator has a sufficiently large amplitude to meet the Schmitt trigger specifications. This Schmitt trigger is not part of the oscillator closed loop. The startup timeout from the IDLE timer enables the clock signals to be routed to the rest of the chip.

Power Save Modes (Continued)

If an RC clock option is being used, the fixed delay is introduced optionally. A control bit, CLKDLY, mapped as configuration bit G7, controls whether the delay is to be introduced or not. The delay is included if CLKDLY is set, and excluded if CLKDLY is reset. The CLKDLY bit is cleared on reset.

The WATCHDOG detector circuit is inhibited during the HALT mode. However, the clock monitor circuit, if enabled, remains active during HALT mode in order to ensure a clock monitor error if the device inadvertently enters the HALT mode as a result of a runaway program or power glitch.

IDLE MODE

The device is placed in the IDLE mode by writing a "1" to the IDLE flag (G6 data bit). In this mode, all activity, except the associated on-board oscillator circuitry, the WATCHDOG logic, the clock monitor and the IDLE Timer T0, is stopped.

As with the HALT mode, the device can be returned to normal operation with a reset, or with a Multi-Input Wake-up from the L Port. Alternately, the microcontroller resumes normal operation from the IDLE mode when the thirteenth bit (representing 4.096 ms at internal clock frequency of 1 MHz, t_C = 1 μs) of the IDLE Timer toggles.

This toggle condition of the thirteenth bit of the IDLE Timer T0 is latched into the T0PND pending flag.

The user has the option of being interrupted with a transition on the thirteenth bit of the IDLE Timer T0. The interrupt can be enabled or disabled via the T0EN control bit. Setting the T0EN flag enables the interrupt and vice versa.

The user can enter the IDLE mode with the Timer T0 interrupt enabled. In this case, when the T0PND bit gets set, the device will first execute the Timer T0 interrupt service routine and then return to the instruction following the "Enter Idle Mode" instruction.

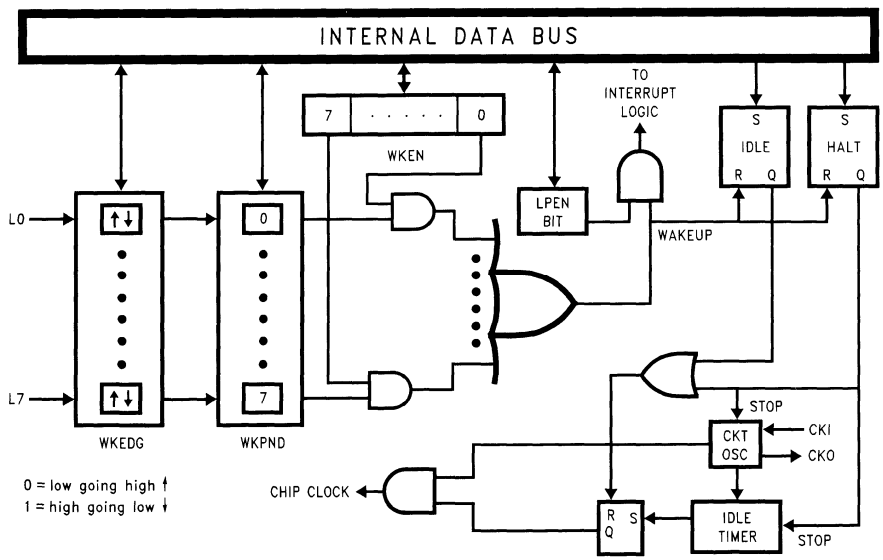
Alternatively, the user can enter the IDLE mode with the IDLE Timer T0 interrupt disabled. In this case, the device will resume normal operation with the instruction immediately following the "Enter IDLE Mode" instruction.

Note: It is necessary to program two NOP instructions following both the set HALT mode and set IDLE mode instructions. These NOP instructions are necessary to allow clock resynchronization following the HALT or IDLE modes. Due to the on-board 8k EPROM with port recreation logic, the HALT/IDLE current is much higher compared to the equivalent masked device (COP888CL/COP884CL).

Multi-Input Wakeup

The Multi-Input Wakeup feature is used to return (wakeup) the device from either the HALT or IDLE modes. Alternately Multi-Input Wakeup/Interrupt feature may also be used to generate up to 8 edge selectable external interrupts.

Figure 9 shows the Multi-Input Wakeup logic.



TL/DD/12524–12

FIGURE 9. Multi-Input Wake Up Logic

Multi-Input Wakeup (Continued)

The Multi-Input Wakeup feature utilizes the L Port. The user selects which particular L port bit (or combination of L Port bits) will cause the device to exit the HALT or IDLE modes. The selection is done through the Reg: WKEN. The Reg: WKEN is an 8-bit read/write register, which contains a control bit for every L port bit. Setting a particular WKEN bit enables a Wakeup from the associated L port pin.

The user can select whether the trigger condition on the selected L Port pin is going to be either a positive edge (low to high transition) or a negative edge (high to low transition). This selection is made via the Reg: WKEDG, which is an 8-bit control register with a bit assigned to each L Port pin. Setting the control bit will select the trigger condition to be a negative edge on that particular L Port pin. Resetting the bit selects the trigger condition to be a positive edge. Changing an edge select entails several steps in order to avoid a pseudo Wakeup condition as a result of the edge change. First, the associated WKEN bit should be reset, followed by the edge select change in WKEDG. Next, the associated WKPND bit should be cleared, followed by the associated WKEN bit being re-enabled.

An example may serve to clarify this procedure. Suppose we wish to change the edge select from positive (low going high) to negative (high going low) for L Port bit 5, where bit 5 has previously been enabled for an input interrupt. The program would be as follows:

```
RBIT  5, WKEN
SBIT  5, WKEDG
RBIT  5, WKPND
SBIT  5, WKEN
```

If the L port bits have been used as outputs and then changed to inputs with Multi-Input Wakeup/Interrupt, a safety procedure should also be followed to avoid inherited pseudo wakeup conditions. After the selected L port bits have been changed from output to input but before the associated WKEN bits are enabled, the associated edge select bits in WKEDG should be set or reset for the desired edge selects, followed by the associated WKPND bits being cleared.

This same procedure should be used following reset, since the L port inputs are left floating as a result of reset.

The occurrence of the selected trigger condition for Multi-Input Wakeup is latched into a pending register called WKPND. The respective bits of the WKPND register will be set on the occurrence of the selected trigger edge on the corresponding Port L pin. The user has the responsibility of clearing these pending flags. Since WKPND is a pending register for the occurrence of selected wakeup conditions, the device will not enter the HALT mode if any Wakeup bit is both enabled and pending. Consequently, the user has the responsibility of clearing the pending flags before attempting to enter the HALT mode.

The WKEN, WKPND and WKEDG are all read/write registers, and are cleared at reset.

PORT L INTERRUPTS

Port L provides the user with an additional eight fully selectable, edge sensitive interrupts which are all vectored into the same service subroutine.

The interrupt from Port L shares logic with the wake up circuitry. The register WKEN allows interrupts from Port L to be individually enabled or disabled. The register WKEDG specifies the trigger condition to be either a positive or a negative edge. Finally, the register WKPND latches in the pending trigger conditions.

The GIE (Global Interrupt Enable) bit enables the interrupt function. A control flag, LPEN, functions as a global interrupt enable for Port L interrupts. Setting the LPEN flag will enable interrupts and vice versa. A separate global pending flag is not needed since the register WKPND is adequate.

Since Port L is also used for waking the device out of the HALT or IDLE modes, the user can elect to exit the HALT or IDLE modes either with or without the interrupt enabled. If he elects to disable the interrupt, then the device will restart execution from the instruction immediately following the instruction that placed the microcontroller in the HALT or IDLE modes. In the other case, the device will first execute the interrupt service routine and then revert to normal operation.

The Wakeup signal will not start the chip running immediately since crystal oscillators or ceramic resonators have a finite start up time. The IDLE Timer (T0) generates a fixed delay to ensure that the oscillator has indeed stabilized before allowing the execution of instructions. In this case, upon detecting a valid Wakeup signal, only the oscillator circuitry and the IDLE Timer T0 are enabled. The IDLE Timer is loaded with a value of 256 and is clocked from the t_C instruction cycle clock. The t_C clock is derived by dividing down the oscillator clock by a factor of 10. A Schmitt trigger following the CKI on-chip inverter ensures that the IDLE timer is clocked only when the oscillator has a sufficiently large amplitude to meet the Schmitt trigger specifications. This Schmitt trigger is not part of the oscillator closed loop. The startup timeout from the IDLE timer enables the clock signals to be routed to the rest of the chip.

If the RC clock option is used, the fixed delay is under software control. A control flag, CLKDLY, in the G7 configuration bit allows the clock start up delay to be optionally inserted. Setting CLKDLY flag high will cause clock start up delay to be inserted and resetting it will exclude the clock start up delay. The CLKDLY flag is cleared during reset, so the clock start up delay is not present following reset with the RC clock options.

Interrupts

The device supports a vectored interrupt scheme. It supports a total of ten interrupt sources. The following table lists all the possible interrupt sources, their arbitration ranking and the memory locations reserved for the interrupt vector for each source.

Interrupts (Continued)

Arbitration Ranking	Source	Description	Vector Address Hi-Low Byte
(1) Highest	Software	INTR Instruction	0yFE–0yFF
(2)	Reserved	for Future Use	0yFC–0yFD
(3)	External	Pin G0 Edge	0yFA–0yFB
(4)	Timer T0	Underflow	0yF8–0yF9
(5)	Timer T1	T1A/Underflow	0yF6–0yF7
(6)	Timer T1	T1B	0yF4–0yF5
(7)	MICROWIRE/PLUS	BUSY Goes Low	0yF2–0yF3
(8)	Reserved	for Future Use	0yF0–0yF1
(9)	Reserved	for UART	0yEE–0yEF
(10)	Reserved	for UART	0yEC–0yED
(11)	Timer T2	T2A/Underflow	0yEA–0yEB
(12)	Timer T2	T2B	0yE8–0yE9
(13)	Reserved	for Future Use	0yE6–0yE7
(14)	Reserved	for Future Use	0yE4–0yE5
(15)	Port L/Wakeup	Port L Edge	0yE2–0yE3
(16) Lowest	Default	VIS Instr. Execution without Any Interrupts	0yE0–0yE1

y is VIS page, $y \neq 0$.

Two bytes of program memory space are reserved for each interrupt source. All interrupt sources except the software interrupt are maskable. Each of the maskable interrupts have an Enable bit and a Pending bit. A maskable interrupt is active if its associated enable and pending bits are set. If GIE = 1 and an interrupt is active, then the processor will be interrupted as soon as it is ready to start executing an instruction except if the above conditions happen during the Software Trap service routine. This exception is described in the Software Trap sub-section.

The interruption process is accomplished with the INTR instruction (opcode 00), which is jammed inside the Instruction Register and replaces the opcode about to be executed. The following steps are performed for every interrupt:

1. The GIE (Global Interrupt Enable) bit is reset.
2. The address of the instruction about to be executed is pushed into the stack.
3. The PC (Program Counter) branches to address 00FF. This procedure takes 7 t_C cycles to execute.

At this time, since GIE = 0, other maskable interrupts are disabled. The user is now free to do whatever context switching is required by saving the context of the machine in the stack with PUSH instructions. The user would then program a VIS (Vector Interrupt Select) instruction in order to branch to the interrupt service routine of the highest priority interrupt enabled and pending at the time of the VIS. Note that this is not necessarily the interrupt that caused the branch to address location 00FF Hex prior to the context switching.

Thus, if an interrupt with a higher rank than the one which caused the interruption becomes active before the decision of which interrupt to service is made by the VIS, then the interrupt with the higher rank will override any lower ones and will be acknowledged. The lower priority interrupt(s) are still pending, however, and will cause another interrupt immediately following the completion of the interrupt service routine associated with the higher priority interrupt just serviced. This lower priority interrupt will occur immediately following the RETI (Return from Interrupt) instruction at the end of the interrupt service routine just completed.

Interrupts (Continued)

Inside the interrupt service routine, the associated pending bit has to be cleared by software. The RETI (Return from Interrupt) instruction at the end of the interrupt service routine will set the GIE (Global Interrupt Enable) bit, allowing the processor to be interrupted again if another interrupt is active and pending.

The VIS instruction looks at all the active interrupts at the time it is executed and performs an indirect jump to the beginning of the service routine of the one with the highest rank.

The addresses of the different interrupt service routines, called vectors, are chosen by the user and stored in ROM in a table starting at 01E0 (assuming that VIS is located between 00FF and 01DF). The vectors are 15-bit wide and therefore occupy 2 ROM locations.

VIS and the vector table must be located in the same 256-byte block (0y00 to 0yFF) except if VIS is located at the last address of a block. In this case, the table must be in the next block. The vector table cannot be inserted in the first 256-byte block.

The vector of the maskable interrupt with the lowest rank is located at 0yE0 (Hi-Order byte) and 0yE1 (Lo-Order byte) and so forth in increasing rank number. The vector of the maskable interrupt with the highest rank is located at 0yFA (Hi-Order byte) and 0yFB (Lo-Order byte).

The Software Trap has the highest rank and its vector is located at 0yFE and 0yFF.

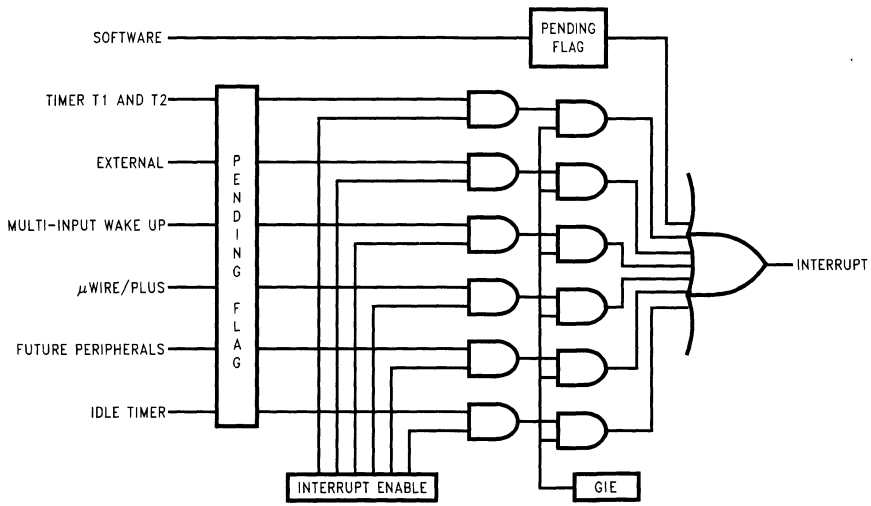
If, by accident, a VIS gets executed and no interrupt is active, then the PC (Program Counter) will branch to a vector located at 0yE0–0yE1.

WARNING

A Default VIS interrupt handler routine must be present. As a minimum, this handler should confirm that the GIE bit is cleared (this indicates that the interrupt sequence has been taken), take care of any required housekeeping, restore context and return. Some sort of Warm Restart procedure should be implemented. These events can occur without any error on the part of the system designer or programmer.

Note: There is always the possibility of an interrupt occurring during an instruction which is attempting to reset the GIE bit or any other interrupt enable bit. If this occurs when a single cycle instruction is being used to reset the interrupt enable bit, the interrupt enable bit will be reset but an interrupt may still occur. This is because interrupt processing is started at the same time as the interrupt bit is being reset. To avoid this scenario, the user should always use a two, three, or four cycle instruction to reset interrupt enable bits.

Figure 10 shows the Interrupt block diagram.



TL/DD/12524–13

FIGURE 10. COP888CL Interrupt Block Diagram

Interrupts (Continued)

SOFTWARE TRAP

The Software Trap (ST) is a special kind of non-maskable interrupt which occurs when the INTR instruction (used to acknowledge interrupts) is fetched from ROM and placed inside the instruction register. This may happen when the PC is pointing beyond the available ROM address space or when the stack is over-popped.

When an ST occurs, the user can re-initialize the stack pointer and do a recovery procedure (similar to reset, but not necessarily containing all of the same initialization procedures) before restarting.

The occurrence of an ST is latched into the ST pending bit. The GIE bit is not affected and the ST pending bit **(not accessible by the user)** is used to inhibit other interrupts and to direct the program to the ST service routine with the VIS instruction. The RPND instruction is used to clear the software interrupt pending bit. This bit is also cleared on reset.

The ST has the highest rank among all interrupts.

Nothing (except another ST) can interrupt an ST being serviced.

WATCHDOG

The device contains a WATCHDOG and clock monitor. The WATCHDOG is designed to detect the user program getting stuck in infinite loops resulting in loss of program control or "runaway" programs. The Clock Monitor is used to detect the absence of a clock or a very slow clock below a specified rate on the CKI pin.

The WATCHDOG consists of two independent logic blocks: WD UPPER and WD LOWER. WD UPPER establishes the upper limit on the service window and WD LOWER defines the lower limit of the service window.

Servicing the WATCHDOG consists of writing a specific value to a WATCHDOG Service Register named WDSVR which is memory mapped in the RAM. This value is composed of three fields, consisting of a 2-bit Window Select, a 5-bit Key Data field, and the 1-bit Clock Monitor Select field. Table III shows the WDSVR register.

TABLE III. WATCHDOG Service Register (WDSVR)

Window Select		Key Data					Clock Monitor
X	X	0	1	1	0	0	Y
7	6	5	4	3	2	1	0

The lower limit of the service window is fixed at 2048 instruction cycles. Bits 7 and 6 of the WDSVR register allow the user to pick an upper limit of the service window.

Table IV shows the four possible combinations of lower and upper limits for the WATCHDOG service window. This flexibility in choosing the WATCHDOG service window prevents any undue burden on the user software.

Bits 5, 4, 3, 2 and 1 of the WDSVR register represent the 5-bit Key Data field. The key data is fixed at 01100. Bit 0 of the WDSVR Register is the Clock Monitor Select bit.

TABLE IV. WATCHDOG Service Window Select

WDSVR Bit 7	WDSVR Bit 6	Service Window (Lower-Upper Limits)
0	0	2k–8k t_C Cycles
0	1	2k–16k t_C Cycles
1	0	2k–32k t_C Cycles
1	1	2k–64k t_C Cycles

Clock Monitor

The Clock Monitor aboard the device can be selected or deselected under program control. The Clock Monitor is guaranteed not to reject the clock if the instruction cycle clock ($1/t_C$) is greater or equal to 10 kHz. This equates to a clock input rate on CKI of greater or equal to 100 kHz.

WATCHDOG Operation

The WATCHDOG and Clock Monitor are disabled during reset. The device comes out of reset with the WATCHDOG armed, the WATCHDOG Window Select (bits 6, 7 of the WDSVR Register) set, and the Clock Monitor bit (bit 0 of the WDSVR Register) enabled. Thus, a Clock Monitor error will occur after coming out of reset, if the instruction cycle clock frequency has not reached a minimum specified value, including the case where the oscillator fails to start.

The WDSVR register can be written to only once after reset and the key data (bits 5 through 1 of the WDSVR Register) must match to be a valid write. This write to the WDSVR register involves two irrevocable choices: (i) the selection of the WATCHDOG service window (ii) enabling or disabling of the Clock Monitor. Hence, the first write to WDSVR Register involves selecting or deselecting the Clock Monitor, select the WATCHDOG service window and match the WATCHDOG key data. Subsequent writes to the WDSVR register will compare the value being written by the user to the WATCHDOG service window value and the key data (bits 7 through 1) in the WDSVR Register. Table V shows the sequence of events that can occur.

The user must service the WATCHDOG at least once before the upper limit of the serivce window expires. The WATCHDOG may not be serviced more than once in every lower limit of the service window. The user may service the WATCHDOG as many times as wished in the time period between the lower and upper limits of the service window. The first write to the WDSVR Register is also counted as a WATCHDOG service.

The WATCHDOG has an output pin associated with it. This is the WDOUT pin, on pin 1 of the port G. WDOUT is active low. The WDOUT pin is in the high impedance state in the inactive state. Upon triggering the WATCHDOG, the logic will pull the WDOUT (G1) pin low for an additional 16 t_C–32 t_C cycles after the signal level on WDOUT pin goes below the lower Schmitt trigger threshold. After this delay, the device will stop forcing the WDOUT output low.

The WATCHDOG service window will restart when the WDOUT pin goes high It is recommended that the user tie the WDOUT pin back to V_{CC} through a resistor in order to pull WDOUT high.

A WATCHDOG service while the WDOUT signal is active will be ignored. The state of the WDOUT pin is not guaranteed on reset, but if it powers up low then the WATCHDOG will time out and WDOUT will enter high impedance state.

WATCHDOG Operation (Continued)

TABLE V. WATCHDOG Service Actions

Key Data	Window Data	Clock Monitor	Action
Match	Match	Match	Valid Service: Restart Service Window
Don't Care	Mismatch	Don't Care	Error: Generate WATCHDOG Output
Mismatch	Don't Care	Don't Care	Error: Generate WATCHDOG Output
Don't Care	Don't Care	Mismatch	Error: Generate WATCHDOG Output

TABLE VI. MICROWIRE/PLUS Master Mode Clock Select

SL1	SL0	SK
0	0	$2 \times t_C$
0	1	$4 \times t_C$
1	x	$8 \times t_C$

Where t_C is the instruction cycle clock

The CLOCK MONITOR forces the G1 pin low upon detecting a clock frequency error. The CLOCK MONITOR error will continue until the clock frequency has reached the minimum specified value, after which the G1 output will enter the high impedance TRI-STATE mode following 16 t_C–32 t_C clock cycles. The CLOCK MONITOR generates a continual CLOCK MONITOR error if the oscillator fails to start, or fails to reach the minimum specified frequency. The specification for the CLOCK MONITOR is as follows:

$1/t_C > 10$ kHz—No clock rejection.

$1/t_C < 10$ Hz—Guaranteed clock rejection.

WATCHDOG AND CLOCK MONITOR SUMMARY

The following salient points regarding the WATCHDOG and CLOCK MONITOR should be noted:

- Both WATCHDOG and CLOCK MONITOR detector circuits are inhibited during RESET.
- Following RESET, the WATCHDOG and CLOCK MONITOR are both enabled, with the WATCHDOG having the maximum service window selected.
- The WATCHDOG service window and CLOCK MONITOR enable/disable option can only be changed once, during the initial WATCHDOG service following RESET.
- The initial WATCHDOG service must match the key data value in the WATCHDOG Service register WDSVR in order to avoid a WATCHDOG error.
- Subsequent WATCHDOG services must match all three data fields in WDSVR in order to avoid WATCHDOG errors.
- The correct key data value cannot be read from the WATCHDOG Service register WDSVR. Any attempt to read this key data value of 01100 from WDSVR will read as key data value of all 0's.
- The WATCHDOG detector circuit is inhibited during both the HALT and IDLE modes.
- The CLOCK MONITOR detector circuit is active during both the HALT and IDLE modes. Consequently, the device inadvertently entering the HALT mode will be detected as a CLOCK MONITOR error (provided that the CLOCK MONITOR enable option has been selected by the program).
- With the single-pin R/C oscillator mask option selected and the CLKDLY bit reset, the WATCHDOG service window will resume following HALT mode from where it left off before entering the HALT mode.
- With the crystal oscillator mask option selected, or with the single-pin R/C oscillator mask option selected and the CLKDLY bit set, the WATCHDOG service window will be set to its selected value from WDSVR following HALT. Consequently, the WATCHDOG should not be serviced for at least 2048 instruction cycles following HALT, but must be serviced within the selected window to avoid a WATCHDOG error.
- The IDLE timer T0 is not initialized with RESET.
- The user can sync in to the IDLE counter cycle with an IDLE counter (T0) interrupt or by monitoring the T0PND flag. The T0PND flag is set whenever the thirteenth bit of the IDLE counter toggles (every 4096 instruction cycles). The user is responsible for resetting the T0PND flag.
- A hardware WATCHDOG service occurs just as the device exits the IDLE mode. Consequently, the WATCHDOG should not be serviced for at least 2048 instruction cycles following IDLE, but must be serviced within the selected window to avoid a WATCHDOG error.
- Following RESET, the initial WATCHDOG service (where the service window and the Clock Monitor enable/disable must be selected) may be programmed anywhere within the maximum service window (65,536 instruction cycles) initialized by RESET. Note that this initial WATCHDOG service may be programmed within the initial 2048 instruction cycles without causing a WATCHDOG error.

Detection of Illegal Conditions

The device can detect various illegal conditions resulting from coding errors, transient noise, power supply voltage drops, runaway programs, etc.

Reading of undefined ROM gets zeros. The opcode for software interrupt is zero. If the program fetches instructions from undefined ROM, this will force a software interrupt, thus signaling that an illegal condition has occurred.

Detection of Illegal Conditions (Continued)

The subroutine stack grows down for each call (jump to subroutine), interrupt, or PUSH, and grows up for each return or POP. The stack pointer is initialized to RAM location 06F Hex during reset. Consequently, if there are more returns than calls, the stack pointer will point to addresses 070 and 071 Hex (which are undefined RAM). Undefined RAM from addresses 070 to 07F Hex is read as all 1's, which in turn will cause the program to return to address 7FFF Hex. This is an undefined ROM location and the instruction fetched (all 0's) from this location will generate a software interrupt signaling an illegal condition.

Thus, the chip can detect the following illegal conditions:

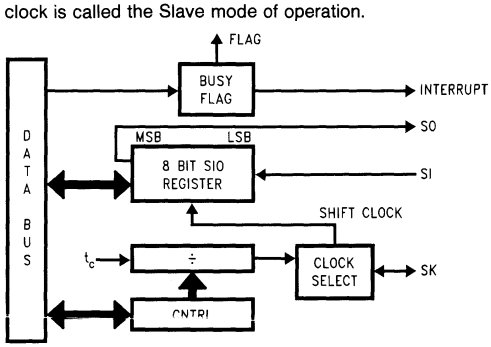
1. Executing from undefined ROM
2. Over "POP"ing the stack by having more returns than calls.

When the software interrupt occurs, the user can re-initialize the stack pointer and do a recovery procedure before restarting (this recovery program is probably similar to that following reset, but might not contain the same program initialization procedures).

MICROWIRE/PLUS

MICROWIRE/PLUS is a serial synchronous communications interface. The MICROWIRE/PLUS capability enables the device to interface with any of National Semiconductor's MICROWIRE peripherals (i.e. A/D converters, display drivers, E^2PROMs etc.) and with other microcontrollers which support the MICROWIRE interface. It consists of an 8-bit serial shift register (SIO) with serial data input (SI), serial data output (SO) and serial shift clock (SK). *Figure 11* shows a block diagram of the MICROWIRE logic.

The shift clock can be selected from either an internal source or an external source. Operating the MICROWIRE/PLUS arrangement with the internal clock source is called the Master mode of operation. Similarly, operating the MICROWIRE/PLUS arrangement with an external shift clock is called the Slave mode of operation.



TL/DD/12524-14

FIGURE 11. MICROWIRE/PLUS Block Diagram

The CNTRL register is used to configure and control the MICROWIRE/PLUS mode. To use the MICROWIRE/PLUS, the MSEL bit in the CNTRL register is set to one. In the master mode, the SK clock rate is selected by the two bits, SL0 and SL1, in the CNTRL register. Table VI details the different clock rates that may be selected.

MICROWIRE/PLUS OPERATION

Setting the BUSY bit in the PSW register causes the MICROWIRE/PLUS to start shifting the data. It gets reset when eight data bits have been shifted. The user may reset the BUSY bit by software to allow less than 8 bits to shift. If enabled, an interrupt is generated when eight data bits have been shifted. The device may enter the MICROWIRE/PLUS mode either as a Master or as a Slave. *Figure 12* shows how two COP888 microcontrollers and several peripherals may be interconnected using the MICROWIRE/PLUS arrangements.

Warning

The SIO register should only be loaded when the SK clock is low. Loading the SIO register while the SK clock is high will result in undefined data in the SIO register. SK clock is normally low when not shifting.

Setting the BUSY flag when the input SK clock is high in the MICROWIRE/PLUS slave mode may cause the current SK clock for the SIO shift register to be narrow. For safety, the BUSY flag should only be set when the input SK clock is low.

MICROWIRE/PLUS Master Mode Operation

In the MICROWIRE/PLUS Master mode of operation the shift clock (SK) is generated internally. The MICROWIRE Master always initiates all data exchanges. The MSEL bit in the CNTRL register must be set to enable the SO and SK functions onto the G Port. The SO and SK pins must also be selected as outputs by setting appropriate bits in the Port G configuration register. Table VII summarizes the bit settings required for Master mode of operation.

MICROWIRE/PLUS Slave Mode Operation

In the MICROWIRE/PLUS Slave mode of operation the SK clock is generated by an external source. Setting the MSEL bit in the CNTRL register enables the SO and SK functions onto the G Port. The SK pin must be selected as an input and the SO pin is selected as an output pin by setting and resetting the appropriate bit in the Port G configuration register. Table VII summarizes the settings required to enter the Slave mode of operation.

The user must set the BUSY flag immediately upon entering the Slave mode. This will ensure that all data bits sent by the Master will be shifted properly. After eight clock pulses the BUSY flag will be cleared and the sequence may be repeated.

Alternate SK Phase Operation

The device allows either the normal SK clock or an alternate phase SK clock to shift data in and out of the SIO register. In both the modes the SK is normally low. In the normal mode data is shifted in on the rising edge of the SK clock and the data is shifted out on the falling edge of the SK clock. The SIO register is shifted on each falling edge of the SK clock in the normal mode. In the alternate SK phase mode the SIO register is shifted on the rising edge of the SK clock.

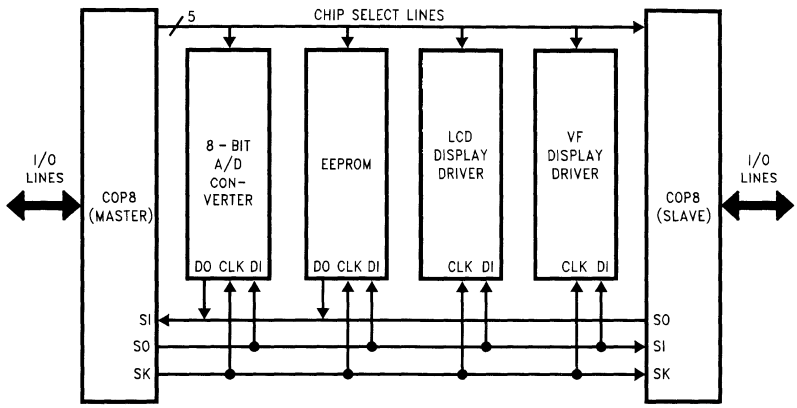
A control flag, SKSEL, allows either the normal SK clock or the alternate SK clock to be selected. Resetting SKSEL causes the MICROWIRE/PLUS logic to be clocked from the normal SK signal. Setting the SKSEL flag selects the alternate SK clock. The SKSEL is mapped into the G6 configuration bit. The SKSEL flag will power up in the reset condition, selecting the normal SK signal.

MICROWIRE/PLUS (Continued)

TABLE VII

G4 (SO) Config. Bit	G5 (SK) Config. Bit	G4 Fun.	G5 Fun.	Operation
1	1	SO	Int. SK	MICROWIRE/PLUS Master
0	1	TRI-STATE	Int. SK	MICROWIRE/PLUS Master
1	0	SO	Ext. SK	MICROWIRE/PLUS Slave
0	0	TRI-STATE	Ext. SK	MICROWIRE/PLUS Slave

This table assumes that the control flag MSEL is set.



TL/DD/12524–15

FIGURE 12. MICROWIRE/PLUS Application

Memory Map

All RAM, ports and registers (except A and PC) are mapped into data memory address space

Address	Contents
00 to 6F	On-Chip RAM bytes
70 to BF	Unused RAM Address Space
C0	Timer T2 Lower Byte
C1	Timer T2 Upper Byte
C2	Timer T2 Autoload Register T2RA Lower Byte
C3	Timer T2 Autoload Register T2RA Upper Byte
C4	Timer T2 Autoload Register T2RB Lower Byte
C5	Timer T2 Autoload Register T2RB Upper Byte
C6	Timer T2 Control Register
C7	WATCHDOG Service Register (Reg:WDSVR)
C8	MIWU Edge Select Register (Reg:WKEDG)
C9	MIWU Enable Register (Reg:WKEN)
CA	MIWU Pending Register (Reg:WKPND)
CB to CF	Reserved
D0	Port L Data Register
D1	Port L Configuration Register
D2	Port L Input Pins (Read Only)
D3	Reserved for Port L
D4	Port G Data Register
D5	Port G Configuration Register
D6	Port G Input Pins (Read Only)
D7	Port I Input Pins (Read Only)
D8	Port C Data Register
D9	Port C Configuration Register
DA	Port C Input Pins (Read Only)
DB	Reserved for Port C
DC	Port D Data Register
DD to DF	Reserved for Port D
E0 to E5	Reserved
E6	Timer T1 Autoload Register T1RB Lower Byte
E7	Timer T1 Autoload Register T1RB Upper Byte
E8	ICNTRL Register
E9	MICROWIRE Shift Register
EA	Timer T1 Lower Byte
EB	Timer T1 Upper Byte
EC	Timer T1 Autoload Register T1RA Lower Byte
ED	Timer T1 Autoload Register T1RA Upper Byte
EE	CNTRL Control Register
EF	PSW Register
F0 to FB	On-Chip RAM Mapped as Registers
FC	X Register
FD	SP Register
FE	B Register
FF	Reserved

Note: Reading memory locations 70-7F Hex will return all ones. Reading other unused memory locations will return undefined data.

Addressing Modes

There are ten addressing modes, six for operand addressing and four for transfer of control.

OPERAND ADDRESSING MODES

Register Indirect

This is the "normal" addressing mode. The operand is the data memory addressed by the B pointer or X pointer.

Register Indirect (with auto post increment or decrement of pointer)

This addressing mode is used with the LD and X instructions. The operand is the data memory addressed by the B pointer or X pointer. This is a register indirect mode that automatically post increments or decrements the B or X register after executing the instruction.

Direct

The instruction contains an 8-bit address field that directly points to the data memory for the operand.

Immediate

The instruction contains an 8-bit immediate field as the operand.

Short Immediate

This addressing mode is used with the Load B Immediate instruction. The instruction contains a 4-bit immediate field as the operand.

Indirect

This addressing mode is used with the LAID instruction. The contents of the accumulator are used as a partial address (lower 8 bits of PC) for accessing a data operand from the program memory.

TRANSFER OF CONTROL ADDRESSING MODES

Relative

This mode is used for the JP instruction, with the instruction field being added to the program counter to get the new program location. JP has a range from −31 to +32 to allow a 1-byte relative jump (JP + 1 is implemented by a NOP instruction). There are no "pages" when using JP, since all 15 bits of PC are used.

Absolute

This mode is used with the JMP and JSR instructions, with the instruction field of 12 bits replacing the lower 12 bits of the program counter (PC). This allows jumping to any location in the current 4k program memory segment.

Absolute Long

This mode is used with the JMPL and JSR instructions, with the instruction field of 12 bits replacing the lower 12 bits of the program counter (PC). This allows jumping to any location up to 32k in the program memory space.

Indirect

This mode is used with the JID instruction. The contents of the accumulator are used as a partial address (lower 8 bits of PC) for accessing a location in the program memory. The contents of this program memory location serve as a partial address (lower 8 bits of PC) for the jump to the next instruction.

Note: The VIS is a special case of the Indirect Transfer of Control addressing mode, where the double byte vector associated with the interrupt is transferred from adjacent addresses in the program memory into the program counter (PC) in order to jump to the associated interrupt service routine.

Instruction Set

Register and Symbol Definition

Registers	
A	8-Bit Accumulator Register
B	8-Bit Address Register
X	8-Bit Address Register
SP	8-Bit Stack Pointer Register
PC	15-Bit Program Counter Register
PU	Upper 7 Bits of PC
PL	Lower 8 Bits of PC
C	1 Bit of PSW Register for Carry
HC	1 Bit of PSW Register for Half Carry
GIE	1 Bit of PSW Register for Global Interrupt Enable
VU	Interrupt Vector Upper Byte
VL	Interrupt Vector Lower Byte

Symbols	
[B]	Memory Indirectly Addressed by B Register
[X]	Memory Indirectly Addressed by X Register
MD	Direct Addressed Memory
Mem	Direct Addressed Memory or [B]
Meml	Direct Addressed Memory or [B] or Immediate Data
Imm	8-Bit Immediate Data
Reg	Register Memory: Addresses F0 to FF (Includes B, X and SP)
Bit	Bit Number (0 to 7)
←	Loaded with
↔	Exchanged with

Instruction Set (Continued)

INSTRUCTION SET

ADD	A,Meml	ADD	A ← A + Meml
ADC	A,Meml	ADD with Carry	A ← A + Meml + C, C ← Carry, HC ← Half Carry
SUBC	A,Meml	Subtract with Carry	A ← A − Meml + C, C ← Carry, HC ← Half Carry
AND	A,Meml	Logical AND	A ← A and Meml
ANDSZ	A,Imm	Logical AND Immed., Skip if Zero	Skip next if (A and Imm) = 0
OR	A,Meml	Logical OR	A ← A or Meml
XOR	A,Meml	Logical EXclusive OR	A ← A xor Meml
IFEQ	MD,Imm	IF EQual	Compare MD and Imm, Do next if MD = Imm
IFEQ	A,Meml	IF EQual	Compare A and Meml, Do next if A = Meml
IFNE	A,Meml	IF Not Equal	Compare A and Meml, Do next if A ≠ Meml
IFGT	A,Meml	IF Greater Than	Compare A and Meml, Do next if A > Meml
IFBNE	#	If B Not Equal	Do next if lower 4 bits of B ≠ Imm
DRSZ	Reg	Decrement Reg., Skip if Zero	Reg ← Reg − 1, Skip if Reg = 0
SBIT	#,Mem	Set BIT	1 to bit, Mem (bit = 0 to 7 immediate)
RBIT	#,Mem	Reset BIT	0 to bit, Mem
IFBIT	#,Mem	IF BIT	If bit in A or Mem is true do next instruction
RPND		Reset PeNDing Flag	Reset Software Interrupt Pending Flag
X	A,Mem	EXchange A with Memory	A ⟷ Mem
X	A,[X]	EXchange A with Memory [X]	A ⟷ [X]
LD	A,Meml	LoaD A with Memory	A ← Meml
LD	A,[X]	LoaD A with Memory [X]	A ← [X]
LD	B,Imm	LoaD B with Immed.	B ← Imm
LD	Mem,Imm	LoaD Memory Immed.	Mem ← Imm
LD	Reg,Imm	LoaD Register Memory Immed.	Reg ← Imm
X	A, [B ±]	EXchange A with Memory [B]	A ⟷ [B], (B ← B ± 1)
X	A, [X ±]	EXchange A with Memory [X]	A ⟷ [X], (X ← ± 1)
LD	A, [B ±]	LoaD A with Memory [B]	A ← [B], (B ← B ± 1)
LD	A, [X ±]	LoaD A with Memory [X]	A ← [X], (X ← X ± 1)
LD	[B ±],Imm	LoaD Memory [B] Immed.	[B] ← Imm, (B ← ± 1)
CLR	A	CLeaR A	A ← 0
INC	A	INCrement A	A ← A + 1
DEC	A	DECrementA	A ← A − 1
LAID		Load A InDirect from ROM	A ← ROM (PU,A)
DCOR	A	Decimal CORrect A	A ← BCD correction of A (follows ADC, SUBC)
RRC	A	Rotate A Right thru C	C ⟷ A7 ⟷ . . . ⟷ A0 ⟷ C
RLC	A	Rotate A Left thru C	C ← A7 ← . . . ← A0 ← C
SWAP	A	SWAP nibbles of A	A7 . . . A4 ⟷ A3 . . . A0
SC		Set C	C ← 1, HC ← 1
RC		Reset C	C ← 0, HC ← 0
IFC		IF C	IF C is true, do next instruction
IFNC		IF Not C	If C is not true, do next instruction
POP	A	POP the stack into A	SP ← SP + 1, A ← [SP]
PUSH	A	PUSH A onto the stack	[SP] ← A, SP ← SP − 1
VIS		Vector to Interrupt Service Routine	PU ← [VU], PL ← [VL]
JMPL	Addr.	Jump absolute Long	PC ← ii (ii = 15 bits, 0 to 32k)
JMP	Addr.	Jump absolute	PC9 . . . 0 ← i (i = 12 bits)
JP	Disp.	Jump relative short	PC ← PC + r (r is −31 to +32, except 1)
JSRL	Addr.	Jump SubRoutine Long	[SP] ← PL, [SP−1] ← PU,SP−2, PC ← ii
JSR	Addr.	Jump SubRoutine	[SP] ← PL, [SP−1] ← PU,SP−2, PC9 . . . 0 ← i
JID		Jump InDirect	PL ← ROM (PU,A)
RET		RETurn from subroutine	SP+2, PL ← [SP], PU ← [SP−1]
RETSK		RETurn and SKip	SP+2, PL ← [SP],PU ← [SP−1]
RETI		RETurn from Interrupt	SP+2, PL ← [SP],PU ← [SP−1],GIE ← 1
INTR		Generate an Interrupt	[SP] ← PL, [SP−1] ← PU, SP−2, PC ← 0FF
NOP		No OPeration	PC ← PC+1

Instruction Execution Time

Most instructions are single byte (with immediate addressing mode instructions taking two bytes).

Most single byte instructions take one cycle time to execute.

Skipped instructions require x number of cycles to be skipped, where x equals the number of bytes in the skipped instruction opcode.

See the BYTES and CYCLES per INSTRUCTION table for details.

Bytes and Cycles per Instruction

The following table shows the number of bytes and cycles for each instruction in the format of byte/cycle.

Logic and Arithmetic Instructions

	[B]	Direct	Immed.
ADD	1/1	3/4	2/2
ADC	1/1	3/4	2/2
SUBC	1/1	3/4	2/2
AND	1/1	3/4	2/2
OR	1/1	3/4	2/2
XOR	1/1	3/4	2/2
IFEQ	1/1	3/4	2/2
IFGT	1/1	3/4	2/2
IFBNE	1/1		
DRSZ		1/3	
SBIT	1/1	3/4	
RBIT	1/1	3/4	
IFBIT	1/1	3/4	

RPND	1/1

Instructions Using A and C

CLRA	1/1
INCA	1/1
DECA	1/1
LAID	1/3
DCORA	1/1
RRCA	1/1
RLCA	1/1
SWAPA	1/1
SC	1/1
RC	1/1
IFC	1/1
IFNC	1/1
PUSHA	1/3
POPA	1/3
ANDSZ	2/2

Transfer of Control Instructions

JMPL	3/4
JMP	2/3
JP	1/3
JSRL	3/5
JSR	2/5
JID	1/3
VIS	1/5
RET	1/5
RETSK	1/5
RETI	1/5
INTR	1/7
NOP	1/1

Memory Transfer Instructions

	Register Indirect		Direct	Immed.	Register Indirect Auto Incr. and Decr.		
	[B]	[X]			[B+, B−]	[X+, X−]	
X A,*	1/1	1/3	2/3		1/2	1/3	
LD A,*	1/1	1/3	2/3	2/2	1/2	1/3	
LD B, Imm				1/1			(IF B < 16)
LD B, Imm				2/3			(IF B > 15)
LD Mem, Imm	2/2		3/3		2/2		
LD Reg, Imm			2/3				
IFEQ MD, Imm			3/3				

* = > Memory location addressed by B or X or directly.

COP87L88CL/COP87L84CL Opcode Table

UPPER NIBBLE															
F	E	D	C	B	A	9	8	7	6	5	4	3	2	1	0
JP −15	JP −31	LD 0F0, #i	DRSZ 0F0	RRCA	RC	ADC A, #i	ADC A,[B]	IFBITL 0,[B]	ANDSZ A, #i	LD B,#0F	IFBNE 0	JSR x000–x0FF	JMP x000–x0FF	JP +17	INTR
JP −14	JP −30	LD 0F1, #i	DRSZ 0F1	*	SC	SUBC A, #i	SUBC A,[B]	IFBIT 1,[B]	*	LD B,#0E	IFBNE 1	JSR x100–x1FF	JMP x100–x1FF	JP +18	JP + 2
JP −13	JP −29	LD 0F2, #i	DRSZ 0F2	X A, [X+]	X A, [B+]	IFEQ A, #i	IFEQ A,[B]	IFBIT 2,[B]	*	LD B,#0D	IFBNE 2	JSR x200–x2FF	JMP x200–x2FF	JP +19	JP + 3
JP −12	JP −28	LD 0F3, #i	DRSZ 0F3	X A, [X−]	X A, [B−]	IFGT A, #i	IFGT A,[B]	IFBIT 3,[B]	*	LD B,#0C	IFBNE 3	JSR x300–x3FF	JMP x300–x3FF	JP +20	JP + 4
JP −11	JP −27	LD 0F4, #i	DRSZ 0F4	VIS	LAID	ADD A, #i	ADD A,[B]	IFBIT 4,[B]	CLRA	LD B,#0B	IFBNE 4	JSR x400–x4FF	JMP x400–x4FF	JP +21	JP + 5
JP −10	JP −26	LD 0F5, #i	DRSZ 0F5	RPND	JID	AND A, #i	AND A,[B]	IFBIT 5,[B]	SWAPA	LD B,#0A	IFBNE 5	JSR x500–x5FF	JMP x500–x5FF	JP +22	JP + 6
JP −9	JP −25	LD 0F6, #i	DRSZ 0F6	X A,[X]	X A,[B]	XOR A, #i	XOR A,[B]	IFBIT 6,[B]	DCORA	LD B,#09	IFBNE 6	JSR x600–x6FF	JMP x600–x6FF	JP +23	JP + 7
JP −8	JP −24	LD 0F7, #i	DRSZ 0F7	*	*	OR A,#i	OR A,[B]	IFBIT 7,[B]	PUSHA	LD B,#08	IFBNE 7	JSR x700–x7FF	JMP x700–x7FF	JP +24	JP + 8
JP −7	JP −23	LD 0F8, #i	DRSZ 0F8	NOP	RLCA	LD A,#i	IFC	SBIT 0,[B]	RBIT 0,[B]	LD B,#07	IFBNE 8	JSR x800–x8FF	JMP x800–x8FF	JP +25	JP + 9
JP −6	JP −22	LD 0F9, #i	DRSZ 0F9	IFNE A,[B]	IFEQ Md,#i	IFNE A,#i	IFNC	SBIT 1,[B]	RBIT 1,[B]	LD B,#06	IFBNE 9	JSR x900–x9FF	JMP x900–x9FF	JP +26	JP + 10
JP −5	JP −21	LD 0FA, #i	DRSZ 0FA	LD A, [X+]	LD A, [B+]	LD [B+], #i	INCA	SBIT 2,[B]	RBIT 2,[B]	LD B,#05	IFBNE 0A	JSR xA00–xAFF	JMP xA00–xAFF	JP +27	JP + 11
JP −4	JP −20	LD 0FB, #i	DRSZ 0FB	LD A, [X−]	LD A, [B−]	LD [B−], #i	DECA	SBIT 3,[B]	RBIT 3,[B]	LD B,#04	IFBNE 0B	JSR xB00–xBFF	JMP xB00–xBFF	JP +28	JP + 12
JP −3	JP −19	LD 0FC, #i	DRSZ 0FC	LD Md,#i	JMPL	X A,Md	POPA	SBIT 4,[B]	RBIT 4,[B]	LD B,#03	IFBNE 0C	JSR xC00–xCFF	JMP xC00–xCFF	JP +29	JP + 13
JP −2	JP −18	LD 0FD, #i	DRSZ 0FD	DIR	JSRL	LD A,Md	RETSK	SBIT 5,[B]	RBIT 5,[B]	LD B,#02	IFBNE 0D	JSR xD00–xDFF	JMP xD00–xDFF	JP +30	JP + 14
JP −1	JP −17	LD 0FE, #i	DRSZ 0FE	LD A,[X]	LD A,[B]	LD [B],#i	RET	SBIT 6,[B]	RBIT 6,[B]	LD B,#01	IFBNE 0E	JSR xE00–xEFF	JMP xE00–xEFF	JP +31	JP + 15
JP −0	JP −16	LD 0FF, #i	DRSZ 0FF	*	*	LD B,#i	RETI	SBIT 7,[B]	RBIT 7,[B]	LD B,#00	IFBNE 0F	JSR xF00–xFFF	JMP xF00–xFFF	JP +32	JP + 16

LOWER NIBBLE

Where,

i is the immediate data

Md is a directly addressed memory location

* is an unused opcode

Note: The opcode 60 Hex is also the opcode for IFBIT #i,A.

Development Support

SUMMARY

- iceMASTER™: IM-COP8/400—Full feature in-circuit emulation for all COP8 products. A full set of COP8 Basic and Feature Family device and package specific probes are available.
- COP8 Debug Module: Moderate cost in-circuit emulation and development programming unit.
- COP8 Evaluation and Programming Unit: EPU-COP888GG—low cost in-circuit simulation and development programming unit.
- Assembler: COP8-DEV-IBMA. A DOS installable cross development Assembler, Linker, Librarian and Utility Software Development Tool Kit.
- C Compiler: COP8C. A DOS installable cross development Software Tool Kit.
- OTP/EPROM Programmer Support: Covering needs from engineering prototype, pilot production to full production environments.

iceMASTER (IM) IN-CIRCUIT EMULATION

The iceMASTER IM-COP8/400 is a full feature, PC based, in-circuit emulation tool developed and marketed by Meta-Link Corporation to support the whole COP8 family of products. National is a resale vendor for these products.

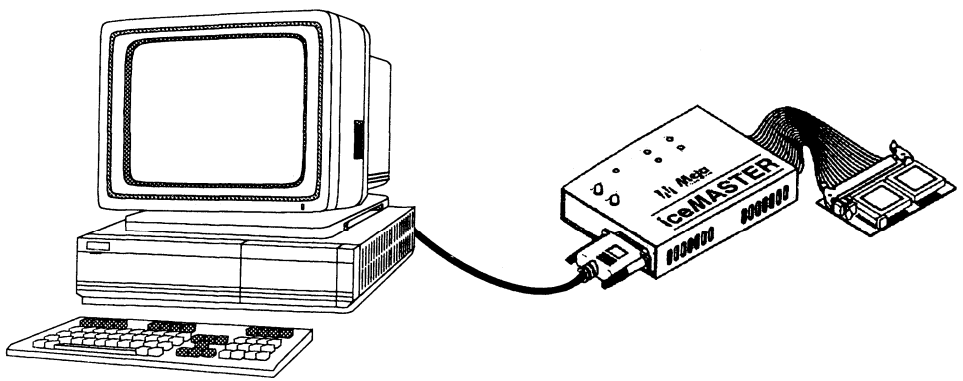
See *Figure 13* for configuration.

The iceMASTER IM-COP8/400 with its device specific COP8 Probe provides a rich feature set for developing, testing and maintaining product:

- Real-time in-circuit emulation; full 2.4V–5.5V operation range, full DC-10 MHz clock. Chip options are programmable or jumper selectable.
- Direct connection to application board by package compatible socket or surface mount assembly.
- Full 32 kbytes of loadable programming space that overlays (replaces) the on-chip ROM or EPROM. On-chip RAM and I/O blocks are used directly or recreated on the probe as necessary.
- Full 4k frame synchronous trace memory. Address, instruction, and 8 unspecified, circuit connectable trace lines. Display can be HLL source (e.g., C source), assembly or mixed.
- A full 64k hardware configurable break, trace on, trace off control, and pass count increment events.
- Tool set integrated interactive symbolic debugger—supports both assembler (COFF) and C Compiler (.COD) linked object formats.
- Real time performance profiling analysis; selectable bucket definition.
- Watch windows, content updated automatically at each execution break.
- Instruction by instruction memory/register changes displayed on source window when in single step operation.
- Single base unit and debugger software reconfigurable to support the entire COP8 family; only the probe personality needs to change. Debugger software is processor customized, and reconfigured from a master model file.
- Processor specific symbolic display of registers and bit level assignments, configured from master model file.
- Halt/Idle mode notification.
- On-line HELP customized to specific processor using master model file.
- Includes a copy of COP8-DEV-IBMA assembler and linker SDK.

IM Order Information

Base Unit	
IM-COP8/400-1	iceMASTER Base Unit, 110V Power Supply
IM-COP8/400-2	iceMASTER Base Unit, 220V Power Supply
iceMASTER Probe	
MHW-884CL28DWPC	28 DIP
MHW-888CL40DWPC	40 DIP
MHW-888CL44PWPC	44 PLCC
28 DIP to 28 SO Adapter	
MHW-SOIC28	28 SO



TL/DD/12524–17

FIGURE 13. COP8 iceMASTER Environment

Development Support (Continued)

iceMASTER DEBUG MODULE (DM)

The iceMASTER Debug Module is a PC based, combination in-circuit emulation tool and COP8 based OTP/EPROM programming tool developed and marketed by MetaLink Corporation to support the whole COP8 family of products. National is a resale vendor for these products.

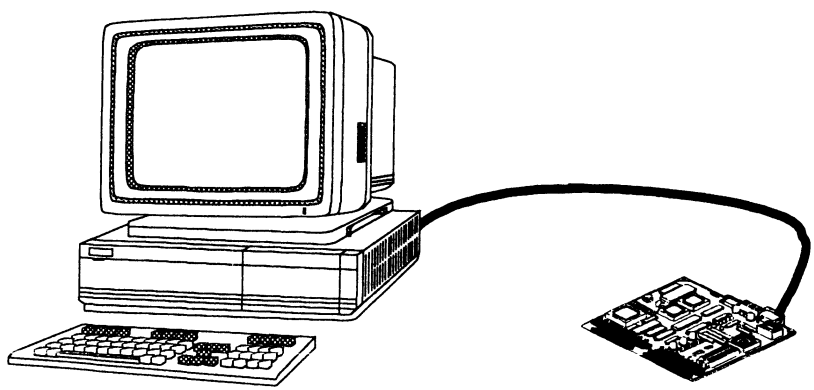
See *Figure 14* for configuration.

The iceMASTER Debug Module is a moderate cost development tool. It has the capability of in-circuit emulation for a specific COP8 microcontroller and in addition serves as a programming tool for COP8 OTP and EPROM product families. Summary of features is as follows:

- Real-time in-circuit emulation; full operating voltage range operation, full DC-10 MHz clock.
- All processor I/O pins can be cabled to an application development board with package compatible cable to socket and surface mount assembly.
- Full 32 kbytes of loadable programming space that overlays (replaces) the on-chip ROM or EPROM. On-chip RAM and I/O blocks are used directly or recreated as necessary.
- 100 frames of synchronous trace memory. The display can be HLL source (C source), assembly or mixed. The most recent history prior to a break is available in the trace memory.
- Configured break points; uses INTR instruction which is modestly intrusive.
- Software—only supported features are selectable.
- Tool set integrated interactive symbolic debugger—supports both assembler (COFF) and C Compiler (.COD) SDK linked object formats.
- Instruction by instruction memory/register changes displayed when in single step operation.
- Debugger software is processor customized, and reconfigured from a master model file.
- Processor specific symbolic display of registers and bit level assignments, configured from master model file.
- Halt/Idle mode notification.
- Programming menu supports full product line of programmable OTP and EPROM COP8 products. Program data is taken directly from the overlay RAM.
- Programming of 44 PLCC and 68 PLCC parts requires external programming adapters.
- Includes wall mount power supply.
- On-board V_{PP} generator from 5V input or connection to external supply supported. Requires V_{PP} level adjustment per the family programming specification (correct level is provided on an on-screen pop-down display).
- On-line HELP customized to specific processor using master model file.
- Includes a copy of COP8-DEV-IBMA assembler and linker SDK.

DM Order Information

Debug Model Unit	
COP8-DM/888CL	
Cable Adapters	
DM-COP8/28D	28 DIP
DM-COP8/40D	40 DIP
DM-COP8/44P	44 PLCC
28 DIP to 28 SO Adapter	
MHW-SOIC28	28 SO



TL/DD/12524–18

FIGURE 14. COP8-DM Environment

Development Support (Continued)

iceMASTER EVALUATION PROGRAMMING UNIT (EPU)

The iceMASTER EPU-COP888GG is a PC based, in-circuit simulation tool to support the feature family COP8 products.

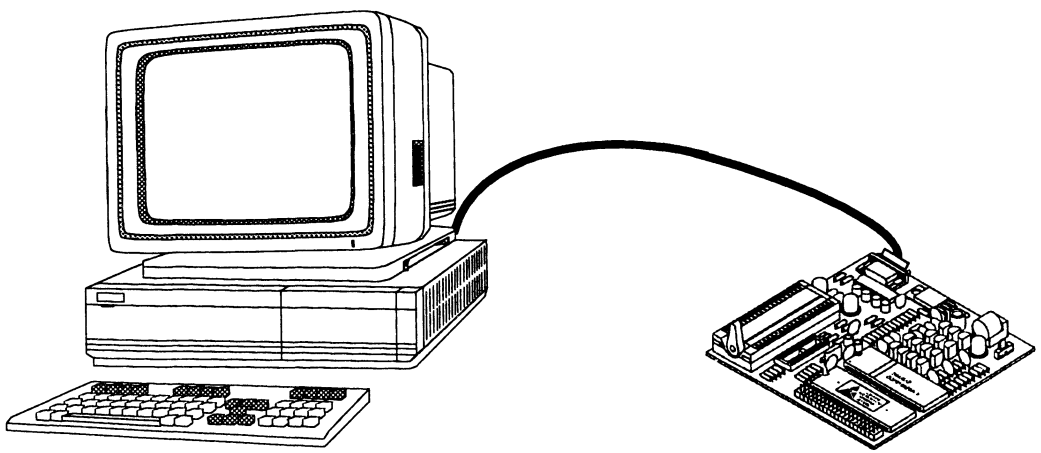
See *Figure 15* for configuration.

The simulation capability is a very low cost means of evaluating the general COP8 architecture. In addition, the EPU has programming capability, with added adapters, for programming the whole COP8 product family of OTP and EPROM products. The product includes the following features:

- Non-real-time in-circuit simulation. Program overlay memory is PC resident; instructions are downloaded over RS-232 as executed. Approximate performance is 20 kHz.
- Includes a 40-pin DIP cable adapter. Other target packages are not supported. All processor I/O pins are cabled to the application development environment.
- Full 32 kbytes of loadable programming space that overlays (replaces) the on-chip ROM or EPROM. On-chip RAM and I/O blocks are used directly or recreated as necessary.
- On-chip timer and WATCHDOG execution are not well synchronized to the instruction simulation.
- 100 frames of synchronous trace memory. The display can be HLL source (e.g., C source), assembly or mixed. The most recent history prior to a break is available in the trace memory.
- Up to eight software configured break points; uses INTR instruction which is modestly intrusive.
- Common look-feel debugger software across all MetaLink products—only supported features are selectable.
- Tool set integrated interactive symbolic debugger—supports both assembler (COFF) and C Compiler (.COD) SDK linked object formats.
- Instruction by instruction memory/register changes displayed when in single step operation.
- Processor specific symbolic display of registers and bit level assignments, configured from master model file.
- Halt/Idle mode notification. Restart requires special handling.
- Programming menu supports full product line of programmable OTP and EPROM COP8 products. Only a 40 ZIF socket is available on the EPU unit. Adapters are available for other part package configurations.
- Integral wall mount power supply provides 5V and develops the required V_{PP} to program parts.
- Includes a copy of COP8-DEV-IBMA assembler, linker SDK.

EPU Order Information

Evaluation Programming Unit	
EPU-COP888GG	Evaluation Programming Unit with debugger and programmer control software with 40 ZIF programming socket.
General Programming Adapters	
COP8-PGMA-DS	28 and 20 DIP and SOIC adapter
COP8-PGMA-DS44P	28 and 20 DIP and SOIC plus 44 PLCC adapter



TL/DD/12524-19

FIGURE 15. EPU-COP8 Tool Environment

Development Support (Continued)

COP8 ASSEMBLER/LINKER SOFTWARE DEVELOPMENT TOOL KIT

National Semiconductor offers a relocateable COP8 macro cross assembler, linker, librarian and utility software development tool kit. Features are summarized as follows:

- Basic and Feature Family instruction set by "device" type.
- Nested macro capability.
- Extensive set of assembler directives.
- Supported on PC/DOS platform.
- Generates National standard COFF output files.
- Integrated Linker and Librarian.
- Integrated utilities to generate ROM code file outputs.
- DUMPCOFF utility.

This product is integrated as a part of MetaLink tools as a development kit, fully supported by the MetaLink debugger. It may be ordered separately or it is bundled with the MetaLink products at no additional cost.

Order Information

Assembler SDK	
COP8-DEV-IBMA	Assembler SDK on installable 3.5" PC/DOS Floppy Disk Drive format. Periodic upgrades and most recent version is available on National's BBS and Internet.

COP8 C COMPILER

A C Compiler is developed and marketed by Byte Craft Limited. The COP8C compiler is a fully integrated development tool specifically designed to support the compact embedded configuration of the COP8 family of products.

Features are summarized as follows:

- ANSI C with some restrictions and extensions that optimize development for the COP8 embedded application.
- BITS data type extension. Register declaration #pragma with direct bit level definitions.
- C language support for interrupt routines.
- Expert system, rule based code generation and optimization.
- Performs consistency checks against the architectural definitions of the target COP8 device.
- Generates program memory code.
- Supports linking of compiled object or COP8 assembled object formats.
- Global optimization of linked code.
- Symbolic debug load format fully sourced level supported by the MetaLink debugger.

OTP/EMULATOR SUPPORT

The COP87L88CL/COP87L84CL devices provide emulation and OTP support for the COP888CL/COP884CL mask programmable devices.

OTP Emulator Ordering Information

Device Number	Clock Option	Package	Emulates
COP87L88CLV-XE	Crystal/HALT En	44 PLCC	COP888CL
COP87L88CLN-XE	Crystal/HALT En	40 DIP	COP888CL
COP87L84CLN-XE	Crystal/HALT En	28 DIP	COP884CL
COP87L84CLM-XE	Crystal/HALT En	28 SO	COP884CL

INDUSTRY WIDE OTP/EPROM PROGRAMMING SUPPORT

Programming support, in addition to the MetaLink development tools, is provided by a full range of independent approved vendors to meet the needs from the engineering laboratory to full production.

Approved List

Manufacturer	North America	Europe	Asia
BP Microsystems	(800) 225-2102 (713) 688-4600 Fax: (713) 688-0920	+49-8152-4183 +49-8856-932616	+852-234-16611 +852-2710-8121
Data I/O	(800) 426-1045 (206) 881-6444 Fax: (206) 882-1043	+44-0734-440011	Call North America
HI-LO	(510) 623-8860	Call Asia	+886-2-764-0215 Fax: +886-2-756-6403
ICE Technology	(800) 624-8949 (919) 430-7915	+44-1226-767404 Fax: 0-1226-370-434	
MetaLink	(800) 638-2423 (602) 926-0797 Fax: (602) 693-0681	+49-80 9156 96-0 Fax: +49-80 9123 86	+852-737-1800
Systems General	(408) 263-6667	+41-1-9450300	+886-2-9173005 Fax: +886-2-911-1283
Needhams	(916) 924-8037 Fax: (916) 924-8065		

Development Support (Continued)

AVAILABLE LITERATURE

For more information, please see the COP8 Basic Family User's Manual, Literature Number 620895, COP8 Feature Family User's Manual, Literature Number 620897 and National's Family of 8-bit Microcontrollers COP8 Selection Guide, Literature Number 630009.

DIAL-A-HELPER SERVICE

Dial-A-Helper is a service provided by the Microcontroller Applications group. The Dial-A-Helper is an Electronic Information System that may be accessed as a Bulletin Board System (BBS) via data modem, as an FTP site on the Internet via standard FTP client application or as an FTP site on the Internet using a standard Internet browser such as Netscape or Mosaic.

Information System

The Dial-A-Helper system provides access to an automated information storage and retrieval system. The system capabilities include a MESSAGE SECTION (electronic mail, when accessed as a BBS) for communications to and from the Microcontroller Applications Group and a FILE SECTION which consists of several file areas where valuable application software and utilities could be found.

DIAL-A-HELPER BBS via a Standard Modem

Modem:	CANADA/U.S.:	(800) NSC-MICRO (800) 672-6427
	EUROPE:	(+49) 0-8141-351332
	Baud:	14.4k
	Set-Up:	Length: 8-Bit Parity: None Stop Bit: 1
	Operation:	24 Hours, 7 Days

DIAL-A-HELPER via FTP

ftp nscmicro.nsc.com	
user:	anonymous
password:	username@yourhost.site.domain

DIAL-A-HELPER via a WorldWide Web Browser

ftp://nscmicro.nsc.com

National Semiconductor on the WorldWide Web

See us on the WorldWide Web at: http://www.national.com

CUSTOMER RESPONSE CENTER

Complete product information and technical support is available from National's customer response centers.

CANADA/US:	Tel:	(800) 272-9959
	email:	support @tevm2.nsc.com
EUROPE:	email:	europe.support@nsc.com
	Deutsch Tel:	+49 (0) 180-530 85 85
	English Tel:	+49 (0) 180-532 78 32
	Français Tel:	+49 (0) 180-532 93 58
	Italiano Tel:	+49 (0) 180-534 16 80
JAPAN:	Tel:	+81-043-299-2309
S.E. ASIA:	Beijing Tel:	(+86) 10-6856-8601
	Shanghai Tel:	(+86) 21-6415-4092
	Hong Kong Tel:	(+852) 2737-1600
	Korea Tel:	(+82) 2-3771-6909
	Malaysia Tel:	(+60-4)-644-9061
	Singapore Tel:	(+65) 255-2226
	Taiwan Tel:	+886-2-521-3288
AUSTRALIA:	Tel:	(+61) 3-9558-9999
INDIA:	Tel:	(+91) 80-559-9467

National Semiconductor

PRELIMINARY

COP87L88CF/COP87L84CF 8-Bit One-Time Programmable (OTP) Microcontroller with A/D Converter

General Description

The COP87L88CF/COP87L84CF OTP microcontrollers are members of the COP8™ feature family using an 8-bit core architecture. It is pin and software compatible to the mask ROM COP888CF/COP884CF product family.

(Continued)

Key Features

- Two 16-bit timers, each with two 16-bit registers supporting:
  - — Processor Independent PWM mode
  - — External Event counter mode
  - — Input Capture mode
- 8-channel A/D converter with prescaler and both differential and single ended modes
- 4 kbytes on-board EPROM with security feature
- 128 bytes on-board RAM

Additional Peripheral Features

- Idle Timer
- Multi-Input Wake Up (MIWU) with optional interrupts (8)
- WATCHDOG™ and Clock Monitor logic
- MICROWIRE/PLUS™ serial I/O

I/O Features

- Memory mapped I/O
- Software selectable I/O options
  - — TRI-STATE® Output (push-pull output, weak pull up input, high impedance input)
- Schmitt trigger inputs on ports G and L
- Packages:
  - — 44 PLCC with 38 I/O pins
  - — 40 DIP with 34 I/O pins
  - — 28 DIP with 22 I/O pins
  - — 28 SO with 22 I/O pins (contact local sales office for availability)

CPU/Instruction Set Features

- 1 μs instruction cycle time
- Ten multi-source vectored interrupts servicing
  - — External interrupt
  - — Idle timer T0
  - — Two timers (each with 2 interrupts)
  - — MICROWIRE/PLUS
  - — Multi-Input Wake Up
  - — Software trap
  - — Default VIS (default interrupt)
- Versatile and easy to use instruction set
- 8-bit Stack Pointer SP—stack in RAM
- Two 8-bit Register Indirect Data Memory Pointers (B and X)

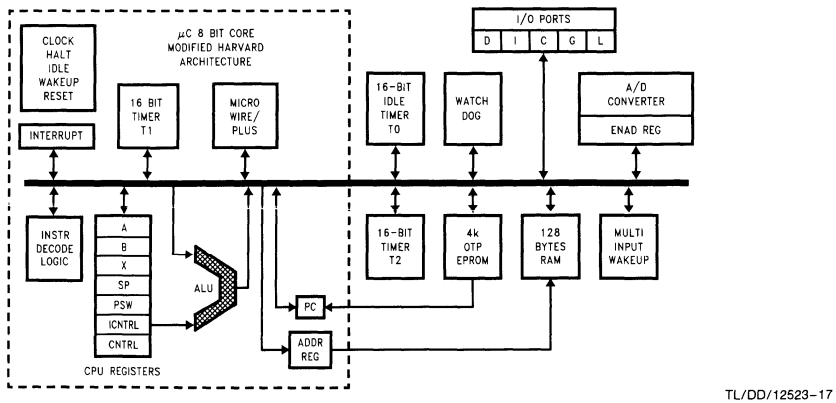
Fully Static CMOS

- Two power saving modes: HALT and IDLE
- Single supply operation: 2.7V–5.5V
- Temperature range: −40°C to +85°C

Development Support

- Emulation device for the COP888CF/COP884CF
- Real time emulation and full program debug offered by MetaLink Development System

Block Diagram



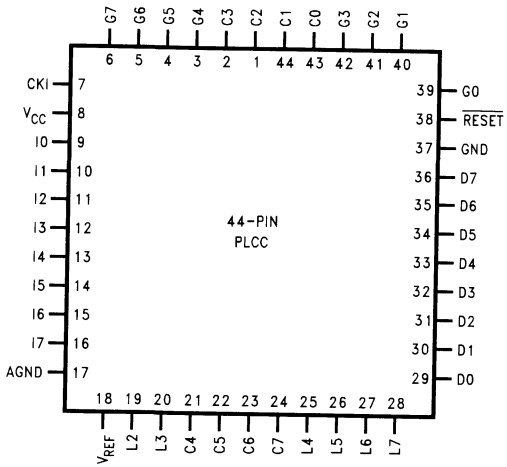
TL/DD/12523–17

General Description (Continued)

The device is a fully static part, fabricated using double-metal silicon gate microCMOS technology. Features include an 8-bit memory mapped architecture, MICROWIRE/PLUS™ serial I/O, two 16-bit timer/counters supporting three modes (Processor Independent PWM generation, External Event counter, and Input Capture mode capabilities), an 8-channel, 8-bit A/D converter with both differential and single ended modes. Each I/O pin has software selectable configurations. The device operates over a voltage range of 2.7V to 5.5V. High throughput is achieved with an efficient, regular instruction set operating at a maximum of 1 μs per instruction rate.

Connection Diagrams

Plastic Chip Carrier



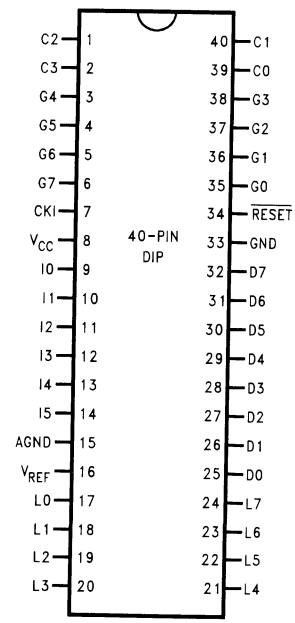
TL/DD/12523-1

Top View

Order Number COP87L88CFV-XE
See NS Package Number V44A

Note: -X Crystal Oscillator
-E Halt Enable

Dual-In-Line Package

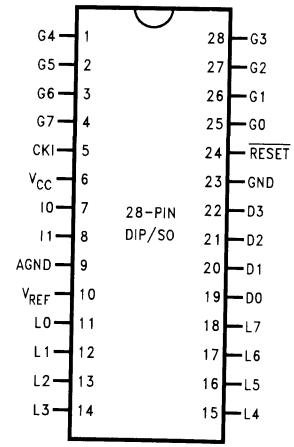


TL/DD/12523-2

Top View

Order Number COP87L84CFN-XE
See NS Package Number N40A

Dual-In-Line Package



TL/DD/12523-3

Top View

Order Number COP87L84CFN-XE,
or COP87L84CFM-XE
See NS Package Number M28B or N28B

FIGURE 1. COP87L88CF/COP87L84CF Connection Diagrams

Connection Diagrams (Continued)

Pinouts for 28-Pin, 40-Pin and 44-Pin Packages

Port	Type	Alt. Fun	Alt. Fun	28-Pin Pkg.	40-Pin Pkg.	44-Pin Pkg.
L0	I/O	MIWU		11	17	
L1	I/O	MIWU		12	18	
L2	I/O	MIWU		13	19	19
L3	I/O	MIWU		14	20	20
L4	I/O	MIWU	T2A	15	21	25
L5	I/O	MIWU	T2B	16	22	26
L6	I/O	MIWU		17	23	27
L7	I/O	MIWU		18	24	28
G0	I/O	INT		25	35	39
G1	WDOUT			26	36	40
G2	I/O	T1B		27	37	41
G3	I/O	T1A		28	38	42
G4	I/O	SO		1	3	3
G5	I/O	SK		2	4	4
G6	I	SI		3	5	5
G7	I/CKO	HALT Restart		4	6	6
D0	O			19	25	29
D1	O			20	26	30
D2	O			21	27	31
D3	O			22	28	32
I0	I	ACH0		7	9	9
I1	I	ACH1		8	10	10
I2	I	ACH2			11	11
I3	I	ACH3			12	12
I4	I	ACH4			13	13
I5	I	ACH5			14	14
I6	I	ACH6				15
I7	I	ACH7				16
D4	O				29	33
D5	O				30	34
D6	O				31	35
D7	O				32	36
C0	I/O				39	43
C1	I/O				40	44
C2	I/O				1	1
C3	I/O				2	2
C4	I/O					21
C5	I/O					22
C6	I/O					23
C7	I/O					24
V_{REF}	$+V_{REF}$			10	16	18
AGND	AGND			9	15	17
V_{CC}				6	8	8
GND				23	33	37
CKI				5	7	7
$\overline{\text{RESET}}$				24	34	38

Absolute Maximum Ratings (Note)

If Military/Aerospace specified devices are required, please contact the National Semiconductor Sales Office/Distributors for availability and specifications.

Supply Voltage (V_{CC})	7V
Voltage at Any Pin	−0.3V to V_{CC} + 0.3V
Total Current into V_{CC} Pin (Source)	100 mA
Total Current out of GND Pin (Sink)	110 mA
Storage Temperature Range	−65°C to +140°C

Note: *Absolute maximum ratings indicate limits beyond which damage to the device may occur. DC and AC electrical specifications are not ensured when operating the device at absolute maximum ratings.*

DC Electrical Characteristics −40°C ≤ T_A ≤ +85°C unless otherwise specified

Parameter	Conditions	Min	Typ	Max	Units
Operating Voltage		2.7		5.5	V
Power Supply Ripple (Note 1)	Peak-to-Peak			0.1 V_{CC}	V
Supply Current (Note 2)					
CKI = 10 MHz	V_{CC} = 5.5V, t_c = 1 μs			16.5	mA
CKI = 4 MHz	V_{CC} = 4.0V, t_c = 2.5 μs			6.5	mA
HALT Current (Note 3)	V_{CC} = 5.5V, CKI = 0 MHz			12	μA
IDLE Current					
CKI = 10 MHz	V_{CC} = 5.5V, t_c = 1 μs			3.5	mA
CKI = 1 MHz	V_{CC} = 4.0V, t_c = 10 μs			0.7	mA
Input Levels					
$\overline{\text{RESET}}$					
Logic High		0.8 V_{CC}			V
Logic Low				0.2 V_{CC}	V
CKI (External and Crystal Osc. Modes)					
Logic High		0.7 V_{CC}			V
Logic Low				0.2 V_{CC}	V
All Other Inputs					
Logic High		0.7 V_{CC}			V
Logic Low				0.2 V_{CC}	V
Hi-Z Input Leakage	V_{CC} = 5.5V	−2		+2	μA
Input Pullup Current	V_{CC} = 5.5V	40		250	μA
G and L Port Input Hysteresis			0.05 V_{CC}	0.35 V_{CC}	V
Output Current Levels					
D Outputs					
Source	V_{CC} = 4.5V, V_{OH} = 3.3V	0.4			mA
Sink (Note 4)	V_{CC} = 4.5V, V_{OL} = 1V	10			mA
All Others					
Source (Weak Pull-Up Mode)	V_{CC} = 4.5V, V_{OH} = 2.7V	10		100	μA
Source (Push-Pull Mode)	V_{CC} = 4.5V, V_{OH} = 3.3V	0.4			mA
Sink (Push-Pull Mode)	V_{CC} = 4.5V, V_{OL} = 0.4V	1.6			mA
TRI-STATE Leakage	V_{CC} = 5.5V	−2		+2	μA
Allowable Sink/Source Current per Pin					
D Outputs (Sink)				15	mA
All Others				3	mA
Maximum Input Current without Latchup (Note 7)	T_A = 25°C			±100	mA
RAM Retention Voltage, V_r	500 ns Rise and Fall Time (Min)	2			V
Input Capacitance				7	pF
Load Capacitance on D2				1000	pF

Note 1: Rate of voltage change must be less then 0.5 V/ms.

Note 2: Supply current is measured after running 2000 cycles with a square wave CKI input, CKO open, inputs at rails and outputs open.

Note 3: The HALT mode will stop CKI from oscillating in the RC and the Crystal configurations by bringing CKI high. Test conditions: All inputs tied to V_{CC}, L and G ports in the TRI-STATE mode and tied to ground, all outputs low and tied to ground. The A/D is disabled. V_{REF} is tied to AGND (effectively shorting the Reference resistor). The clock monitor is disabled.

Note 4: The user must guarantee that D2 pin does not source more than 10 mA during RESET. If D2 sources more than 10 mA during reset, the device will go into programming mode.

A/D Converter Specifications $V_{CC} = 5V \pm 10\%$ $(V_{SS} - 0.050V) \leq$ Any Input $\leq (V_{CC} + 0.050V)$

Parameter	Conditions	Min	Typ	Max	Units
Resolution				8	Bits
Reference Voltage Input	AGND = 0V	3		V_{CC}	V
Absolute Accuracy	$V_{REF} = V_{CC}$			± 2	LSB
Non-Linearity	$V_{REF} = V_{CC}$ Deviation from the Best Straight Line			$\pm \frac{1}{2}$	LSB
Differential Non-Linearity	$V_{REF} = V_{CC}$			$\pm \frac{1}{2}$	LSB
Input Reference Resistance		1.6		4.8	kΩ
Common Mode Input Range (Note 8)		AGND		V_{REF}	V
DC Common Mode Error				$\pm \frac{1}{4}$	LSB
Off Channel Leakage Current			1		μA
On Channel Leakage Current			1		μA
A/D Clock Frequency (Note 6)		0.1		1.67	MHz
Conversion Time (Note 5)			12		A/D Clock Cycles

Note 5: Conversion Time includes sample and hold time.

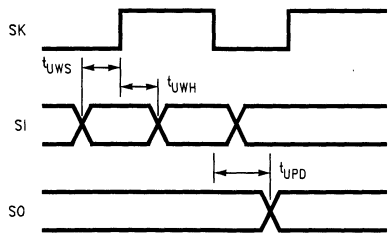
Note 6: See Prescaler description.

Note 7: Pins G6 and $\overline{\text{RESET}}$ are designed with a high voltage input network for factory testing. These pins allow input voltages greater than V_{CC} and the pins will have sink current to V_{CC} when biased at voltages greater than V_{CC} (the pins do not have source current when biased at a voltage below V_{CC}). The effective resistance to V_{CC} is 750Ω (typical). These two pins will not latch up. The voltage at the pins must be limited to less than 14V.

Note 8: For $V_{IN(-)} \geq V_{IN(+)}$, the digital output code will be 0000 0000. Two on-chip diodes are tied to each analog input. The diodes will forward conduct for analog input voltages below ground or above the V_{CC} supply. Be careful, during testing at low V_{CC} levels (4.5V), as high level analog inputs (5V) can cause this input diode to conduct—especially at elevated temperatures. and cause errors for analog inputs near full-scale. The spec allows 50 mV forward bias of either diode. This means that as long as the analog V_{IN} does not exceed the supply voltage by more than 50 mV, the output code will be correct. To achieve an absolute 0 V_{DC} to 5 V_{DC} input voltage range will therefore require a minimum supply voltage of 4.950 V_{DC} over temperature variations, initial tolerance and loading.

AC Electrical Characteristics $-40°C \leq T_A \leq +85°C$ unless otherwise specified

Parameter	Conditions	Min	Typ	Max	Units
Instruction Cycle Time (t_c)					
Crystal, Resonator		1		DC	μs
R/C Oscillator		3		DC	μs
Inputs					
t_{SETUP}		200			ns
t_{HOLD}		60			ns
Output Propagation Delay	$R_L = 2.2k, C_L = 100$ pF				
t_{PD1}, t_{PD0}					
SO, SK	$4V \leq V_{CC} \leq 6V$			0.7	μs
All Others	$4V \leq V_{CC} \leq 6V$			1	μs
MICROWIRE™ Setup Time (t_{UWS})		20			ns
MICROWIRE Hold Time (t_{UWH})		56			ns
MICROWIRE Output Propagation Delay (t_{UPD})				220	ns
Input Pulse Width					
Interrupt Input High Time		1			t_c
Interrupt Input Low Time		1			t_c
Timer Input High Time		1			t_c
Timer Input Low Time		1			t_c
Reset Pulse Width		1			μs



TL/DD/12523-4

FIGURE 2. MICROWIRE/PLUS Timing

Pin Descriptions

V_{CC} and GND are the power supply pins.

V_{REF} and AGND are the reference voltage pins for the on-board A/D converter.

CKI is the clock input. This can come from an R/C generated oscillator, or a crystal oscillator (in conjunction with CKO). See Oscillator Description section.

$\overline{\text{RESET}}$ is the master reset input. See Reset Description section.

The device contains three bidirectional 8-bit I/O ports (C, G and L), where each individual bit may be independently configured as an input (Schmitt trigger inputs on ports G and L), output or TRI-STATE under program control. Three data memory address locations are allocated for each of these I/O ports. Each I/O port has two associated 8-bit memory mapped registers, the CONFIGURATION register and the output DATA register. A memory mapped address is also reserved for the input pins of each I/O port. (See the memory map for the various addresses associated with the I/O ports.) *Figure 3* shows the I/O port configurations. The DATA and CONFIGURATION registers allow for each port bit to be individually configured under software control as shown below:

Configuration Register	Data Register	Port Set-Up
0	0	Hi-Z Input (TRI-STATE Output)
0	1	Input with Weak Pull-Up
1	0	Push-Pull Zero Output
1	1	Push-Pull One Output

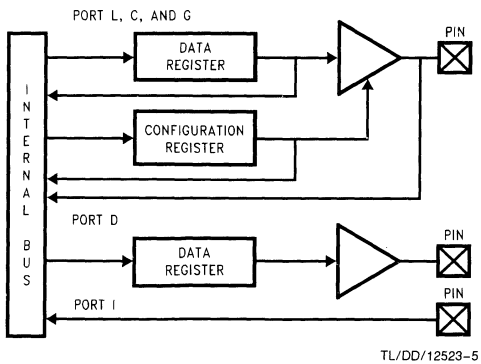


FIGURE 3. I/O Port Configurations

PORT L is an 8-bit I/O port. All L-pins have Schmitt triggers on the inputs.

Port L supports Multi-Input Wakeup (MIWU) on all eight pins. L4 and L5 are used for the timer input functions T2A and T2B. L0 and L1 are not available on the 44-pin version, since they are replaced by V_{REF} and AGND. L0 and L1 are not terminated on the 44-pin version. Consequently, reading L0 or L1 as inputs will return unreliable data with the 44-pin package, so this data should be masked out with user software when the L port is read for input data. It is recommended that the pins be configured as outputs.

Port L has the following alternate features:

- L0 MIWU
- L1 MIWU
- L2 MIWU
- L3 MIWU
- L4 MIWU or T2A
- L5 MIWU or T2B
- L6 MIWU
- L7 MIWU

Port G is an 8-bit port with 5 I/O pins (G0, G2–G5), an input pin (G6), and two dedicated output pins (G1 and G7). Pins G0 and G2–G6 all have Schmitt Triggers on their inputs. Pin G1 serves as the dedicated WDOUT WATCHDOG output, while pin G7 is either input or output depending on the oscillator mask option selected. With the crystal oscillator option selected, G7 serves as the dedicated output pin for the CKO clock output. With the single-pin R/C oscillator mask option selected, G7 serves as a general purpose input pin, but is also used to bring the device out of HALT mode with a low to high transition on G7. There are two registers associated with the G Port, a data register and a configuration register. Therefore, each of the 5 I/O bits (G0, G2–G5) can be individually configured under software control.

Since G6 is an input only pin and G7 is the dedicated CKO clock output pin or general purpose input (R/C clock configuration), the associated bits in the data and configuration registers for G6 and G7 are used for special purpose functions as outlined below. Reading the G6 and G7 data bits will return zeros.

Note that the chip will be placed in the HALT mode by writing a "1" to bit 7 of the Port G Data Register. Similarly the chip will be placed in the IDLE mode by writing a "1" to bit 6 of the Port G Data Register.

Writing a "1" to bit 6 of the Port G Configuration Register enables the MICROWIRE/PLUS to operate with the alternate phase of the SK clock. The G7 configuration bit, if set high, enables the clock start up delay after HALT when the R/C clock configuration is used.

	Config Reg.	Data Reg.
G7	CLKDLY	HALT
G6	Alternate SK	IDLE

Port G has the following alternate features:

- G0 INTR (External Interrupt Input)
- G2 T1B (Timer T1 Capture Input)
- G3 T1A (Timer T1 I/O)
- G4 SO (MICROWIRE Serial Data Output)
- G5 SK (MICROWIRE Serial Clock)
- G6 SI (MICROWIRE Serial Data Input)

Pin Descriptions (Continued)

Port G has the following dedicated functions:

G1 WDOUT WATCHDOG and/or Clock Monitor dedicated output

G7 CKO Oscillator dedicated output or general purpose input

Port C is an 8-bit I/O port. The 40-pin device does not have a full complement of Port C pins. The unavailable pins are not terminated. A read operation for these unterminated pins will return unpredictable values.

Port I is an 8-bit Hi-Z input port, and also provides the analog inputs to the A/D converter. The 28-pin device does not have a full complement of Port I pins. The unavailable pins are not terminated (i.e. they are floating). A read operation from these unterminated pins will return unpredictable values. The user should ensure that the software takes this into account by either masking out these inputs, or else restricting the accesses to bit operations only. If unterminated, Port I pins will draw power only when addressed. The I port leakage current may be higher in 28-pin devices.

Port D is a recreated 8-bit output port that is preset high when RESET goes low. D port recreation is one clock cycle behind the normal port timing. The user can tie two or more D port outputs (except D2 pin) together in order to get a higher drive.

Functional Description

The architecture of the device is modified Harvard architecture. With the Harvard architecture, the control store program memory (ROM) is separated from the data store memory (RAM). Both ROM and RAM have their own separate addressing space with separate address buses. The architecture, though based on Harvard architecture, permits transfer of data from ROM to RAM.

CPU REGISTERS

The CPU can do an 8-bit addition, subtraction, logical or shift operation in one instruction (t_C) cycle time.

There are five CPU registers:

A is the 8-bit Accumulator Register

PC is the 15-bit Program Counter Register

PU is the upper 7 bits of the program counter (PC)

PL is the lower 8 bits of the program counter (PC)

B is an 8-bit RAM address pointer, which can be optionally post auto incremented or decremented.

X is an 8-bit altemate RAM address pointer, which can be optionally post auto incremented or decremented.

SP is the 8-bit stack pointer, which points to the subroutine/interrupt stack (in RAM). The SP is initialized to RAM address 06F with reset.

All the CPU registers are memory mapped with the exception of the Accumulator (A) and the Program Counter (PC).

PROGRAM MEMORY

Program memory consists of 4 kbytes of OTP EPROM. These bytes may hold program instructions or constant data (data tables for the LAID instruction, jump vectors for the JID instruction, and interrupt vectors for the VIS instruction). The program memory is addressed by the 15-bit program counter (PC). All interrupts vector to program memory location 0FF Hex.

The device can be configured to inhibit external reads of the program memory. This is done by programming the Security Byte.

SECURITY FEATURE

The program memory array has an associate Security Byte that is located outside of the program address range. This byte can be addressed only from programming mode by a programmer tool.

Security is an optional feature and can only be asserted after the memory array has been programmed and verified. A secured part will read all 00(hex) by a programmer. The part will fail Blank Check and will fail Verify operations. A Read operation will fill the programmer's memory with 00(hex). The Security Byte itself is always readable with value of 00(hex) if unsecure and FF(hex) if secure.

DATA MEMORY

The data memory address space includes the on-chip RAM and data registers, the I/O registers (Configuration, Data and Pin), the control registers, the MICROWIRE/PLUS SIO shift register, and the various registers, and counters associated with the timers (with the exception of the IDLE timer). Data memory is addressed directly by the instruction or indirectly by the B, X and SP pointers.

The device has 128 bytes of RAM. Sixteen bytes of RAM are mapped as "registers" at addresses 0F0 to 0FF Hex. These registers can be loaded immediately, and also decremented and tested with the DRSZ (decrement register and skip if zero) instruction. The memory pointer registers X, SP, and B are memory mapped into this space at address locations 0FC to 0FE Hex respectively, with the other registers (other than reserved register 0FF) being available for general usage.

The instruction set permits any bit in memory to be set, reset or tested. All I/O and registers (except A and PC) are memory mapped; therefore, I/O bits and register bits can be directly and individually set, reset and tested. The accumulator (A) bits can also be directly and individually tested.

Reset

The $\overline{\text{RESET}}$ input when pulled low initializes the microcontroller. Initialization will occur whenever the $\overline{\text{RESET}}$ input is pulled low. Upon initialization, the data and configuration registers for Ports L, G, and C are cleared, resulting in these Ports being initialized to the TRI-STATE mode. Pin G1 of the G Port is an exception (as noted below) since pin G1 is dedicated as the WATCHDOG and/or Clock Monitor error output pin. Port D is initialized high with $\overline{\text{RESET}}$. The PC, PSW, CNTRL, ICNTRL, and T2CNTRL control registers are cleared. The Multi-Input Wakeup registers WKEN, WKEDG, and WKPND are cleared. The A/D control register ENAD is cleared, resulting in the ADC being powered down initially. The Stack Pointer, SP, is initialized to 06F Hex.

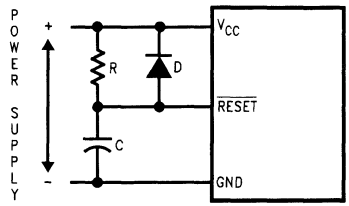
The device comes out of reset with both the WATCHDOG logic and the Clock Monitor detector armed, and with both the WATCHDOG service window bits set and the Clock Monitor bit set. The WATCHDOG and Clock Monitor detector circuits are inhibited during reset. The WATCHDOG service window bits are initialized to the maximum WATCHDOG service window of 64k t_C clock cycles. The Clock Monitor bit

Reset (Continued)

is initialized high, and will cause a Clock Monitor error following reset if the clock has not reached the minimum specified frequency at the termination of reset. A Clock Monitor error will cause an active low error output on pin G1. This error output will continue until 16 t_c–32 t_c clock cycles following the clock frequency reaching the minimum specified value, at which time the G1 output will enter the TRI-STATE mode.

The external RC network shown in *Figure 4* should be used to ensure that the $\overline{\text{RESET}}$ pin is held low until the power supply to the chip stabilizes.

Note: In continued state of reset, the device will draw excessive current.



TL/DD/12523–6

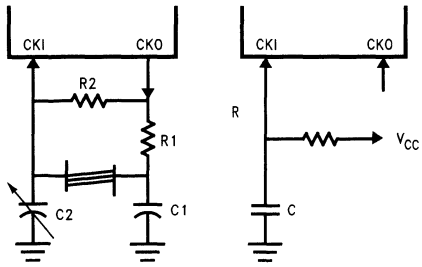
RC > 5 × Power Supply Rise Time

FIGURE 4. Recommended Reset Circuit

Oscillator Circuits

The chip can be driven by a clock input on the CKI input pin which can be between DC and 10 MHz. The CKO output clock is on pin G7 (crystal configuration). The CKI input frequency is divided down by 10 to produce the instruction cycle clock ($1/t_c$).

Figure 5 shows the Crystal and R/C diagrams.



TL/DD/12523–7

FIGURE 5. Crystal and R/C Oscillator Diagrams

CRYSTAL OSCILLATOR

CKI and CKO can be connected to make a closed loop crystal (or resonator) controlled oscillator.

Table I shows the component values required for various standard crystal values.

TABLE I. Crystal Oscillator Configuration, T_A = 25°C

R1 (kΩ)	R2 (MΩ)	C1 (pF)	C2 (pF)	CKI Freq (MHz)	Conditions
0	1	30	30–36	10	V_{CC} = 5V
0	1	30	30–36	4	V_{CC} = 5V
0	1	200	100–150	0.455	V_{CC} = 5V

R/C OSCILLATOR

By selecting CKI as a single pin oscillator input, a single pin R/C oscillator circuit can be connected to it. CKO is available as a general purpose input, and/or HALT restart pin.

Table II shows the variation in the oscillator frequencies as functions of the component (R and C) values.

TABLE II. R/C Oscillator Configuration, T_A = 25°C

R (kΩ)	C (pF)	CKI Freq (MHz)	Instr. Cycle (μs)	Conditions
3.3	82	2.2 to 2.7	3.7 to 4.6	V_{CC} = 5V
5.6	100	1.1 to 1.3	7.4 to 9.0	V_{CC} = 5V
6.8	100	0.9 to 1.1	8.8 to 10.8	V_{CC} = 5V

Note: 3k ≤ R ≤ 200k

50 pF ≤ C ≤ 200 pF

Control Registers

CNTRL REGISTER (ADDRESS X'00EE)

The Timer1 (T1) and MICROWIRE/PLUS control register contains the following bits:

SL1 & SL0 Select the MICROWIRE/PLUS clock divide by (00 = 2, 01 = 4, 1x = 8)

IEDG External interrupt edge polarity select (0 = Rising edge, 1 = Falling edge)

MSEL Selects G5 and G4 as MICROWIRE/PLUS signals SK and SO respectively

T1C0 Timer T1 Start/Stop control in timer

Timer T1 Underflow Interrupt Pending Flag in timer mode 3

T1C1 Timer T1 mode control bit

T1C2 Timer T1 mode control bit

T1C3 Timer T1 mode control bit

T1C3	T1C2	T1C1	T1C0	MSEL	IEDG	SL1	SL0

Bit 7 Bit 0

PSW REGISTER (ADDRESS X'00EF)

The PSW register contains the following select bits:

GIE Global interrupt enable (enables interrupts)

EXEN Enable external interrupt

BUSY MICROWIRE/PLUS busy shifting flag

EXPND External interrupt pending

T1ENA Timer T1 Interrupt Enable for Timer Underflow or T1A Input capture edge

T1PNDA Timer T1 Interrupt Pending Flag (Autoreload RA in mode 1, T1 Underflow in Mode 2, T1A capture edge in mode 3)

C Carry Flag

HC Half Carry Flag

HC	C	T1PNDA	T1ENA	EXPND	BUSY	EXEN	GIE

Bit 7 Bit 0

The Half-Carry bit is also affected by all the instructions that affect the Carry flag. The SC (Set Carry) and RC (Reset Carry) instructions will respectively set or clear both the carry flags. In addition to the SC and RC instructions, ADC, SUBC, RRC and RLC instructions affect the Carry and Half Carry flags.

ICNTRL REGISTER (ADDRESS X'00E8)

The ICNTRL register contains the following bits:

T1ENB Timer T1 Interrupt Enable for T1B Input capture edge

T1PNDB Timer T1 Interrupt Pending Flag for T1B capture edge

WEN Enable MICROWIRE/PLUS interrupt

WPND MICROWIRE/PLUS interrupt pending

T0EN Timer T0 Interrupt Enable (Bit 12 toggle)

T0PND Timer T0 Interrupt pending

LPENL Port Interrupt Enable (Multi-Input Wakeup/ Interrupt)

Bit 7 could be used as a flag

T2CNTRL Register (Address X'00C6)

Unused	LPEN	T0PND	T0EN	WPND	WEN	T1PNDB	T1ENB

Bit 7 Bit 0

The T2CNTRL register contains the following bits:

T2ENB Timer T2 Interrupt Enable for T2B Input capture edge

T2PNDB Timer T2 Interrupt Pending Flag for T2B capture edge

T2ENA Timer T2 Interrupt Enable for Timer Underflow or T2A Input capture edge

T2PNDA Timer T2 Interrupt Pending Flag (Autoreload RA in mode 1, T2 Underflow in mode 2, T2A capture edge in mode 3)

T2C0 Timer T2 Start/Stop control in timer modes 1 and 2 Timer T2 Underflow Interrupt Pending Flag in timer mode 3

T2C1 Timer T2 mode control bit

T2C2 Timer T2 mode control bit

T2C3 Timer T2 mode control bit

T2C3	T2C2	T2C1	T2C0	T2PNDA	T2ENA	T2PNDB	T2ENB

Bit 7 Bit 0

Timers

The device contains a very versatile set of timers (T0, T1, T2). All timers and associated autoreload/capture registers power up containing random data.

TIMER T0 (IDLE TIMER)

The device supports applications that require maintaining real time and low power with the IDLE mode. This IDLE mode support is furnished by the IDLE timer T0, which is a 16-bit timer. The Timer T0 runs continuously at the fixed rate of the instruction cycle clock, t_C. The user cannot read or write to the IDLE Timer T0, which is a count down timer.

The Timer T0 supports the following functions:

Exit out of the Idle Mode (See Idle Mode description)

WATCHDOG logic (See WATCHDOG description)

Start up delay out of the HALT mode

The IDLE Timer T0 can generate an interrupt when the thirteenth bit toggles. This toggle is latched into the T0PND pending flag, and will occur every 4 ms at the maximum clock frequency (t_C = 1s). A control flag T0EN allows the interrupt from the thirteenth bit of Timer T0 to be enabled or disabled. Setting T0EN will enable the interrupt, while resetting it will disable the interrupt.

TIMER T1 AND TIMER T2

The device has a set of two powerful timer/counter blocks, T1 and T2. The associated features and functioning of a timer block are described by referring to the timer block Tx. Since the two timer blocks, T1 and T2, are identical, all comments are equally applicable to either timer block.

Each timer block consists of a 16-bit timer, Tx, and two supporting 16-bit autoreload/capture registers, RxA and RxB. Each timer block has two pins associated with it, TxA and TxB. The pin TxA supports I/O required by the timer block, while the pin TxB is an input to the timer block. The powerful and flexible timer block allows the device to easily perform all timer functions with minimal software overhead. The timer block has three operating modes: Processor Independent PWM mode, External Event Counter mode, and Input Capture mode.

The control bits TxC3, TxC2, and TxC1 allow selection of the different modes of operation.

Mode 1. Processor Independent PWM Mode

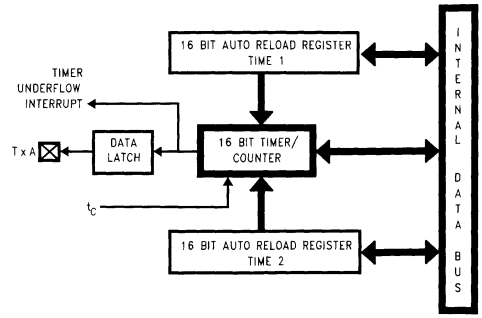
As the name suggests, this mode allows the device to generate a PWM signal with very minimal user intervention.

The user only has to define the parameters of the PWM signal (ON time and OFF time). Once begun, the timer block will continuously generate the PWM signal completely independent of the microcontroller. The user software services the timer block only when the PWM parameters require updating.

In this mode the timer Tx counts down at a fixed rate of t_C. Upon every underflow the timer is alternately reloaded with the contents of supporting registers, RxA and RxB. The very first underflow of the timer causes the timer to reload from the register RxA. Subsequent underflows cause the timer to be reloaded from the registers alternately beginning with the register RxB.

The Tx Timer control bits, TxC3, TxC2 and TxC1 set up the timer for PWM mode operation.

Figure 6 shows a block diagram of the timer in PWM mode.



TL/DD/12523–9

FIGURE 6. Timer in PWM Mode

The underflows can be programmed to toggle the TxA output pin. The underflows can also be programmed to generate interrupts.

Underflows from the timer are alternately latched into two pending flags, TxPNDA and TxPNDB. The user must reset these pending flags under software control. Two control enable flags, TxENA and TxENB, allow the interrupts from the timer underflow to be enabled or disabled. Setting the timer enable flag TxENA will cause an interrupt when a timer underflow causes the RxA register to be reloaded into the timer. Setting the timer enable flag TxENB will cause an interrupt when a timer underflow causes the RxB register to be reloaded into the timer. Resetting the timer enable flags will disable the associated interrupts.

Either or both of the timer underflow interrupts may be enabled. This gives the user the flexibility of interrupting once per PWM period on either the rising or falling edge of the PWM output. Alternatively, the user may choose to interrupt on both edges of the PWM output.

Mode 2. External Event Counter Mode

This mode is quite similar to the processor independent PWM mode described above. The main difference is that the timer, Tx, is clocked by the input signal from the TxA pin. The Tx timer control bits, TxC3, TxC2 and TxC1 allow the timer to be clocked either on a positive or negative edge from the TxA pin. Underflows from the timer are latched into the TxPNDA pending flag. Setting the TxENA control flag will cause an interrupt when the timer underflows.

In this mode the input pin TxB can be used as an independent positive edge sensitive interrupt input if the TxENB control flag is set. The occurrence of a positive edge on the TxB input pin is latched into the TxPNDB flag.

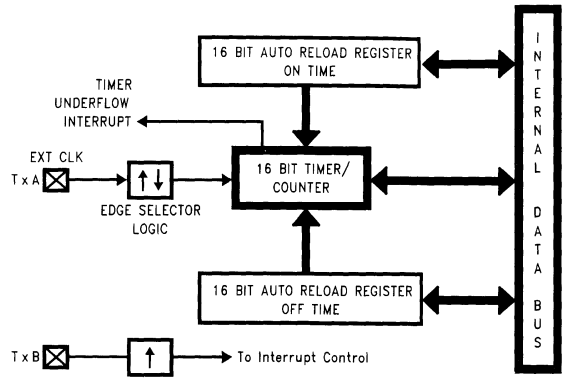
Figure 7 shows a block diagram of the timer in External Event Counter mode.

Note: The PWM output is not available in this mode since the TxA pin is being used as the counter input clock.

Mode 3. Input Capture Mode

The device can precisely measure external frequencies or time external events by placing the timer block, Tx, in the input capture mode.

Timers (Continued)



TL/DD/12523–10

FIGURE 7. Timer in External Event Counter Mode

In this mode, the timer Tx is constantly running at the fixed t_C rate. The two registers, RxA and RxB, act as capture registers. Each register acts in conjunction with a pin. The register RxA acts in conjunction with the TxA pin and the register RxB acts in conjunction with the TxB pin.

The timer value gets copied over into the register when a trigger event occurs on its corresponding pin. Control bits, TxC3, TxC2 and TxC1, allow the trigger events to be specified either as a positive or a negative edge. The trigger condition for each input pin can be specified independently.

The trigger conditions can also be programmed to generate interrupts. The occurrence of the specified trigger condition on the TxA and TxB pins will be respectively latched into the pending flags, TxPNDA and TxPNDB. The control flag TxENA allows the interrupt on TxA to be either enabled or disabled. Setting the TxENA flag enables interrupts to be generated when the selected trigger condition occurs on the TxA pin. Similarly, the flag TxENB controls the interrupts from the TxB pin.

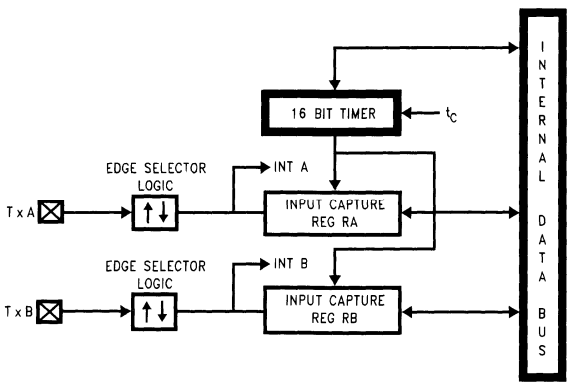
Underflows from the timer can also be programmed to generate interrupts. Underflows are latched into the timer TxC0 pending flag (the TxC0 control bit serves as the timer underflow interrupt pending flag in the Input Capture mode). Consequently, the TxC0 control bit should be reset when entering the Input Capture mode. The timer underflow interrupt is enabled with the TxENA control flag. When a TxA interrupt occurs in the Input Capture mode, the user must check both whether a TxA input capture or a timer underflow (or both) caused the interrupt.

Figure 8 shows a block diagram of the timer in Input Capture mode.

TIMER CONTROL FLAGS

The timers T1 and T2 have indentical control structures. The control bits and their functions are summarized below.

TxC0	Timer Start/Stop control in Modes 1 and 2 (Processor Independent PWM and External Event Counter), where 1 = Start, 0 = Stop Timer Underflow Interrupt Pending Flag in Mode 3 (Input Capture)
TxPNDA	Timer Interrupt Pending Flag
TxPNDB	Timer Interrupt Pending Flag
TxENA	Timer Interrupt Enable Flag
TxENB	Timer Interrupt Enable Flag 1 = Timer Interrupt Enabled 0 = Timer Interrupt Disabled
TxC3	Timer mode control
TxC2	Timer mode control
TxC1	Timer mode control



TL/DD/12523–11

FIGURE 8. Timer in Input Capture Mode

Timers (Continued)

The timer mode control bits (TxC3, TxC2 and TxC1) are detailed below.

TxC3	TxC2	TxC1	Timer Mode	Interrupt A Source	Interrupt B Source	Timer Counts On
0	0	0	MODE 2 (External Event Counter)	Timer Underflow	Pos. TxB Edge	TxA Pos. Edge
0	0	1	MODE 2 (External Event Counter)	Timer Underflow	Pos. TxB Edge	TxA Neg. Edge
1	0	1	MODE 1 (PWM) TxA Toggle	Autoreload RA	Autoreload RB	t_C
1	0	0	MODE 1 (PWM) No TxA Toggle	Autoreload RA	Autoreload RB	t_C
0	1	0	MODE 3 (Capture) Captures: TxA Pos. Edge TxB Pos. Edge	Pos. TxA Edge or Timer Underflow	Pos. TxB Edge	t_C
1	1	0	MODE 3 (Capture) Captures: TxA Pos. Edge TxB Neg. Edge	Pos. TxA Edge or Timer Underflow	Neg. TxB Edge	t_C
0	1	1	MODE 3 (Capture) Captures: TxA Neg. Edge TxB Pos. Edge	Neg. TxB Edge or Timer Underflow	Pos. TxB Edge	t_C
1	1	1	MODE 3 (Capture) Captures: TxA Neg. Edge TxB Neg. Edge	Neg. TxA Edge or Timer Underflow	Neg. TxB Edge	t_C

Power Save Modes

The device offers the user two power save modes of operation: HALT and IDLE. In the HALT mode, all microcontroller activities are stopped. In the IDLE mode, the on-board oscillator circuitry and timer T0 are active but all other microcontroller activities are stopped. In either mode, all on-board RAM, registers, I/O states, and timers (with the exception of T0) are unaltered.

HALT MODE

The device is placed in the HALT mode by writing a "1" to the HALT flag (G7 data bit). All microcontroller activities, including the clock, timers, and A/D converter, are stopped. The WATCHDOG logic is disabled during the HALT mode. However, the clock monitor circuitry if enabled remains active and will cause the WATCHDOG output pin (WDOUT) to go low. If the HALT mode is used and the user does not want to activate the WDOUT pin, the Clock Monitor should be disabled after the device comes out of reset (resetting the Clock Monitor control bit with the first write to the WDSVR register). In the HALT mode, the power requirements of the device are minimal and the applied voltage (V_{CC}) may be decreased to V_r (V_r = 2.0V) without altering the state of the machine.

The device supports three different ways of exiting the HALT mode. The first method of exiting the HALT mode is with the Multi-Input Wakeup feature on the L port. The second method is with a low to high transition on the CKO (G7) pin. This method precludes the use of the crystal clock configuration (since CKO becomes a dedicated output), and so may be used with an RC clock configuration. The third method of exiting the HALT mode is by pulling the RESET pin low.

Since a crystal or ceramic resonator may be selected as the oscillator, the Wakeup signal is not allowed to start the chip running immediately since crystal oscillators and ceramic resonators have a delayed start up time to reach full amplitude and frequency stability. The IDLE timer is used to generate a fixed delay to ensure that the oscillator has indeed stabilized before allowing instruction execution. In this case, upon detecting a valid Wakeup signal, only the oscillator circuitry is enabled. The IDLE timer is loaded with a value of 256 and is clocked with the t_C instruction cycle clock. The t_C clock is derived by dividing the oscillator clock down by a factor of 10. The Schmitt trigger following the CKI inverter on the chip ensures that the IDLE timer is clocked only when the oscillator has a sufficiently large amplitude to meet the Schmitt trigger specifications. This Schmitt trigger is not part of the oscillator closed loop. The startup timeout from the IDLE timer enables the clock signals to be routed to the rest of the chip.

If an RC clock option is being used, the fixed delay is introduced optionally. A control bit, CLKDLY, mapped as configuration bit G7, controls whether the delay is to be introduced or not. The delay is included if CLKDLY is set, and excluded if CLKDLY is reset. The CLKDLY bit is cleared on reset.

The WATCHDOG detector circuit is inhibited during the HALT mode. However, the clock monitor circuit if enabled remains active during HALT mode in order to ensure a clock monitor error if the device inadvertently enters the HALT mode as a result of a runaway program or power glitch.

IDLE MODE

The device is placed in the IDLE mode by writing a "1" to the IDLE flag (G6 data bit). In this mode, all activity, except

Power Save Modes (Continued)

the associated on-board oscillator circuitry, the WATCHDOG logic, the clock monitor and the IDLE Timer T0, is stopped.

As with the HALT mode, the device can be returned to normal operation with a reset, or with a Multi-Input Wake Up from the L Port. Alternately, the microcontroller resumes normal operation from the IDLE mode when the thirteenth bit (representing 4.096 ms at internal clock frequency of 1 MHz, $t_C = 1\ \mu s$) of the IDLE Timer toggles.

This toggle condition of the thirteenth bit of the IDLE Timer T0 is latched into the T0PND pending flag.

The user has the option of being interrupted with a transition on the thirteenth bit of the IDLE Timer T0. The interrupt can be enabled or disabled via the T0EN control bit. Setting the T0EN flag enables the interrupt and vice versa.

The user can enter the IDLE mode with the Timer T0 interrupt enabled. In this case, when the T0PND bit gets set, the device will first execute the Timer T0 interrupt service routine and then return to the instruction following the "Enter Idle Mode" instruction.

Alternatively, the user can enter the IDLE mode with the IDLE Timer T0 interrupt disabled. In this case, the device will resume normal operation with the instruction immediately following the "Enter IDLE Mode" instruction.

Note: It is necessary to program two NOP instructions following both the set HALT mode and set IDLE mode instructions. These NOP instructions are necessary to allow clock resynchronization following the HALT or IDLE modes.

Due to the onboard 8k EPROM with port recreation logic, the HALT/IDLE current is much higher compared to the equivalent masked device.

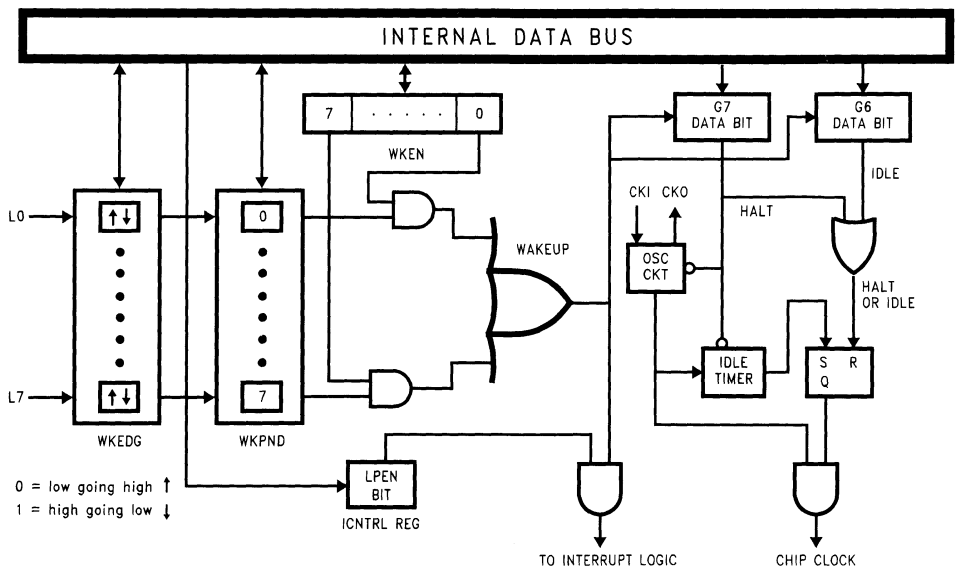
Multi-Input Wake Up

The Multi-Input Wake Up feature is used to return (Wake Up) the device from either the HALT or IDLE modes. Alternately Multi-Input Wake Up/Interrupt feature may also be used to generate up to 8 edge selectable external interrupts.

Figure 9 shows the Multi-Input Wake Up logic.

The Multi-Input Wakeup feature utilizes the L Port. The user selects which particular L port bit (or combination of L Port bits) will cause the device to exit the HALT or IDLE modes. The selection is done through the Reg: WKEN. The Reg: WKEN is an 8-bit read/write register, which contains a control bit for every L port bit. Setting a particular WKEN bit enables a Wake Up from the associated L port pin.

The user can select whether the trigger condition on the selected L Port pin is going to be either a positive edge (low to high transition) or a negative edge (high to low transition). This selection is made via the Reg: WKEDG, which is an 8-bit control register with a bit assigned to each L Port pin. Setting the control bit will select the trigger condition to be a negative edge on that particular L Port pin. Resetting the bit selects the trigger condition to be a positive edge. Changing an edge select entails several steps in order to avoid a pseudo Wake Up condition as a result of the edge change. First, the associated WKEN bit should be reset, followed by the edge select change in WKEDG. Next, the associated WKPND bit should be cleared, followed by the associated WKEN bit being re-enabled.



TL/DD/12523–12

FIGURE 9. Multi-Input Wake Up Logic

Multi-Input Wake Up (Continued)

An example may serve to clarify this procedure. Suppose we wish to change the edge select from positive (low going high) to negative (high going low) for L Port bit 5, where bit 5 has previously been enabled for an input interrupt. The program would be as follows:

```
RBIT 5, WKEN
SBIT 5, WKEDG
RBIT 5, WKPND
SBIT 5, WKEN
```

If the L port bits have been used as outputs and then changed to inputs with Multi-Input Wake Up/Interrupt, a safety procedure should also be followed to avoid inherited pseudo Wake Up conditions. After the selected L port bits have been changed from output to input but before the associated WKEN bits are enabled, the associated edge select bits in WKEDG should be set or reset for the desired edge selects, followed by the associated WKPND bits being cleared.

This same procedure should be used following reset, since the L port inputs are left floating as a result of reset.

The occurrence of the selected trigger condition for Multi-Input Wake Up is latched into a pending register called WKPND. The respective bits of the WKPND register will be set on the occurrence of the selected trigger edge on the corresponding Port L pin. The user has the responsibility of clearing these pending flags. Since WKPND is a pending register for the occurrence of selected wakeup conditions, the device will not enter the HALT mode if any Wake Up bit is both enabled and pending. Consequently, the user has the responsibility of clearing the pending flags before attempting to enter the HALT mode.

The WKEN, WKPND and WKEDG are all read/write registers, and are cleared at reset.

PORT L INTERRUPTS

Port L provides the user with an additional eight fully selectable, edge sensitive interrupts which are all vectored into the same service subroutine.

The interrupt from Port L shares logic with the Wake Up circuitry. The register WKEN allows interrupts from Port L to be individually enabled or disabled. The register WKEDG specifies the trigger condition to be either a positive or a negative edge. Finally, the register WKPND latches in the pending trigger conditions.

The GIE (Global Interrupt Enable) bit enables the interrupt function. A control flag, LPEN, functions as a global interrupt enable for Port L interrupts. Setting the LPEN flag will enable interrupts and vice versa. A separate global pending flag is not needed since the register WKPND is adequate.

Since Port L is also used for waking the device out of the HALT or IDLE modes, the user can elect to exit the HALT or IDLE modes either with or without the interrupt enabled. If he elects to disable the interrupt, then the device will restart execution from the instruction immediately following the instruction that placed the microcontroller in the HALT or IDLE modes. In the other case, the device will first execute the interrupt service routine and then revert to normal operation.

The Wake Up signal will not start the chip running immediately since crystal oscillators or ceramic resonators have a finite start up time. The IDLE Timer (T0) generates a fixed delay to ensure that the oscillator has indeed stabilized before allowing the device 40 execute instructions. In this case, upon detecting a valid Wake Up signal, only the oscillator circuitry and the IDLE Timer T0 are enabled. The IDLE Timer is loaded with a value of 256 and is clocked from the t_C instruction cycle clock. The t_C clock is derived by dividing down the oscillator clock by a factor of 10. A Schmitt trigger following the CKI on-chip inverter ensures that the IDLE timer is clocked only when the oscillator has a sufficiently large amplitude to meet the Schmitt trigger specifications. This Schmitt trigger is not part of the oscillator closed loop. The startup timeout from the IDLE timer enables the clock signals to be routed to the rest of the chip.

If the RC clock option is used, the fixed delay is under software control. A control flag, CLKDLY, in the G7 configuration bit allows the clock start up delay to be optionally inserted. Setting CLKDLY flag high will cause clock start up delay to be inserted and resetting it will exclude the clock start up delay. The CLKDLY flag is cleared during reset, so the clock start up delay is not present following reset with the RC clock options.

A/D Converter

The device contains an 8-channel, multiplexed input, successive approximation, A/D converter. Two dedicated pins, V_{REF} and AGND are provided for voltage reference.

OPERATING MODES

The A/D converter supports ratiometric measurements. It supports both Single Ended and Differential modes of operation.

Four specific analog channel selection modes are supported. These are as follows:

Allow any specific channel to be selected at one time. The A/D converter performs the specific conversion requested and stops.

Allow any specific channel to be scanned continuously. In other words, the user will specify the channel and the A/D converter will keep on scanning it continuously. The user can come in at any arbitrary time and immediately read the result of the last conversion. The user does not have to wait for the current conversion to be completed.

Allow any differential channel pair to be selected at one time. The A/D converter performs the specific differential conversion requested and stops.

Allow any differential channel pair to be scanned continuously. In other words, the user will specify the differential channel pair and the A/D converter will keep on scanning it continuously. The user can come in at any arbitrary time and immediately read the result of the last differential conversion. The user does not have to wait for the current conversion to be completed.

The A/D converter is supported by two memory mapped registers, the result register and the mode control register. When the device is reset, the control register is cleared and the A/D is powered down. The A/D result register has unknown data following reset.

A/D Converter (Continued)

A/D Control Register

A control register, Reg: ENAD, contains 3 bits for channel selection, 3 bits for prescaler selection, and 2 bits for mode selection. An A/D conversion is initiated by writing to the ENAD control register. The result of the conversion is available to the user from the A/D result register, Reg: ADRSLT.

Reg: ENAD

Channel Select	Mode Select	Prescaler Select
Bits 7, 6, 5	Bits 4, 3	Bits 2, 1, 0

CHANNEL SELECT

This 3-bit field selects one of eight channels to be the V_{IN+}. The mode selection determines the V_{IN-} input.

Single Ended mode:

Bit 7	Bit 6	Bit 5	Channel No.
0	0	0	0
0	0	1	1
0	1	0	2
0	1	1	3
1	0	0	4
1	0	1	5
1	1	0	6
1	1	1	7

Differential mode:

Bit 7	Bit 6	Bit 5	Channel Pairs (+, −)
0	0	0	0, 1
0	0	1	1, 0
0	1	0	2, 3
0	1	1	3, 2
1	0	0	4, 5
1	0	1	5, 4
1	1	0	6, 7
1	1	1	7, 6

MODE SELECT

This 2-bit field is used to select the mode of operation (single conversion, continuous conversions, differential, single ended) as shown in the following table.

Bit 4	Bit 3	Mode
0	0	Single Ended mode, single conversion
0	1	Single Ended mode, continuous scan of a single channel into the result register
1	0	Differential mode, single conversion
1	1	Differential mode, continuous scan of a channel pair into the result register

PRESCALER SELECT

This 3-bit field is used to select one of the seven prescaler clocks for the A/D converter. The prescaler also allows the A/D clock inhibit power saving mode to be selected. The following table shows the various prescaler options.

Bit 2	Bit 1	Bit 0	Clock Select
0	0	0	Inhibit A/D clock
0	0	1	Divide by 1
0	1	0	Divide by 2
0	1	1	Divide by 4
1	0	0	Divide by 6
1	0	1	Divide by 12
1	1	0	Divide by 8
1	1	1	Divide by 16

ADC Operation

The A/D converter interface works as follows. Writing to the A/D control register ENAD initiates an A/D conversion unless the prescaler value is set to 0, in which case the ADC clock is stopped and the ADC is powered down. The conversion sequence starts at the beginning of the write to ENAD operation powering up the ADC. At the first falling edge of the converter clock following the write operation (not counting the falling edge if it occurs at the same time as the write operation ends), the sample signal turns on for two clock cycles. The ADC is selected in the middle of the sample period. If the ADC is in single conversion mode, the conversion complete signal from the ADC will generate a power down for the A/D converter. If the ADC is in continuous mode, the conversion complete signal will restart the conversion sequence by deselecting the ADC for one converter clock cycle before starting the next sample. The ADC 8-bit result is loaded into the A/D result register (ADRSLT) except during LOAD clock high, which prevents transient data (resulting from the ADC writing a new result over an old one) being read from ADRSLT.

Inadvertant changes to the ENAD register during conversion are prevented by the control logic of the A/D. Any attempt to write any bit of the ENAD Register except ADBSY, while ADBSY is a one, is ignored. ADBSY must be cleared either by completion of an A/D conversion or by the user before the prescaler, conversion mode or channel select values can be changed. After stopping the current conversion, the user can load different values for the prescaler, conversion mode or channel select and start a new conversion in one instruction.

It is important for the user to realize that, when used in differential mode, only the positive input to the A/D converter is sampled and held. The negative input is constantly connected and should be held stable for the duration of the conversion. Failure to maintain a stable negative input will result in incorrect conversion.

PRESCALER

The A/D Converter (ADC) contains a prescaler option which allows seven different clock selections. The A/D clock frequency is equal to CKI divided by the prescaler value. Note that the prescaler value must be chosen such that the A/D clock falls within the specified range. The maximum A/D frequency is 1.67 MHz. This equates to a 600 ns ADC clock cycle.

The A/D converter takes 12 ADC clock cycles to complete a conversion. Thus the minimum ADC conversion time is 7.2 μs when a prescaler of 6 has been selected. These 12 ADC clock cycles necessary for a conversion consist of 1 cycle at the beginning for reset, 2 cycles for sampling, 8 cycles for converting, and 1 cycle for loading the result into the A/D result register (ADRSLT). This A/D result register is a read-only register. The user cannot write into ADRSLT.

A/D Converter (Continued)

The prescaler also allows an A/D clock inhibit option, which saves power by powering down the A/D when it is not in use.

Note: The A/D converter is also powered down when the device is in either the HALT or IDLE modes. If the ADC is running when the device enters the HALT or IDLE modes, the ADC will power down during the HALT or IDLE, and then will reinitialize the conversion when the device comes out of the HALT or IDLE modes.

Analog Input and Source Resistance Considerations

Figure 10 shows the A/D pin model in single-ended mode. The differential mode has a similiar A/D pin model. The leads to the analog inputs should be kept as short as possible. Both noise and digital clock coupling to an A/D input can cause conversion errors. The clock lead should be kept away from the analog input line to reduce coupling. The A/D channel input pins do not have any internal output driver circuitry connected to them because this circuitry would load the analog input signals due to output buffer leakage current.

Source impedances greater than 1 kΩ on the analog input lines will adversely affect internal RC charging time during input sampling. As shown in *Figure 10,* the analog switch to the DAC array is closed only during the 2 A/D cycle sample time. Large source impedances on the analog inputs may result in the DAC array not being charged to the correct voltage levels, causing scale errors.

If large source resistance is necessary, the recommended solution is to slow down the A/D clock speed in proportion to the source resistance. The A/D converter may be operated at the maximum speed for R_S less than 1 kΩ. For R_S greater than 1 kΩ, A/D clock speed needs to be reduced.

For example, with R_S = 2 kΩ, the A/D converter may be operated at half the maximum speed. A/D converter clock speed may be slowed down by either increasing the A/D prescaler divide-by or decreasing the CKI clock frequency. The A/D clock speed may be reduced to its minimum frequency of 100 kHz.

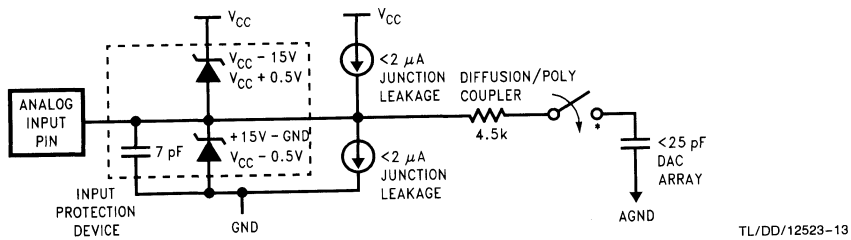
Interrupts

The device supports a vectored interrupt scheme. It supports a total of ten interrupt sources. The following table lists all the possible interrupt sources, their arbitration ranking and the memory locations reserved for the interrupt vector for each source.

Two bytes of program memory space are reserved for each interrupt source. All interrupt sources except the software interrupt are maskable. Each of the maskable interrupts have an Enable bit and a Pending bit. A maskable interrupt is active if its associated enable and pending bits are set. If GIE = 1 and an interrupt is active, then the processor will be interrupted as soon as it is ready to start executing an instruction except if the above conditions happen during the Software Trap service routine. This exception is described in the Software Trap sub-section.

The interruption process is accomplished with the INTR instruction (opcode 00), which is jammed inside the Instruction Register and replaces the opcode about to be executed. The following steps are performed for every interrupt:

1. The GIE (Global Interrupt Enable) bit is reset.
2. The address of the instruction about to be executed is pushed into the stack.
3. The PC (Program Counter) branches to address 00FF. This procedure takes 7 t_c cycles to execute.



*The analog switch is closed only during the sample time.

FIGURE 10. A/D Pin Model (Single Ended Mode)

Interrupts (Continued)

At this time, since GIE = 0, other maskable interrupts are disabled. The user is now free to do whatever context switching is required by saving the context of the machine in the stack with PUSH instructions. The user would then program a VIS (Vector Interrupt Select) instruction in order to branch to the interrupt service routine of the highest priority interrupt enabled and pending at the time of the VIS. Note that this is not necessarily the interrupt that caused the branch to address location 00FF Hex prior to the context switching.

Thus, if an interrupt with a higher rank than the one which caused the interruption becomes active before the decision of which interrupt to service is made by the VIS, then the interrupt with the higher rank will override any lower ones and will be acknowledged. The lower priority interrupt(s) are still pending, however, and will cause another interrupt immediately following the completion of the interrupt service routine associated with the higher priority interrupt just serviced. This lower priority interrupt will occur immediately following the RETI (Return from Interrupt) instruction at the end of the interrupt service routine just completed.

Inside the interrupt service routine, the associated pending bit has to be cleared by software. The RETI (Return from Interrupt) instruction at the end of the interrupt service routine will set the GIE (Global Interrupt Enable) bit, allowing the processor to be interrupted again if another interrupt is active and pending.

The VIS instruction looks at all the active interrupts at the time it is executed and performs an indirect jump to the beginning of the service routine of the one with the highest rank. The addresses of the different interrupt service routines, called vectors, are chosen by the user and stored in ROM in a table starting at 01E0 (assuming that VIS is located between 00FF and 01DF). The vectors are 15-bit wide and therefore occupy 2 ROM locations.

Arbitration Ranking	Source	Description	Vector Address Hi-Low Byte
(1) Highest	Software	INTR Instruction	0yFE–0yFF
(2)	Reserved	for Future Use	0yFC–0yFD
(3)	External	Pin G0 Edge	0yFA–0yFB
(4)	Timer T0	Underflow	0yF8–0yF9
(5)	Timer T1	T1A/Underflow	0yF6-0yF7
(6)	Timer T1	T1B	0yF4–0yF5
(7)	MICROWIRE/PLUS	BUSY Goes Low	0yF2–0yF3
(8)	Reserved	for Future Use	0yF0–0yF1
(9)	Reserved	for UART	0yEE–0yEF
(10)	Reserved	for UART	0yEC–0yED
(11)	Timer T2	T2A/Underflow	0yEA–0yEB
(12)	Timer T2	T2B	0yE8–0yE9
(13)	Reserved	for Future Use	0yE6–0yE7
(14)	Reserved	for Future Use	0yE4–0yE5
(15)	Port L/Wakeup	Port L Edge	0yE2–0yE3
(16) Lowest	Default	VIS Instr. Execution without Any Interrupts	0yE0–0yE1

y is VIS page, $y \neq 0$

Interrupts (Continued)

VIS and the vector table must be located in the same 256-byte block (0y00 to 0yFF) except if VIS is located at the last address of a block. In this case, the table must be in the next block. The vector table cannot be inserted in the first 256-byte block.

The vector of the maskable interrupt with the lowest rank is located at 0yE0 (Hi-Order byte) and 0yE1 (Lo-Order byte) and so forth in increasing rank number. The vector of the maskable interrupt with the highest rank is located at 0yFA (Hi-Order byte) and OyFB (Lo-Order byte).

The Software Trap has the highest rank and its vector is located at 0yFE and 0yFF.

If, by accident, a VIS gets executed and no interrupt is active, then the PC (Program Counter) will branch to a vector located at 0yE0–0yE1.

WARNING

A Default VIS interrupt handler routine must be present. As a minimum, this handler should confirm that the GIE bit is cleared (this indicates that the interrupt sequence has been taken), take care of any required housekeeping, restore context and return. Some sort of Warm Restart procedure should be implemented. These events can occur without any error on the part of the system designer or programmer.

Note: There is always the possibility of an interrupt occurring during an instruction which is attempting to reset the GIE bit or any other interrupt enable bit. If this occurs when a single cycle instruction is being used to reset the interrupt enable bit, the interrupt enable bit will be reset but an interrupt may still occur. This is because interrupt processing is started at the same time as the interrupt bit is being reset. To avoid this scenario, the user should always use a two, three, or four cycle instruction to reset interrupt enable bits.

Figure 11 shows the device Interrupt block diagram.

SOFTWARE TRAP

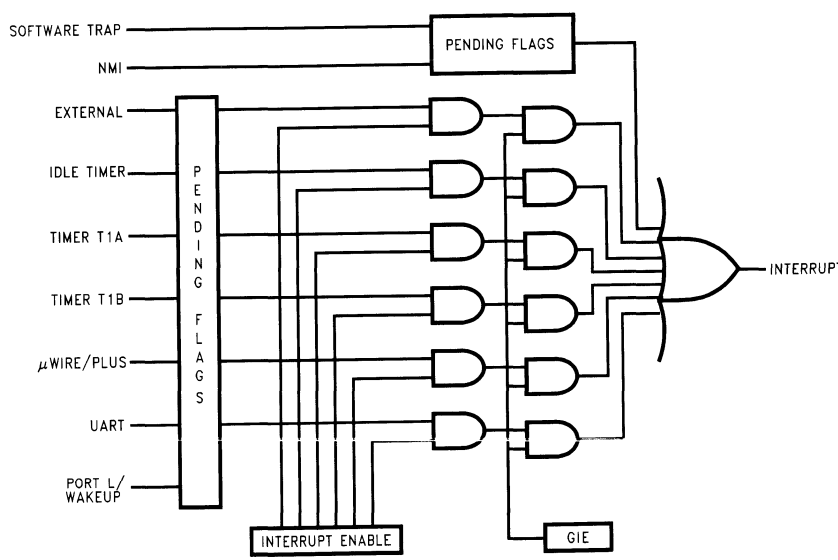
The Software Trap (ST) is a special kind of non-maskable interrupt which occurs when the INTR instruction (used to acknowledge interrupts) is fetched from ROM and placed inside the instruction register. This may happen when the PC is pointing beyond the available ROM address space or when the stack is over-popped.

When an ST occurs, the user can re-initialize the stack pointer and do a recovery procedure (similar to RESET, but not necessarily containing all of the same initialization procedures) before restarting.

The occurrence of an ST is latched into the ST pending bit. The GIE bit is not affected and the ST pending bit **(not accessible by the user)** is used to inhibit other interrupts and to direct the program to the ST service routine with the VIS instruction. The RPND instruction is used to clear the software interrupt pending bit. This bit is also cleared on reset.

The ST has the highest rank among all interrupts.

Nothing (except another ST) can interrupt an ST being serviced.



TL/DD/12523–14

FIGURE 11. Interrupt Block Diagram

WATCHDOG

The device contains a WATCHDOG and clock monitor. The WATCHDOG is designed to detect the user program getting stuck in infinite loops resulting in loss of program control or "runaway" programs. The Clock Monitor is used to detect the absence of a clock or a very slow clock below a specified rate on the CKI pin.

The WATCHDOG consists of two independent logic blocks: WD UPPER and WD LOWER. WD UPPER establishes the upper limit on the service window and WD LOWER defines the lower limit of the service window.

Servicing the WATCHDOG consists of writing a specific value to a WATCHDOG Service Register named WDSVR which Is memory mapped in the RAM. This value is composed of three fields, consisting of a 2-bit Window Select, a 5-bit Key Data field, and the 1-bit Clock Monitor Select field. Table III shows the WDSVR register.

TABLE III. WATCHDOG Service Register (WDSVR)

Window Select		Key Data					Clock Monitor
X	X	0	1	1	0	0	Y
7	6	5	4	3	2	1	0

The lower limit of the service window is fixed at 2048 instruction cycles. Bits 7 and 6 of the WDSVR register allow the user to pick an upper limit of the service window.

Table IV shows the four possible combinations of lower and upper limits for the WATCHDOG service window. This flexibility in choosing the WATCHDOG service window prevents any undue burden on the user software.

TABLE IV. WATCHDOG Service Window Select

WDSVR Bit 7	WDSVR Bit 6	Service Window (Lower-Upper Limits)
0	0	2k–8k t_C Cycles
0	1	2k–16k t_C Cycles
1	0	2k–32k t_C Cycles
1	1	2k–64k t_C Cycles

Bits 5, 4, 3, 2 and 1 of the WDSVR register represent the 5-bit Key Data field. The key data is fixed at 01100. Bit 0 of the WDSVR Register is the Clock Monitor Select bit.

Clock Monitor

The Clock Monitor aboard the device can be selected or deselected under program control. The Clock Monitor is guaranteed not to reject the clock if the instruction cycle clock ($1/t_C$) is greater or equal to 10 kHz. This equates to a clock input rate on CKI of greater or equal to 100 kHz.

WATCHDOG Operation

The WATCHDOG and Clock Monitor are disabled during reset. The device comes out of reset with the WATCHDOG armed, the WATCHDOG Window Select (bits 6, 7 of the WDSVR Register) set, and the Clock Monitor bit (bit 0 of the WDSVR Register) enabled. Thus, a Clock Monitor error will occur after coming out of reset, if the instruction cycle clock frequency has not reached a minimum specified value, including the case where the oscillator fails to start.

The WDSVR register can be written to only once after reset and the key data (bits 5 through 1 of the WDSVR Register) must match to be a valid write. This write to the WDSVR register involves two irrevocable choices: (i) the selection of the WATCHDOG service window (ii) enabling or disabling of the Clock Monitor. Hence, the first write to WDSVR Register involves selecting or deselecting the Clock Monitor, select the WATCHDOG service window and match the WATCHDOG key data. Subsequent writes to the WDSVR register will compare the value being written by the user to the WATCHDOG service window value and the key data (bits 7 through 1) in the WDSVR Register. Table V shows the sequence of events that can occur.

The user must service the WATCHDOG at least once before the upper limit of the service window expires. The WATCHDOG may not be serviced more than once in every lower limit of the service window. The user may service the WATCHDOG as many times as wished in the time period between the lower and upper limits of the service window. The first write to the WDSVR Register is also counted as a WATCHDOG service.

TABLE V. WATCHDOG Service Actions

Key Data	Window Data	Clock Monitor	Action
Match	Match	Match	Valid Service: Restart Service Window
Don't Care	Mismatch	Don't Care	Error: Generate WATCHDOG Output
Mismatch	Don't Care	Don't Care	Error: Generate WATCHDOG Output
Don't Care	Don't Care	Mismatch	Error: Generate WATCHDOG Output

WATCHDOG Operation (Continued)

The WATCHDOG has an output pin associated with it. This is the WDOUT pin, on pin 1 of the port G. WDOUT is active low. The WDOUT pin is in the high impedance state in the inactive state. Upon triggering the WATCHDOG, the logic will pull the WDOUT (G1) pin low for an additional 16 t_C–32 t_C cycles after the signal level on WDOUT pin goes below the lower Schmitt trigger threshold. After this delay, the device will stop forcing the WDOUT output low.

The WATCHDOG service window will restart when the WDOUT pin goes high. It is recommended that the user tie the WDOUT pin back to V_{CC} through a resistor in order to pull WDOUT high.

A WATCHDOG service while the WDOUT signal is active will be ignored. The state of the WDOUT pin is not guaranteed on reset, but if it powers up low then the WATCHDOG will time out and WDOUT will enter high impedance state.

The Clock Monitor forces the G1 pin low upon detecting a clock frequency error. The Clock Monitor error will continue until the clock frequency has reached the minimum specified value, after which the G1 output will enter the high impedance TRI-STATE mode following 16 t_C–32 t_C clock cycles. The Clock Monitor generates a continual Clock Monitor error if the oscillator fails to start, or fails to reach the minimum specified frequency. The specification for the Clock Monitor is as follows:

$1/t_C > 10$ kHz—No clock rejection.

$1/t_C < 10$ Hz—Guaranteed clock rejection.

WATCHDOG AND CLOCK MONITOR SUMMARY

The following salient points regarding the WATCHDOG and CLOCK MONITOR should be noted:

- Both the WATCHDOG and Clock Monitor detector circuits are inhibited during RESET.
- Following RESET, the WATCHDOG and CLOCK MONITOR are both enabled, with the WATCHDOG having the maximum service window selected.
- The WATCHDOG service window and Clock Monitor enable/disable option can only be changed once, during the initial WATCHDOG service following RESET.
- The initial WATCHDOG service must match the key data value in the WATCHDOG Service register WDSVR in order to avoid a WATCHDOG error.
- Subsequent WATCHDOG services must match all three data fields in WDSVR in order to avoid WATCHDOG errors.
- The correct key data value cannot be read from the WATCHDOG Service register WDSVR. Any attempt to read this key data value of 01100 from WDSVR will read as key data value of all 0's.
- The WATCHDOG detector circuit is inhibited during both the HALT and IDLE modes.
- The Clock Monitor detector circuit is active during both the HALT and IDLE modes. Consequently, the device inadvertently entering the HALT mode will be detected as a Clock Monitor error (provided that the Clock Monitor enable option has been selected by the program).
- With the single-pin R/C oscillator mask option selected and the CLKDLY bit reset, the WATCHDOG service window will resume following HALT mode from where it left off before entering the HALT mode.
- With the crystal oscillator mask option selected, or with the single-pin R/C oscillator mask option selected and the CLKDLY bit set, the WATCHDOG service window will be set to its selected value from WDSVR following HALT. Consequently, the WATCHDOG should not be serviced for at least 2048 instruction cycles following HALT, but must be serviced within the selected window to avoid a WATCHDOG error.
- The IDLE timer T0 is not initialized with RESET.
- The user can sync in to the IDLE counter cycle with an IDLE counter (T0) interrupt or by monitoring the T0PND flag. The T0PND flag is set whenever the thirteenth bit of the IDLE counter toggles (every 4096 instruction cycles). The user is responsible for resetting the T0PND flag.
- A hardware WATCHDOG service occurs just as the device exits the IDLE mode. Consequently, the WATCHDOG should not be serviced for at least 2048 instruction cycles following IDLE, but must be serviced within the selected window to avoid a WATCHDOG error.
- Following RESET, the initial WATCHDOG service (where the service window and the CLOCK MONITOR enable/disable must be selected) may be programmed anywhere within the maximum service window (65,536 instruction cycles) initialized by RESET. Note that this initial WATCHDOG service may be programmed within the initial 2048 instruction cycles without causing a WATCHDOG error.

Detection of Illegal Conditions

The device can detect various illegal conditions resulting from coding errors, transient noise, power supply voltage drops, runaway programs, etc.

Reading of undefined ROM gets zeros. The opcode for software interrupt is zero. If the program fetches instructions from undefined ROM, this will force a software interrupt, thus signaling that an illegal condition has occurred.

The subroutine stack grows down for each call (jump to subroutine), interrupt, or PUSH, and grows up for each return or POP, the stack pointer is initialized to RAM location 06F Hex during reset. Consequently, if there are more returns than calls, the stack pointer will point to addresses 070 and 071 Hex (which are undefined RAM). Undefined RAM from addresses 070 to 07F Hex is read as all 1's, which in turn will cause the program to return to address 7FFF Hex. This is an undefined ROM location and the instruction fetched (all 0's) from this location will generate a software interrupt signaling an illegal condition.

Detection of Illegal Conditions

(Continued)

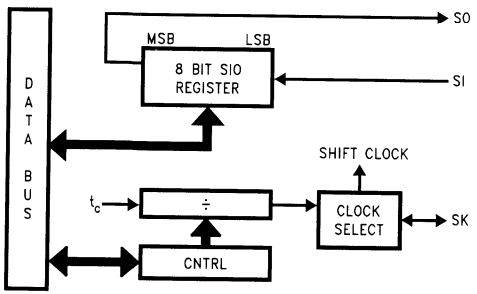
Thus, the chip can detect the following illegal conditions:

1. Executing from undefined ROM.
2. Over "POP"ing the stack by having more returns than calls.

When the software interrupt occurs, the user can re-initialize the stack pointer and do a recovery procedure before re-starting (this recovery program is probably similar to that following reset, but might not contain the same program initialization procedures).

MICROWIRE/PLUS

MICROWIRE/PLUS is a serial synchronous communications interface. The MICROWIRE/PLUS capability enables the device to interface with any of National Semiconductor's MICROWIRE peripherals (i.e. A/D converters, display drivers, E2PROMs etc.) and with other microcontrollers which support the MICROWIRE interface. It consists of an 8-bit serial shift register (SIO) with serial data input (SI), serial data output (SO) and serial shift clock (SK). *Figure 12* shows a block diagram of the MICROWIRE/PLUS logic.



TL/DD/12523–15

FIGURE 12. MICROWIRE/PLUS Block Diagram

The shift clock can be selected from either an internal source or an external source. Operating the MICROWIRE/PLUS arrangement with the internal clock source is called the Master mode of operation. Similarly, operating the MICROWIRE/PLUS arrangement with an external shift clock is called the Slave mode of operation.

The CNTRL register is used to configure and control the MICROWIRE/PLUS mode. To use the MICROWIRE/PLUS, the MSEL bit in the CNTRL register is set to one. In the master mode the SK clock rate is selected by the two bits, SL0 and SL1, in the CNTRL register. Table VI details the different clock rates that may be selected.

TABLE VI. MICROWIRE/PLUS Master Mode Clock Selection

SL1	SL0	SK
0	0	$2 \times t_c$
0	1	$4 \times t_c$
1	x	$8 \times t_c$

Where t_c is the instruction cycle clock

Thus, the chip can detect the following illegal conditions:

1. Executing from undefined ROM.
2. Over "POP"ing the stack by having more returns than calls.

MICROWIRE/PLUS OPERATION

Setting the BUSY bit in the PSW register causes the MICROWIRE/PLUS to start shifting the data. It gets reset when eight data bits have been shifted. The user may reset the BUSY bit by software to allow less than 8 bits to shift. If enabled, an interrupt is generated when eight data bits have been shifted. The device may enter the MICROWIRE/PLUS mode either as a Master or as a Slave. *Figure 13* shows how two COP888 microcontrollers and several peripherals may be interconnected using the MICROWIRE/PLUS arrangements.

Warning

The SIO register should only be loaded when the SK clock is low. Loading the SIO register while the SK clock is high will result in undefined data in the SIO register. SK clock is normally low when not shifting.

Setting the BUSY flag when the input SK clock is high in the MICROWIRE/PLUS slave mode may cause the current SK clock for the SIO shift register to be narrow. For safety, the BUSY flag should only be set when the input SK clock is low.

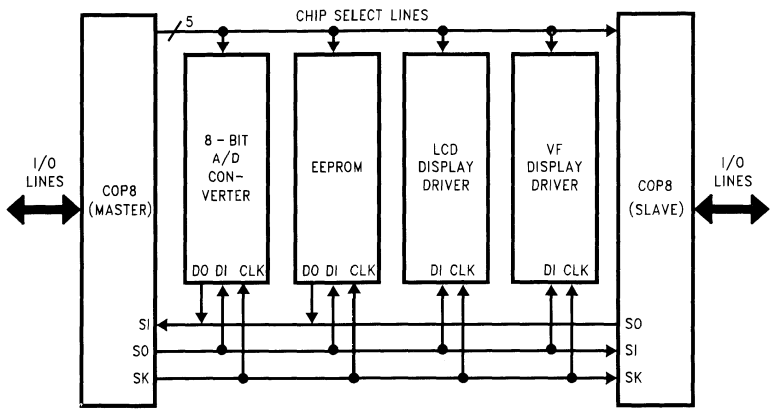
MICROWIRE/PLUS Master Mode Operation

In the MICROWIRE/PLUS Master mode of operation the shift clock (SK) is generated internally. The MICROWIRE Master always initiates all data exchanges. The MSEL bit in the CNTRL register must be set to enable the SO and SK functions onto the G Port. The SO and SK pins must also be selected as outputs by setting appropriate bits in the Port G configuration register. Table VI summarizes the bit settings required for Master mode of operation.

MICROWIRE/PLUS Slave Mode Operation

In the MICROWIRE/PLUS Slave mode of operation the SK clock is generated by an external source. Setting the MSEL bit in the CNTRL register enables the SO and SK functions onto the G Port. The SK pin must be selected as an input and the SO pin is selected as an output pin by setting and resetting the appropriate bit in the Port G configuration register. Table VII summarizes the settings required to enter the Slave mode of operation.

MICROWIRE/PLUS (Continued)



TL/DD/12523–16

FIGURE 13. MICROWIRE/PLUS Application

TABLE VII. MICROWIRE/PLUS Mode Selection

G4 (SO) Config. Bit	G5 (SK) Config. Bit	G4 Fun.	G5 Fun.	Operation
1	1	SO	Int. SK	MICROWIRE/PLUS Master
0	1	TRI-STATE	Int. SK	MICROWIRE/PLUS Master
1	0	SO	Ext. SK	MICROWIRE/PLUS Slave
0	0	TRI-STATE	Ext. Sk	MICROWIRE/PLUS Slave

This table assumes that the control flag MSEL is set.

The user must set the BUSY flag immediately upon entering the Slave mode. This will ensure that all data bits sent by the Master will be shifted properly. After eight clock pulses the BUSY flag will be cleared and the sequence may be repeated.

Alternate SK Phase Operation

The device allows either the normal SK clock or an alternate phase SK clock to shift data in and out of the SIO register. In both the modes the SK is normally low. In the normal mode data is shifted in on the rising edge of the SK clock and the data is shifted out on the falling edge of the SK clock. The SIO register is shifted on each falling edge of the SK clock in the normal mode. In the alternate SK phase mode the SIO register is shifted on the rising edge of the SK clock.

A control flag, SKSEL, allows either the normal SK clock or the alternate SK clock to be selected. Resetting SKSEL causes the MICROWIRE/PLUS logic to be clocked from the normal SK signal. Setting the SKSEL flag selects the alternate SK clock. The SKSEL is mapped into the G6 configuration bit. The SKSEL flag will power up in the reset condition, selecting the normal SK signal.

Memory Map

All RAM, ports and registers (except A and PC) are mapped into data memory address space.

Address	Contents
00 to 6F	On-Chip RAM bytes
70 to BF	Unused RAM Address Space
C0	Timer T2 Lower Byte
C1	Timer T2 Upper Byte
C2	Timer T2 Autoload Register T2RA Lower Byte
C3	Timer T2 Autoload Register T2RA Upper Byte
C4	Timer T2 Autoload Register T2RB Lower Byte
C5	Timer T2 Autoload Register T2RB Upper Byte
C6	Timer T2 Control Register
C7	WATCHDOG Service Register (Reg:WDSVR)
C8	MIWU Edge Select Register (Reg:WKEDG)
C9	MIWU Enable Register (Reg:WKEN)
CA	MIWU Pending Register (Reg:WKPND)
CB	A/D Converter Control Register (Reg:ENAD)
CC	A/D Converter Result Register (Reg:ADRSLT)
CD to CF	Reserved
D0	Port L Data Register
D1	Port L Configuration Register
D2	Port L Input Pins (Read Only)
D3	Reserved for Port L
D4	Port G Data Register
D5	Port G Configuration Register
D6	Port G Input Pins (Read Only)
D7	Port I Input Pins (Read Only)
D8	Port C Data Register
D9	Port C Configuration Register
DA	Port C Input Pins (Read Only)
DB	Reserved for Port C
DC	Port D Data Register
DD to DF	Reserved for Port D
E0 to E5	Reserved
E6	Timer T1 Autoload Register T1RB Lower Byte
E7	Timer T1 Autoload Register T1RB Upper Byte
E8	ICNTRL Register
E9	MICROWIRE Shift Register
EA	Timer T1 Lower Byte
EB	Timer T1 Upper Byte
EC	Timer T1 Autoload Register T1RA Lower Byte
ED	Timer T1 Autoload Register T1RA Upper Byte
EE	CNTRL Control Register
EF	PSW Register
F0 to FB	On-Chip RAM Mapped as Registers
FC	X Register
FD	SP Register
FE	B Register
FF	Reserved

Note: Reading memory locations 70–7F Hex will return all ones. Reading other unused memory locations will return undefined data.

Addressing Modes

There are ten addressing modes, six for operand addressing and four for transfer of control.

OPERAND ADDRESSING MODES

Register Indirect

This is the "normal" addressing mode. The operand is the data memory addressed by the B pointer or X pointer.

Register Indirect (with auto post increment or decrement of pointer)

This addressing mode is used with the LD and X instructions. The operand is the data memory addressed by the B pointer or X pointer. This is a register indirect mode that automatically post increments or decrements the B or X register after executing the instruction.

Direct

The instruction contains an 8-bit address field that directly points to the data memory for the operand.

Immediate

The instruction contains an 8-bit immediate field as the operand.

Short Immediate

This addressing mode is used with the Load B Immediate instruction. The instruction contains a 4-bit immediate field as the operand.

Indirect

This addressing mode is used with the LAID instruction. The contents of the accumulator are used as a partial address (lower 8 bits of PC) for accessing a data operand from the program memory.

TRANSFER OF CONTROL ADDRESSING MODES

Relative

This mode is used for the JP instruction, with the instruction field being added to the program counter to get the new program location. JP has a range from −31 to +32 to allow a 1-byte relative jump (JP+1 is implemented by a NOP instruction). There are no "pages" when using JP, since all 15 bits of PC are used.

Absolute

This mode is used with the JMP and JSR instructions, with the instruction field of 12 bits replacing the lower 12 bits of the program counter (PC). This allows jumping to any location in the current 4k program memory segment.

Addressing Modes (Continued)

Absolute Long

This mode is used with the JMPL and JSRL instructions, with the instruction field of 15 bits replacing the entire 15 bits of the program counter (PC). This allows jumping to any location up to 32k in the program memory space.

Indirect

This mode is used with the JID instruction. The contents of the accumulator are used as a partial address (lower 8 bits of PC) for accessing a location in the program memory. The contents of this program memory location serve as a partial address (lower 8 bits of PC) for the jump to the next instruction.

Note: The VIS is a special case of the Indirect Transfer of Control addressing mode, where the double byte vector associated with the interrupt is transferred from adjacent addresses in the program memory into the program counter (PC) in order to jump to the associated interrupt service routine.

Instruction Set

REGISTER AND SYMBOL DEFINITION

Registers

A	8-Bit Accumulator Register
B	8-Bit Address Register
X	8-Bit Address Register
SP	8-Bit Stack Pointer Register
PC	15-Bit Program Counter Register
PU	Upper 7 Bits of PC
PL	Lower 8 Bits of PC
C	1 Bit of PSW Register for Carry
HC	1 Bit of PSW Register for Half Carry
GIE	1 Bit of PSW Register for Global Interrupt Enable
VU	Interrupt Vector Upper Byte
VL	Interrupt Vector Lower Byte

Symbols

[B]	Memory Indirectly Addressed by B Register
[X]	Memory Indirectly Addressed by X Register
MD	Direct Addressed Memory
Mem	Direct Addressed Memory or [B]
Meml	Direct Addressed Memory or [B] or Immediate Data
Imm	8-Bit Immediate Data
Reg	Register Memory: Addresses F0 to FF (Includes B, X and SP)
Bit	Bit Number (0 to 7)
←	Loaded with
⟷	Exchanged with

Instruction Set (Continued)

INSTRUCTION SET

ADD	A,Meml	ADD	A ← A + Meml
ADC	A,Meml	ADD with Carry	A ← A + Meml + C, C ← Carry, HC ← Half Carry
SUBC	A,Meml	Subtract with Carry	A ← A − Meml + C, C ← Carry, HC ← Half Carry
AND	A,Meml	Logical AND	A ← A and $\overline{\text{Meml}}$
ANDSZ	A,Imm	Logical AND Immed., Skip if Zero	Skip next if (A and Imm) = 0
OR	A,Meml	Logical OR	A ← A or Meml
XOR	A,Meml	Logical EXclusive OR	A ← A xor Meml
IFEQ	MD,Imm	IF EQual	Compare MD and Imm, Do next if MD = Imm
IFEQ	A,Meml	IF EQual	Compare A and Meml, Do next if A = Meml
IFNE	A,Meml	IF Not Equal	Compare A and Meml, Do next if A ≠ Meml
IFGT	A,Meml	IF Greater Than	Compare A and Meml, Do next if A > Meml
IFBNE	#	IF B Not Equal	Do next if lower 4 bits of B ≠ Imm
DRSZ	Reg	Decrement Reg., Skip if Zero	Reg ← Reg − 1, Skip if Reg = 0
SBIT	#,Mem	Set BIT	1 to bit, Mem (bit = 0 to 7 immediate)
RBIT	#,Mem	Reset BIT	0 to bit, Mem
IFBIT	#,Mem	IF BIT	IF bit in A or Mem is two do next instruction
RPND		Reset PeNDing Flag	Reset Software Interrupt Pending Flag
X	A,Mem	EXchange A with Memory	A ⟷ Mem
X	A,[X]	EXchange A with Memory [X]	A ⟷ [X]
LD	A,Meml	LoaD A with Memory	A ← Meml
LD	A,[X]	LoaD A with Memory [X]	A ← [X]
LD	B,Imm	LoaD B with Immed.	B ← Imm
LD	Mem,Imm	LoaD Memory Immed.	Mem ← Imm
LD	Reg,Imm	LoaD Register Memory Immed.	Reg ← Imm
X	A,[B]	EXchange A with Memory [B]	A ⟷ [B], (B ← B 1)
X	A,[X]	EXchange A with Memory [X]	A ⟷ [X], (X ← 1)
LD	A,[B]	LoaD A with Memory [B]	A ← [B], (B ← B 1)
LD	A,[X]	LoaD A with Memory [X]	A ← [X], (X ← X 1)
LD	[B],Imm	LoaD Memory [B] Immed	[B] ← Imm, (B ← B 1)
CLR	A	CLeaR A	A ← 0
INC	A	INCrement A	A ← A + 1
DEC	A	DECrement A	A ← A − 1
LAID		Load A InDirect from ROM	A ← ROM (PU,A)
DCOR	A	Decimal CORrect A	A ← BCD correction of A (follows ADC, SUBC)
RRC	A	Rotate A Right thru C	C → A7 → . . . → A0 → C
RLC	A	Rotate A Left thru C	C ← A7 ← . . . ← A0 ← C
SWAP	A	SWAP nibbles of A	A7 . . . A4 ⟷ A3 . . . A0
SC		Set C	C ← 1, HC ← 1
RC		Reset C	C ← 0, HC ← 0
IF C		IF C	IF C is true, do next instruction
IFNC		IF Not C	IF C is not true, do next instruction
POP	A	POP the stack into A	SP ← SP + 1, A ← [SP]
PUSH	A	PUSH A onto the stack	[SP] ← A, SP ← SP − 1
VIS		Vector to Interrupt Service Routine	PU ← [VU], PL ← [VL]
JMPL	Addr.	Jump absolute Long	PC ← ii (ii = 15 bits, 0 to 32k)
JMP	Addr.	Jump absolute	PC9 . . . 0 ← i (i = 12 bits)
JP	Disp.	Jump relative short	PC ← PC + r (r is −31 to +32, except 1)
JSRL	Addr.	Jump SubRoutne Long	[SP] ← PL, [SP−1] ← PU,SP−2, PC ← ii
JSR	Addr	Jump SubRoutine	[SP] ← PL, [SP−1] ← PU,SP−2, PC9 . . . 0 ← i
JID		Jump InDirect	PL ← ROM (PU,A)
RET		RETurn from subroutine	SP + 2, PL ← [SP], PU ← [SP−1]
RETSK		RETurn and SKip	SP + 2, PL ← [SP], PU ← [SP−1]
RETI		RETurn from Interrupt	SP + 2, PL ← [SP], PU ← [SP−1], GIE ← 1
INTR		Generate an Interrupt	[SP] ← PL, [SP−1] ← PU, SP−2, PC ← 0FF
NOP		No OPeration	PC ← PC + 1

Instruction Execution Time

Most instructions are single byte (with immediate addressing mode instructions taking two bytes).

Most single byte instructions take one cycle time to execute.

Skipped instructions require x number of cycles to be skipped, where x equals the number of bytes in the skipped instruction opcode.

See the BYTES and CYCLES per INSTRUCTION table for details.

Bytes and Cycles per Instruction

The following table shows the number of bytes and cycles for each instruction in the format of byte/cycle.

Logic and Arithmetic Instructions

Instr.	[B]	Direct	Immed.
ADD	1/1	3/4	2/2
ADC	1/1	3/4	2/2
SUBC	1/1	3/4	2/2
AND	1/1	3/4	2/2
OR	1/1	3/4	2/2
XOR	1/1	3/4	2/2
IFEQ	1/1	3/4	2/2
IFGT	1/1	3/4	2/2
IFBNE	1/1		
DRSZ		1/3	
SBIT	1/1	3/4	
RBIT	1/1	3/4	
IFBIT	1/1	3/4	

RPND	1/1

Instructions Using A and C

CLRA	1/1
INCA	1/1
DECA	1/1
LAID	1/3
DCORA	1/1
RRCA	1/1
RLCA	1/1
SWAPA	1/1
SC	1/1
RC	1/1
IFC	1/1
IFNC	1/1
PUSHA	1/3
POPA	1/3
ANDSZ	2/2

Transfer of Control Instructions

JMPL	3/4
JMP	2/3
JP	1/3
JSRL	3/5
JSR	2/5
JID	1/3
VIS	1/5
RET	1/5
RETSK	1/5
RETI	1/5
INTR	1/7
NOP	1/1

Memory Transfer Instructions

	Register Indirect		Direct	Immed.	Register Indirect Auto Incr & Decr		
	[B]	[X]			[B+, B−]	[X+, X−]	
X A,*	1/1	1/3	2/3		1/2	1/3	
LD A,*	1/1	1/3	2/3	2/2	1/2	1/3	
LD B,Imm				1/1			(If B < 16)
LD B,Imm				2/3			(If B > 15)
LD Mem,Imm	2/2		3/3		2/2		
LD Reg,Imm			2/3				
IFEQ MD,Imm			3/3				

* ≥ Memory location addressed by B or X or directly

C0P8788CF/COP8784CF Opcode Table

Upper Nibble																Lower Nibble
F	E	D	C	B	A	9	8	7	6	5	4	3	2	1	0	
JP−15	JP−31	LD 0F0,#i	DRSZ 0F0	RRCA	RC	ADC A,#i	ADC A,[B]	IFBIT 0,[B]	ANDSZ A,#i	LD B,#0F	IFBNE 0	JSR x000−x0FF	JMP x000−x0FF	JP+17	INTR	0
JP−14	JP−30	LD 0F1,#i	DRSZ 0F1	*	SC	SUBC A,#i	SUBCA,[B]	IFBIT 1,[B]	*	LD B,#0E	IFBNE 1	JSR x100−x1FF	JMP x100−x1FF	JP+18	JP+2	1
JP−13	JP−29	LD 0F2,#i	DRSZ 0F2	X A,[X+]	X A,[B+]	IFEQ A,#i	IFEQ A,[B]	IFBIT 2,[B]	*	LD B,#0D	IFBNE 2	JSR x200−x2FF	JMP x200−x2FF	JP+19	JP+3	2
JP−12	JP−28	LD 0F3,#i	DRSZ 0F3	X A,[X−]	X A,[B−]	IFGT A,#i	IFGT A,[B]	IFBIT 3,[B]	*	LD B,#0C	IFBNE 3	JSR x300−x3FF	JMP x300−x3FF	JP+20	JP+4	3
JP−11	JP−27	LD 0F4,#i	DRSZ 0F4	VIS	LAID	ADD A,#i	ADD A,[B]	IFBIT 4,[B]	CLRA	LD B,#0B	IFBNE 4	JSR x400−x4FF	JMP x400−x4FF	JP+21	JP+5	4
JP−10	JR-26	LD 0F5,#i	DRSZ 0F5	RPND	JID	AND A,#i	AND A,[B]	IFBIT 5,[B]	SWAPA	LD B,#0A	IFBNE 5	JSR x500−x5FF	JMR x500−x5FF	JR+22	JP+6	5
JP−9	JP−25	LD 0F6,#i	DRSZ 0F6	X A,[X]	X A,[B]	XOR A,#i	XOR A,[B]	IFBIT 6,[B]	DCORA	LD B,#09	IFBNE 6	JSR x600−x6FF	JMP x600−x6FF	JP+23	JP+7	6
JP−8	JP−24	LD 0F7,#i	DRSZ 0F7	*	*	OR A,#i	OR A,[B]	IFBIT 7,[B]	PUSHA	LD B,#08	IFBNE 7	JSR x700−x7FF	JMR x700−x7FF	JR+24	JP+8	7
JP−7	JP−23	LD 0F8,#i	DRSZ 0F8	NOP	RLCA	LD A,#i	IFC	SBIT 0,[B]	RBIT 0,[B]	LD B,#07	IFBNE 8	JSR x800−x8FF	JMP x800−x8FF	JP+25	JP+9	8
JP−6	JP−22	LD 0F9,#i	DRSZ 0F9	IFNE A,[B]	IFEQ Md,#i	IFNE A,#i	IFNC	SBIT 1,[B]	RBIT 1,[B]	LD B,#06	IFBNE 9	JSR x900−x9FF	JMP x900−x9FF	JP+26	JP+10	9
JP−5	JP−21	LD 0FA,#i	DRSZ 0FA	LD A,[X+]	LD B,[B+]	LD [B+],#i	INCA	SBIT 2,[B]	RBIT 2,[B]	LD B,#05	IFBNE 0A	JSR xA00−xAFF	JMP xA00−xAFF	JP+27	JP+11	A
JP−4	JP−20	LD 0FB,#i	DRSZ 0FB	LD A,[X−]	LD A[B−]	LD [B−],#i	DECA	SBIT 3,[B]	RBIT 3,[B]	LD B,#04	IFBNE 0B	JSR xB00−xBFF	JMP xB00−xBFF	JP+28	JP+12	B
JP−3	JP−19	LD 0FC,#i	DRSZ 0FC	LD Md,#i	JMPL	X A,Md	POPA	SBIT 4,[B]	RBIT 4,[B]	LD B,#03	IFBNE 0C	JSR xC00−xCFF	JMP xC00−xCFF	JP+29	JP+13	C
JP−2	JP−18	LD 0FD,#i	DRSZ 0FD	DIR	JSRL	LD A,Md	RETSK	SBIT 5,[B]	RBIT 5,[B]	LD B,#02	IFBNE 0D	JSR xD00−xDFF	JMP xD00−xDFF	JP+30	JP+14	D
JP−1	JP−17	LD 0FE,#i	DRSZ 0FE	LD A,[X]	LD A,[B]	LD [B],#i	RET	SBIT 6,[B]	RBIT 6,[B]	LD B,#01	IFBNE 0E	JSR xE00−xEFF	JMP xE00-xEFF	JP+31	JP+15	E
JP−0	JP−16	LD 0FF,#i	DRSZ 0FF	*	*	LD B,#i	RETI	SBIT 7,[B]	RBIT 7,[B]	LD B,#00	IFBNE 0F	JSR xF00−xFFF	JMP xF00-xFFF	JP+32	JP+16	F

where, i is the immediate data
Md is a directly addressed memory location
* is an unused opcode
The opcode 60 Hex is also the opcode for IFBIT #i,A

Development Support

SUMMARY

- iceMASTER™: IM-COP8/400—Full feature in-circuit emulation for all COP8 products. A full set of COP8 Basic and Feature Family device and package specific probes are available.
- COP8 Debug Module: Moderate cost in-circuit emulation and development programming unit.
- COP8 Evaluation and Programming Unit: EPU-COP888GG—low cost In-circuit simulation and development programming unit.
- Assembler: COP8-DEV-IBMA. A DOS installable cross development Assembler, Linker, Librarian and Utility Software Development Tool Kit.
- C Compiler: COP8C. A DOS installable cross development Software Tool Kit.
- OTP/EPROM Programmer Support: Covering needs from engineering prototype, pilot production to full production environments.

iceMASTER (IM) IN-CIRCUIT EMULATION

The iceMASTER IM-COP8/400 is a full feature, PC based, in-circuit emulation tool developed and marketed by Meta-Link Corporation to support the whole COP8 family of products. National is a resale vendor for these products.

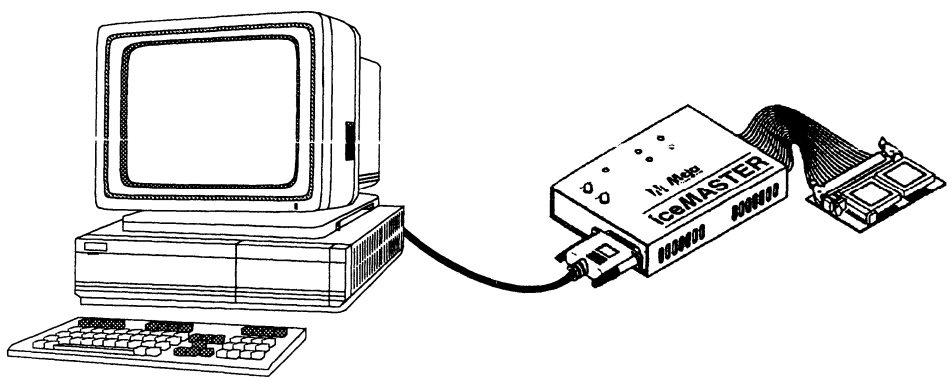
See *Figure 14* for configuration.

The iceMASTER IM-COP8/400 with its device specific COP8 Probe provides a rich feature set for developing, testing and maintaining product:

- Real-time in-circuit emulation; full 2.4V–5.5V operation range, full DC-10 MHz clock. Chip options are programmable or jumper selectable.
- Direct connection to application board by package compatible socket or surface mount assembly.
- Full 32 kbytes of loadable programming space that overlays (replaces) the on-chip ROM or EPROM. On-chip RAM and I/O blocks are used directly or recreated on the probe as necessary.
- Full 4k frame synchronous trace memory. Address, instruction, and 8 unspecified, circuit connectable trace lines. Display can be HLL source (e.g., C source), assembly or mixed.
- A full 64k hardware configurable break, trace on, trace off control, and pass count increment events.
- Tool set integrated interactive symbolic debugger—supports both assembler (COFF) and C Compiler (.COD) linked object formats.
- Real time performance profiling analysis; selectable bucket definition.
- Watch windows, content updated automatically at each execution break.
- Instruction by instruction memory/register changes displayed on source window when in single step operation.
- Single base unit and debugger software reconfigurable to support the entire COP8 family; only the probe personality needs to change. Debugger software is processor customized, and reconfigured from a master model file.
- Processor specific symbolic display of registers and bit level assignments, configured from master model file.
- Halt/Idle mode notification.
- On-line HELP customized to specific processor using master model file.
- Includes a copy of COP8-DEV-IBMA assembler and linker SDK.

IM Order Information

Base Unit	
IM-COP8/400-1	iceMASTER Base Unit, 110V Power Supply
IM-COP8/400-2	iceMASTER Base Unit, 220V Power Supply
iceMASTER Probe	
MHW-884CF28DWPC	28 DIP
MHW-888CF40DWPC	40 DIP
MHW-888CF44PWPC	44 PLCC
28 DIP to 28 SO Adapter	
MHW-SOIC28	28 SO



TL/DD/12523–18

FIGURE 14. COP8 iceMASTER Environment

Development Support (Continued)

iceMASTER DEBUG MODULE (DM)

The iceMASTER Debug Module is a PC based, combination in-circuit emulation tool and COP8 based OTP/EPROM programming tool developed and marketed by MetaLink Corporation to support the whole COP8 family of products. National is a resale vendor for these products.

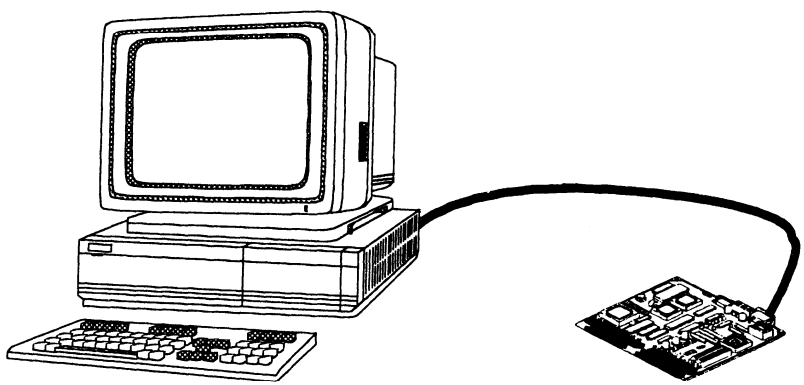
See *Figure 15* for configuration.

The iceMASTER Debug Module is a moderate cost development tool. It has the capability of in-circuit emulation for a specific COP8 microcontroller and in addition serves as a programming tool for COP8 OTP and EPROM product families. Summary of features is as follows:

- Real-time in-circuit emulation; full operating voltage range operation, full DC-10 MHz clock.
- All processor I/O pins can be cabled to an application development board with package compatible cable to socket and surface mount assembly.
- Full 32 kbytes of loadable programming space that overlays (replaces) the on-chip ROM or EPROM. On-chip RAM and I/O blocks are used directly or recreated as necessary.
- 100 frames of synchronous trace memory. The display can be HLL source (C source), assembly or mixed. The most recent history prior to a break is available in the trace memory.
- Configured break points; uses INTR instruction which is modestly intrusive.
- Software—only supported features are selectable.
- Tool set integrated interactive symbolic debugger—supports both assembler (COFF) and C Compiler (.COD) SDK linked object formats.
- Instruction by instruction memory/register changes displayed when in single step operation.
- Debugger software is processor customized, and reconfigured from a master model file.
- Processor specific symbolic display of registers and bit level assignments, configured from master model file.
- Halt/Idle mode notification.
- Programming menu supports full product line of programmable OTP and EPROM COP8 products. Program data is taken directly from the overlay RAM.
- Programming of 44 PLCC and 68 PLCC parts requires external programming adapters.
- Includes wallmount power supply.
- On-board V_{PP} generator from 5V input or connection to external supply supported. Requires V_{PP} level adjustment per the family programming specification (correct level is provided on an on-screen pop-down display).
- On-line HELP customized to specific processor using master model file.
- Includes a copy of COP8-DEV-IBMA assembler and linker SDK.

DM Order Information

Debug Model Unit	
COP8-DM/888CF	
Cable Adapters	
DM-COP8/28D	28 DIP
DM-COP8/40D	40 DIP
DM-COP8/44P	44 PLCC
28 DIP to 28 SO Adapter	
DM-COP8/28D-SO	28 SO



TL/DD/12523–19

FIGURE 15. COP8-DM Environment

Development Support (Continued)

COP8 ASSEMBLER/LINKER SOFTWARE DEVELOPMENT TOOL KIT

National Semiconductor offers a relocateable COP8 macro cross assembler, linker, librarian and utility software development tool kit. Features are summarized as follows:

- Basic and Feature Family instruction set by "device" type.
- Nested macro capability.
- Extensive set of assembler directives.
- Supported on PC/DOS platform.
- Generates National standard COFF output files.
- Integrated Linker and Librarian.
- Integrated utilities to generate ROM code file outputs.
- DUMPCOFF utility.

This product is integrated as a part of MetaLink tools as a development kit, fully supported by the MetaLink debugger. It may be ordered separately or it is bundled with the MetaLink products at no additional cost.

Order Information

Assembler SDK	
COP8-DEV-IBMA	Assembler SDK on installable 3.5" PC/DOS Floppy Disk Drive format. Periodic upgrades and most recent version is available on National's BBS and Internet.

COP8 C COMPILER

A C Compiler is developed and marketed by Byte Craft Limited. The COP8C compiler is a fully integrated development tool specifically designed to support the compact embedded configuration of the COP8 family of products.

Features are summarized as follows:

- ANSI C with some restrictions and extensions that optimize development for the COP8 embedded application.
- BITS data type extension. Register declaration #pragma with direct bit level definitions.
- C language support for interrupt routines.
- Expert system, rule based code generation and optimization.
- Performs consistency checks against the architectural definitions of the target COP8 device.
- Generates program memory code.
- Supports linking of compiled object or COP8 assembled object formats.
- Global optimization of linked code.
- Symbolic debug load format fully sourced level supported by the MetaLink debugger.

OTP EMULATOR SUPPORT

The COP87L88CF/COP87L84CF devices provide emulation and OTP support for the COP888CF/COP884CF mask programmable devices.

Approved List

Manufacturer	North America	Europe	Asia
BP Microsystems	(800) 225-2102 (713) 688-4600 Fax: (713) 688-0920	+49-8152-4183 +49-8856-932616	+852-234-16611 +852-2710-8121
Data I/O	(800) 426-1045 (206) 881-6444 Fax: (206) 882-1043	+44-0734-440011	Call North America
HI-LO	(510) 623-8860	Call Asia	+886-2-764-0215 Fax: +886-2-756-6403
ICE Technology	(800) 624-8949 (919) 430-7915	+44-1226-767404 Fax: 0-1226-370-434	
MetaLink	(800) 638-2423 (602) 926-0797 Fax: (602) 693-0681	+49-80 9156 96-0 Fax: +49-80 9123 86	+852-737-1800
Systems General	(408) 263-6667	+41-1-9450300	+886-2-917-3005 Fax: +886-2-911-1263
Needhams	(916) 924-8037 Fax: (916) 924-8065		

Development Support (Continued)

OTP Emulator Ordering Information

Device Number	Clock Option	Package	Emulates
COP87L88CFV-XE	Crystal/Halt En	44 PLCC	COP888CF
COP87L88CFN-XE	Crystal/Halt En	40 DIP	COP888CF
COP87L84CFN-XE	Crystal/Halt En	28 DIP	COP884CF
COP87L84CFM-XE	Crystal/Halt En	28 SO	COP884CF

INDUSTRY WIDE OTP/EPROM PROGRAMMING SUPPORT

Programming support, in addition to the MetaLink development tools, is provided by a full range of independent approved vendors to meet the needs from the engineering laboratory to full production.

AVAILABLE LITERATURE

For more information, please see the COP8 Basic Family User's Manual, Literature Number 620895, COP8 Feature Family User's Manual, Literature Number 620897 and National's Family of 8-bit Microcontrollers COP8 Selection Guide, Literature Number 630009.

DIAL-A-HELPER SERVICE

Dial-A-Helper is a service provided by the Microcontroller Applications group. The Dial-A-Helper is an Electronic Information System that may be accessed as a Bulletin Board System (BBS) via data modem, as an FTP site on the Internet via standard FTP client application or as an FTP site on the Internet using a standard Internet browser such as Netscape or Mosaic.

The Dial-A-Helper system provides access to an automated information storage and retrieval system . The system capabilities include a MESSAGE SECTION (electronic mail, when accessed as a BBS) for communications to and from the Microcontroller Applications Group and a FILE SECTION which consists of several file areas where valuable application software and utilities could be found.

DIAL-A-HELPER BBS via a Standard Modem

Modem: CANADA/U.S.: (800) NSC-MICRO
(800) 672-6427
EUROPE: (+49) 0-8141-351332
Baud: 14.4k
Set-Up: Length: 8-Bit
Parity: None
Stop Bit: 1
Operation: 24 Hours, 7 Days

DIAL-A-HELPER via FTP

ftp nscmicro.nsc.com
user: anonymous
password: username@yourhost.site.domain

DIAL-A-HELPER via WorldWide Web Browser

ftp://nscmicro.nsc.com

National Semiconductor on the WorldWide Web

See us on the WorldWide Web at: http://www.national.com

CUSTOMER RESPONSE CENTER

Complete product information and technical support is available from National's customer response centers.

CANADA/U.S.:	Tel:	(800)272-9959
	email:	support@tevm2.nsc.com
EUROPE:	email:	europe.support@nsc.com
	Deutsch Tel:	+49 (0) 180-530 85 85
	English Tel:	+49 (0) 180-532 78 32
	Français Tel:	+49 (0) 180-532 93 58
	Italiano Tel:	+49 (0) 180-534 16 80
JAPAN:	Tel:	+81-043-299-2309
S.E. ASIA:	Beijing Tel:	(+86) 10-6856-8601
	Shanghai Tel:	(+86) 21-6415-4092
	Hong Kong Tel:	(+852) 2737-1600
	Korea Tel:	(+82) 2-3771-6909
	Malaysia Tel:	(+60-4) 644-9061
	Singapore Tel:	(+65) 255-2226
	Taiwan Tel:	+886-2-521-3288
AUSTRALIA:	Tel:	(+61) 3-9558-9999
INDIA:	Tel:	(+91) 80-559-9467

National Semiconductor

ADVANCE INFORMATION

COP8ACC7
8-Bit One Time Programmable (OTP) Microcontroller with High Resolution A/D Conversion

General Description

The COP8ACC7 OTP microcontroller is a member of the COP8™ feature family using an 8-bit core architecture. It is pin and software compatible to the mask ROM COP8ACC5 product family. (Continued)

Key Features

- Analog Function Block for high resolution A/D including
  - — Analog comparator with seven input muxes
  - — Constant Current Source and $V_{CC/2}$ Reference
  - — 16-bit capture timer (upcounter) clocked from CKI with auto reset on timer startup
- Quiet design (reduced radiated emissions)
- 4096 bytes on-board OTP EPROM with security feature
- 128 bytes on-board RAM

Additional Peripheral Features

- Idle Timer
- One 16-bit timer with two 16-bit registers supporting:
  - — Processor Independent PWM mode
  - — External Event counter mode
  - — Input Capture mode
- Multi-Input Wake-Up (MIWU) with optional interrupts (4)
- WATCHDOG™ and clock monitor logic
- MICROWIRE/PLUS™ serial I/O with programmable shift clock-polarity

I/O Features

- Memory mapped I/O
- Software selectable I/O options (TRI-STATE® Output, Push-Pull Output, Weak Pull-Up Input, High Impedance Input)
- High current outputs
- Schmitt Trigger inputs on ports G and L
- Packages: 28 DIP/SO with 24 I/O pins
  20 SO with 16 I/O pins

CPU/Instruction Set Features

- 1 μs instruction cycle time
- Eight multi-source vectored interrupt servicing
  - — External Interrupt
  - — Idle Timer T0
  - — Timer T1 associated Interrupts
  - — MICROWIRE/PLUS
  - — Multi-Input Wake Up
  - — Software Trap
  - — Default VIS
  - — A/D (Capture Timer)
- Versatile and easy to use instruction set
- 8-bit Stack Pointer (SP)—stack in RAM
- Two 8-bit Registers Indirect Data Memory Pointers (B and X)

Fully Static CMOS

- Low current drain (typically < 5 μA)
- Two power saving modes: HALT and IDLE
- Single supply operation: 2.7V to 5.5V
- Temperature ranges: 0°C to +70°C, −40°C to +85°C,

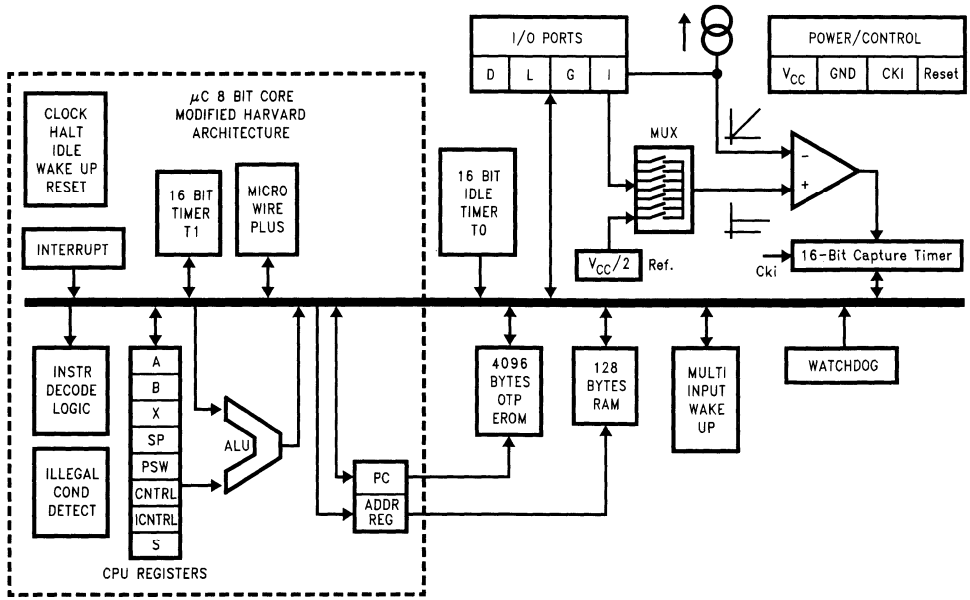
Development System

- Emulation device for COP8ACC5
- Real time emulation and full program debug offered by MetaLink development system

Applications

- Battery Chargers
- Appliances
- Data Acquisition systems

Block Diagram



TL/DD/12869–1

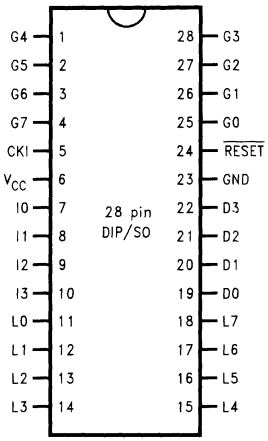
FIGURE 1. Block Diagram

General Description (Continued)

The device provides up to 6 channels of A/D performing a measurement with 12 bits of resolution in less than 0.5 ms at a clock-rate of 10 MHz. There is only an external capacitor required to complete the measurement setup and establishing low cost, high-resolution (up to 16 bits) and accurate A/D.

This device is a complete microcomputer containing all system timing, interrupt logic, ROM, RAM, and I/O necessary to implement dedicated control functions in a variety of applications. Features include an 8-bit memory mapped architecture, MICROWIRE/PLUS serial I/O, one 16-bit PWM-timer with two autoreload registers, multi-sourced interrupts an Analog Function Block and an idle timer WATCHDOG. Each I/O pin has software selectable options to adapt the device to the specific application. The device operates over a voltage range of 2.7V to 5.5V. High throughput is achieved with an efficient, regular instruction set operating at a minimum of 2 ms per instruction cycle.

Connection Diagrams

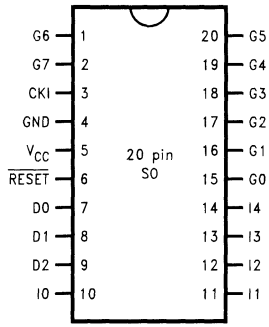


TL/DD/12869-2

Top View

Order Number COP8ACC728N9, COP8ACC728N8 or COP8ACC728N6
See NS Molded Package Number N28A

Order Number COP8ACC728M9, COP8ACC728M8 or COP8ACC728M6
See NS Molded Package Number M28B



TL/DD/12869-3

Top View

Order Number COP8ACC720M9, COP8ACC720N8 or COP8ACC720M6
See NS Molded Package Number M20B

FIGURE 2. Connection Diagrams

Connection Diagrams (Continued)

Pinouts for 28-Pin, 20-Pin Packages

Port	Type	Alt. Fun	Alt. Fun	28-Pin DIP/SO	20-Pin SO
L4	I/O	MIWU	Ext. Int.	4	
L5	I/O	MIWU	Ext. Int.	5	
L6	I/O	MIWU	Ext. Int.	6	
L7	I/O	MIWU	Ext. Int.	7	
G0	I/O	INT		23	15
G1	WDOUT			24	16
G2	I/O	T1B		25	17
G3	I/O	T1A		26	18
G4	I/O	SO		27	19
G5	I/O	SK		28	20
G6	I	SI		1	1
G7	I/CKO	HALT Restart		2	2
D0	O			11	7
D1	O			12	8
D2	O			13	9
D3	O			14	
I0	I	Analog CH1		15	10
I1	I	I_{SRC}		16	11
I2	I	Analog CH2		17	12
I3	I	Analog CH3		18	13
I4	I	Analog CH4		19	14
I5	I	Analog CH5		20	
I6	I	Analog CH6		21	
I7	I	C_{OUT}		22	
V_{CC}				9	5
GND				8	4
CKI				3	3
$\overline{\text{RESET}}$				10	6

Absolute Maximum Ratings

If Military/Aerospace specified devices are required, please contact the National Semiconductor Sales Office/Distributors for availability and specifications.

Supply Voltage (V_{CC})	7V
Voltage at Any Pin	−0.3V to V_{CC} +0.3V
Total Current into V_{CC} Pin (Source)	100 mA
Total Current out of GND Pin (Sink)	110 mA
Storage Temperature Range	−65°C to +140°C

Note: *Absolute maximum ratings indicate limits beyond which damage to the device may occur. DC and AC electrical specifications are not ensured when operating the device at absolute maximum ratings.*

DC Electrical Characteristics $0°C \le T_A \le +70°C$ unless otherwise specified

Parameter	Conditions	Min	Typ	Max	Units
Operating Voltage	Peak-to-Peak	2.7		5.5	V
Power Supply Ripple (Note 1)				0.1 V_{CC}	V
Supply Current (Note 2)					
CKI = 4 MHz	V_{CC} = 5.5V, t_C = 2.5 μs			9.5	mA
CKI = 4 MHz	V_{CC} = 4V, t_C = 2.5 μs			6.5	mA
CKI = 1 MHz	V_{CC} = 4V, t_C = 10 μs			5.4	mA
HALT Current (Note 3)	V_{CC} = 5.5V, CKI = 0 MHz		< 5	10	μA
	V_{CC} = 4V, CKI = 0 MHz		<3	6	μA
IDLE Current					
CKI = 4 MHz	V_{CC} = 5.5V, t_C = 2.5 μs			1.5	mA
CKI = 1 MHz	V_{CC} = 4V, t_C = 10 μs			0.5	mA
Input Levels (V_{IH}, V_{IL})					
$\overline{\text{RESET}}$					
Logic High		0.8 V_{CC}			V
Logic Low				0.2 V_{CC}	V
CKI, All Other Inputs					
Logic High		0.7 V_{CC}			V
Logic Low				0.2 V_{CC}	V
Hi-Z Input Leakage	V_{CC} = 5.5V	1		1	μA
Input Pullup Current	V_{CC} = 5.5V, V_{IN} = 0V	−40		−250	μA
G and L Port Input Hysteresis	(Note 5)			0.35 V_{CC}	V
Output Current Levels					
D Outputs					
Source	V_{CC} = 4V, V_{OH} = 3.3V	−0.4			mA
	V_{CC} = 2.7V, V_{OH} = 1.8V	−0.2			mA
Sink	V_{CC} = 4V, V_{OL} = 1V	10			mA
	V_{CC} = 2.7V, V_{OL} = 0.4V	2.0			mA
All Others					
Source (Weak Pull-Up Mode)	V_{CC} = 4V, V_{OH} = 2.7V	−10		−110	μA
	V_{CC} = 2.7V, V_{OH} = 1.8V	−2.5		−33	μA
Source (Push-Pull Mode)	V_{CC} = 4V, V_{OH} = 3.3V	−0.4			mA
	V_{CC} = 2.7V, V_{OH} = 1.8V	−0.2			mA
Sink (Push-Pull Mode)	V_{CC} = 4V, V_{OL} = 0.4V	1.6			mA
	V_{CC} = 2.7V, V_{OL} = 0.4V	0.7			mA
TRI-STATE Leakage	V_{CC} = 5.5V	1		1	μA
Allowable Sink/Source Current per Pin					
D Outputs (Sink)				15	mA
All others				3	mA
Maximum Input Current without Latchup (Note 4)	Room Temp			±200	mA
RAM Retention Voltage, V_r	500 ns Rise and Fall Time (min)	2			V
Input Capacitance	(Note 5)			7	pF
Load Capacitance on D2	(Note 5)			1000	pF

AC Electrical Characteristics $0°C \le T_A \le +70°C$ unless otherwise specified

Parameter	Conditions	Min	Typ	Max	Units
Instruction Cycle Time (t_C)					
Crystal, Resonator	$2.7V \le V_{CC} \le 4V$	2.5		DC	μs
	$4V \le V_{CC} \le 5.5V$	1.0		DC	μs
R/C Oscillator	$2.7V \le V_{CC} \le 4V$	7.5		DC	μs
	$4V \le V_{CC} \le 5.5V$	3.0		DC	μs
Inputs					
t_{SETUP}	$4V \le V_{CC} \le 5.5V$	200			ns
	$2.7V \le V_{CC} \le 4V$	500			ns
t_{HOLD}	$4V \le V_{CC} \le 5.5V$	60			ns
	$2.7V \le V_{CC} \le 4V$	150			ns
Output Propagation Delay (Note 5)	$R_L = 2.2k$, $C_L = 100$ pF				
t_{PD1}, t_{PD0}					
SO, SK	$4V \le V_{CC} \le 5.5V$			0.7	μs
	$2.7V \le V_{CC} \le 4V$			1.75	μs
All Others	$4V \le V_{CC} \le 5.5V$				1μs
	$2.7V \le V_{CC} \le 4V$			2.5	μs
MICROWIRE Setup Time (t_{UWS}) (Note 5)	$V_{CC} \ge 4V$	20			ns
MICROWIRE Hold Time (t_{UWH}) (Note 5)	$V_{CC} \ge 4V$	56			ns
MICROWIRE Output Propagation Delay (t_{UPD})	$V_{CC} \ge 4V$			220	ns
Input Pulse Width (Note 6)					
Interrupt Input High Time		1			t_C
Interrupt Input Low Time		1			t_C
Timer 1, 2, 3 Input High Time		1			t_C
Timer 1, 2, 3 Input Low Time		1			t_C
Reset Pulse Width		1			μs

Note 1: Maximum rate of voltage change must be < 0.5V/ms.

Note 2: Supply current is measured after running 2000 cycles with a square wave CKI input, CKO open, inputs at rails and outputs open.

Note 3: The HALT mode will stop CKI from oscillating in the RC and the Crystal configurations. Measurement of I_{DD} HALT is done with device neither sourcing or sinking current; with L, C, and G0–G5 programmed as low outputs and not driving a load; all outputs programmed low and not driving a load; all inputs tied to V_{CC}; clock monitor and comparator disabled. Parameter refers to HALT mode entered via setting bit 7 of the G Port data register. Part will pull up CKI during HALT in crystal clock mode.

Note 4: Pins G6 and $\overline{\text{RESET}}$ are designed with a high voltage input network. These pins allow input voltages > V_{CC} and the pins will have sink current to V_{CC} when biased at voltages > V_{CC} (the pins do not have source current when biased at a voltage below V_{CC}). The effective resistance to V_{CC} is 750Ω (typical). These two pins will not latch up. The voltage at the pins must be limited to less than 14V. **WARNING: Voltages in excess of 14V will cause damage to the pins. This warning excludes ESD transients.**

Note 5: The output propagation delay is referenced to the end of the instruction cycle where the output change occurs.

Note 6: Parameter characterized but not tested.

Note 7: t_C = Instruction Cycle Time.

Absolute Maximum Ratings

If Military/Aerospace specified devices are required, please contact the National Semiconductor Sales Office/Distributors for availability and specifications.

Supply Voltage (V_{CC})	7V
Voltage at Any Pin	−0.3V to V_{CC} +0.3V
Total Current into V_{CC} Pin (Source)	100 mA
Total Current out of GND Pin (Sink)	110 mA
Storage Temperature Range	−65°C to +140°C

Note: *Absolute maximum ratings indicate limits beyond which damage to the device may occur. DC and AC electrical specifications are not ensured when operating the device at absolute maximum ratings.*

DC Electrical Characteristics $-40°C \le T_A \le +85°C$ unless otherwise specified

Parameter	Conditions	Min	Typ	Max	Units
Operating Voltage		2.7		5.5	V
Power Supply Ripple (Note 1)	Peak-to-Peak			0.1 V_{CC}	V
Supply Current (Note 2)					
CKI = 4 MHz	V_{CC} = 5.5V, t_C = 2.5 μs			9.5	mA
CKI = 4 MHz	V_{CC} = 4V, t_C = 2.5 μs			6.5	mA
CKI = 1 MHz	V_{CC} = 4V, t_C = 10 μs			5.4	mA
HALT Current (Note 3)	V_{CC} = 5.5V, CKI = 0 MHz		< 5	12	μA
	V_{CC} = 4V, CKI = 0 MHz		< 3	8	μA
IDLE Current					
CKI = 4 MHz	V_{CC} = 5.5V, t_C = 2.5 μs			1.5	mA
CKI = 1 MHz	V_{CC} = 4V, t_C = 10 μs			0.5	mA
Input Levels (V_{IH}, V_{IL})					
$\overline{\text{RESET}}$					
Logic High		0.8 V_{CC}			V
Logic Low				0.2 V_{CC}	V
CKI, All Other Inputs					
Logic High		0.7 V_{CC}			V
Logic Low				0.2 V_{CC}	V
Hi-Z Input Leakage	V_{CC} = 5.5V	−2		+2	μA
Input Pullup Current	V_{CC} = 5.5V, V_{IN} = 0V	−40		−250	μA
G and L Port Input Hysteresis	(Note 5)			0.35 V_{CC}	V
Output Current Levels					
D Outputs					
Source	V_{CC} = 4V, V_{OH} = 3.3V	−0.4			mA
	V_{CC} = 2.7V, V_{OH} = 1.8V	−0.2			mA
Sink	V_{CC} = 4V, V_{OL} = 1V	10			mA
	V_{CC} = 2.7V, V_{OL} = 0.4V	2.0			mA
All Others					
Source (Weak Pull-Up Mode)	V_{CC} = 4V, V_{OH} = 2.7V	−10		−110	μA
	V_{CC} = 2.7V, V_{OH} = 1.8V	−2.5		−33	μA
Source (Push-Pull Mode)	V_{CC} = 4V, V_{OH} = 3.3V	−0.4			mA
	V_{CC} = 2.7V, V_{OH} = 1.8V	−0.2			mA
Sink (Push-Pull Mode)	V_{CC} = 4V, V_{OL} = 0.4V	1.6			mA
	V_{CC} = 2.7V, V_{OL} = 0.4V	0.7			mA
TRI-STATE Leakage	V_{CC} = 5.5V	−2		+2	μA
Allowable Sink/Source Current per Pin					
D Outputs (Sink)				15	mA
All others				3	mA
Maximum Input Current without Latchup (Note 4)	Room Temp			±200	mA
RAM Retention Voltage, V_r	500 ns Rise and Fall Time (min)	2			V
Input Capacitance	(Note 5)			7	pF
Load Capacitance on D2	(Note 5)			1000	pF

AC Electrical Characteristics $-40°C \le T_A \le +85°C$ unless otherwise specified

Parameter	Conditions	Min	Typ	Max	Units
Instruction Cycle Time (t_C)					
Crystal, Resonator	$2.7V \le V_{CC} < 4V$	2.5		DC	μs
	$4V \le V_{CC} \le 5.5V$	1.0		DC	μs
R/C Oscillator	$2.7V \le V_{CC} < 4V$	7.5		DC	μs
	$4V \le V_{CC} < 5.5V$	3.0		DC	μs
Inputs					
t_{SETUP}	$4V \le V_{CC} \le 5.5V$	200			ns
	$2.7V \le V_{CC} < 4V$	500			ns
t_{HOLD}	$4V \le V_{CC} \le 5.5V$	60			ns
	$2.7V \le V_{CC} < 4V$	150			ns
Output Propagation Delay (Note 5)	$R_L = 2.2k$, $C_L = 100$ pF				
t_{PD1}, t_{PD0}					
SO, SK	$4V \le V_{CC} \le 5.5V$			0.7	μs
	$2.7V \le V_{CC} < 4V$			1.75	μs
All Others	$4V \le V_{CC} \le 5.5V$			1	μs
	$2.7V \le V_{CC} < 4V$			2.5	μs
MICROWIRE Setup Time (t_{UWS}) (Note 5)	$V_{CC} \ge 4V$	20			ns
MICROWIRE Hold Time (t_{UWH}) (Note 5)	$V_{CC} \ge 4V$	56			ns
MICROWIRE Output Propagation Delay (t_{UPD})	$V_{CC} \ge 4V$			220	ns
Input Pulse Width (Note 6)					
Interrupt Input High Time		1			t_C
Interrupt Input Low Time		1			t_C
Timer 1, 2, 3 Input High Time		1			t_C
Timer 1, 2, 3 Input Low Time		1			t_C
Reset Pulse Width		1			μs

Note 1: Maximum rate of voltage change must be < 0.5 V/ms.

Note 2: Supply current is measured after running 2000 cycles with a square wave CKI input, CKO open, inputs at rails and outputs open.

Note 3: The HALT mode will stop CKI from oscillating in the RC and the Crystal configurations. Measurement of I_{DD} HALT is done with device neither sourcing or sinking current; with L, C, and G0–G5 programmed as low outputs and not driving a load; all outputs programmed low and not driving a load; all inputs tied to V_{CC}; clock monitor and comparator disabled. Parameter refers to HALT mode entered via setting bit 7 of the G Port data register. Part will pull up CKI during HALT in crystal clock mode.

Note 4: Pins G6 and $\overline{\text{RESET}}$ are designed with a high voltage input network. These pins allow input voltages > V_{CC} and the pins will have sink current to V_{CC} when biased at voltages greater than V_{CC} (the pins do not have source current when biased at a voltage below V_{CC}). The effective resistance to V_{CC} is 750Ω (typical). These two pins will not latch up. The voltage at the pins must be limited to less than 14V. **WARNING: Voltages in excess of 14V will cause damage to the pins. This warning excludes ESD transients.**

Note 5: The output propagation delay is referenced to the end of the instruction cycle where the output change occurs.

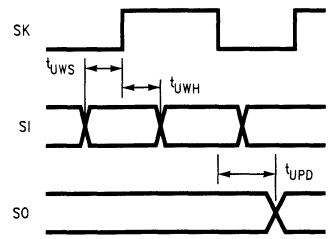
Note 6: Parameter characterized but not tested.

Note 7: t_C = Instruction Cycle Time.

Comparator AC and DC Characteristics V_{CC} = 5V, −40°C ≤ T_A ≤ +85°C

Parameter	Conditions	Min	Typ	Max	Units
Input Offset Voltage	0.4V < V_{IN} < V_{CC} −1.5V		10	25	mV
Input Common Mode Voltage Range (Note 8)		0.4		V_{CC} −1.5	V
Voltage Gain			300k		V/V
V_{CC}/2 Reference	4.0V < V_{CC} < 5.5V	0.5 V_{CC} −0.04	0.5V_{CC}	0.5V_{CC} +0.04	V
DC Supply Current For Comparator (when enabled)	V_{CC} = 5.5V			250	μA
DC Supply Current For V_{CC}/2 reference (when enabled)	V_{CC} 5.5V		50	80	μA
DC Supply Current For Constant Current Source (when enabled)	V_{CC} = 5.5V			200	μA
Constant Current Source	4.0V < V_{CC} < 5.5V	10	20	40	μA
Current Source Variation	4.0V < V_{CC} < 5.5V Temp = Constant			2	μA
Current Source Enable Time			1.5	2	μs
Comparator Response Time	10 mV overdrive, 100 pF load			1	μs

Note 8: The device is capable of operating over a common mode voltage range of 0 to V_{CC} − 1.5V, however increased offset voltage will be observed between 0V and 0.4V.



TL/DD/12869–4

FIGURE 2. MICROWIRE/PLUS Timing

Pin Descriptions

V_{CC} and GND are the power supply pins. All V_{CC} and GND pins must be connected.

CKI is the clock input. This can come from an R/C generated oscillator, or a crystal oscillator (in conjunction with CKO). See Oscillator Description section.

$\overline{\text{RESET}}$ is the master reset input. See Reset description section.

The device contains two bidirectional (one 8-bit, one 4-bit) I/O ports (G and L), where each individual bit may be independently configured as an input (Schmitt trigger inputs on ports L and G), output or TRI-STATE under program control. Three data memory address locations are allocated for each of these I/O ports. Each I/O port has two associated 8-bit memory mapped registers, the CONFIGURATION register and the output DATA register. A memory mapped address is also reserved for the input pins of each I/O port. (See the memory map for the various addresses associated with the I/O ports.) *Figure 3* shows the I/O port configurations. The DATA and CONFIGURATION registers allow for each port bit to be individually configured under software control as shown below:

PORT L is a 4-bit I/O port. All L-pins have Schmitt triggers on the inputs.

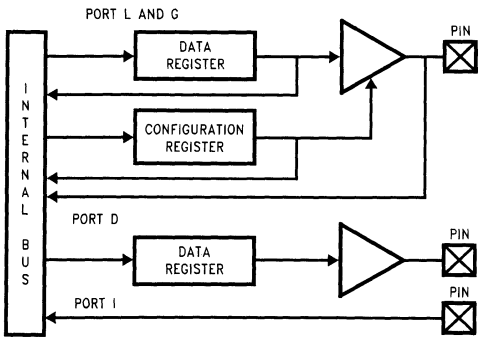
The Port L supports Multi-Input Wake Up on all four pins. The Port L has the following alternate features:

L4 MIWU or external interrupt

L5 MIWU or external interrupt

L6 MIWU or external interrupt

L7 MIWU or external interrupt



TL/DD/12869–5

FIGURE 3. I/P Port Configurations

Configuration Register	Data Register	Port Set-Up
0	0	Hi-Z Input (TRI-STATE Output)
0	1	Input with Weak Pull-Up
1	0	Push-Pull Zero Output
1	1	Push-Pull One Output

Please note: The lower 4 L-bits read all ones (L0:L3). This is independant from the states of the associated bits in the L-port Data- and Configuration register. The lower 4 bits in the L-port Data- and Configuration register can be used as general purpose status indicators (flags).

Port G is an 8-bit port with 5 I/O pins (G0, G2–G5), an input pin (G6), and a dedicated output pin (G7). Pins G0 and G2–G6 all have Schmitt Triggers on their inputs. Pin G1 serves as the dedicated WDOUT WATCHDOG output, while pin G7 is either input or output depending on the oscillator mask option selected. With the crystal oscillator option selected, G7 serves as the dedicated output pin for the CKO clock output. With the single-pin R/C oscillator mask option selected, G7 serves as a general purpose input pin but is also used to bring the device out of HALT mode with a low to high transition on G7. There are two registers associated with the G Port, a data register and a configuration register. Therefore, each of the 5 I/O bits (G0, G2–G5) can be individually configured under software control.

Since G6 is an input only pin and G7 is the dedicated CKO clock output pin (crystal clock option) or general purpose input (R/C clock option), the associated bits in the data and configuration registers for G6 and G7 are used for special purpose functions as outlined below. Reading the G6 and G7 data bits will return zeros.

Note that the chip will be placed in the HALT mode by writing a "1" to bit 7 of the Port G Data Register. Similarly the chip will be placed in the IDLE mode by writing a "1" to bit 6 of the Port G Data Register.

Writing a "1" to bit 6 of the Port G Configuration Register enables the MICROWIRE/PLUS to operate with the alternate phase of the SK clock. The G7 configuration bit, if set high, enables the clock start up delay after HALT when the R/C clock configuration is used.

	Config Reg.	Data Reg.
G7	CLKDLY	HALT
G6	Alternate SK	IDLE

Port G has the following alternate features:

G0 INTR (External Interrupt Input)

G2 T1B (Timer T1 Capture Input)

G3 T1A (Timer T1 I/O)

G4 SO (MICROWIRE Serial Data Output)

G5 SK (MICROWIRE Serial Clock)

G6 SI (MICROWIRE Serial Data Input)

Port G has the following dedicated functions:

G1 WDOUT WATCHDOG and/or Clock Monitor dedicated output.

G7 CKO Oscillator dedicated output or general purpose input

Port I is an eight-bit Hi-Z input port.

Port I0–I7 are used for the analog function block.

The Port I has the following alternate features:

I0 COMPIN1+ (Comparator Positive Input 1)

I1 COMPIN− (Comparator Negative Input/Current Source Out)

I2 COMPIN0+ (Comparator Positive Input 0)

Pin Descriptions (Continued)

I3 COMPOUT/COMPIN2+ (Comparator Output/Comparator Positive Input 2)

I4 COMPIN3+ (Comparator Positive Input 3)

I5 COMPIN4+ (Comparator Positive Input 4)

I6 COMPIN5+ (Comparator Positive Input 5)

I7 COMPOUT (Comparator Output)

Port D is a 4-bit output port that is preset high when $\overline{\text{RESET}}$ goes low. The user can tie two or more D port outputs (except D2) together in order to get a higher drive.

Functional Description

The architecture of the device is a modified Harvard architecture. With the Harvard architecture, the control store program memory (ROM) is separated from the data store memory (RAM). Both ROM and RAM have their own separate addressing space with separate address buses. The architecture, though based on the Harvard architecture, permits transfer of data from ROM to RAM.

CPU REGISTERS

The CPU can do an 8-bit addition, subtraction, logical or shift operation in one instruction (t_C) cycle time.

There are six CPU registers:

A is the 8-bit Accumulator Register

PC is the 15-bit Program Counter Register

PU is the upper 7 bits of the program counter (PC)

PL is the lower 8 bits of the program counter (PC)

B is an 8-bit RAM address pointer, which can be optionally post auto incremented or decremented.

X is an 8-bit alternate RAM address pointer, which can be optionally post auto incremented or decremented.

SP is the 8-bit stack pointer, which points to the subroutine/interrupt stack (in RAM). The SP is initialized to RAM address 06F with reset.

All the CPU registers are memory mapped with the exception of the Accumulator (A) and the Program Counter (PC).

PROGRAM MEMORY

The program memory consists of 4096 bytes of OTP EPROM. These bytes may hold program instructions or constant data (data tables for the LAID instruction, jump vectors for the JID instruction, and interrupt vectors for the VIS instruction). The program memory is addressed by the 15-bit program counter (PC). All interrupts in the device vector to program memory location 0FF Hex.

The device can be configured to inhibit external reads of the program memory. This is done by programming the Security Byte.

SECURITY FEATURE

The program memory array has an associate Security Byte that is located outside of the program address range. This byte can be addressed only from programming mode by a programmer tool.

Security is an optional feature and can only be asserted after the memory array has been programmed and verified. A secured part will read all 00(hex) by a programmer. The part will fail Blank Check and will fail Verify operations. A Read operation will fill the programmer's memory with 00(hex). The Security Byte itself is always readable with value of 00(hex) if unsecure and FF(hex) if secure.

DATA MEMORY

The data memory address space includes the on-chip RAM and data registers, the I/O registers (Configuration, Data and Pin), the control registers, the MICROWIRE/PLUS SIO shift register, and the various registers, and counters associated with the timers (with the exception of the IDLE timer). Data memory is addressed directly by the instruction or indirectly by the B, X, SP pointers and S register.

The data memory consists of 128 bytes of RAM. Sixteen bytes of RAM are mapped as "registers" at addresses 0F0 to 0FF Hex. These registers can be loaded immediately, and also decremented and tested with the DRSZ (decrement register and skip if zero) instruction. The memory pointer registers X, B and SP are memory mapped into this space at address locations 0FC to 0FF Hex respectively, with the other registers being available for general usage.

The instruction set permits any bit in memory to be set, reset or tested. All I/O and registers (except A and PC) are memory mapped; therefore, I/O bits and register bits can be directly and individually set, reset and tested. The accumulator (A) bits can also be directly and individually tested.

Note: RAM contents are undefined upon power-up.

Reset

The $\overline{\text{RESET}}$ input when pulled low initializes the microcontroller. Initialization will occur whenever the $\overline{\text{RESET}}$ input is pulled low. Upon initialization, the data and configuration registers for ports L and G are cleared, resulting in these Ports being initialized to the TRI-STATE mode. Pin G1 of the G Port is an exception (as noted below) since pin G1 is dedicated as the WATCHDOG and/or Clock Monitor error output pin. Port D is set high. The PC, PSW, ICNTRL and CNTRL-control registers are cleared. The Comparator Select Register is cleared. The S register is initialized to zero. The Multi-Input Wakeup registers WKEN and WKEDG are cleared. Wakeup register WKPND is unknown. The stack pointer, SP, is initialized to 6F Hex.

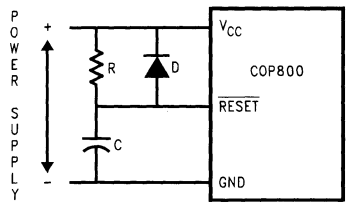
The device comes out of reset with both the WATCHDOG logic and the Clock Monitor detector armed, with the WATCHDOG service window bits set and the Clock Monitor bit set. The WATCHDOG and Clock Monitor circuits are inhibited during reset. The WATCHDOG service window bits being initialized high default to the maximum WATCHDOG service window of 64k t_C clock cycles. The Clock Monitor bit being initialized high will cause a Clock Monitor error following reset if the clock has not reached the minimum specified frequency at the termination of reset. A Clock Monitor error will cause an active low error output on pin G1. This error output will continue until 16 t_C-32 t_C clock cycles following the clock frequency reaching the minimum specified value, at which time the G1 output will enter the TRI-STATE mode.

The external RC network shown in *Figure 4* should be used to ensure that the $\overline{\text{RESET}}$ pin is held low until the power supply to the chip stabilizes.

Oscillator Circuits

The chip can be driven by a clock input on the CKI input pin which can be between DC and 10 MHz. The CKO output clock is on pin G7 (crystal configuration). The CKI input frequency is divided down by 10 to produce the instruction cycle clock (t_C).

Oscillator Circuits (Continued)



TL/DD/12869-6

RC > 5 x POWER SUPPLY RISE TIME

FIGURE 4. Recommended Reset Circuit

Figure 5 shows the Crystal and R/C Oscillator diagrams.

CRYSTAL OSCILLATOR

CKI and CKO can be connected to make a closed loop crystal (or resonator) controlled oscillator.

Table I shows the component values required for various standard crystal values.

TABLE I. Crystal Oscillator Configuration, T_A = 25°C

R1 (kΩ)	R2 (MΩ)	C1 (pF)	C2 (pF)	CKI Freq (MHz)	Conditions
0	1	30	30-36	10	V_{CC} = 5V
0	1	30	30-36	4	V_{CC} = 5V
0	1	200	100-150	0.455	V_{CC} = 5V

R/C OSCILLATOR

By selecting CKI as a single pin oscillator input, a single pin R/C oscillator circuit can be connected to it. CKO is available as a general purpose input, and/or HALT restart input.

Note: Use of the R/C oscillator option will result in higher electromagnetic emissions.

Table II shows the variation in the oscillator frequencies as functions of the component (R and C) values.

TABLE II. RC Oscillator Configuration, T_A = 25°C

R (kΩ)	C (pF)	CKI Freq (MHz)	Instr. Cycle (μs)	Conditions
3.3	82	2.2 to 2.7	3.7 to 4.6	V_{CC} = 5V
5.6	100	1.1 to 1.3	7.4 to 9.0	V_{CC} = 5V
6.8	100	0.9 to 1.1	8.8 to 10.8	V_{CC} = 5V

Note: 3k ≤ R ≤ 200k
50 pF ≤ C ≤ 200 pF

Control Registers

CNTRL Register (Address X'00EE)

The Timer1 (T1) and MICROWIRE/PLUS control register contains the following bits:

SL1 and SL0 Select the MICROWIRE/PLUS clock divide by (00 = 2, 01 = 4, 1x = 8)

IEDG External interrupt edge polarity select (0 = Rising edge, 1 = Falling edge)

MSEL Selects G5 and G4 as MICROWIRE/PLUS signals SK and SO respectively

T1C0 Timer T1 Start/Stop control in timer modes 1 and 2. T1 Underflow Interrupt Pending Flag in timer mode 3

T1C1 Timer T1 mode control bit

T1C2 Timer T1 mode control bit

T1C3 Timer T1 mode control bit

T1C3	T1C2	T1C1	T1C0	MSEL	IEDG	SL1	SL0

Bit 7 Bit 0

PSW Register (Address X'00EF)

The PSW register contains the following select bits:

GIE Global interrupt enable (enables interrupts)

EXEN Enable external interrupt

BUSY MICROWIRE/PLUS busy shifting flag

EXPND External interrupt pending

T1ENA Timer T1 Interrupt Enable for Timer Underflow or T1A Input capture edge

T1PNDA Timer T1 Interrupt Pending Flag (Autoreload RA in mode 1, T1 Underflow in Mode 2, T1A capture edge in mode 3)

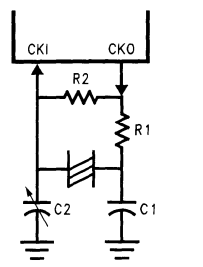
C Carry Flag

HC Half Carry Flag

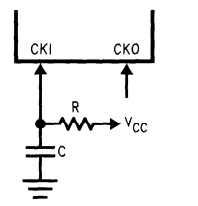
HC	C	T1PNDA	T1ENA	EXPND	BUSY	EXEN	GIE

Bit 7 Bit 0

The Half-Carry flag is also affected by all the instructions that affect the Carry flag. The SC (Set Carry) and RC (Reset Carry) instructions will respectively set or clear both the carry flags. In addition to the SC and RC instructions, ADC, SUBC, RRC and RLC instructions affect the Carry and Half Carry flags.



TL/DD/12869-7



TL/DD/12869-8

FIGURE 5. Crystal and R/C Oscillator Diagrams

Control Registers (Continued)

ICNTRL Register (Address X'00E8)

The ICNTRL register contains the following bits:

T1ENB	Timer T1 Interrupt Enable for T1B Input capture edge
T1PNDB	Timer T1 Interrupt Pending Flag for T1B capture edge
μWEN	Enable MICROWIRE/PLUS interrupt
μWPND	MICROWIRE/PLUS interrupt pending
T0EN	Timer T0 Interrupt Enable (Bit 12 toggle)
T0PND	Timer T0 Interrupt pending
LPEN	L Port Interrupt Enable (Multi-Input Wakeup/Interrupt)

Bit 7 could be used as a flag

Unused	LPEN	T0PND	T0EN	WPND	WEN	T1PNDB	T1ENB
Bit 7							Bit 0

CAPCNTL Register (Address X'00)

The CAPCNTL register contains the following bits:

CAPIEN	Capture Interrupts enable
CAPPND	Capture pending
CAPOVL	Capture Timer overflow
CAPRUN	Capture Timer Run
CAPMOD	Reset Timer

Unused	CAPMOD	CAPRUN	CAPOVL	CAPPND	CAPIEN
Bit 7					Bit 0

Timers

The device contains a very versatile set of timers (T0 and T1). All timers and associated autoreload/capture registers power up containing random data.

TIMER T0 (IDLE TIMER)

The device supports applications that require maintaining real time and low power with the IDLE mode. This IDLE mode support is furnished by the IDLE timer T0, which is a 16-bit timer. The Timer T0 runs continuously at the fixed rate of the instruction cycle clock, t_C. The user cannot read or write to the IDLE Timer T0, which is a count down timer.

The Timer T0 supports the following functions:

- Exit out of the Idle Mode (See Idle Mode description)
- WATCHDOG logic (See WATCHDOG description)
- Start up delay out of the HALT mode

Figure 6 is a functional block diagram showing the structure of the IDLE Timer and its associated interrupt logic.

Bits 11 through 15 of the ITMR register can be selected for triggering the IDLE Timer interrupt. Each time the selected bit underflows (every 4k, 8k, 16k, 32k or 64k instruction cycles), the IDLE Timer interrupt pending bit T0PND is set, thus generating an interrupt (if enabled), and bit 6 of the Port G data register is reset, thus causing an exit from the IDLE mode if the device is in that mode.

In order for an interrupt to be generated, the IDLE Timer interrupt enable bit T0EN must be set, and the GIE (Global Interrupt Enable) bit must also be set. The T0PND flag and T0EN bit are bits 5 and 4 of the ICNTRL register, respectively. The interrupt can be used for any purpose. Typically, it is used to perform a task upon exit from the IDLE mode. For more information on the IDLE mode, refer to the Power Save Modes section.

The Idle Timer period is selected by bits 0–2 of the ITMR register Bits 3–7 of the ITMR Register are reserved and should not be used as software flags.

TABLE III. Idle Timer Window Length

ITSEL2	ITSEL1	ITSEL0	Idle Timer Period (Instruction Cycles)
0	0	0	4,096
0	0	1	8,192
0	1	0	16,384
0	1	1	32,768
1	X	X	65,536

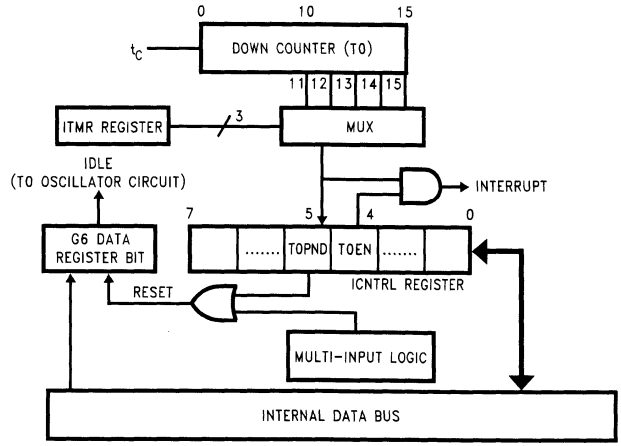
The ITMR register is cleared on Reset and the Idle Timer period is reset to 4,096 instruction cycles.

ITMR Register (Address X'0xCF)

Reserved	ITSEL2	ITSEL1	ITSEL0
Bit 7			Bit 0

Any time the IDLE Timer period is changed there is the possibility of generating a spurious IDLE Timer interrupt by setting the T0PND bit. The user is advised to disable IDLE Timer interrupts prior to changing the value of the ITSEL bits of the ITMR Register and then clear the T0PND bit before attempting to synchronize operation to the IDLE Timer.

Timers (Continued)



TL/DD/12869–9

FIGURE 6. Functional Block Diagram for Idle Timer T0

TIMER T1

The device has a powerful timer/counter block. The timer consists of a 16-bit timer, T1, and two supporting 16-bit autoreload/capture registers, R1A and R1B. The timer block has two pins associated with it, T1A and T1B. The pin T1A supports I/O required by the timer block, while the pin T1B is an input to the timer block. The powerful and flexible timer block allows the device to easily perform all timer functions with minimal software overhead. The timer block has three operating modes: Processor Independent PWM mode, External Event Counter mode, and Input Capture mode.

The control bits T1C3, T1C2, and T1C1 allow selection of the different modes of operation.

Mode 1. Processor Independent PWM Mode

As the name suggests, this mode allows the device to generate a PWM signal with very minimal user intervention. The user only has to define the parameters of the PWM signal (ON time and OFF time). Once begun, the timer block will continuously generate the PWM signal completely independent of the microcontroller. The user software services the timer block only when the PWM parameters require updating.

In this mode the timer T1 counts down at a fixed rate of t_C. Upon every underflow the timer is alternately reloaded with the contents of supporting registers, R1A and R1B. The very first underflow of the timer causes the timer to reload from the register R1A. Subsequent underflows cause the timer to be reloaded from the registers alternately beginning with the register R1B.

The T1 Timer control bits, T1C3, T1C2 and T1C1 set up the timer for PWM mode operation.

Figure 7 shows a block diagram of the timer in PWM mode.

The underflows can be programmed to toggle the T1A output pin. The underflows can also be programmed to generate interrupts.

Underflows from the timer are alternately latched into two pending flags, T1PNDA and T1PNDB. The user must reset these pending flags under software control. Two control enable flags, T1ENA and T1ENB, allow the interrupts from the timer underflow to be enabled or disabled. Setting the timer enable flag T1ENA will cause an interrupt when a timer underflow causes the R1A register to be reloaded into the timer. Setting the timer enable flag T1ENB will cause an interrupt when a timer underflow causes the R1B register to be reloaded into the timer. Resetting the timer enable flags will disable the associated interrupts.

Either or both of the timer underflow interrupts may be enabled. This gives the user the flexibility of interrupting once per PWM period on either the rising or falling edge of the PWM output. Alternatively, the user may choose to interrupt on both edges of the PWM output.

Timers (Continued)

Mode 2. External Event Counter Mode

This mode is quite similar to the processor independent PWM mode previously described. The main difference is that the timer, T1, is clocked by the input signal from the T1A pin. The T1 timer control bits, T1C3, T1C2 and T1C1 allow the timer to be clocked either on a positive or negative edge from the T1A pin. Underflows from the timer are latched into the T1PNDA pending flag. Setting the T1ENA control flag will cause an interrupt when the timer underflows.

In this mode the input pin T1B can be used as an independent positive edge sensitive interrupt input if the T1ENB control flag is set. The occurrence of a positive edge on the T1B input pin is latched into the T1PNDB flag.

Figure 8 shows a block diagram of the timer in External Event Counter mode.

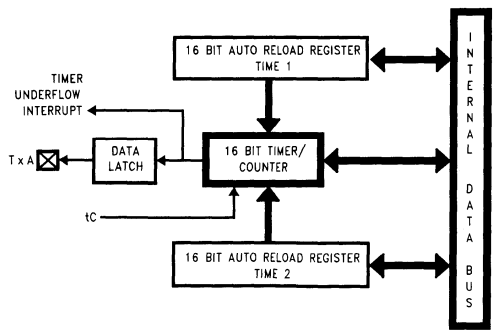
Note: The PWM output is not available in this mode since the T1A pin is being used as the counter input clock.

Mode 3. Input Capture Mode

The device can precisely measure external frequencies or time external events by placing the timer block, T1, in the input capture mode.

In this mode, the timer T1 is constantly running at the fixed t_C rate. The two registers, R1A and R1B, act as capture registers. Each register acts in conjunction with a pin. The register R1A acts in conjunction with the T1A pin and the register R1B acts in conjunction with the T1B pin.

The timer value gets copied over into the register when a trigger event occurs on its corresponding pin. Control bits, T1C3, T1C2 and T1C1, allow the trigger events to be specified either as a positive or a negative edge. The trigger condition for each input pin can be specified independently.

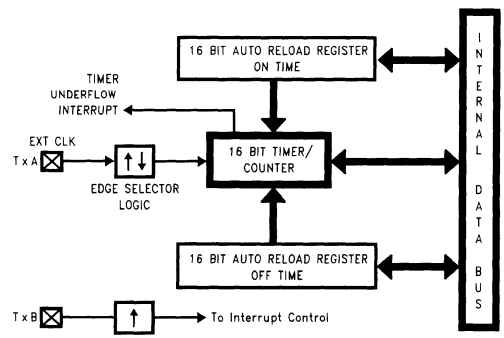


TL/DD/12869–10

FIGURE 7. Timer in PWM Mode

The trigger conditions can also be programmed to generate interrupts. The occurrence of the specified trigger condition on the T1A and T1B pins will be respectively latched into the pending flags, T1PNDA and T1PNDB. The control flag T1ENA allows the interrupt on T1A to be either enabled or disabled. Setting the T1ENA flag enables interrupts to be generated when the selected trigger condition occurs on the T1A pin. Similarly, the flag T1ENB controls the interrupts from the T1B pin.

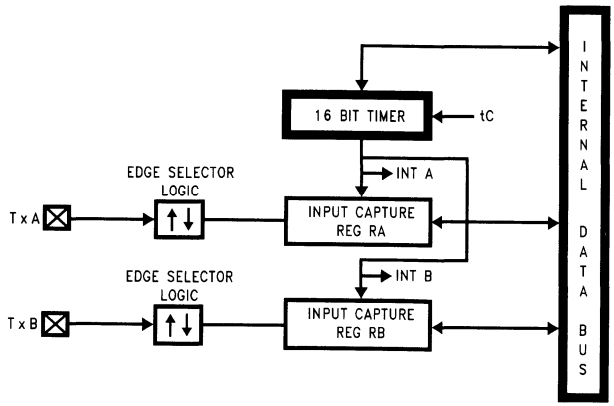
Underflows from the timer can also be programmed to generate interrupts. Underflows are latched into the timer T1C0 pending flag (the T1C0 control bit serves as the timer underflow interrupt pending flag in the Input Capture mode). Consequently, the T1C0 control bit should be reset when entering the Input Capture mode. The timer underflow interrupt is enabled with the T1ENA control flag. When a T1A interrupt occurs in the Input Capture mode, the user must check both the T1PNDA and T1C0 pending flags in order to determine whether a T1A input capture or a timer underflow (or both) caused the interrupt.



TL/DD/12869–11

FIGURE 8. Timer in External Event Counter Mode

Timers (Continued)



TL/DD/12869–12

FIGURE 9. Timer in Input Capture Mode

Figure 9 shows a block diagram of the timer in Input Capture mode.

TIMER CONTROL FLAGS

The Timer T1 control bits and their functions are summarized below.

T1C0 Timer Start/Stop control in Modes 1 and 2 (Processor Independent PWM and External Event Counter), where 1 = Start, 0 = Stop Timer Underflow Interrupt Pending Flag in Mode 3 (Input Capture)

T1PNDA Timer Interrupt Pending Flag

T1PNDB Timer Interrupt Pending Flag

T1ENA Timer Interrupt Enable Flag

T1ENB Timer Interrupt Enable Flag

1 = Timer Interrupt Enabled

0 = Timer Interrupt Disabled

T1C3 Timer mode control

T1C2 Timer mode control

T1C1 Timer mode control

HIGH SPEED CAPTURE TIMER

The device provides a 16-bit high-speed capture timer. The timer consists of a 16-bit up-counter that is clocked with the device clock input frequency (CKI) and an 8-bit control register. The 16-bit counter is mapped as two read/write 8-bit registers. This timer is specifically designed to be used in conjunction with the Analog Function Block (comparator, analog multiplexer, constant current source) to implement a low-cost, high-resolution, single-slope A/D.

The timer is automatically stopped in the event of a capture to allow the software to read the timer value. Coming out of reset the counter is disabled (stopped) and reads all "0".

Setting the Capture Timer Run bit CAPRUN bit in the Capture Control Register (CAPCNTL) will start the counter. The counter will count up until a capture event (negative edge) is received. Upon a capture the counter will be stopped, the Capture Pending bit (CAPPND) is set, and the CAPRUN bit is automatically reset. If capture interrupts are enabled (CAPIEN = 1), the capture event will generate an interrupt. Setting the CAPRUN bit again by software will start a new counting cycle. If the Capture Mode bit is reset (CAPMOD = 0) the capture timer will be automatically initialized to all "0" with each setting of the CAPRUN bit. If CAPMOD = 1 the timer will not be cleared when setting the CAPRUN bit, thus allowing the user's software to pre-load the timer registers with any desired value. This mode can be used in conjunction with the timer's overflow to implement for example a programmable delay counter.

Timers (Continued)

The timer mode control bits (T1C3, T1C2 and T1C1) are detailed below:

TABLE IV. Timer Mode Control

T1C3	T1C2	T1C1	Timer Mode	Interrupt A Source	Interrupt B Source	Timer Counts On
0	0	0	MODE 2 (External Event Counter)	Timer Underflow	Pos. T1B Edge	T1A Pos. Edge
0	0	1	MODE 2 (External Event Counter)	Timer Underflow	Pos. T1B Edge	T1A Neg. Edge
1	0	1	MODE 1 (PWM) T1A Toggle	Autoreload RA	Autoreload RB	t_C
1	0	0	MODE 1 (PWM) No T1A Toggle	Autoreload RA	Autoreload RB	t_C
0	1	0	MODE 3 (Capture) Captures: T1A Pos. Edge T1B Pos. Edge	Pos. T1A Edge Timer Underflow	Pos. T1B Edge	t_C
1	1	0	MODE 3 (Capture) Captures: T1A Pos. Edge T1B Neg. Edge	Pos. T1A Edge or Timer Underflow	Neg. T1B Edge	t_C
0	1	1	MODE 3 (Capture) Captures: T1A Neg. Edge T1B Pos. Edge	Neg. T1A Edge or Timer Underflow	Pos. T1B Edge	t_C
1	1	1	MODE 3 (Capture) Captures: T1A Neg. Edge T1B Neg. Edge	Neg. T1A Edge or Timer Underflow	Neg. T1B Edge	t_C

"CAPTURE MODE" is only active when the CAPRUN bit is set, i.e. any capture events received while the timer is stopped (CAPRUN = 0) will be ignored and will not cause the CAPPND bit to be set. The capture counter can also be stopped (frozen) by the user's software resetting the CAPRUN bit.

If the user program tries to set the CAPRUN bit at the same time that the hardware gets a capture event and tries to reset the CAPRUN bit, the hardware will have precedence.

Should the counter overflow before a capture condition occurs, the Capture Overflow bit (CAPOVL) bit in the CAPCNTL register will be set. If Capture interrupts are enabled (CAPIEN = 1) an overflow will generate an interrupt. The user software should reset this bit before the next overflow occurs, otherwise subsequent overflow conditions cannot be detected.

Capture Overflow interrupt and Capture Pending interrupt share the same interrupt vector.

Power Save Modes

The device offers the user two power save modes of operation: HALT and IDLE. In the HALT mode, all microcontroller activities are stopped. In the IDLE mode, the on-board oscillator circuitry and timer T0 are active but all other microcontroller activities are stopped. In either mode, all on-board RAM, registers, I/O states, and timers (with the exception of T0) are unaltered.

HALT MODE

The device can be placed in the HALT mode by writing a "1" to the HALT flag (G7 data bit). All microcontroller activities, including the clock and timers, are stopped. The WATCHDOG logic on the device is disabled during the HALT mode. However, the clock monitor circuitry, if enabled, remains active and will cause the WATCHDOG output pin (WDOUT) to go low. If the HALT mode is used and the user does not want to activate the WDOUT pin, the Clock Monitor should be disabled after the device comes out of reset (resetting the Clock Monitor control bit with the first write to the WDSVR register). In the HALT mode, the power requirements of the device are minimal and the applied voltage (V_{CC}) may be decreased to V_r (V_r = 2.0V) without altering the state of the machine.

The device supports three different ways of exiting the HALT mode. The first method of exiting the HALT mode is with the Multi-Input Wakeup feature on the Port L.

The second method is with a low to high transition on the CKO (G7) pin. This method precludes the use of the crystal clock configuration (since CKO becomes a dedicated output), and so may only be used with an RC clock configuration. The third method of exiting the HALT mode is by pulling the $\overline{\text{RESET}}$ pin low.

Since a crystal or ceramic resonator may be selected as the oscillator, the Wakeup signal is not allowed to start the chip running immediately since crystal oscillators and ceramic resonators have a delayed start up time to reach full ampli-

Power Save Modes (Continued)

tude and frequency stability. The IDLE timer is used to generate a fixed delay to ensure that the oscillator has indeed stabilized before allowing instruction execution. In this case, upon detecting a valid Wakeup signal, only the oscillator circuitry is enabled. The IDLE timer is loaded with a value of 256 and is clocked with the t_C instruction cycle clock. The t_C clock is derived by dividing the oscillator clock down by a factor of 10. The Schmitt trigger following the CKI inverter on the chip ensures that the IDLE timer is clocked only when the oscillator has a sufficiently large amplitude to meet the Schmitt trigger specifications. This Schmitt trigger is not part of the oscillator closed loop. The startup timeout from the IDLE timer enables the clock signals to be routed to the rest of the chip.

If an RC clock option is being used, the fixed delay is introduced optionally. A control bit, CLKDLY, mapped as configuration bit G7, controls whether the delay is to be introduced or not. The delay is included if CLKDLY is set, and excluded if CLKDLY is reset. The CLKDLY bit is cleared on reset.

The device has two mask options associated with the HALT mode. The first mask option enables the HALT mode feature, while the second mask option disables the HALT mode. With the HALT mode enable mask option, the device will enter and exit the HALT mode as described above. With the HALT disable mask option, the device cannot be placed in the HALT mode (writing a "1" to the HALT flag will have no effect, the HALT flag will remain "0").

IDLE MODE

The device is placed in the IDLE mode by writing a "1" to the IDLE flag (G6 data bit). In this mode, all activities, except the associated on-board oscillator circuitry and the IDLE Timer T0, are stopped.

As with the HALT mode, the device can be returned to normal operation with a reset, or with a Multi-Input Wakeup from the L Port. Alternately, the microcontroller resumes normal operation from the IDLE mode when the thirteenth bit (representing 4.096 ms at internal clock frequency of 10 MHz, t_C = 1 μs) of the IDLE Timer toggles.

This toggle condition of the thirteenth bit of the IDLE Timer T0 is latched into the T0PND pending flag.

The user has the option of being interrupted with a transition on the thirteenth bit of the IDLE Timer T0. The interrupt can be enabled or disabled via the T0EN control bit. Setting the T0EN flag enables the interrupt and vice versa.

The user can enter the IDLE mode with the Timer T0 interrupt enabled. In this case, when the T0PND bit gets set, the device will first execute the Timer T0 interrupt service routine and then return to the instruction following the "Enter Idle Mode" instruction.

Alternatively, the user can enter the IDLE mode with the IDLE Timer T0 interrupt disabled. In this case, the device will resume normal operation with the instruction immediately following the "Enter IDLE Mode" instruction.

Note: It is necessary to program two NOP instructions following both the set HALT mode and set IDLE mode instructions. These NOP instructions are necessary to allow clock resynchronization following the HALT or IDLE modes.

Multi-Input Wakeup

The Multi-Input Wakeup feature is used to return (wakeup) the device from either the HALT or IDLE modes. Alternately Multi-Input Wakeup/Interrupt feature may also be used to generate up to 4 edge selectable external interrupts.

Figure 10 shows the Multi-Input Wakeup logic.

The Multi-Input Wakeup feature utilizes the L Port. The user selects which particular L port bit (or combination of L Port bits) will cause the device to exit the HALT or IDLE modes. The selection is done through the register WKEN. The register WKEN is an 8-bit read/write register, which contains a control bit for every L port bit. Setting a particular WKEN bit enables a Wakeup from the associated L port pin.

The user can select whether the trigger condition on the selected L Port pin is going to be either a positive edge (low to high transition) or a negative edge (high to low transition). This selection is made via the register WKEDG, which is an 8-bit control register with a bit assigned to each L Port pin. Setting the control bit will select the trigger condition to be a negative edge on that particular L Port pin. Resetting the bit selects the trigger condition to be a positive edge. Changing an edge select entails several steps in order to avoid a Wakeup condition as a result of the edge change. First, the associated WKEN bit should be reset, followed by the edge select change in WKEDG. Next, the associated WKPND bit should be cleared, followed by the associated WKEN bit being re-enabled.

An example may serve to clarify this procedure. Suppose we wish to change the edge select from positive (low going high) to negative (high going low) for L Port bit 5, where bit 5 has previously been enabled for an input interrupt. The program would be as follows:

```
RBIT 5, WKEN
SBIT 5, WKEDG
RBIT 5, WKPND
SBIT 5, WKEN
```

If the L port bits have been used as outputs and then changed to inputs with Multi-Input Wakeup/Interrupt, a safety procedure should also be followed to avoid wakeup conditions. After the selected L port bits have been changed from output to input but before the associated WKEN bits are enabled, the associated edge select bits in WKEDG should be set or reset for the desired edge selects, followed by the associated WKPND bits being cleared.

This same procedure should be used following reset, since the L port inputs are left floating as a result of reset.

The occurrence of the selected trigger condition for Multi-Input Wakeup is latched into a pending register called WKPND. The respective bits of the WKPND register will be set on the occurrence of the selected trigger edge on the corresponding Port L pin. The user has the responsibility of clearing these pending flags. Since WKPND is a pending register for the occurrence of selected wakeup conditions, the device will not enter the HALT mode if any Wakeup bit is both enabled and pending. Consequently, the user must clear the pending flags before attempting to enter the HALT mode.

WKEN, WKPND and WKEDG are all read/write registers, and are cleared at reset.

Multi-Input Wakeup (Continued)

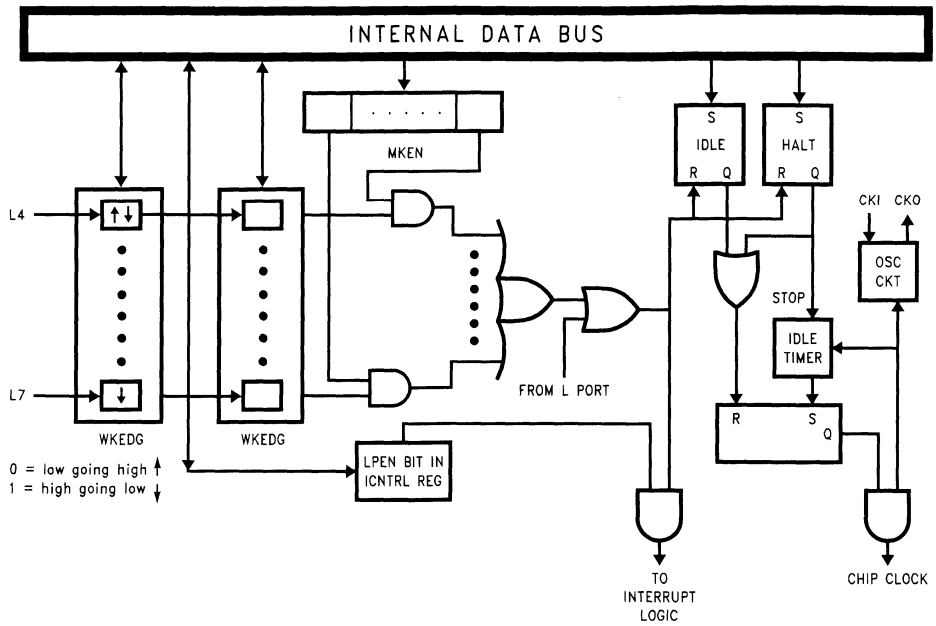


FIGURE 10. Multi-Input Wake Up Logic

PORT L INTERRUPTS

Port L provides the user with an additional eight fully selectable, edge sensitive interrupts which are all vectored into the same service subroutine.

The interrupt from Port L shares logic with the wake up circuitry. The register WKEN allows interrupts from Port L to be individually enabled or disabled. The register WKEDG specifies the trigger condition to be either a positive or a negative edge. Finally, the register WKPND latches in the pending trigger conditions.

The GIE (Global Interrupt Enable) bit enables the interrupt function.

A control flag, LPEN, functions as a global interrupt enable for Port L interrupts. Setting the LPEN flag will enable interrupts and vice versa. A separate global pending flag is not needed since the register WKPND is adequate.

Since Port L is also used for waking the device out of the HALT or IDLE modes, the user can elect to exit the HALT or IDLE modes either with or without the interrupt enabled. If he elects to disable the interrupt, then the device will restart execution from the instruction immediately following the instruction that placed the microcontroller in the HALT or IDLE modes. In the other case, the device will first execute the interrupt service routine and then revert to normal operation. (See HALT MODE for clock option wakeup information.)

Analog Function Block

This device contains an analog function block with the intent to provide a function which allows for single slope, low cost, A/D conversion of up to 6 channels.

CMPSL REGISTER (ADDRESS X'00B7)

The CMPSL register contains the following bits:

CMPNEG — Will drive I1 to a low level. This bit can be used to discharge an external capacitor. This bit is disabled if the comparator is not enabled (CMPEN = 0).

CMPEN — Enable the comparator ("1" = enable)

CSEN — Enables the internal constant current source. This current source provides a nominal 20 μA constant current at the I1 pin. This current can be used to ensure a linear charging rate on an external capacitor. This bit has no affect and the current source is disabled if the comparator is not enabled (CMPEN = 0).

CMPOE — Enables the comparator output to either pin I3 or pin I7 ("1" = enable) depending on the value of CMPISEL0/1/2.

CMPISEL0/1/2 — Will select one of seven possible sources (I0/I2/I3/I4/I5/I6/internal reference) as a positive input to the comparator (see Table V for more information.)

CMPT2B — Selects the timer T2B input to be driven directly by the comparator output. If the comparator is disabled (CMPEN = 0), this function is disabled, i.e. the T2B input is connected to Port L5.

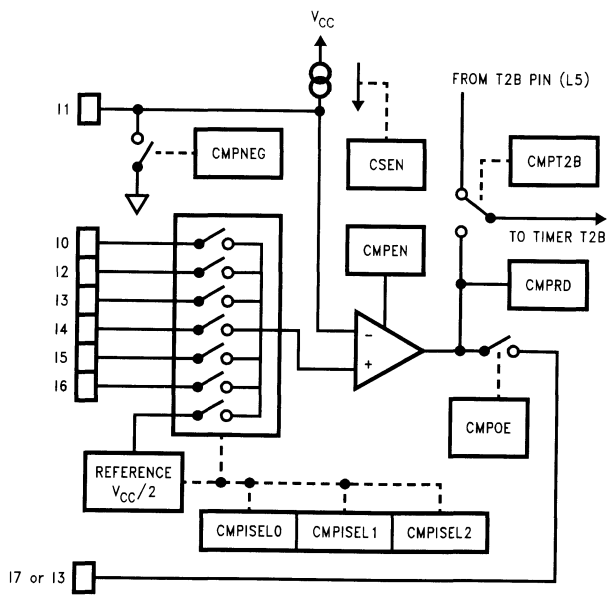
CMPT2B	CMPISEL2	CMPISEL1	CMPISEL0	CMPOE	CSEN	CMPEN	CMPNEG

Bit 7 Bit 0

The Comparator Select Register is cleared on RESET (the comparator is disabled). To save power the program should also disable the comparator before the μC enters the HALT/IDLE modes. Disabling the comparator will turn off the constant current source and the $V_{CC}/2$ reference, disconnect the comparator output from the T2B input and pin I3 or I7 and remove the low on I1 caused by CMPNEG.

It is often useful for the user's program to read the result of a comparator operation. Since I1 is always selected to be COMPIN—when the comparator is enabled (CMPEN = 1), the comparator output can be read internally by reading bit 1 (CMPRD) of register PORTI (RAM address 0xD7).

The following table lists the comparator inputs and outputs versus the value of the CMPISEL0/1/2 bits. The output will only be driven if the CMPOE bit is set to 1.



TL/DD/12869–14

FIGURE 11. COP884CT Analog Function Block

Analog Function Block (Continued)

TABLE V. Comparator Input Selection

Control Bit			Comparator Input Source		Comparator Output
CMPISEL2	CMPISEL1	CMPISEL0	Neg. Input	Pos. Input	
0	0	0	I1	I2	I3
0	0	1	I1	I2	I7
0	1	0	I1	I3	I7
0	1	1	I1	I0	I7
1	0	0	I1	I4	I7
1	0	1	I1	I5	I7
1	1	0	I1	I6	I7
1	1	1	I1	$V_{CC}/2$ Ref.	I7

Reset

The state of the Comparator Block immediately after RESET is as follows:

1. The CMPSL Register is set to all zeros
2. The Comparator is disabled
3. The Constant Current Source is disabled
4. CMPNEG is turned off
5. The Port I inputs are electrically isolated from the comparator
6. The T2B input is as normally selected by the T2CNTRL Register
7. CMPISEL0–CMPISEL2 are set to zero
8. All Port I inputs are selected to the default digital input mode

The comparator outputs have the same specification as Ports L and G except that the rise and fall times are symmetrical.

Interrupts

The device supports a vectored interrupt scheme. It supports a total of twelve interrupt sources. The following table lists all the possible device interrupt sources, their arbitration rankings and the memory locations reserved for the interrupt vector for each source.

Two bytes of program memory space are reserved for each interrupt source. All interrupt sources except the software interrupt are maskable. Each of the maskable interrupts have an Enable bit and one or more Pending bits. A maskable interrupt is active if its associated enable and pending bits are set. If GIE = 1 and an interrupt is active, then the processor will be interrupted as soon as it is ready to start executing an instruction except if the above conditions happen during the Software Trap service routine. This exception is described in the Software Trap sub-section.

The interruption process is accomplished with the INTR instruction (opcode 00), which is jammed inside the Instruction Register and replaces the opcode about to be executed. The following steps are performed for every interrupt:

1. The GIE (Global Interrupt Enable) bit is reset.
2. The address of the instruction about to be executed is pushed into the stack.
3. The PC (Program Counter) branches to address 00FF.

This procedure takes 7 t_C cycles to execute.

TABLE VI. Interrupt Vector Table

ARBITRATION RANKING	SOURCE DESCRIPTION		VECTOR* ADDRESS (Hi-Low Byte)
(1) Highest	Software	INTR Instruction	0yFE–0yFF
(2)	Reserved		0yFC–0yFD
(3)	External	G0	0yFA–0yFB
(4)	Timer T0	Idle Timer	0yF8–0yF9
(5)	Timer T1	T1A/Underflow	0yF6–0yF7
(6)	Timer T1	T1B	0yF4–0yF5
(7)	Microwire/Plus	Busy Low	0yF2–0yF3
(8)	Reserved		0yF0–0yF1
(9)	Reserved		0yEE–0yEF
(10)	Reserved		0yEC–0yED
(11)	High Speed Capture Timer	Capture Overflow/ Capture Pending	0yEA–0yEB
(12)	Reserved		0yE8–0yE9
(13)	Reserved		0yE6–0yE7
(14)	Reserved		0yE4–0yE5
(15)	Port L/Wakeup	Port L Edge	0yE2–0yE3
(16) Lowest	Default VIS	Reserved	0yE0–0yE1

*y is a variable which represents the VIS block. VIS and the vector table must be located in the same 256-byte block except if VIS is located at the last address of a block. In this case, the table must be in the next block.

Interrupts (Continued)

At this time, since GIE = 0, other maskable interrupts are disabled. The user is now free to do whatever context switching is required by saving the context of the machine in the stack with PUSH instructions. The user would then program a VIS (Vector Interrupt Select) instruction in order to branch to the interrupt service routine of the highest priority interrupt enabled and pending at the time of the VIS. Note that this is not necessarily the interrupt that caused the branch to address location 00FF Hex prior to the context switching.

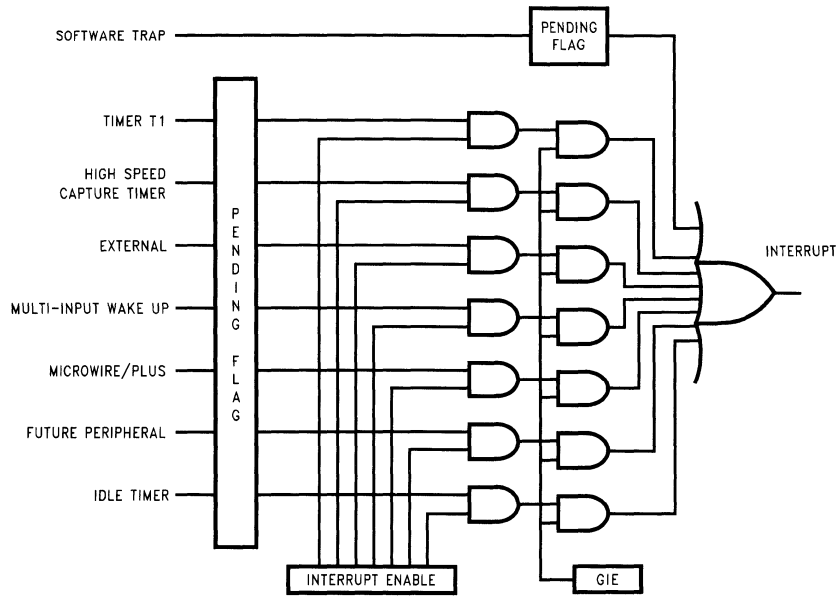
Thus, if an interrupt with a higher rank than the one which caused the interruption becomes active before the decision of which interrupt to service is made by the VIS, then the interrupt with the higher rank will override any lower ones and will be acknowledged. The lower priority interrupt(s) are still pending, however, and will cause another interrupt immediately following the completion of the interrupt service routine associated with the higher priority interrupt just serviced. This lower priority interrupt will occur immediately following the RETI (Return from Interrupt) instruction at the end of the interrupt service routine just completed.

Inside the interrupt service routine, the associated pending bit has to be cleared by software. The RETI (Return from Interrupt) instruction at the end of the interrupt service routine will set the GIE (Global Interrupt Enable) bit, allowing the processor to be interrupted again if another interrupt is active and pending.

The VIS instruction looks at all the active interrupts at the time it is executed and performs an indirect jump to the beginning of the service routine of the one with the highest rank.

The addresses of the different interrupt service routines, called vectors, are chosen by the user and stored in ROM in a table starting at 01E0 (assuming that VIS is located between 00FF and 01DF). The vectors are 15-bit wide and therefore occupy 2 ROM locations.

VIS and the vector table must be located in the same 256-byte block (0y00 to 0yFF) except if VIS is located at the last address of a block. In this case, the table must be in the next block. The vector table cannot be inserted in the first 256-byte block (y ≠ 0).



TL/DD/12869–15

FIGURE 12. Interrupt Block Diagram

Interrupts (Continued)

The vector of the maskable interrupt with the lowest rank is located at 0yE0 (Hi-Order byte) and 0yE1 (Lo-Order byte) and so forth in increasing rank number. The vector of the maskable interrupt with the highest rank is located at 0yFA (Hi-Order byte) and 0yFB (Lo-Order byte).

The Software Trap has the highest rank and its vector is located at 0yFE and 0yFF.

If, by accident, a VIS gets executed and no interrupt is active, then the PC (Program Counter) will branch to a vector located at 0yE0-0yE1.

WARNING

A Default VIS interrupt handler routine must be present. As a minimum, this handler should confirm that the GIE bit is cleared (this indicates that the interrupt sequence has been taken), take care of any required housekeeping, restore context and return. Some sort of Warm Restart procedure should be implemented. These events can occur without any error on the part of the system designer or programmer.

Note: There is always the possibility of an interrupt occurring during an instruction which is attempting to reset the GIE bit or any other interrupt enable bit. If this occurs when a single cycle instruction is being used to reset the interrupt enable bit, the interrupt enable bit will be reset but an interrupt may still occur. This is because interrupt processing is started at the same time as the interrupt bit is being reset. To avoid this scenario, the user should always use a two, three, or four cycle instruction to reset interrupt enable bits.

Figure 12 shows the Interrupt block diagram.

SOFTWARE TRAP

The Software Trap (ST) is a special kind of non-maskable interrupt which occurs when the INTR instruction (used to acknowledge interrupts) is fetched from ROM and placed inside the instruction register. This may happen when the PC is pointing beyond the available ROM address space or when the stack is over-popped.

When an ST occurs, the user can re-initialize the stack pointer and do a recovery procedure (similar to reset, but not necessarily containing all of the same initialization procedures) before restarting.

The occurrence of an ST is latched into the ST pending bit. The GIE bit is not affected and the ST pending bit (**not accessible by the user**) is used to inhibit other interrupts and to direct the program to the ST service routine with the VIS instruction. The RPND instruction is used to clear the software interrupt pending bit. This pending bit is also cleared on reset.

The ST has the highest rank among all interrupts.

Nothing (except another ST) can interrupt an ST being serviced.

WATCHDOG

The devices contain a WATCHDOG and clock monitor. The WATCHDOG is designed to detect the user program getting stuck in infinite loops resulting in loss of program control or "runaway" programs. The Clock Monitor is used to detect the absence of a clock or a very slow clock below a specified rate on the CKI pin.

The WATCHDOG consists of two independent logic blocks: WD UPPER and WD LOWER. WD UPPER establishes the upper limit on the service window and WD LOWER defines the lower limit of the service window.

Servicing the WATCHDOG consists of writing a specific value to a WATCHDOG Service Register named WDSVR which is memory mapped in the RAM. This value is composed of three fields, consisting of a 2-bit Window Select, a 5-bit Key Data field, and the 1-bit Clock Monitor Select field. Table VII shows the WDSVR register.

TABLE VII. WATCHDOG Service Register (WDSVR)

Window Select		Key Data					Clock Monitor
X	X	0	1	1	0	0	Y
7	6	5	4	3	2	1	0

The lower limit of the service window is fixed at 2048 instruction cycles. Bits 7 and 6 of the WDSVR register allow the user to pick an upper limit of the service window.

Table VIII shows the four possible combinations of lower and upper limits for the WATCHDOG service window. This flexibility in choosing the WATCHDOG service window prevents any undue burden on the user software.

Bits 5, 4, 3, 2 and 1 of the WDSVR register represent the 5-bit Key Data field. The key data is fixed at 01100. Bit 0 of the WDSVR Register is the Clock Monitor Select bit.

TABLE VIII. WATCHDOG Service Window Select

WDSVR Bit 7	WDSVR Bit 6	Service Window (Lower-Upper Limits)
0	0	2k–8k t_C Cycles
0	1	2k–16k t_C Cycles
1	0	2k–32k t_C Cycles
1	1	2k–64k t_C Cycles

Clock Monitor

The Clock Monitor aboard the device can be selected or deselected under program control. The Clock Monitor is guaranteed not to reject the clock if the instruction cycle clock ($1/t_C$) is greater or equal to 10 kHz. This equates to a clock input rate on CKI of greater or equal to 100 kHz.

WATCHDOG Operation

The WATCHDOG and Clock Monitor are disabled during reset. The device comes out of reset with the WATCHDOG armed, the WATCHDOG Window Select bits (bits 6, 7 of the WDSVR Register) set, and the Clock Monitor bit (bit 0 of the WDSVR Register) enabled. Thus, a Clock Monitor error will occur after coming out of reset, if the instruction cycle clock frequency has not reached a minimum specified value, including the case where the oscillator fails to start.

The WDSVR register can be written to only once after reset and the key data (bits 5 through 1 of the WDSVR Register) must match to be a valid write. This write to the WDSVR register involves two irrevocable choices: (i) the selection of

WATCHDOG Operation (Continued)

the WATCHDOG service window (ii) enabling or disabling of the Clock Monitor. Hence, the first write to WDSVR Register involves selecting or deselecting the Clock Monitor, select the WATCHDOG service window and match the WATCHDOG key data. Subsequent writes to the WDSVR register will compare the value being written by the user to the WATCHDOG service window value and the key data (bits 7 through 1) in the WDSVR Register. Table IX shows the sequence of events that can occur.

The user must service the WATCHDOG at least once before the upper limit of the service window expires. The WATCHDOG may not be serviced more than once in every lower limit of the service window. The user may service the WATCHDOG as many times as wished in the time period between the lower and upper limits of the service window. The first write to the WDSVR Register is also counted as a WATCHDOG service.

The WATCHDOG has an output pin associated with it. This is the WDOUT pin, on pin 1 of the port G. WDOUT is active low. The WDOUT pin is in the high impedance state in the inactive state. Upon triggering the WATCHDOG, the logic will pull the WDOUT (G1) pin low for an additional 16 t_C–32 t_C cycles after the signal level on WDOUT pin goes below the lower Schmitt trigger threshold. After this delay, the device will stop forcing the WDOUT output low. The WATCHDOG service window will restart when the WDOUT pin goes high. It is recommended that the user tie the WDOUT pin back to V_{CC} through a resistor in order to pull WDOUT high.

A WATCHDOG service while the WDOUT signal is active will be ignored. The state of the WDOUT pin is not guaranteed on reset, but if it powers up low then the WATCHDOG will time out and WDOUT will enter high impedance state.

The Clock Monitor forces the G1 pin low upon detecting a clock frequency error. The Clock Monitor error will continue until the clock frequency has reached the minimum specified value, after which the G1 output will enter the high impedance TRI-STATE mode following 16 t_C–32 t_C clock cycles. The Clock Monitor generates a continual Clock Monitor error if the oscillator fails to start, or fails to reach the minimum specified frequency. The specification for the Clock Monitor is as follows:

$1/t_C > 10$ kHz—No clock rejection.
$1/t_C < 10$ Hz—Guaranteed clock rejection.

WATCHDOG AND CLOCK MONITOR SUMMARY

The following salient points regarding the WATCHDOG and CLOCK MONITOR should be noted:

- Both the WATCHDOG and CLOCK MONITOR detector circuits are inhibited during RESET.
- Following RESET, the WATCHDOG and CLOCK MONITOR are both enabled, with the WATCHDOG having the maximum service window selected.
- The WATCHDOG service window and CLOCK MONITOR enable/disable option can only be changed once, during the initial WATCHDOG service following RESET.
- The initial WATCHDOG service must match the key data value in the WATCHDOG Service register WDSVR in order to avoid a WATCHDOG error.
- Subsequent WATCHDOG services must match all three data fields in WDSVR in order to avoid WATCHDOG errors.
- The correct key data value cannot be read from the WATCHDOG Service register WDSVR. Any attempt to read this key data value of 01100 from WDSVR will read as key data value of all 0's.
- The WATCHDOG detector circuit is inhibited during both the HALT and IDLE modes.
- The CLOCK MONITOR detector circuit is active during both the HALT and IDLE modes. Consequently, the device inadvertently entering the HALT mode will be detected as a CLOCK MONITOR error (provided that the CLOCK MONITOR enable option has been selected by the program).
- With the single-pin R/C oscillator mask option selected and the CLKDLY bit reset, the WATCHDOG service window will resume following HALT mode from where it left off before entering the HALT mode.
- With the crystal oscillator mask option selected, or with the single-pin R/C oscillator mask option selected and the CLKDLY bit set, the WATCHDOG service window will be set to its selected value from WDSVR following HALT. Consequently, the WATCHDOG should not be serviced for at least 2048 instruction cycles following HALT, but must be serviced within the selected window to avoid a WATCHDOG error.
- The IDLE timer T0 is not initialized with RESET.
- The user can sync in to the IDLE counter cycle with an IDLE counter (T0) interrupt or by monitoring the T0PND flag. The T0PND flag is set whenever the thirteenth bit of the IDLE counter toggles (every 4096 instruction cycles). The user is responsible for resetting the T0PND flag.
- A hardware WATCHDOG service occurs just as the device exits the IDLE mode. Consequently, the WATCHDOG should not be serviced for at least 2048 instruction cycles following IDLE, but must be serviced within the selected window to avoid a WATCHDOG error.
- Following RESET, the initial WATCHDOG service (where the service window and the CLOCK MONITOR enable/disable must be selected) may be programmed anywhere within the maximum service window (65,536 instruction cycles) initialized by RESET. Note that this initial WATCHDOG service may be programmed within the initial 2048 instruction cycles without causing a WATCHDOG error.

TABLE IX. WATCHDOG Service Actions

Key Data	Window Data	Clock Monitor	Action
Match	Match	Match	Valid Service: Restart Service Window
Don't Care	Mismatch	Don't Care	Error: Generate WATCHDOG Output
Mismatch	Don't Care	Don't Care	Error: Generate WATCHDOG Output
Don't Care	Don't Care	Mismatch	Error: Generate WATCHDOG Output

Detection of Illegal Conditions

The device can detect various illegal conditions resulting from coding errors, transient noise, power supply voltage drops, runaway programs, etc.

Reading of undefined ROM gets zeros. The opcode for software interrupt is 00. If the program fetches instructions from undefined ROM, this will force a software interrupt, thus signaling that an illegal condition has occurred.

The subroutine stack grows down for each call (jump to subroutine), interrupt, or PUSH, and grows up for each return or POP. The stack pointer is initialized to RAM location 06F Hex during reset. Consequently, if there are more returns than calls, the stack pointer will point to addresses 070 and 071 Hex (which are undefined RAM). Undefined RAM from addresses 070 to 07F (Segment 0), and all other segments (i.e., Segments 4... etc.) is read as all 1's, which in turn will cause the program to return to address 7FFF Hex. This is an undefined ROM location and the instruction fetched (all 0's) from this location will generate a software interrupt signaling an illegal condition.

Thus, the chip can detect the following illegal conditions:

1. Executing from undefined ROM
2. Over "POP"ing the stack by having more returns than calls.

When the software interrupt occurs, the user can re-initialize the stack pointer and do a recovery procedure before restarting (this recovery program is probably similar to that following reset, but might not contain the same program initialization procedures). The recovery program should reset the software interrupt pending bit using the RPND instruction.

MICROWIRE/PLUS

MICROWIRE/PLUS is a serial synchronous communications interface. The MICROWIRE/PLUS capability enables the device to interface with any of National Semiconductor's MICROWIRE peripherals (i.e. A/D converters, display drivers, E2PROMs etc.) and with other microcontrollers which support the MICROWIRE interface. It consists of an 8-bit serial shift register (SIO) with serial data input (SI), serial data output (SO) and serial shift clock (SK). *Figure 13* shows a block diagram of the MICROWIRE/PLUS logic.

The shift clock can be selected from either an internal source or an external source. Operating the MICROWIRE/PLUS arrangement with the internal clock source is called the Master mode of operation. Similarly, operating the MICROWIRE/PLUS arrangement with an external shift clock is called the Slave mode of operation.

The CNTRL register is used to configure and control the MICROWIRE/PLUS mode. To use the MICROWIRE/PLUS, the MSEL bit in the CNTRL register is set to one. In the master mode, the SK clock rate is selected by the two bits, SL0 and SL1, in the CNTRL register. Table X details the different clock rates that may be selected.

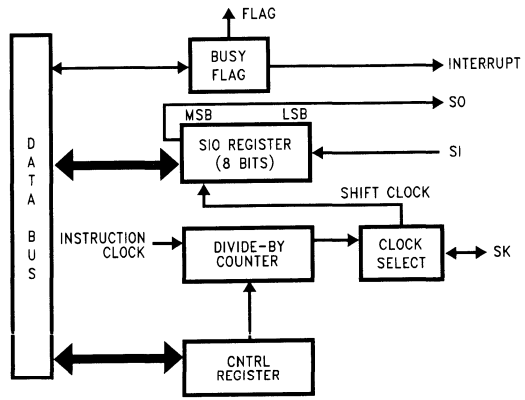
TABLE X. MICROWIRE/PLUS Master Mode Clock Select

SL1	SL0	SK period
0	0	2 X t_C
0	1	4 X t_C
1	x	8 X t_C

Where t_C is the instruction cycle clock

MICROWIRE/PLUS OPERATION

Setting the BUSY bit in the PSW register causes the MICROWIRE/PLUS to start shifting the data. It gets reset when eight data bits have been shifted. The user may reset the BUSY bit by software to allow less than 8 bits to shift. If enabled, an interrupt is generated when eight data bits have been shifted. The device may enter the MICROWIRE/PLUS mode either as a Master or as a Slave. *Figure 14* shows how two devices, microcontrollers and several peripherals may be interconnected using the MICROWIRE/PLUS arrangements.



TL/DD/12869–16

FIGURE 13. MICROWIRE/PLUS Block Diagram

MICROWIRE/PLUS (Continued)

WARNING

The SIO register should only be loaded when the SK clock is low. Loading the SIO register while the SK clock is high will result in undefined data in the SIO register. SK clock is normally low when not shifting.

Setting the BUSY flag when the input SK clock is high in the MICROWIRE/PLUS slave mode may cause the current SK clock for the SIO shift register to be narrow. For safety, the BUSY flag should only be set when the input SK clock is low.

MICROWIRE/PLUS Master Mode Operation

In the MICROWIRE/PLUS Master mode of operation the shift clock (SK) is generated internally. The MICROWIRE Master always initiates all data exchanges. The MSEL bit in the CNTRL register must be set to enable the SO and SK functions onto the G Port. The SO and SK pins must also be selected as outputs by setting appropriate bits in the Port G configuration register. Table XI summarizes the bit settings required for Master mode of operation.

MICROWIRE/PLUS Slave Mode Operation

In the MICROWIRE/PLUS Slave mode of operation the SK clock is generated by an external source. Setting the MSEL bit in the CNTRL register enables the SO and SK functions onto the G Port. The SK pin must be selected as an input and the SO pin is selected as an output pin by setting and resetting the appropriate bits in the Port G configuration register. Table XI summarizes the settings required to enter the Slave mode of operation.

The user must set the BUSY flag immediately upon entering the Slave mode. This will ensure that all data bits sent by the Master will be shifted properly. After eight clock pulses the BUSY flag will be cleared and the sequence may be repeated.

TABLE XI. MICROWIRE Mode Settings

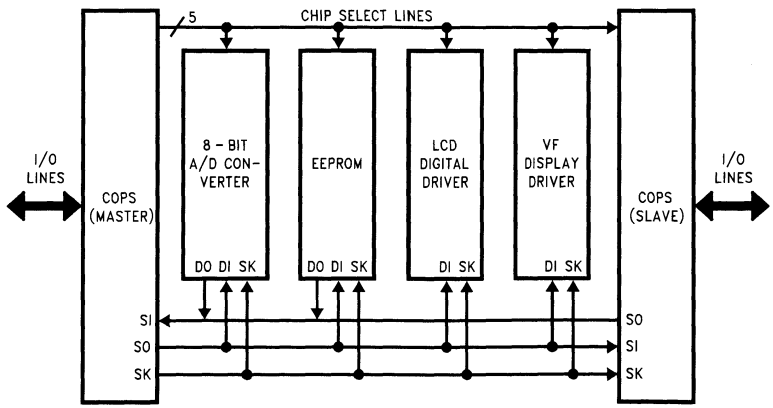
G4 (SO) Config. Bit	G5 (SK) Config. Bit	G4 Fun.	G5 Fun.4	Operation
1	1	SO	Int. SK	MICROWIRE/PLUS Master
0	1	TRI-STATE	Int. SK	MICROWIRE/PLUS Master
1	0	SO	Ext. SK	MICROWIRE/PLUS Slave
0	0	TRI-STATE	Ext. SK	MICROWIRE/PLUS Slave

This table assumes that the control flag MSEL is set.

Alternate SK Phase Operation

The device allows either the normal SK clock or an alternate phase SK clock to shift data in and out of the SIO register. In both the modes the SK is normally low. In the normal mode data is shifted in on the rising edge of the SK clock and the data is shifted out on the falling edge of the SK clock. The SIO register is shifted on each falling edge of the SK clock. In the alternate SK phase operation, data is shifted in on the falling edge of the SK clock and shifted out on the rising edge of the SK clock.

A control flag, SKSEL, allows either the normal SK clock or the alternate SK clock to be selected. Resetting SKSEL causes the MICROWIRE/PLUS logic to be clocked from the normal SK signal. Setting the SKSEL flag selects the alternate SK clock. The SKSEL is mapped into the G6 configuration bit. The SKSEL flag will power up in the reset condition, selecting the normal SK signal.



TL/DD/12869–17

FIGURE 14. MICROWIRE/PLUS Application

Memory Map

All RAM, ports and registers (except A and PC) are mapped into data memory address space.

Address S/ADD REG	Contents
0000 to 006F	On-Chip RAM bytes (112 bytes)
0070 to 007F	Unused RAM Address Space (Reads As All Ones)
xx80 to xxAF	Unused RAM Address Space (Reads Undefined Data)
xxB0	Reserved
XXB1	Reserved
xxB2	Reserved
xxB3	Reserved
xxB4	Reserved
xxB5	Reserved
xxB6	Reserved
xxB7	Comparator Select Register (CMPSL)
xxB8 to xxBF	Reserved
xxC0	Reserved
xxC1	Reserved
xxC2	Reserved
xxC3	Reserved
xxC4	Reserved
xxC5	Reserved
xxC6	Reserved
xxC7	WATCHDOG Service Register (Reg:WDSVR)
xxC8	MIWU Edge Select Register (Reg:WKEDG)
xxC9	MIWU Enable Register (Reg:WKEN)
xxCA	MIWU Pending Register (Reg:WKPND)
xxCB	Reserved
xxCC	CAPTLO (Capture Timer Low-Byte)
xxCD	CAPTHI (Capture Timer High-Byte)
xxCE	CAPCNTL (Capture Timer Control Register)
xxCF	Idle Timer Control Register
xxD0	Port L Data Register
xxD1	Port L Configuration Register
xxD2	Port L Input Pins (Read Only)
xxD3	Reserved
xxD4	Port G Data Register
xxD5	Port G Configuration Register
xxD6	Port G Input Pins (Read Only)
xxD7	Port I Input Pins (Read Only)
xxD8	Reserved
xxD9	Reserved
xxDA	Reserved
xxDB	Reserved
xxDC	Port D
xxDD to DF	Reserved
xxE0 to xxE5	Reserved
xxE6	Timer T1 Autoload Register T1RB Lower Byte
xxE7	Timer T1 Autoload Register T1RB Upper Byte
xxE8	ICNTRL Register
xxE9	MICROWIRE/PLUS Shift Register
xxEA	Timer T1 Lower Byte
xxEB	Timer T1 Upper Byte
xxEC	Timer T1 Autoload Register T1RA Lower Byte
xxED	Timer T1 Autoload Register T1RA Upper Byte
xxEE	CNTRL Control Register
xxEF	PSW Register
xxF0 to FB	On-Chip RAM Mapped as Registers
xxFC	X Register
xxFD	SP Register
xxFE	B Register
xxFF	Reserved
0100-017F	Reserved

Reading memory locations 0070H-007FH (Segment 0) will return all ones. Reading unused memory locations 0080H-00AFH (Segment 0) will return undefined data. Reading memory locations from other Segments (i.e., Segment 2, Segment 3, . . . etc.) will return all ones.

Addressing Modes

There are ten addressing modes, six for operand addressing and four for transfer of control.

OPERAND ADDRESSING MODES

Register Indirect

This is the "normal" addressing mode. The operand is the data memory addressed by the B pointer or X pointer.

Register Indirect (with auto post increment or decrement of pointer)

This addressing mode is used with the LD and X instructions. The operand is the data memory addressed by the B pointer or X pointer. This is a register indirect mode that automatically post increments or decrements the B or X register after executing the instruction.

Direct

The instruction contains an 8-bit address field that directly points to the data memory for the operand.

Immediate

The instruction contains an 8-bit immediate field as the operand.

Short Immediate

This addressing mode is used with the Load B Immediate instruction. The instruction contains a 4-bit immediate field as the operand.

Indirect

This addressing mode is used with the LAID instruction. The contents of the accumulator are used as a partial address (lower 8 bits of PC) for accessing a data operand from the program memory.

TRANSFER OF CONTROL ADDRESSING MODES

Relative

This mode is used for the JP instruction, with the instruction field being added to the program counter to get the new program location. JP has a range from −31 to +32 to allow a 1-byte relative jump (JP + 1 is implemented by a NOP instruction). There are no "pages" when using JP, since all 15 bits of PC are used.

Absolute

This mode is used with the JMP and JSR instructions, with the instruction field of 12 bits replacing the lower 12 bits of the program counter (PC). This allows jumping to any location in the current 4k program memory segment.

Absolute Long

This mode is used with the JMPL and JSRL instructions, with the instruction field of 15 bits replacing the entire 15 bits of the program counter (PC). This allows jumping to any location up to 32k in the program memory space.

Indirect

This mode is used with the JID instruction. The contents of the accumulator are used as a partial address (lower 8 bits of PC) for accessing a location in the program memory. The contents of this program memory location serve as a partial address (lower 8 bits of PC) for the jump to the next instruction.

Note: The VIS is a special case of the Indirect Transfer of Control addressing mode, where the double byte vector associated with the interrupt is transferred from adjacent addresses in the program memory into the program counter (PC) in order to jump to the associated interrupt service routine.

Instruction Set

Register and Symbol Definition

	Registers
A	8-Bit Accumulator Register
B	8-Bit Address Register
X	8-Bit Address Register
SP	8-Bit Stack Pointer Register
PC	15-Bit Program Counter Register
PU	Upper 7 Bits of PC
PL	Lower 8 Bits of PC
C	1-Bit of PSW Register for Carry
HC	1-Bit of PSW Register for Half Carry
GIE	1-Bit of PSW Register for Global Interrupt Enable
VU	Interrupt Vector Upper Byte
VL	Interrupt Vector Lower Byte
	Symbols
[B]	Memory Indirectly Addressed by B Register
[X]	Memory Indirectly Addressed by X Register
MD	Direct Addressed Memory
Mem	Direct Addressed Memory or [B]
Meml	Direct Addressed Memory or [B] or Immediate Data
Imm	8-Bit Immediate Data
Reg	Register Memory: Addresses F0 to FF (Includes B, X and SP)
Bit	Bit Number (0 to 7)
←	Loaded with
⟷	Exchanged with

Instruction Set

ADD	A,Meml	ADD	A ← A + Meml
ADC	A,Meml	ADD with Carry	A ← A + Meml + C, C ← Carry, HC ← Half Carry
SUBC	A,Meml	Subtract with Carry	A ← A − Meml + C, C ← Carry, HC ← Half Carry
AND	A,Meml	Logical AND	A ← A and $\overline{\text{Meml}}$
ANDSZ	A,Imm	Logical AND Immed., Skip if Zero	Skip next if (A and Imm) = 0
OR	A,Meml	Logical OR	A ← A or Meml
XOR	A,Meml	Logical EXclusive OR	A ← A xor Meml
IFEQ	MD,Imm	IF EQual	Compare MD and Imm, Do next if MD = Imm
IFEQ	A,Meml	IF EQual	Compare A and Meml, Do next if A = Meml
IFNE	A,Meml	IF Not Equal	Compare A and Meml, Do next if A ≠ Meml
IFGT	A,Meml	IF Greater Than	Compare A and Meml, Do next if A > Meml
IFBNE	#	If B Not Equal	Do next if lower 4 bits of B ≠ Imm
DRSZ	Reg	Decrement Reg., Skip if Zero	Reg ← Reg − 1, Skip if Reg = 0
SBIT	#,Mem	Set BIT	1 to bit, Mem (bit = 0 to 7 immediate)
RBIT	#,Mem	Reset BIT	0 to bit, Mem
IFBIT	#,Mem	IF BIT	If bit #,A or Mem is true do next instruction
RPND		Reset PeNDing Flag	Reset Software Interrupt Pending Flag
X	A,Mem	EXchange A with Memory	A ⟷ Mem
X	A,[X]	EXchange A with Memory [X]	A ⟷ [X]
LD	A,Meml	LoaD A with Memory	A ← Meml
LD	A,[X]	LoaD A with Memory [X]	A ← [X]
LD	B,Imm	LoaD B with Immed.	B ← Imm
LD	Mem,Imm	LoaD Memory Immed	Mem ← Imm
LD	Reg,Imm	LoaD Register Memory Immed.	Reg ← Imm
X	A, [B±]	EXchange A with Memory [B]	A ⟷ [B], (B ← B ±1)
X	A, [X±]	EXchange A with Memory [X]	A ⟷ [X], (X ← X±1)
LD	A, [B±]	LoaD A with Memory [B]	A ← [B], (B ← B±1)
LD	A, [X±]	LoaD A with Memory [X]	A ← [X], (X ← X±1)
LD	[B±],Imm	LoaD Memory [B] Immed.	[B] ← Imm, (B ← B±1)
CLR	A	CLeaR A	A ← 0
INC	A	INCrement A	A ← A + 1
DEC	A	DECrement A	A ← A − 1
LAID		Load A InDirect from ROM	A ← ROM (PU,A)
DCOR	A	Decimal CORrect A	A ← BCD correction of A (follows ADC, SUBC)
RRC	A	Rotate A Right thru C	C → A7 → ... → A0 → C
RLC	A	Rotate A Left thru C	C ← A7 ← ... ← A0 ← C
SWAP	A	SWAP nibbles of A	A7...A4 ⟷ A3...A0
SC		Set C	C ← 1, HC ← 1
RC		Reset C	C ← 0, HC ← 0
IFC		IF C	IF C is true, do next instruction
IFNC		IF Not C	If C is not true, do next instruction
POP	A	POP the stack into A	SP ← SP + 1, A ← [SP]
PUSH	A	PUSH A onto the stack	[SP] ← A, SP ← SP − 1
VIS		Vector to Interrupt Service Routine	PU ← [VU], PL ← [VL]
JMPL	Addr.	Jump absolute Long	PC ← ii (ii = 15 bits, 0 to 32k)
JMP	Addr	Jump absolute	PC9...0 ← i (i = 12 bits)
JP	Disp.	Jump relative short	PC ← PC + r (r is −31 to +32, except 1)
JSRL	Addr.	Jump SubRoutine Long	[SP] ← PL, [SP-1] ← PU,SP-2, PC ← ii
JSR	Addr	Jump SubRoutine	[SP] ← PL, [SP-1] ← PU,SP-2, PC9...0 ← i
JID		Jump InDirect	PL ← ROM (PU,A)
RET		RETurn from subroutine	SP + 2, PL ← [SP], PU ← [SP-1]
RETSK		RETurn and SKip	SP + 2, PL ← [SP],PU ← [SP-1], skip next instruction
RETI		RETurn from Interrupt	SP + 2, PL ← [SP],PU ← [SP-1],GIE ← 1
INTR		Generate an Interrupt	[SP] ← PL, [SP-1] ← PU, SP-2, PC ← 0FF
NOP		No OPeration	PC ← PC + 1

Instruction Execution Time

Most instructions are single byte (with immediate addressing mode instructions taking two bytes).

Most single byte instructions take one cycle time to execute.

Skipped instructions require x number of cycles to be skipped, where x equals the number of bytes in the skipped instruction opcode.

See the BYTES and CYCLES per INSTRUCTION table for details.

Bytes and Cycles per Instruction

The following table shows the number of bytes and cycles for each instruction in the format of byte/cycle.

Arithmetic and Logic Instructions

	[B]	Direct	Immed
ADD	1/1	3/4	2/2
ADC	1/1	3/4	2/2
SUBC	1/1	3/4	2/2
AND	1/1	3/4	2/2
OR	1/1	3/4	2/2
XOR	1/1	3/4	2/2
IFEQ	1/1	3/4	2/2
IFGT	1/1	3/4	2/2
IFBNE	1/1		
DRSZ	1/1	1/3	
SBIT	1/1	3/4	
RBIT	1/1	3/4	
IFBIT	1/1	3/4	

RPND	1/1

Instructions Using A and C

CLRA	1/1
INCA	1/1
DECA	1/1
LAID	1/3
DCORA	1/1
RRCA	1/1
RLCA	1/1
SWAPA	1/1
SC	1/1
RC	1/1
IFC	1/1
IFNC	1/1
PUSHA	1/3
POPA	1/3
ANDSZ	2/2

Transfer of Control Instructions

JMPL	3/4
JMP	2/3
JP	1/3
JSRL	3/5
JSR	2/5
JID	1/3
VIS	1/5
RET	1/5
RETSK	1/5
RETI	1/5
INTR	1/7
NOP	1/1

Memory Transfer Instructions

	Register Indirect		Direct	Immed.	Register Indirect Auto Incr and Decr		
	[B]	[X]			[B+, B−]	[X+, X−]	
X A,*	1/1	1/3	2/3		1/2	1/3	
LD A,*	1/1	1/3	2/3	2/2	1/2	1/3	
LD B,Imm				1/1			(If B < 16)
LD B,Imm				2/2			(If B > 15)
LD Mem,Imm	2/2		3/3		2/2		
LD Reg,Imm			2/3				
IFEQ MD,Imm			3/3				

*Memory location addressed by B or X or directly.

Opcode Table

UPPER NIBBLE																LOWER NIBBLE
F	E	D	C	B	A	9	8	7	6	5	4	3	2	1	0	
JP −15	JP −31	LD 0F0, #i	DRSZ 0F0	RRCA	RC	ADC A, #i	ADC A,[B]	IFBIT 0,[B]	ANDSZ A, #i	LD B, #0F	IFBNE 0	JSR x000−x0FF	JMP x000−x0FF	JP + 17	INTR	0
JP −14	JP −30	LD 0F1, #i	DRSZ 0F1	*	SC	SUBC A, #i	SUBC A,[B]	IFBIT 1,[B]	*	LD B, #0E	IFBNE 1	JSR x100−x1FF	JMP x100−x1FF	JP + 18	JP + 2	1
JP −13	JP −29	LD 0F2, #i	DRSZ 0F2	X A, [X+]	X A, [B+]	IFEQ A, #i	IFEQ A,[B]	IFBIT 2,[B]	*	LD B, #0D	IFBNE 2	JSR x200−x2FF	JMP x200−x2FF	JP + 19	JP + 3	2
JP −12	JP −28	LD 0F3, #i	DRSZ 0F3	X A, [X−]	X A, [B−]	IFGT A, #i	IFGT A,[B]	IFBIT 3,[B]	*	LD B, #0C	IFBNE 3	JSR x300−x3FF	JMP x300−x3FF	JP + 20	JP + 4	3
JP −11	JP −27	LD 0F4, #i	DRSZ 0F4	VIS	LAID	ADD A, #i	ADD A,[B]	IFBIT 4,[B]	CLRA	LD B, #0B	IFBNE 4	JSR x400−x4FF	JMP x400−x4FF	JP + 21	JP + 5	4
JP −10	JP −26	LD 0F5, #i	DRSZ 0F5	RPND	JID	AND A, #i	AND A,[B]	IFBIT 5,[B]	SWAPA	LD B, #0A	IFBNE 5	JSR x500−x5FF	JMP x500−x5FF	JP + 22	JP + 6	5
JP −9	JP −25	LD 0F6, #i	DRSZ 0F6	X A,[X]	X A,[B]	XOR A, #i	XOR A,[B]	IFBIT 6,[B]	DCORA	LD B, #09	IFBNE 6	JSR x600−x6FF	JMP x600−x6FF	JP + 23	JP + 7	6
JP −8	JP −24	LD 0F7, #i	DRSZ 0F7	*	*	OR A, #i	OR A,[B]	IFBIT 7,[B]	PUSHA	LD B, #08	IFBNE 7	JSR x700−x7FF	JMP x700−x7FF	JP + 24	JP + 8	7
JP −7	JP −23	LD 0F8, #i	DRSZ 0F8	NOP	RLCA	LD A, #i	IFC	SBIT 0,[B]	RBIT 0,[B]	LD B, #07	IFBNE 8	JSR x800−x8FF	JMP x800−x8FF	JP + 25	JP + 9	8
JP −6	JP −22	LD 0F9, #i	DRSZ 0F9	IFNE A,[B]	IFEQ Md, #i	IFNE A, #i	IFNC	SBIT 1,[B]	RBIT 1,[B]	LD B, #06	IFBNE 9	JSR x900−x9FF	JMP x900−x9FF	JP + 26	JP + 10	9
JP −5	JP −21	LD 0FA, #i	DRSZ 0FA	LD A, [X+]	LD A, [B+]	LD [B+], #i	INCA	SBIT 2,[B]	RBIT 2,[B]	LD B, #05	IFBNE 0A	JSR xA00−xAFF	JMP xA00−xAFF	JP + 27	JP + 11	A
JP −4	JP −20	LD 0FB, #i	DRSZ 0FB	LD A, [X−]	LD A, [B−]	LD [B−], #i	DECA	SBIT 3,[B]	RBIT 3,[B]	LD B, #04	IFBNE 0B	JSR xB00−xBFF	JMP xB00−xBFF	JP + 28	JP + 12	B
JP −3	JP −19	LD 0FC, #i	DRSZ 0FC	LD Md, #i	JMPL	X A,Md	POPA	SBIT 4,[B]	RBIT 4,[B]	LD B, #03	IFBNE 0C	JSR xC00−xCFF	JMP xC00−xCFF	JP + 29	JP + 13	C
JP −2	JP −18	LD 0FD, #i	DRSZ 0FD	DIR	JSRL	LD A,Md	RETSK	SBIT 5,[B]	RBIT 5,[B]	LD B, #02	IFBNE 0D	JSR xD00−xDFF	JMP xD00−xDFF	JP + 30	JP + 14	D
JP −1	JP −17	LD 0FE, #i	DRSZ 0FE	LD A,[X]	LD A,[B]	LD [B], #i	RET	SBIT 6,[B]	RBIT 6,[B]	LD B, #01	IFBNE 0E	JSR xE00−xEFF	JMP xE00−xEFF	JP + 31	JP + 15	E
JP −0	JP −16	LD 0FF, #i	DRSZ 0FF	*	*	LD B, #i	RETI	SBIT 7,[B]	RBIT 7,[B]	LD B, #00	IFBNE 0F	JSR xF00−xFFF	JMP xF00−xFFF	JP + 32	JP + 16	F

where,

i is the immediate data

Md is a directly addressed memory location

* is an unused opcode

Note: The opcode 60 Hex is also the opcode for IFBIT #i,A

Development Support

Summary

- iceMASTER: IM-COP8/400—Full feature in-circuit emulation for all COP8 products. A full set of COP8 Basic and Feature Family device and package specific probes are available.
- COP8 Debug Module: Moderate cost in-circuit emulation and development programming unit.
- Assembler: COP8-DEV-IBMA. A DOS installable cross development Assembler, Linker, Librarian and Utility Software Development Tool Kit.
- C Compiler: COP8C. A DOS installable cross development Software Tool Kit.
- OTP/EPROM Programmer Support: Covering needs from engineering prototype, pilot production to full production environments.

IceMASTER (IM) IN-CIRCUIT EMULATION

The iceMASTER IM-COP8/400 is a full feature, PC based, in-circuit emulation tool developed and marketed by MetaLink Corporation to support the whole COP8 family of products. National is a resale vendor for these products.

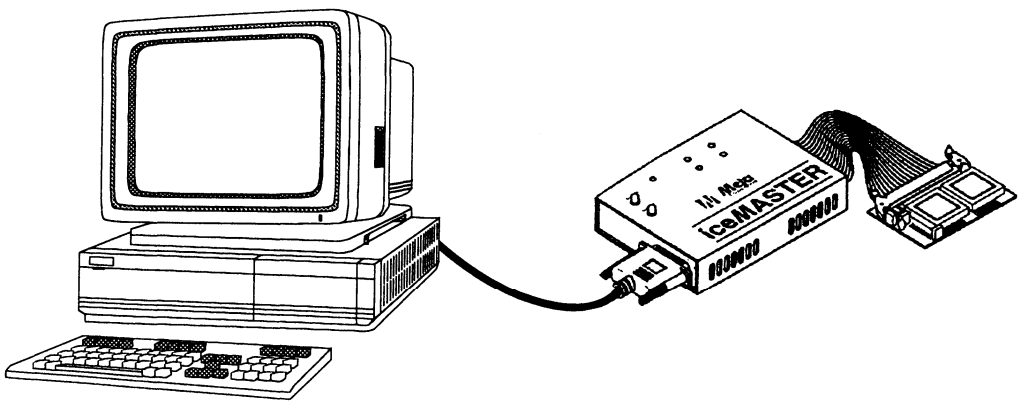
See *Figure 15* for configuration.

The iceMASTER IM-COP8/400 with its device specific COP8 Probe provides a rich feature set for developing, testing and maintaining product:

- Real-time in-circuit emulation; full 2.4V–5.5V operation range, full DC-10 MHz clock. Chip options are programmable or jumper selectable.
- Direct connection to application board by package compatible socket or surface mount assembly.
- Full 32 KByte of loadable programming space that overlays (replaces) the on-chip ROM or EPROM. On-chip RAM and I/O blocks are used directly or recreated on the probe as necessary.
- Full 4k frame synchronous trace memory. Address, instruction, and eight unspecified, circuit connectable trace lines. Display can be HLL source (e.g., C source), assembly or mixed.
- A full 64k hardware configurable break, trace on, trace off control, and pass count increment events.
- Tool set integrated interactive symbolic debugger—supports both assembler (COFF) and C Compiler (.COD) linked object formats.
- Real time performance profiling analysis; selectable bucket definition.
- Watch windows, content updated automatically at each execution break.
- Instruction by instruction memory/register changes displayed on source window when in single step operation.
- Single base unit and debugger software reconfigurable to support the entire COP8 family; only the probe personality needs to change. Debugger software is processor customized, and reconfigured from a master model file.
- Processor specific symbolic display of registers and bit level assignments, configured from master model file.
- Halt/Idle mode notification.
- On-line HELP customized to specific processor using master model file.
- Includes a copy of COP8-DEV-IBMA assembler and linker SDK.

IM Order Information

Base Unit	
IM-COP8/400-1	iceMASTER Base Unit, 110V Power Supply
IM-COP8/400-2	iceMASTER Base Unit, 220V Power Supply
iceMASTER Probe	
COP8AC-IM28N COP8AC-IM20N	28 DIP 20 DIP
Surface Mount Adapter	
MHW-SOIC28 MHW-SOIC20	28 SO 20 SO



TL/DD/12869–18

FIGURE 15. COP8 iceMASTER Environment

Development Support (Continued)

IceMASTER DEBUG MODULE (DM)

The iceMASTER Debug Module is a PC based, combination in-circuit emulation tool and COP8 based OTP/EPROM programming tool developed and marketed by MetaLink Corporation to support the whole COP8 family of products. National is a resale vendor for these products.

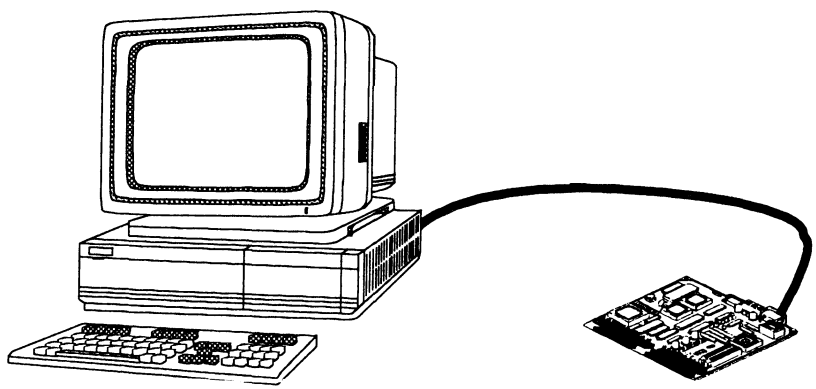
See *Figure 16* for configuration.

The iceMASTER Debug Module is a moderate cost development tool. It has the capability of in-circuit emulation for a specific COP8 microcontroller and in addition serves as a programming tool for COP8 OTP and EPROM product families. Summary of features is as follows:

- Real-time in-circuit emulation; full operating voltage range operation, full DC-10 MHz clock.
- All processor I/O pins can be cabled to an application development board with package compatible cable to socket and surface mount assembly.
- Full 32 kbyte of loadable programming space that overlays (replaces) the on-chip ROM or EPROM. On-chip RAM and I/O blocks are used directly or recreated as necessary.
- 100 frames of synchronous trace memory. The display can be HLL source (C source), assembly or mixed. The most recent history prior to a break is available in the trace memory.
- Configured break points; uses INTR instruction which is modestly intrusive.
- Software—only supported features are selectable.
- Tool set integrated interactive symbolic debugger—supports both assembler (COFF) and C Compiler (.COD) SDK linked object formats.
- Instruction by instruction memory/register changes displayed when in single step operation.
- Debugger software is processor customized, and reconfigured from a master model file.
- Processor specific symbolic display of registers and bit level assignments, configured from master model file.
- Halt/Idle mode notification.
- Programming menu supports full product line of programmable OTP and EPROM COP8 products. Program data is taken directly from the overlay RAM.
- Programming of 44PLCC and 68PLCC parts requires external programming adapters.
- Includes wallmount power supply.
- On-board V_{PP} generator from 5V input or connection to external supply supported. Requires V_{PP} level adjustment per the family programming specification (correct level is provided on an on-screen pop-down display).
- On-line HELP customized to specific processor using master model file.
- Includes a copy of COP8-DEV-IBMA assembler and linker SDK.

DM Order Information

Debug Module Unit	
COP8AC-DM	
Cable Adapters	
DM-COP8/28D	28 DIP
DM-COP8/20D	20 DIP
Surface Mount Adapters	
DM-COP8/28D-SO	28 DIP to SO
DM-COP8/20D-SO	20 DIP to SO



TL/DD/12869–19

FIGURE 16. COP-DM Environment

Development Support (Continued)

COP8 ASSEMBLER/LINKER SOFTWARE DEVELOPMENT TOOL KIT

National Semiconductor offers a relocatable COP8 macro cross assembler, linker, librarian and utility software development tool kit. Features are summarized as follows:

- Basic and Feature Family instruction set by "device" type.
- Nested macro capability.
- Extensive set of assembler directives.
- Supported on PC/DOS platform.
- Generates National standard COFF output files.
- Integrated Linker and Librarian.
- Integrated utilities to generate ROM code file outputs.
- DUMPCOFF utility.

This product is integrated as a part of MetaLink tools as a development kit, fully supported by the MetaLink debugger. It may be ordered separately or it is bundled with the MetaLink products at no additional cost.

Order Information

Assembler SDK:	
COP8-DEV-IBMA	Assembler SDK on installable 3.5" PC/DOS Floppy Disk Drive format. Periodic upgrades and most recent version is available on National's BBS and Internet.

COP8 C COMPILER

A C Compiler is developed and marketed by Byte Craft Limited. The COP8C compiler is a fully integrated development tool specifically designed to support the compact embedded configuration of the COP8 family of products.

Features are summarized as follows:

- ANSI C with some restrictions and extensions that optimize development for the COP8 embedded application.
- BITS data type extension. Register declaration #pragma with direct bit level definitions.
- C language support for interrupt routines.
- Expert system, rule based code generation and optimization.
- Performs consistency checks against the architectural definitions of the target COP8 device.
- Generates program memory code.
- Supports linking of compiled object or COP8 assembled object formats.
- Global optimization of linked code.
- Symbolic debug load format fully source level supported by the MetaLink debugger.

SINGLE CHIP OTP/EMULATOR SUPPORT

The COP8 family is supported by single chip OTP emulators. For detailed information refer to the emulator specific datasheet and the emulator selection table below:

OTP Emulator Ordering Information

Device Number	Clock Option	Package	Emulates
COP8ACC720Mx	Crystal	20 SO	COP8ACC720Mx
COP8ACC728Nx	Crystal	20 DIP	COP8ACC728Nx
COP8ACC728Mx	Crystal	20 SO	COP8ACC728Mx

x = temp. range (6, 7, 8)

INDUSTRY WIDE OTP/EPROM PROGRAMMING SUPPORT

Programming support, in addition to the MetaLink development tools, is provided by a full range of independent approved vendors to meet the needs from the engineering laboratory to full production.

Approved List:

Manufacturer	North America	Europe	Asia
BP Microsystems	(800) 225-2102 (713) 688-4600 Fax: (713) 688-0920	+49-8152-4183 +49-8856-932616	+852-234-16611 +852-2710-8121
Data I/O	(800) 426-1045 (206) 881-6444 Fax: (206) 882-1043	+44-0734-440011	Call North America
HI-LO	(510) 623-8860	Call Asia	+886-2-764-0215 Fax: +886-2-756-6403
ICE Technology	(800) 624-8949 (919) 430-7915	+44-1226-767404 Fax: 0-1226-370-434	
MetaLink	(800) 638-2423 (602) 926-0797 Fax: (602) 693-0681	+49-80 9156 96-0 Fax: +49-80 9123 86	+852-737-1800
Systems General	(408) 263-6667	+41-1-9450300	+886-2-917-3005 Fax: +886-2-911-1283
Needhams	(916) 924-8037 Fax: (916) 924-8065		

Development Support (Continued)

AVAILABLE LITERATURE

For more information, please see the COP8 Basic Family User's Manual, Literature Number 620895, COP8 Feature Family User's Manual, Literature Number 620897 and National's Family of 8-bit Microcontrollers COP8 Selection Guide, Literature Number 630009.

DIAL-A-HELPER SERVICE

Dial-A-Helper is a service provided by the Microcontroller Applications group. The Dial-A-Helper is an Electronic Information System that may be accessed as a Bulletin Board System (BBS) via data modem, as an FTP site on the Internet via standard FTP client application or as an FTP site on the Internet using a standard Internet browser such as Netscape or Mosaic.

The Dial-A-Helper system provides access to an automated information storage and retrieval system. The system capabilities include a MESSAGE SECTION (electronic mail, when accessed as a BBS) for communications to and from the Microcontroller Applications Group and a FILE SECTION which consists of several file areas where valuable application software and utilities could be found.

DIAL-A-HELPER BBS via a Standard Modem

Modem: CANADA/U.S.: (800) NSC-MICRO
(800) 672-6427
EUROPE: (+49) 0-814-135 13 32
Baud: 14.4k
Set-Up: Length: 8-Bit
Parity: None
Stop Bit: 1
Operation: 24 Hours, 7 Days

DIAL-A-HELPER via FTP

ftp nscmicro.nsc.com
user: anonymous
password: username@yourhost.site.domain

DIAL-A-HELPER via a WorldWide Web Browser

ftp://nscmicro.nsc.com

National Semiconductor on the WorldWide Web

See us on the WorldWide Web at: http://www.national.com

CUSTOMER RESPONSE CENTER

Complete product information and technical support is available from National's customer response centers.

CANADA/U.S.:	Tel:	(800) 272-9959
	email:	support @tevm2.nsc.com
EUROPE:	email:	europe.support@nsc.com
	Deutsch Tel:	+49 (0) 180-530 85 85
	English Tel:	+49 (0) 180-532 78 32
	Français Tel:	+49 (0) 180-532 93 58
	Italiano Tel:	+49 (0) 180-534 16 80
JAPAN:	Tel:	+81-043-299-2309
S.E. ASIA:	Beijing Tel:	(+86) 10-6856-8601
	Shanghai Tel:	(+86) 21-6415-4092
	Hong Kong Tel:	(+852) 2737-1600
	Korea Tel:	(+82) 2-3771-6909
	Malaysia Tel:	(+60-4) 644-9061
	Singapore Tel:	(+65) 255-2226
	Taiwan Tel:	+886-2-521-3288
AUSTRALIA:	Tel:	(+61) 3-9558-9999
INDIA:	Tel:	(+91) 80-559-9467

National Semiconductor

PRELIMINARY

COP87L88EK/COP87L84EK 8-Bit One Time Programmable (OTP) Microcontroller with Analog Function Block

General Description

The COP87L88EK/COP87L84EK OTP microcontrollers are members of the COP8™ feature family using an 8-bit core architecture. It is pin and software compatible to the mask ROM COP888EK/COP884EK product family.

(Continued)

Key Features

- Analog function block with
  - — Analog comparator with seven input multiplexor
  - — Constant current source and $V_{CC}/2$ reference
- Three 16-bit timers, each with two 16-bit registers supporting:
  - — Processor independent PWM mode
  - — External event counter mode
  - — Input capture mode
- 8 kbytes on-board EPROM with security feature
- 256 bytes on-board RAM

Additional Peripheral Features

- Idle timer
- Multi-Input Wake-Up (MIWU) with optional interrupts (8)
- WATCHDOG™ and clock monitor logic
- MICROWIRE/PLUS™ serial I/O

Features

- Memory mapped I/O
- Software selectable I/O options (TRI-STATE® output, push-pull output, weak pull-up input, high impedance input)
- Schmitt trigger inputs on ports G and L
- Packages:
  - — 44 PLCC with 40 I/O pins
  - — 40 DIP with 36 I/O pins
  - — 28 DIP/SO, each with 24 I/O pins

CPU/Instruction Set Features

- 1 μs instruction cycle time
- Twelve multi-source vectored interrupts servicing
  - — External interrupt
  - — Idle timer T0
  - — Three timers (Each with 2 interrupts)
  - — MICROWIRE/PLUS
  - — Multi-Input Wake Up
  - — Software trap
  - — Default VIS (default interrupt)
- Versatile and easy to use instruction set
- 8-bit Stack Pointer (SP)—stack in RAM
- Two 8-bit register indirect data memory pointers (B and X)

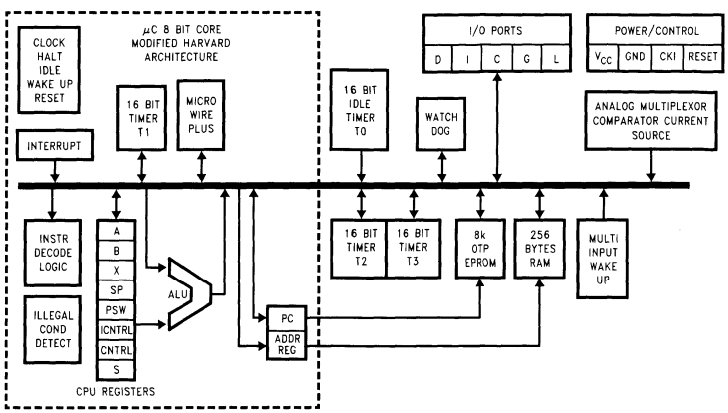
Fully Static CMOS

- Two power saving modes: HALT and IDLE
- Single supply operation: 2.7V to 5.5V
- Temperature ranges: −40° to +85°C

Development Support

- Emulation device for the COP888EK/COP884EK
- Real time emulation and full program debug offered by MetaLink Development System

Block Diagram



TL/DD/12520–1

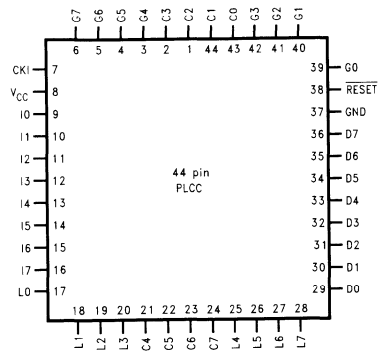
FIGURE 1. Block Diagram

General Description (Continued)

It is a fully static part, fabricated using double-metal silicon gate microCMOS technology. The device is available as One-Time Programmable (OTP). Features include an 8-bit memory mapped architecture, MICROWIRE/PLUS serial I/O, three 16-bit timer/counters supporting three modes (Processor Independent PWM generation, External Event counter, and Input Capture mode capabilities), one analog comparator with seven input multiplexor, and two power saving modes (HALT and IDLE), both with a multi-sourced wakeup/interrupt capability. This multi-sourced interrupt capability may also be used independent of the HALT or IDLE modes. Each I/O pin has software selectable configurations. The devices operate over a voltage range of 2.7V to 5.5V. High throughput is achieved with an efficient, regular instruction set operating at a maximum rate of 1 μs per instruction.

Connection Diagrams

Plastic Chip Carrier

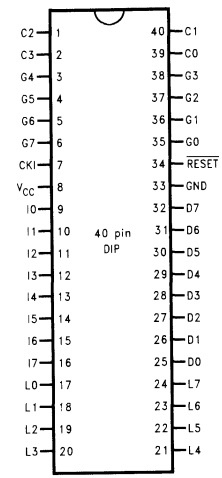


TL/DD/12520-2

Top View

Order Number COP87L88EKV-XE
See NS Plastic Chip Package Number V44A

Dual-In-Line Package

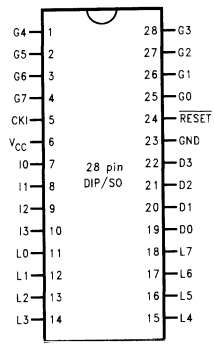


TL/DD/12520-3

Top View

Order Number COP87L84EKN-XE
See NS Molded Package Number N40A

Dual-In-Line Package



TL/DD/12520-4

Top View

Order Number COP87L84EKN-XE
See NS Molded Package Number N28B
Order Number COP87L84EKM-XE
See NS Molded Package Number M28B

Note: -X Crystal Oscillator
-E Halt Enable

FIGURE 2. Connection Diagrams

Connection Diagrams (Continued)

Pinouts for 28-, 40- and 44-Pin Packages

Port	Type	Alt. Fun	Alt. Fun	28-Pin Pack.	40-Pin Pack.	44-Pin Pack.
L0	I/O	MIWU		11	17	17
L1	I/O	MIWU		12	18	18
L2	I/O	MIWU		13	19	19
L3	I/O	MIWU		14	20	20
L4	I/O	MIWU	T2A	15	21	25
L5	I/O	MIWU	T2B	16	22	26
L6	I/O	MIWU	T3A	17	23	27
L7	I/O	MIWU	T3B	18	24	28
G0	I/O	INT		25	35	39
G1	WDOUT			26	36	40
G2	I/O	T1B		27	37	41
G3	I/O	T1A		28	38	42
G4	I/O	SO		1	3	3
G5	I/O	SK		2	4	4
G6	I	SI		3	5	5
G7	I/CKO	HALT Restart		4	6	6
D0	O			19	25	29
D1	O			20	26	30
D2	O			21	27	31
D3	O			22	28	32
I0	I	COMPIN1+		7	9	9
I1	I	COMPIN−/Current Source Out		8	10	10
I2	I	COMPIN0+		9	11	11
I3	I	COMPOUT/COMPIN2+		10	12	12
I4	I	COMPIN3+			13	13
I5	I	COMPIN4+			14	14
I6	I	COMPIN5+			15	15
I7	I	COMPOUT			16	16
D4	O				29	33
D5	O				30	34
D6	O				31	35
D7	O				32	36
C0	I/O				39	43
C1	I/O				40	44
C2	I/O				1	1
C3	I/O				2	2
C4	I/O					21
C5	I/O					22
C6	I/O					23
C7	I/O					24
V_{CC}				6	8	8
GND				23	33	37
CKI				5	7	7
$\overline{\text{RESET}}$				24	34	38

Absolute Maximum Ratings

If Military/Aerospace specified devices are required, please contact the National Semiconductor Sales Office/Distributors for availability and specifications.

Supply Voltage (V_{CC})	7V
Voltage at Any Pin	−0.3V to V_{CC} + 0.3V
Total Current into V_{CC} Pin (Source)	100 mA
Total Current out of GND Pin (Sink)	110 mA
Storage Temperature Range	−65°C to +140°C

Note: *Absolute maximum ratings indicate limits beyond which damage to the device may occur. DC and AC electrical specifications are not ensured when operating the device at absolute maximum ratings.*

DC Electrical Characteristics

−40°C ≤ T_A ≤ +85°C unless otherwise specified

Parameter	Conditions	Min	Typ	Max	Units
Operating Voltage		2.7		5.5	V
Power Supply Ripple (Note 1)	Peak-to-Peak			0.1 V_{CC}	V
Supply Current (Note 2)					
CKI = 10 MHz	V_{CC} = 5.5V, t_c = 1 μs			16.5	mA
CKI = 4 MHz	V_{CC} = 4.0V, t_c = 2.5 μs			6.5	mA
HALT Current (Note 3)	V_{CC} = 5.5V, CKI = 0 MHz			12	μA
	V_{CC} = 4.0V, CKI = 0 MHz			8	μA
IDLE Current (Note 2)					
CKI = 10 MHz	V_{CC} = 5.5V, t_c = 1 μs			3.5	mA
CKI = 1 MHz	V_{CC} = 4.0V, t_c = 10 μs			0.7	mA
Input Levels (V_{IH}, V_{IL})					
$\overline{\text{RESET}}$					
Logic High		0.8 V_{CC}			V
Logic Low				0.2 V_{CC}	V
CKI, All Other Inputs					
Logic High		0.7 V_{CC}			V
Logic Low				0.2 V_{CC}	V
Hi-Z Input Leakage	V_{CC} = 5.5V	−2		+2	μA
Input Pullup Current	V_{CC} = 5.5V, V_{IN} = 0V	−40		−250	μA
G and L Port Input Hysteresis (Note 7)				0.35 V_{CC}	V
Output Current Levels					
D Outputs					
Source	V_{CC} = 4.5V, V_{OH} = 3.3V	−0.4			mA
Sink (Note 4)	V_{CC} = 4.5V, V_{OL} = 1V	10			mA
All Others					
Source (Weak Pull-Up Mode)	V_{CC} = 4.5V, V_{OH} = 2.7V	−10		−110	μA
Source (Push-Pull Mode)	V_{CC} = 4.5V, V_{OH} = 3.3V	−0.4			mA
Sink (Push-Pull Mode)	V_{CC} = 4.5V, V_{OL} = 0.4V	1.6			mA
TRI-STATE Leakage	V_{CC} = 5.5V	−2		+2	μA
Allowable Sink/Source Current per Pin (Note 7)					
D Outputs (Sink)				15	mA
All others				3	mA
Maximum Input Current without Latchup (Note 5)	Room Temp			±200	mA
RAM Retention Voltage, V_r	500 ns Rise and Fall Time (Min)	2			V
Input Capacitance (Note 6)				7	pF
Load Capacitance on D2 (Note 6)				1000	pF

AC Electrical Characteristics $-40°C \le T_A \le +85°C$ unless otherwise specified

Parameter	Conditions	Min	Typ	Max	Units
Instruction Cycle Time (t_c)					
Crystal, Resonator,	$4.5V \le V_{CC} \le 5.5V$	1.0		DC	μs
R/C Oscillator	$4.5V \le V_{CC} \le 5.5V$	3.0		DC	μs
Inputs					
t_{SETUP}	$4.5V \le V_{CC} \le 5.5V$	200			ns
t_{HOLD}	$4.5V \le V_{CC} \le 5.5V$	60			ns
Output Propagation Delay (Note 6)	$R_L = 2.2k, C_L = 100$ pF				
t_{PD1}, t_{PD0}					
SO, SK	$4.5V \le V_{CC} \le 5.5V$			0.7	μs
All Others	$4.5V \le V_{CC} \le 5.5V$			1	μs
MICROWIRE™ Setup Time (t_{UWS}) (Note 7)	$V_{CC} \ge 4.5V$	20			ns
MICROWIRE Hold Time (t_{UWH}) (Note 7)	$V_{CC} \ge 4.5V$	56			ns
MICROWIRE Output Propagation Delay (t_{UPD})	$V_{CC} \ge 4.5V$			220	ns
Input Pulse Width (Note 7)					
Interrupt Input High Time		1.0			t_c
Interrupt Input Low Time		1.0			t_c
Timer 1, 2, 3 Input High Time		1.0			t_c
Timer 1, 2, 3 Input Low Time		1.0			t_c
Reset Pulse Width		1.0			μs

t_c = Instruction Cycle Time

Note 1: Maximum rate of voltage change must be < 0.5 V/ms.

Note 2: Supply and IDLE currents are measured with CKI driven with a square wave Oscillator, CKO driven 180° out of phase with CKI, inputs connected to V_{CC} and outputs driven low but not connected to a load.

Note 3: The HALT mode will stop CKI from oscillating in the RC and the Crystal configurations by bringing CKI high. Measurement of I_{DD} HALT is done with device neither sourcing nor sinking current; with L, C, and G0–G5 programmed as low outputs and not driving a load; all outputs programmed low and not driving a load; all inputs tied to V_{CC}; clock monitor and comparator disabled. Parameter refers to HALT mode entered via setting bit 7 of the G Port data register. Part will pull up CKI during HALT in crystal clock mode.

Note 4: The user must guarantee that D2 pin does not source more than 10 mA during RESET. If D2 sources more than 10 mA during reset, the device will go into programming mode.

Note 5: Pins G6 and $\overline{\text{RESET}}$ are designed with a high voltage input network. These pins allow input voltages > V_{CC} and the pins will have sink current to V_{CC} when biased at voltages > V_{CC} (the pins do not have source current when biased at a voltage below V_{CC}). The effective resistance to V_{CC} is 750Ω (typical). These two pins will not latch up. The voltage at the pins must be limited to < 14V. **WARNING: Voltages in excess of 14V will cause damage to the pins. This warning excludes ESD transients.**

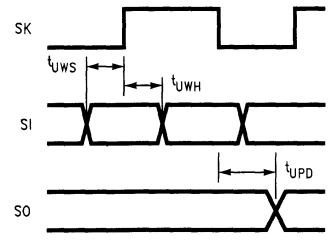
Note 6: The output propagation delay is referenced to the end of the instruction cycle where the output change occurs.

Note 7: Parameter characterized but not tested.

Analog Function Block V_{CC} = 5.0V, −40°C ≤ T_A ≤ +85°C

Parameter	Conditions	Min	Typ	Max	Units
Input Offset Voltage	0.4V < V_{IN} < V_{CC} − 1.5V		±10	±25	mV
Input Common Mode Voltage Range (Note 8)		0.4		V_{CC} − 1.5	V
V_{CC}/2 Reference	4.5V < V_{CC} < 5.5V	0.5 V_{CC} − 0.04	0.5 V_{CC}	0.5 V_{CC} + 0.04	V
DC Supply Current for Comparator (when enabled)	V_{CC} = 5.5V			250	μA
DC Supply Current for V_{CC}/2 Reference (when enabled)	V_{CC} = 5.5V		50	80	μA
DC Supply Current for Constant Current Source (when enabled)	V_{CC} = 5.5V			200	μA
Constant Current Source	4.5V < V_{CC} < 5.5V	10	20	40	μA
Current Source Variation over Common Mode Range	4.5V < V_{CC} < 5.5V Temp = Constant			±2	μA
Current Source Enable Time			1.5	2	μs
Comparator Response Time	100 mV Overdrive, 100 pF Load			1	μs

Note 8: The device is capable of operating over a common mode voltage range of 0 to V_{CC} − 1.5V, however increased offset voltage will be observed between 0V and 0.4V.



TL/DD/12520–5

FIGURE 3. MICROWIRE/PLUS Timing

Pin Descriptions

V_{CC} and GND are the power supply pins. All V_{CC} and GND pins must be connected.

CKI is the clock input. This can come from an R/C generated oscillator, or a crystal oscillator (in conjunction with CKO). See Oscillator Description section.

$\overline{\text{RESET}}$ is the master reset input. See Reset Description section.

The device contains three bidirectional 8-bit I/O ports (C, G and L), where each individual bit may be independently configured as an input (Schmitt Trigger inputs on ports L and G), output or TRI-STATE under program control. Three data memory address locations are allocated for each of these I/O ports. Each I/O port has two associated 8-bit memory mapped registers, the CONFIGURATION register and the output DATA register. A memory mapped address is also reserved for the input pins of each I/O port. (See the memory map for the various addresses associated with the I/O ports.) *Figure 4* shows the I/O port configurations. The DATA and CONFIGURATION registers allow for each port bit to be individually configured under software control as shown below:

CONFIGURATION Register	DATA Register	Port Set-Up
0	0	Hi-Z Input (TRI-STATE Output)
0	1	Input with Weak Pull-Up
1	0	Push-Pull Zero Output
1	1	Push-Pull One Output

PORT L is an 8-bit I/O port. All L-pins have Schmitt triggers on the inputs.

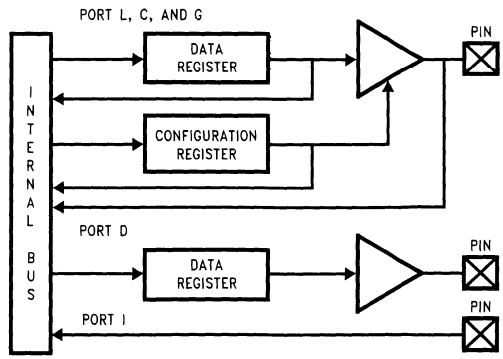
The Port L supports Multi-Input Wake Up on all eight pins. L4 and L5 are used for the timer input functions T2A and T2B. L6 and L7 are used for the timer input functions T3A and T3B.

The Port L has the following alternate features:

- L0 MIWU
- L1 MIWU
- L2 MIWU
- L3 MIWU
- L4 MIWU or T2A
- L5 MIWU or T2B
- L6 MIWU or T3A
- L7 MIWU or T3B

Port G is an 8-bit port with 5 I/O pins (G0, G2–G5), an input pin (G6), and a dedicated output pin (G7). Pins G0 and G2–G6 all have Schmitt Triggers on their inputs. Pin G1 serves as the dedicated WDOUT WATCHDOG output, while pin G7 is either input or output depending on the oscillator mask option selected. With the crystal oscillator option selected, G7 serves as the dedicated output pin for the CKO clock output. With the single-pin R/C oscillator mask option selected, G7 serves as a general purpose input pin but is also used to bring the device out of HALT mode with a low to high transition on G7. There are two registers associated with the G Port, a data register and a configuration register. Therefore, each of the 5 I/O bits (G0, G2–G5) can be individually configured under software control.

Since G6 is an input only pin and G7 is the dedicated CKO clock output pin (crystal clock option) or general purpose input (R/C clock option), the associated bits in the data and configuration registers for G6 and G7 are used for special purpose functions as outlined on the next page. Reading the G6 and G7 data bits will return zeros.



TL/DD/12520–6

FIGURE 4. I/O Port Configurations

Pin Descriptions (Continued)

Note that the chip will be placed in the HALT mode by writing a "1" to bit 7 of the Port G Data Register. Similarly the chip will be placed in the IDLE mode by writing a "1" to bit 6 of the Port G Data Register.

Writing a "1" to bit 6 of the Port G Configuration Register enables the MICROWIRE/PLUS to operate with the alternate phase of the SK clock. The G7 configuration bit, if set high, enables the clock start up delay after HALT when the R/C clock configuration is used.

	Config Reg.	Data Reg.
G7	CLKDLY	HALT
G6	Alternate SK	IDLE

Port G has the following alternate features:

- G0 INTR (External Interrupt Input)
- G2 T1B (Timer T1 Capture Input)
- G3 T1A (Timer T1 I/O)
- G4 SO (MICROWIRE Serial Data Output)
- G5 SK (MICROWIRE Serial Clock)
- G6 SI (MICROWIRE Serial Data Input)

Port G has the following dedicated functions:

- G1 WDOUT WATCHDOG and/or Clock Monitor dedicated output
- G7 CKO Oscillator dedicated output or general purpose input

Port C is an 8-bit I/O port. The 40-pin device does not have a full complement of Port C pins. The unavailable pins are not terminated. A read operation for these unterminated pins will return unpredicatable values.

Port I is an eight-bit Hi-Z input port.

Port I0–I7 are used for the analog function block.

The Port I has the following alternate features:

- I0 COMPIN1+ (Comparator Positive Input 1)
- I1 COMPIN− (Comparator Negative Input/Current Source Out)
- I2 COMPIN0+ (Comparator Positive Input 0)
- I3 COMPOUT/COMPIN2+ (Comparator Output/ Comparator Positive Input 2))
- I4 COMPIN3+ (Comparator Positive Input 3)
- I5 COMPIN4+ (Comparator Positive Input 4)
- I6 COMPIN5+ (Comparator Positive Input 5)
- I7 COMPOUT (Comparator Output)

Port D is an 8-bit output port that is preset high when $\overline{\text{RESET}}$ goes low. The user can tie two or more D port outputs (except D2) together in order to get a higher drive.

Note: Care must be exercised with the D2 pin operation. At RESET, the external loads on this pin must ensure that the output voltages stay above 0.8 V_{CC} to prevent the chip from entering special modes. Also keep the external loading on D2 to <1000 pF.

Functional Description

The architecture of the device is modified Harvard architecture. With the Harvard architecture, the control store program memory (ROM) is separated from the data store memory (RAM). Both ROM and RAM have their own separate addressing space with separate address buses. The architecture, though based on Harvard architecture, permits transfer of data from ROM to RAM.

CPU REGISTERS

The CPU can do an 8-bit addition, subtraction, logical or shift operation in one instruction (t_C) cycle time.

There are six CPU registers:

A is the 8-bit Accumulator Register

PC is the 15-bit Program Counter Register

PU is the upper 7 bits of the program counter (PC)
PL is the lower 8 bits of the program counter (PC)

B is an 8-bit RAM address pointer, which can be optionally post auto incremented or decremented.

X is an 8-bit alternate RAM address pointer, which can be optionally post auto incremented or decremented.

SP is the 8-bit stack pointer, which points to the subroutine/ interrupt stack (in RAM). The SP is initialized to RAM address 06F with reset.

S is the 8-bit Data Segment Address Register used to extend the lower half of the address range (00 to 7F) into 256 data segments of 128 bytes each.

All the CPU registers are memory mapped with the exception of the Accumulator (A) and the Program Counter (PC).

PROGRAM MEMORY

The program memory consists of 8 kbytes of OTP EPROM. These bytes may hold program instructions or constant data (data tables for the LAID instruction, jump vectors for the JID instruction, and interrupt vectors for the VIS instruction). The program memory is addressed by the 15-bit program counter (PC). All interrupts in the devices vector to program memory location 0FF Hex.

The device can be configured to inhibit external reads of the program memory. This is done by programming the Security Byte.

SECURITY FEATURE

The program memory array has an associate Security Byte that is located outside of the program address range. This byte can be addressed only from programming mode by a programmer tool.

Security is an optional feature and can only be asserted after the memory array has been programmed and verified. A secured part will read all 00(hex) by a programmer. The part will fail Blank Check and will fail Verify operations. A Read operation will fill the programmer's memory with 00(hex). The Security Byte itself is always readable with value of 00(hex) if unsecure and FF(hex) if secure.

DATA MEMORY

The data memory address space includes the on-chip RAM and data registers, the I/O registers (Configuration, Data and Pin), the control registers, the MICROWIRE/PLUS SIO shift register, and the various registers, and counters associated with the timers (with the exception of the IDLE timer). Data memory is addressed directly by the instruction or indirectly by the B, X, SP pointers and S register.

The data memory consists of 256 bytes of RAM. Sixteen bytes of RAM are mapped as "registers" at addresses 0F0 to 0FF Hex. These registers can be loaded immediately, and also decremented and tested with the DRSZ (decrement register and skip if zero) instruction. The memory pointer registers X, SP, B and S are memory mapped into this space at address locations 0FC to 0FF Hex respectively, with the other registers being available for general usage.

Functional Description (Continued)

The instruction set permits any bit in memory to be set, reset or tested. All I/O and registers (except A and PC) are memory mapped; therefore, I/O bits and register bits can be directly and individually set, reset and tested. The accumulator (A) bits can also be directly and individually tested.

Note: RAM contents are undefined upon power-up.

Data Memory Segment RAM Extension

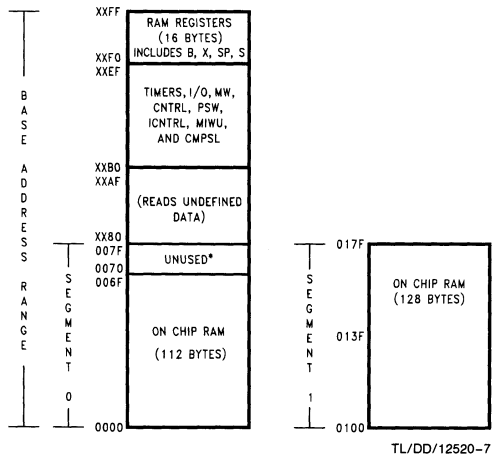
Data memory address 0FF is used as a memory mapped location for the Data Segment Address Register (S).

The data store memory is either addressed directly by a single byte address within the instruction, or indirectly relative to the reference of the B, X, or SP pointers (each contains a single-byte address). This single-byte address allows an addressing range of 256 locations from 00 to FF hex. The upper bit of this single-byte address divides the data store memory into two separate sections as outlined previously. With the exception of the RAM register memory from address locations 00F0 to 00FF, all RAM memory is memory mapped with the upper bit of the single-byte address being equal to zero. This allows the upper bit of the single-byte address to determine whether or not the base address range (from 0000 to 00FF) is extended. If this upper bit equals one (representing address range 0080 to 00FF), then address extension does not take place. Alternatively, if this upper bit equals zero, then the data segment extension register S is used to extend the base address range (from 0000 to 007F) from XX00 to XX7F, where XX represents the 8 bits from the S register. Thus the 128-byte data segment extensions are located from addresses 0100 to 017F for data segment 1, 0200 to 027F for data segment 2, etc., up to FF00 to FF7F for data segment 255. The base address range from 0000 to 007F represents data segment 0.

Figure 5 illustrates how the S register data memory extension is used in extending the lower half of the base address range (00 to 7F hex) into 256 data segments of 128 bytes each, with a total addressing range of 32 kbytes from XX00 to XX7F. This organization allows a total of 256 data segments of 128 bytes each with an additional upper base segment of 128 bytes. Furthermore, all addressing modes are available for all data segments. The S register must be changed under program control to move from one data segment (128 bytes) to another. However, the upper base segment (containing the 16 memory registers, I/O registers, control registers, etc.) is always available regardless of the contents of the S register, since the upper base segment (address range 0080 to 00FF) is independent of data segment extension.

The instructions that utilize the stack pointer (SP) always reference the stack as part of the base segment (Segment 0), regardless of the contents of the S register. The S register is not changed by these instructions. Consequently, the stack (used with subroutine linkage and interrupts) is always located in the base segment. The stack pointer will be initialized to point at data memory location 006F as a result of reset.

The 128 bytes of RAM contained in the base segment are split between the lower and upper base segments. The first 112 bytes of RAM are resident from address 0000 to 006F in the lower base segment, while the remaining 16 bytes of RAM represent the 16 data memory registers located at addresses 00F0 to 00FF of the upper base segment. No RAM is located at the upper sixteen addresses (0070 to 007F) of the lower base segment.



*Reads as all ones.

FIGURE 5. RAM Organization

Additional RAM beyond these initial 128 bytes, however, will always be memory mapped in groups of 128 bytes (or less) at the data segment address extensions (XX00 to XX7F) of the lower base segment. The additional 128 bytes of RAM are memory mapped at address locations 0100 to 017F hex.

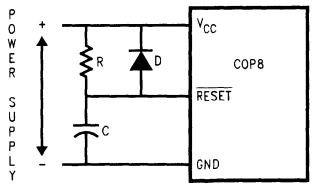
Reset

The $\overline{\text{RESET}}$ input when pulled low initializes the microcontroller. Initialization will occur whenever the $\overline{\text{RESET}}$ input is pulled low. Upon initialization, the data and configuration registers for ports L, G and C are cleared, resulting in these Ports being initialized to the TRI-STATE mode. Pin G1 of the G Port is an exception (as noted below) since pin G1 is dedicated as the WATCHDOG and/or Clock Monitor error output pin. Port D is set high. The PC, PSW, ICNTRL, CNTRL, T2CNTRL and T3CNTRL control registers are cleared. The Comparator Select Register is cleared. The S register is initialized to zero. The Multi-Input Wakeup registers WKEN and WKEDG are cleared. Wakeup register WKPND is unknown. The stack pointer, SP, is initialized to 6F hex.

The device comes out of reset with both the WATCHDOG logic and the Clock Monitor detector armed, with the WATCHDOG service window bits set and the Clock Monitor bit set. The WATCHDOG and Clock Monitor circuits are inhibited during reset. The WATCHDOG service window bits being initialized high default to the maximum WATCHDOG service window of 64k t_C clock cycles. The Clock Monitor bit being initialized high will cause a Clock Monitor error following reset if the clock has not reached the minimum specified frequency at the termination of reset. A Clock Monitor error will cause an active low error output on pin G1. This error output will continue until 16 t_C–32 t_C clock cycles following the clock frequency reaching the minimum specified value, at which time the G1 output will enter the TRI-STATE mode.

The external RC network shown in *Figure 6* should be used to ensure that the $\overline{\text{RESET}}$ pin is held low until the power supply to the chip stabilizes.

Reset (Continued)



TL/DD/12520–8

RC > 5 × Power Supply Rise Time

FIGURE 6. Recommended Reset Circuit

Oscillator Circuits

The chip can be driven by a clock input on the CKI input pin which can be between DC and 10 MHz. The CKO output clock is on pin G7 (crystal configuration). The CKI input frequency is divided down by 10 to produce the instruction cycle clock ($1/t_C$).

Figure 7 shows the Crystal and R/C oscillator diagrams.

CRYSTAL OSCILLATOR

CKI and CKO can be connected to make a closed loop crystal (or resonator) controlled oscillator.

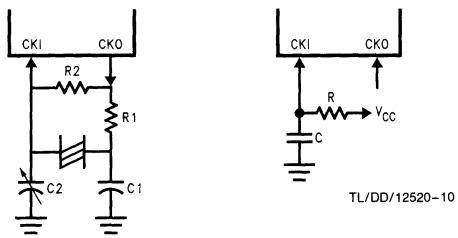
Table A shows the component values required for various standard crystal values.

R/C OSCILLATOR

By selecting CKI as a single pin oscillator input, a single pin R/C oscillator circuit can be connected to it. CKO is available as a general purpose input, and/or HALT restart input.

Note: Use of the R/C oscillator option will result in higher electromagnetic emissions.

Table B shows the variation in the oscillator frequencies as functions of the component (R and C) values.



TL/DD/12520–10

TL/DD/12520–9

FIGURE 7. Crystal and R/C Oscillator Diagrams

TABLE A. Crystal Oscillator Configuration, T_A = 25°C

R1 (kΩ)	R2 (MΩ)	C1 (pF)	C2 (pF)	CKI Freq (MHz)	Conditions
0	1	30	30–36	10	V_{CC} = 5V
0	1	30	30–36	4	V_{CC} = 5V
0	1	200	100–150	0.455	V_{CC} = 5V

TABLE B. RC Oscillator Configuration, T_A = 25°C

R (kΩ)	C (pF)	CKI Freq (MHz)	Instr. Cycle (μs)	Conditions
3.3	82	2.2 to 2.7	3.7 to 4.6	V_{CC} = 5V
5.6	100	1.1 to 1.3	7.4 to 9.0	V_{CC} = 5V
6.8	100	0.9 to 1.1	8.8 to 10.8	V_{CC} = 5V

Note: 3k ≤ R ≤ 200k

50 pF ≤ C ≤ 200 pF

Control Registers

CNTRL Register (Address X'00EE)

The Timer1 (T1) and MICROWIRE/PLUS control register contains the following bits:

SL1 & SL0	Select the MICROWIRE/PLUS clock divide by (00 = 2, 01 = 4, 1x = 8)
IEDG	External interrupt edge polarity select (0 = Rising edge, 1 = Falling edge)
MSEL	Selects G5 and G4 as MICROWIRE/PLUS signals SK and SO respectively
T1C0	Timer T1 Start/Stop control in timer modes 1 and 2
	Timer T1 Underflow Interrupt Pending Flag in timer mode 3
T1C1	Timer T1 mode control bit
T1C2	Timer T1 mode control bit
T1C3	Timer T1 mode control bit

T1C3	T1C2	T1C1	T1C0	MSEL	IEDG	SL1	SL0

Bit 7 Bit 0

Control Registers (Continued)

PSW Register (Address X'00EF)

The PSW register contains the following select bits:

- GIE Global interrupt enable (enables interrupts)
- EXEN Enable external interrupt
- BUSY MICROWIRE/PLUS busy shifting flag
- EXPND External interrupt pending
- T1ENA Timer T1 Interrupt Enable for Timer Underflow or T1A Input capture edge
- T1PNDA Timer T1 Interrupt Pending Flag (Autoreload RA in mode 1, T1 Underflow in Mode 2, T1A capture edge in mode 3)
- C Carry Flag
- HC Half Carry Flag

HC	C	T1PNDA	T1ENA	EXPND	BUSY	EXEN	GIE

Bit 7 Bit 0

The Half-Carry bit is also affected by all the instructions that affect the Carry flag. The SC (Set Carry) and RC (Reset Carry) instructions will respectively set or clear both the carry flags. In addition to the SC and RC instructions, ADC, SUBC, RRC and RLC instructions affect the carry and Half Carry flags.

ICNTRL Register (Address X'00E8)

The ICNTRL register contains the following bits:

- T1ENB Timer T1 Interrupt Enable for T1B Input capture edge
- T1PNDB Timer T1 Interrupt Pending Flag for T1B capture edge
- μWEN Enable MICROWIRE/PLUS interrupt
- μWPND MICROWIRE/PLUS interrupt pending
- T0EN Timer T0 Interrupt Enable (Bit 12 toggle)
- T0PND Timer T0 Interrupt pending
- LPEN L Port Interrupt Enable (Multi-Input Wakeup/Interrupt)

  Bit 7 could be used as a flag

Unused	LPEN	T0PND	T0EN	μWPND	μWEN	T1PNDB	T1ENB

Bit 7 Bit 0

T2CNTRL Register (Address X'00C6)

The T2CNTRL register contains the following bits:

- T2ENB Timer T2 Interrupt Enable for T2B Input capture edge
- T2PNDB Timer T2 Interrupt Pending Flag for T2B capture edge
- T2ENA Timer T2 Interrupt Enable for Timer Underflow or T2A Input capture edge
- T2PNDA Timer T2 Interrupt Pending Flag (Autoreload RA in mode 1, T2 Underflow in mode 2, T2A capture edge in mode 3)
- T2C0 Timer T2 Start/Stop control in timer modes 1 and 2 Timer T2 Underflow Interrupt Pending Flag in timer mode 3
- T2C1 Timer T2 mode control bit
- T2C2 Timer T2 mode control bit
- T2C3 Timer T2 mode control bit

T2C3	T2C2	T2C1	T2C0	T2PNDA	T2ENA	T2PNDB	T2ENB

Bit 7 Bit 0

T3CNTRL Register (Address X'00B6)

The T3CNTRL register contains the following bits:

- T3ENB Timer T3 Interrupt Enable for T3B
- T3PNDB Timer T3 Interrupt Pending Flag for T3B pin (T3B capture edge)
- T3ENA Timer T3 Interrupt Enable for Timer Underflow or T3A pin
- T3PNDA Timer T3 Interrupt Pending Flag (Autoload RA in mode 1, T3 Underflow in mode 2, T3a capture edge in mode 3)
- T3C0 Timer T3 Start/Stop control in timer modes 1 and 2

  Timer T3 Underflow Interrupt Pending Flag in timer mode 3
- T3C1 Timer T3 mode control bit
- T3C2 Timer T3 mode control bit
- T3C3 Timer T3 mode control bit

T3C3	T3C2	T3C1	T3C0	T3PNDA	T3ENA	T3PNDB	T3ENB

Bit 7 Bit 0

Timers

The device contains a very versatile set of timers (T0, T1, T2, T3). All timers and associated autoreload/capture registers power up containing random data.

TIMER T0 (IDLE TIMER)

The device supports applications that require maintaining real time and low power with the IDLE mode. This IDLE mode support is furnished by the IDLE timer T0, which is a 16-bit timer. The Timer T0 runs continuously at the fixed rate of the instruction cycle clock, t_C. The user cannot read or write to the IDLE Timer T0, which is a count down timer. The Timer T0 supports the following functions:

- Exit out of the Idle Mode (See Idle Mode description)
- WATCHDOG logic (See WATCHDOG description)
- Start up delay out of the HALT mode

The IDLE Timer T0 can generate an interrupt when the thirteenth bit toggles. This toggle is latched into the T0PND pending flag, and will occur every 4 ms at the maximum clock frequency (t_C = 1 μs). A control flag T0EN allows the interrupt from the thirteenth bit of Timer T0 to be enabled or disabled. Setting T0EN will enable the interrupt, while resetting it will disable the interrupt.

Timers (Continued)

TIMER T1, TIMER T2 AND TIMER T3

The device has a set of three powerful timer/counter blocks, T1, T2 and T3. The associated features and functioning of a timer block are described by referring to the timer block Tx. Since the three timer blocks, T1, T2 and T3 are identical, all comments are equally applicable to any of the three timer blocks.

Each timer block consists of a 16-bit timer, Tx, and two supporting 16-bit autoreload/capture registers, RxA and RxB. Each timer block has two pins associated with it, TxA and TxB. The pin TxA supports I/O required by the timer block, while the pin TxB is an input to the timer block. The powerful and flexible timer block allows the device to easily perform all timer functions with minimal software overhead. The timer block has three operating modes: Processor Independent PWM mode, External Event Counter mode, and Input Capture mode.

The control bits TxC3, TxC2, and TxC1 allow selection of the different modes of operation.

Mode 1. Processor Independent PWM Mode

As the name suggests, this mode allows the device to generate a PWM signal with very minimal user intervention. The user only has to define the parameters of the PWM signal (ON time and OFF time). Once begun, the timer block will continuously generate the PWM signal completely independent of the microcontroller. The user software services the timer block only when the PWM parameters require updating.

In this mode the timer Tx counts down at a fixed rate of t_C. Upon every underflow the timer is alternately reloaded with the contents of supporting registers, RxA and RxB. The very first underflow of the timer causes the timer to reload from the register RxA. Subsequent underflows cause the timer to be reloaded from the registers alternately beginning with the register RxB.

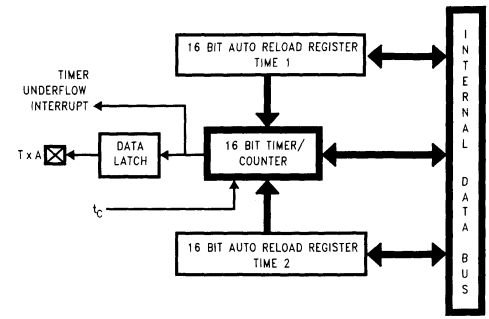
The Tx Timer control bits, TxC3, TxC2 and TxC1 set up the timer for PWM mode operation.

Figure 8 shows a block diagram of the timer in PWM mode.

The underflows can be programmed to toggle the TxA output pin. The underflows can also be programmed to generate interrupts.

Underflows from the timer are alternately latched into two pending flags, TxPNDA and TxPNDB. The user must reset these pending flags under software control. Two control enable flags, TxENA and TxENB, allow the interrupts from the timer underflow to be enabled or disabled. Setting the timer enable flag TxENA will cause an interrupt when a timer underflow causes the RxA register to be reloaded into the timer. Setting the timer enable flag TxENB will cause an interrupt when a timer underflow causes the RxB register to be reloaded into the timer. Resetting the timer enable flags will disable the associated interrupts.

Either or both of the timer underflow interrupts may be enabled. This gives the user the flexibility of interrupting once per PWM period on either the rising or falling edge of the PWM output. Alternatively, the user may choose to interrupt on both edges of the PWM output.



TL/DD/12520–11

FIGURE 8. Timer in PWM Mode

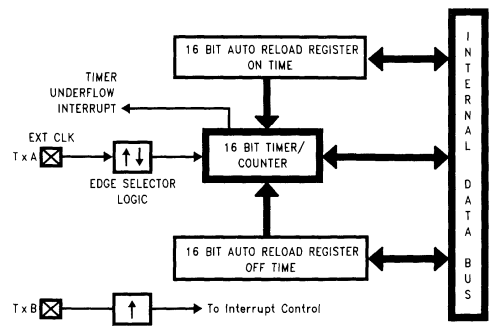
Mode 2. External Event Counter Mode

This mode is quite similar to the processor independent PWM mode described above. The main difference is that the timer, Tx, is clocked by the input signal from the TxA pin. The Tx timer control bits, TxC3, TxC2 and TxC1 allow the timer to be clocked either on a positive or negative edge from the TxA pin. Underflows from the timer are latched into the TxPNDA pending flag. Setting the TxENA control flag will cause an interrupt when the timer underflows.

In this mode the input pin TxB can be used as an independent positive edge sensitive interrupt input if the TxENB control flag is set. The occurrence of a positive edge on the TxB input pin is latched into the TxPNDB flag.

Figure 9 shows a block diagram of the timer in External Event Counter mode.

Note: The PWM output is not available in this mode since the TxA pin is being used as the counter input clock.



TL/DD/12520–12

FIGURE 9. Timer in External Event Counter Mode

Mode 3. Input Capture Mode

The device can precisely measure external frequencies or time external events by placing the timer block, Tx, in the input capture mode.

In this mode, the timer Tx is constantly running at the fixed t_C rate. The two registers, RxA and RxB, act as capture registers. Each register acts in conjunction with a pin. The register RxA acts in conjunction with the TxA pin and the register RxB acts in conjunction with the TxB pin.

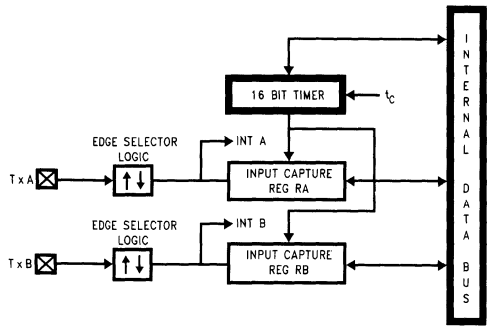
Timers (Continued)

The timer value gets copied over into the register when a trigger event occurs on its corresponding pin. Control bits, TxC3, TxC2 and TxC1, allow the trigger events to be specified either as a positive or a negative edge. The trigger condition for each input pin can be specified independently.

The trigger conditions can also be programmed to generate interrupts. The occurrence of the specified trigger condition on the TxA and TxB pins will be respectively latched into the pending flags, TxPNDA and TxPNDB. The control flag TxENA allows the interrupt on TxA to be either enabled or disabled. Setting the TxENA flag enables interrupts to be generated when the selected trigger condition occurs on the TxA pin. Similarly, the flag TxENB controls the interrupts from the TxB pin.

Underflows from the timer can also be programmed to generate interrupts. Underflows are latched into the timer TxC0 pending flag (the TxC0 control bit serves as the timer underflow interrupt pending flag in the Input Capture mode). Consequently, the TxC0 control bit should be reset when entering the Input Capture mode. The timer underflow interrupt is enabled with the TxENA control flag. When a TxA interrupt occurs in the Input Capture mode, the user must check both the TxPNDA and TxC0 pending flags in order to determine whether a TxA input capture or a timer underflow (or both) caused the interrupt.

Figure 10 shows a block diagram of the timer in Input Capture mode.



TL/DD/12520–13

FIGURE 10. Timer in Input Capture Mode

TIMER CONTROL FLAGS

The timers T1, T2 and T3 have indentical control structures. The control bits and their functions are summarized below.

TxC0	Timer Start/Stop control in Modes 1 and 2 (Processor Independent PWM and External Event Counter), where 1 = Start, 0 = Stop Timer Underflow Interrupt Pending Flag in Mode 3 (Input Capture)
TxPNDA	Timer Interrupt Pending Flag
TxPNDB	Timer Interrupt Pending Flag
TxENA	Timer Interrupt Enable Flag
TxENB	Timer Interrupt Enable Flag 1 = Timer Interrupt Enabled 0 = Timer Interrupt Disabled
TxC3	Timer mode control
TxC2	Timer mode control
TxC1	Timer mode control

Timers (Continued)

The timer mode control bits (TxC3, TxC2 and TxC1) are detailed below:

TxC3	TxC2	TxC1	Timer Mode	Interrupt A Source	Interrupt B Source	Timer Counts On
0	0	0	MODE 2 (External Event Counter)	Timer Underflow	Pos. TxB Edge	TxA Pos. Edge
0	0	1	MODE 2 (External Event Counter)	Timer Underflow	Pos. TxB Edge	TxA Neg. Edge
1	0	1	MODE 1 (PWM) TxA Toggle	Autoreload RA	Autoreload RB	t_C
1	0	0	MODE 1 (PWM) No TxA Toggle	Autoreload RA	Autoreload RB	t_C
0	1	0	MODE 3 (Capture) Captures: TxA Pos. Edge TxB Pos. Edge	Pos. TxA Edge or Timer Underflow	Pos. TxB Edge	t_C
1	1	0	MODE 3 (Capture) Captures: TxA Pos. Edge TxB Neg. Edge	Pos. TxA Edge or Timer Underflow	Neg. TxB Edge	t_C
0	1	1	MODE 3 (Capture) Captures: TxA Neg. Edge TxB Pos. Edge	Neg. TxA Edge or Timer Underflow	Pos. TxB Edge	t_C
1	1	1	MODE 3 (Capture) Captures: TxA Neg. Edge TxB Neg. Edge	Neg. TxA Edge or Timer Underflow	Neg. TxB Edge	t_C

Power Save Modes

The device offers the user two power save modes of operation: HALT and IDLE. In the HALT mode, all microcontroller activities are stopped. In the IDLE mode, the on-board oscillator circuitry the WATCHDOG logic, the Clock Monitor and timer T0 are active but all other microcontroller activities are stopped. In either mode, all on-board RAM, registers, I/O states, and timers (with the exception of T0) are unaltered.

HALT MODE

The device can be placed in the HALT mode by writing a "1" to the HALT flag (G7 data bit). All microcontroller activities, including the clock and timers, are stopped. The WATCHDOG logic is disabled during the HALT mode. However, the clock monitor circuitry if enabled remains active and will cause the WATCHDOG output pin (WDOUT) to go low. If the HALT mode is used and the user does not want to activate the WDOUT pin, the Clock Monitor should be disabled after the device comes out of reset (resetting the Clock Monitor control bit with the first write to the WDSVR register). In the HALT mode, the power requirements of the device are minimal and the applied voltage (V_{CC}) may be decreased to V_r (V_r = 2.0V) without altering the state of the machine.

The device supports three different ways of exiting the HALT mode. The first method of exiting the HALT mode is with the Multi-Input Wakeup feature on the L port. The second method is with a low to high transition on the CKO (G7) pin. This method precludes the use of the crystal clock configuration (since CKO becomes a dedicated output), and so may be used with an RC clock configuration. The third method of exiting the HALT mode is by pulling the $\overline{\text{RESET}}$ pin low.

Since a crystal or ceramic resonator may be selected as the oscillator, the Wakeup signal is not allowed to start the chip running immediately since crystal oscillators and ceramic resonators have a delayed start up time to reach full amplitude and frequency stability. The IDLE timer is used to generate a fixed delay to ensure that the oscillator has indeed stabilized before allowing instruction execution. In this case, upon detecting a valid Wakeup signal, only the oscillator circuitry is enabled. The IDLE timer is loaded with a value of 256 and is clocked with the t_C instruction cycle clock. The t_C clock is derived by dividing the oscillator clock down by a factor of 10. The Schmitt trigger following the CKI inverter on the chip ensures that the IDLE timer is clocked only when the oscillator has a sufficiently large amplitude to meet the Schmitt trigger specifications. This Schmitt trigger is not part of the oscillator closed loop. The startup timeout from the IDLE timer enables the clock signals to be routed to the rest of the chip.

If an RC clock option is being used, the fixed delay is introduced optionally. A control bit, CLKDLY, mapped as configuration bit G7, controls whether the delay is to be introduced or not. The delay is included if CLKDLY is set, and excluded if CLKDLY is reset. The CLKDLY bit is cleared on reset.

Power Save Modes (Continued)

The device has two mask options associated with the HALT mode. The first mask option enables the HALT mode feature, while the second mask option disables the HALT mode. With the HALT mode enable mask option, the device will enter and exit the HALT mode as described above. With the HALT disable mask option, the device cannot be placed in the HALT mode (writing a "1" to the HALT flag will have no effect, the HALT flag will remain "0").

IDLE MODE

The device is placed in the IDLE mode by writing a "1" to the IDLE flag (G6 data bit). In this mode, all activities, except the associated on-board oscillator circuitry, the WATCHDOG logic, the clock monitor and the IDLE Timer T0, are stopped.

As with the HALT mode, the device can be returned to normal operation with a reset, or with a Multi-Input Wakeup from the L Port. Alternately, the microcontroller resumes normal operation from the IDLE mode when the thirteenth bit (representing 4.096 ms at internal clock frequency of 1 MHz, $t_C = 1\ \mu s$) of the IDLE Timer toggles.

This toggle condition of the thirteenth bit of the IDLE Timer T0 is latched into the T0PND pending flag.

The user has the option of being interrupted with a transition on the thirteenth bit of the IDLE Timer T0. The interrupt can be enabled or disabled via the T0EN control bit. Setting the T0EN flag enables the interrupt and vice versa.

The user can enter the IDLE mode with the Timer T0 interrupt enabled. In this case, when the T0PND bit gets set, the device will first execute the Timer T0 interrupt service routine and then return to the instruction following the "Enter Idle Mode" instruction.

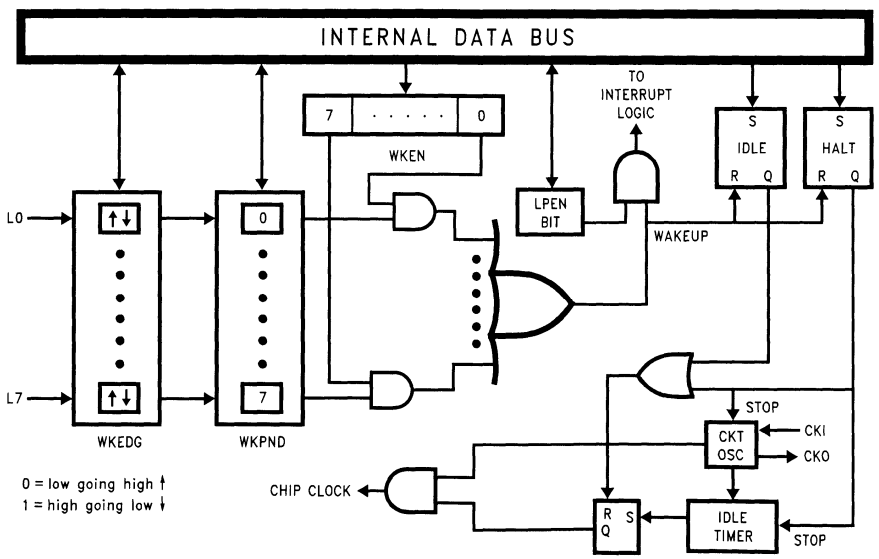
Alternatively, the user can enter the IDLE mode with the IDLE Timer T0 interrupt disabled. In this case, the device will resume normal operation with the instruction immediately following the "Enter IDLE Mode" instruction.

Note: It is necessary to program two NOP instructions following both the set HALT mode and set IDLE mode instructions. These NOP instructions are necessary to allow clock resynchronization following the HALT or IDLE modes.

Multi-Input Wakeup

The Multi-Input Wakeup feature is ued to return (wakeup) the device from either the HALT or IDLE modes. Alternately Multi-Input Wakeup/Interrupt feature may also be used to generate up to 8 edge selectable external interrupts.

Figure 11 shows the Multi-Input Wakeup logic.



TL/DD/12520–14

FIGURE 11. Multi-Input Wake Up Logic

Multi-Input Wakeup (Continued)

The Multi-Input Wakeup feature utilizes the L Port. The user selects which particular L port bit (or combination of L Port bits) will cause the device to exit the HALT or IDLE modes. The selection is done through the Reg: WKEN. The Reg: WKEN is an 8-bit read/write register, which contains a control bit for every L port bit. Setting a particular WKEN bit enables a Wakeup from the associated L port pin.

The user can select whether the trigger condition on the selected L Port pin is going to be either a positive edge (low to high transition) or a negative edge (high to low transition). This selection is made via the Reg: WKEDG, which is an 8-bit control register with a bit assigned to each L Port pin. Setting the control bit will select the trigger condition to be a negative edge on that particular L Port pin. Resetting the bit selects the trigger condition to be a positive edge. Changing an edge select entails several steps in order to avoid a pseudo Wakeup condition as a result of the edge change. First, the associated WKEN bit should be reset, followed by the edge select change in WKEDG. Next, the associated WKPND bit should be cleared, followed by the associated WKEN bit being re-enabled.

An example may serve to clarify this procedure. Suppose we wish to change the edge select from positive (low going high) to negative (high going low) for L Port bit 5, where bit 5 has previously been enabled for an input interrupt. The program would be as follows:

```
RBIT  5, WKEN
SBIT  5, WKEDG
RBIT  5, WKPND
SBIT  5, WKEN
```

If the L port bits have been used as outputs and then changed to inputs with Multi-Input Wakeup/Interrupt, a safety procedure should also be followed to avoid inherited pseudo wakeup conditions. After the selected L port bits have been changed from output to input but before the associated WKEN bits are enabled, the associated edge select bits in WKEDG should be set or reset for the desired edge selects, followed by the associated WKPND bits being cleared.

This same procedure should be used following reset, since the L port inputs are left floating as a result of reset.

The occurrence of the selected trigger condition for Multi-Input Wakeup is latched into a pending register called WKPND. The respective bits of the WKPND register will be set on the occurrence of the selected trigger edge on the corresponding Port L pin. The user has the responsibility of clearing these pending flags. Since WKPND is a pending register for the occurrence of selected wakeup conditions, the device will not enter the HALT mode if any Wakeup bit is both enabled and pending. Consequently, the user has the responsibility of clearing the pending flags before attempting to enter the HALT mode.

WKEN, WKPND and WKEDG are all read/write registers, and are cleared at reset.

PORT L INTERRUPTS

Port L provides the user with an additional eight fully selectable, edge sensitive interrupts which are all vectored into the same service subroutine.

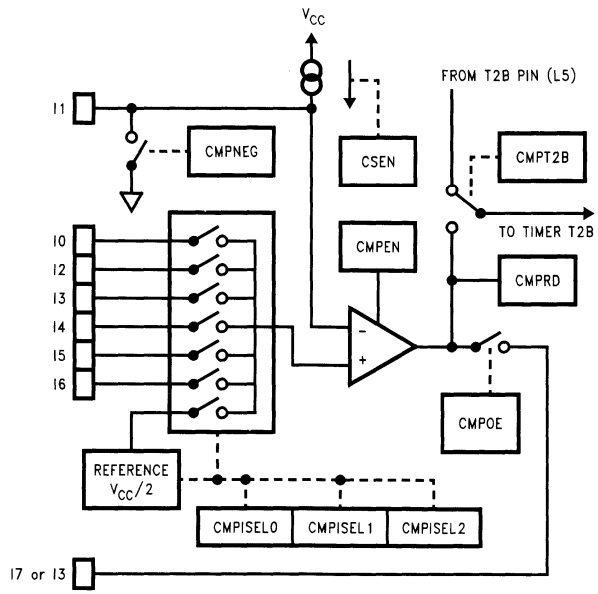
The interrupt from Port L shares logic with the wake up circuitry. The register WKEN allows interrupts from Port L to be individually enabled or disabled. The register WKEDG specifies the trigger condition to be either a positive or a negative edge. Finally, the register WKPND latches in the pending trigger conditions.

The GIE (Global Interrupt Enable) bit enables the interrupt function.

A control flag, LPEN, functions as a global interrupt enable for Port L interrupts. Setting the LPEN flag will enable interrupts and vice versa. A separate global pending flag is not needed since the register WKPND is adequate.

Since Port L is also used for waking the device out of the HALT or IDLE modes, the user can elect to exit the HALT or IDLE modes either with or without the interrupt enabled. If he elects to disable the interrupt, then the device will restart execution from the instruction immediately following the instruction that placed the microcontroller in the HALT or IDLE modes. In the other case, the device will first execute the interrupt service routine and then revert to normal operation. (See HALT MODE for clock option wakeup information.)

Analog Function Block



TL/DD/12520–15

FIGURE 12. COP87L88EK Analog Function Block

This device contains an analog function block with the intent to provide a function which allows for single slope, low cost, A/D conversion of up to 6 channels.

CMPSL REGISTER (ADDRESS X'00B7)

The CMPSL register contains the following bits:

CMPNEG — Will drive I1 to a low level. This bit can be used to discharge an external capacitor. This bit is disabled if the comparator is not enabled (CMPEN = 0).

CMPEN — Enable the comparator ("1" = enable).

CSEN — Enables the internal constant current source. This current source provides a nominal 20 μA constant current at the I1 pin. This current can be used to ensure a linear charging rate on an external capacitor. This bit has no affect and the current source is disabled if the comparator is not enabled (CMPEN = 0).

CMPOE — Enables the comparator output to either pin I3 or pin I7 ("1" = enable) depending on the value of CMPISEL0/1/2.

CMPISEL0/1/2 — Will select one of seven possible sources (I0/I2/I3/I4/I5/I6/internal reference) as a positive input to the comparator (see Table I for more information.)

CMPT2B — Selects the timer T2B input to be driven directly by the comparator output. If the comparator is disabled (CMPEN = 0), this function is disabled, i.e., the T2B input is connected to Port L5.

CMPT2B	CMPISEL2	CMPISEL1	CMPISEL0	CMPOE	CSEN	CMPEN	CMPNEG
Bit 7							Bit 0

The Comparator Select Register is cleared on RESET (the comparator is disabled). To save power the program should also disable the comparator before the μC enters the HALT/IDLE modes. Disabling the comparator will turn off the constant current source and the $V_{CC}/2$ reference, disconnect the comparator output from the T2B input and pin I3 or I7 and remove the low on I1 caused by CMPNEG.

It is often useful for the user's program to read the result of a comparator operation. Since I1 is always selected to be COMPIN− when the comparator is enabled (CMPEN = 1), the comparator output can be read internally by reading bit 1 (CMPRD) of register PORTI (RAM address 0 x D7).

The following table lists the comparator inputs and outputs vs. the value of the CMPISEL0/1/2 bits. The output will only be driven if the CMPOE bit is set to 1.

Analog Function Block (Continued)

TABLE I. Comparator Input Selection

Control Bit			Comparator Input Source		Comparator Output
CMPISEL2	CMPISEL1	CMPISEL0	Neg. Input	Pos. Input	
0	0	0	I1	I2	I3
0	0	1	I1	I2	I7
0	1	0	I1	I3	I7
0	1	1	I1	I0	I7
1	0	0	I1	I4	I7
1	0	1	I1	I5	I7
1	1	0	I1	I6	I7
1	1	1	I1	V_{CC}/2 Ref.	I7

Reset

The state of the Comparator Block immediately after RESET is as follows:

1. The CMPSL Register is set to all zeros
2. The Comparator is disabled
3. The Constant Current Source is disabled
4. CMPNEG is turned off
5. The Port I inputs are electrically isolated from the comparator
6. The T2B input is as normally selected by the T2CNTRL Register
7. CMPISEL0–CMPISEL2 are set to zero
8. All Port I inputs are selected to the default digital input mode

The comparator outputs have the same specification as Ports L and G except that the rise and fall times are symmetrical.

Interrupts

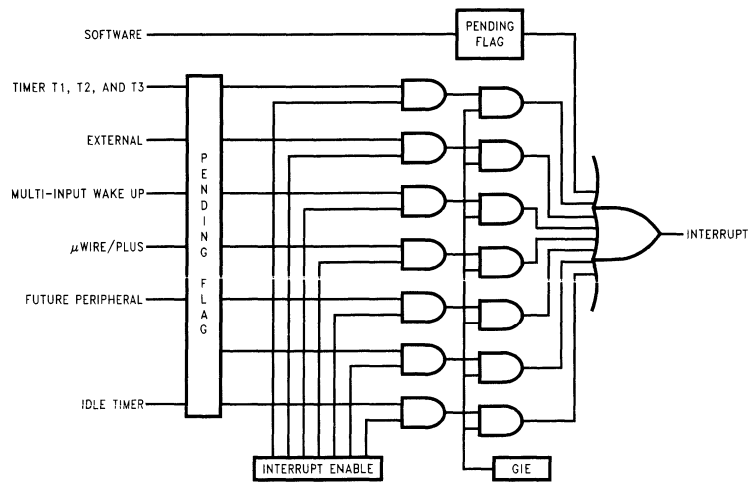
The device supports a vectored interrupt scheme. It supports a total of fourteen interrupt sources. The following table lists all the possible interrupt sources, their arbitration ranking and the memory locations reserved for the interrupt vector for each source.

Two bytes of program memory space are reserved for each interrupt source. All interrupt sources except the software interrupt are maskable. Each of the maskable interrupts have an Enable bit and a Pending bit. A maskable interrupt is active if its associated enable and pending bits are set. If GIE = 1 and an interrupt is active, then the processor will be interrupted as soon as it is ready to start executing an instruction except if the above conditions happen during the Software Trap service routine. This exception is described in the Software Trap sub-section.

The interruption process is accomplished with the INTR instruction (opcode 00), which is jammed inside the Instruction Register and replaces the opcode about to be executed. The following steps are performed for every interrupt:

1. The GIE (Global Interrupt Enable) bit is reset.
2. The address of the instruction about to be executed is pushed into the stack.
3. The PC (Program Counter) branches to address 00FF.

This procedure takes 7 t_C cycles to execute.



TL/DD/12520–16

FIGURE 13. Interrupt Block Diagram

Interrupts (Continued)

Arbitration Ranking	Source	Description	Vector* Address Hi-Low Byte
(1) Highest	Software	INTR Instruction	0yFE–0yFF
(2)	Reserved		0yFC–0yFD
(3)	External	G0	0yFA–0yFB
(4)	Timer T0	Underflow	0yF8–0yF9
(5)	Timer T1	T1A/Underflow	0yF6–0yF7
(6)	Timer T1	T1B	0yF4–0yF5
(7)	MICROWIRE/PLUS	BUSY Low	0yF2–0yF3
(8)	Reserved		0yF0–0yF1
(9)	Reserved		0yEE–0yEF
(10)	Reserved		0yEC–0yED
(11)	Timer T2	T2A/Underflow	0yEA–0yEB
(12)	Timer T2	T2B	0yE8–0yE9
(13)	Timer T3	T3A/Underflow	0yE6–0yE7
(14)	Timer T3	T3B	0yE4–0yE5
(15)	Port L/Wakeup	Port L Edge	0yE2–0yE3
(16) Lowest	Default	VIS Instr. Execution without Any Interrupts	0yE0–0yE1

*y is a variable which represents the VIS block. VIS and the vector table must be located in the same 256-byte block except if VIS is located at the last address of a block. In this case, the table must be in the next block.

At this time, since GIE = 0, other maskable interrupts are disabled. The user is now free to do whatever context switching is required by saving the context of the machine in the stack with PUSH instructions. The user would then program a VIS (Vector Interrupt Select) instruction in order to branch to the interrupt service routine of the highest priority interrupt enabled and pending at the time of the VIS. Note that this is not necessarily the interrupt that caused the branch to address location 00FF Hex prior to the context switching.

Thus, if an interrupt with a higher rank than the one which caused the interruption becomes active before the decision of which interrupt to service is made by the VIS, then the interrupt with the higher rank will override any lower ones and will be acknowledged. The lower priority interrupt(s) are still pending, however, and will cause another interrupt immediately following the completion of the interrupt service routine associated with the higher priority interrupt just serviced. This lower priority interrupt will occur immediately following the RETI (Return from Interrupt) instruction at the end of the interrupt service routine just completed.

Inside the interrupt service routine, the associated pending bit has to be cleared by software. The RETI (Return from Interrupt) instruction at the end of the interrupt service routine will set the GIE (Global Interrupt Enable) bit, allowing the processor to be interrupted again if another interrupt is active and pending.

The VIS instruction looks at all the active interrupts at the time it is executed and performs an indirect jump to the beginning of the service routine of the one with the highest rank.

The addresses of the different interrupt service routines, called vectors, are chosen by the user and stored in ROM in a table starting at 01E0 (assuming that VIS is located between 00FF and 01DF). The vectors are 15-bit wide and therefore occupy 2 ROM locations.

VIS and the vector table must be located in the same 256-byte block (0y00 to 0yFF) except if VIS is located at the last address of a block. In this case, the table must be in the next block. The vector table cannot be inserted in the first 256-byte block (y ≠ 0).

The vector of the maskable interrupt with the lowest rank is located at 0yE0 (Hi-Order byte) and 0yE1 (Lo-Order byte) and so forth in increasing rank number. The vector of the maskable interrupt with the highest rank is located at 0yFA (Hi-Order byte) and 0yFB (Lo-Order byte).

The Software Trap has the highest rank and its vector is located at 0yFE and 0yFF.

If, by accident, a VIS gets executed and no interrupt is active, then the PC (Program Counter) will branch to a vector located at 0yE0–0yE1.

WARNING

A Default VIS interrupt handler routine must be present. As a minimum, this handler should confirm that the GIE bit is cleared (this indicates that the interrupt sequence has been taken), take care of any required housekeeping, restore context and return. Some sort of Warm Restart procedure should be implemented. These events can occur without any error on the part of the system designer or programmer.

Note: There is always the possibility of an interrupt occurring during an instruction which is attempting to reset the GIE bit or any other interrupt enable bit. If this occurs when a single cycle instruction is being used to reset the interrupt enable bit, the interrupt enable bit will be reset but an interrupt may still occur. This is because interrupt processing is started at the same time as the interrupt bit is being reset. To avoid this scenario, the user should always use a two, three, or four cycle instruction to reset interrupt enable bits.

Figure 13 shows the Interrupt block diagram.

Interrupts (Continued)

SOFTWARE TRAP

The Software Trap (ST) is a special kind of non-maskable interrupt which occurs when the INTR instruction (used to acknowledge interrupts) is fetched from ROM and placed inside the instruction register. This may happen when the PC is pointing beyond the available ROM address space or when the stack is over-popped.

When an ST occurs, the user can re-initialize the stack pointer and do a recovery procedure (similar to reset, but not necessarily containing all of the same initialization procedures) before restarting.

The occurrence of an ST is latched into the ST pending bit. The GIE bit is not affected and the ST pending bit **(not accessible by the user)** is used to inhibit other interrupts and to direct the program to the ST service routine with the VIS instruction. The RPND instruction is used to clear the software interrupt pending bit. This pending bit is also cleared on reset.

The ST has the highest rank among all interrupts.

Nothing (except another ST) can interrupt an ST being serviced.

WATCHDOG

The device contains a WATCHDOG and clock monitor. The WATCHDOG is designed to detect the user program getting stuck in infinite loops resulting in loss of program control or "runaway" programs. The Clock Monitor is used to detect the absence of a clock or a very slow clock below a specified rate on the CKI pin.

The WATCHDOG consists of two independent logic blocks: WD UPPER and WD LOWER. WD UPPER establishes the upper limit on the service window and WD LOWER defines the lower limit of the service window.

Servicing the WATCHDOG consists of writing a specific value to a WATCHDOG Service Register named WDSVR which is memory mapped in the RAM. This value is composed of three fields, consisting of a 2-bit Window Select, a 5-bit Key Data field, and the 1-bit Clock Monitor Select field. Table II shows the WDSVR register.

TABLE II. WATCHDOG Service Register (WDSVR)

Window Select		Key Data					Clock Monitor
X	X	0	1	1	0	0	Y
7	6	5	4	3	2	1	0

The lower limit of the service window is fixed at 2048 instruction cycles. Bits 7 and 6 of the WDSVR register allow the user to pick an upper limit of the service window.

Table III shows the four possible combinations of lower and upper limits for the WATCHDOG service window. This flexibility in choosing the WATCHDOG service window prevents any undue burden on the user software.

Bits 5, 4, 3, 2 and 1 of the WDSVR register represent the 5-bit Key Data field. The key data is fixed at 01100. Bit 0 of the WDSVR Register is the Clock Monitor Select bit.

TABLE III. WATCHDOG Service Window Select

WDSVR Bit 7	WDSVR Bit 6	Service Window (Lower-Upper Limits)
0	0	2k–8k t_c Cycles
0	1	2k–16k t_c Cycles
1	0	2k–32k t_c Cycles
1	1	2k–64k t_c Cycles

Clock Monitor

The Clock Monitor aboard the device can be selected or deselected under program control. The Clock Monitor is guaranteed not to reject the clock if the instruction cycle clock ($1/t_c$) is greater or equal to 10 kHz. This equates to a clock input rate on CKI of greater or equal to 100 kHz.

WATCHDOG Operation

The WATCHDOG and Clock Monitor are disabled during reset. The device comes out of reset with the WATCHDOG armed, the WATCHDOG Window Select bits (bits 6, 7 of the WDSVR Register) set, and the Clock Monitor bit (bit 0 of the WDSVR Register) enabled. Thus, a Clock Monitor error will occur after coming out of reset, if the instruction cycle clock frequency has not reached a minimum specified value, including the case where the oscillator fails to start.

The WDSVR register can be written to only once after reset and the key data (bits 5 through 1 of the WDSVR Register) must match to be a valid write. This write to the WDSVR register involves two irrevocable choices: (i) the selection of the WATCHDOG service window (ii) enabling or disabling of the Clock Monitor. Hence, the first write to WDSVR Register involves selecting or deselecting the Clock Monitor, select the WATCHDOG service window and match the WATCHDOG key data. Subsequent writes to the WDSVR register will compare the value being written by the user to the WATCHDOG service window value and the key data (bits 7 through 1) in the WDSVR Register. Table IV shows the sequence of events that can occur.

The user must service the WATCHDOG at least once before the upper limit of the service window expires. The WATCHDOG may not be serviced more than once in every lower limit of the service window. The user may service the WATCHDOG as many times as wished in the time period between the lower and upper limits of the service window. The first write to the WDSVR Register is also counted as a WATCHDOG service.

The WATCHDOG has an output pin associated with it. This is the WDOUT pin, on pin 1 of the port G. WDOUT is active low. The WDOUT pin is in the high impedance state in the inactive state. Upon triggering the WATCHDOG, the logic will pull the WDOUT (G1) pin low for an additional 16 t_c–32 t_c cycles after the signal level on WDOUT pin goes below the lower Schmitt trigger threshold. After this delay, the device will stop forcing the WDOUT output low.

The WATCHDOG service window will restart when the WDOUT pin goes high. It is recommended that the user tie the WDOUT pin back to V_{CC} through a resistor in order to pull WDOUT high.

WATCHDOG Operation (Continued)

A WATCHDOG service while the WDOUT signal is active will be ignored. The state of the WDOUT pin is not guaranteed on reset, but if it powers up low then the WATCHDOG will time out and WDOUT will enter high impedance state.

The Clock Monitor forces the G1 pin low upon detecting a clock frequency error. The Clock Monitor error will continue until the clock frequency has reached the minimum specified value, after which the G1 output will enter the high impedance TRI-STATE mode following 16 t_C–32 t_C clock cycles. The Clock Monitor generates a continual Clock Monitor error if the oscillator fails to start, or fails to reach the minimum specified frequency. The specification for the Clock Monitor is as follows:

$1/t_C > 10$ kHz—No clock rejection.

$1/t_C < 10$ Hz—Guaranteed clock rejection.

WATCHDOG AND CLOCK MONITOR SUMMARY

The following salient points regarding the WATCHDOG and CLOCK MONITOR should be noted:

- Both the WATCHDOG and CLOCK MONITOR detector circuits are inhibited during RESET.
- Following RESET, the WATCHDOG and CLOCK MONITOR are both enabled, with the WATCHDOG having he maximum service window selected.
- The WATCHDOG service window and CLOCK MONITOR enable/disable option can only be changed once, during the initial WATCHDOG service following RESET.
- The initial WATCHDOG service must match the key data value in the WATCHDOG Service register WDSVR in order to avoid a WATCHDOG error.
- Subsequent WATCHDOG services must match all three data fields in WDSVR in order to avoid WATCHDOG errors.
- The correct key data value cannot be read from the WATCHDOG Service register WDSVR. Any attempt to read this key data value of 01100 from WDSVR will read as key data value of all 0's.
- The WATCHDOG detector circuit is inhibited during both the HALT and IDLE modes.
- The CLOCK MONITOR detector circuit is active during both the HALT and IDLE modes. Consequently, the device inadvertently entering the HALT mode will be detected as a CLOCK MONITOR error (provided that the CLOCK MONITOR enable option has been selected by the program).
- With the single-pin R/C oscillator mask option selected and the CLKDLY bit reset, the WATCHDOG service window will resume following HALT mode from where it left off before entering the HALT mode.
- With the crystal oscillator mask option selected, or with the single-pin R/C oscillator mask option selected and the CLKDLY bit set, the WATCHDOG service window will be set to its selected value from WDSVR following HALT. Consequently, the WATCHDOG should not be serviced for at least 2048 instruction cycles following HALT, but must be serviced within the selected window to avoid a WATCHDOG error.
- The IDLE timer T0 is not initialized with RESET.
- The user can sync in to the IDLE counter cycle with an IDLE counter (T0) interrupt or by monitoring the T0PND flag. The T0PND flag is set whenever the thirteenth bit of the IDLE counter toggles (every 4096 instruction cycles). The user is responsible for resetting the T0PND flag.
- A hardware WATCHDOG service occurs just as the device exits the IDLE mode. Consequently, the WATCHDOG should not be serviced for at least 2048 instruction cycles following IDLE, but must be serviced within the selected window to avoid a WATCHDOG error.
- Following RESET, the initial WATCHDOG service (where the service window and the CLOCK MONITOR enable/disable must be selected) may be programmed anywhere within the maximum service window (65,536 instruction cycles) initialized by RESET. Note that this initial WATCHDOG service may be programmed within the initial 2048 instruction cycles without causing a WATCHDOG error.

Detection of Illegal Conditions

The device can detect various illegal conditions resulting from coding errors, transient noise, power supply voltage drops, runaway programs, etc.

Reading of undefined ROM gets zeros. The opcode for software interrupt is zero. If the program fetches instructions from undefined ROM, this will force a software interrupt, thus signaling that an illegal condition has occurred.

The subroutine stack grows down for each call (jump to subroutine), interrupt, or PUSH, and grows up for each return or POP. The stack pointer is initialized to RAM location 06F Hex during reset. Consequently, if there are more returns than calls, the stack pointer will point to addresses 070 and 071 Hex (which are undefined RAM). Undefined RAM from addresses 070 to 07F (Segment 0), 140 to 17F (Segment 1), and all other segments (i.e., Segments 2 . . . etc.) is read as all 1's, which in turn will cause the program to return to address 7FFF Hex. This is an undefined ROM location and the instruction fetched (all 0's) from this location will generate a software interrupt signaling an illegal condition.

Thus, the chip can detect the following illegal conditions:

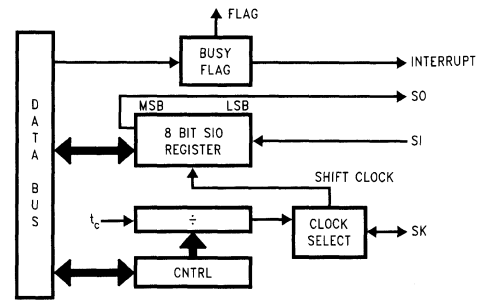
a. Executing from undefined ROM

b. Over "POP"ing the stack by having more returns than calls.

When the software interrupt occurs, the user can re-initialize the stack pointer and do a recovery procedure before restarting (this recovery program is probably similar to that following reset, but might not contain the same program initialization procedures). The recovery program should reset the software interrupt pending bit using the RPND instruction.

MICROWIRE/PLUS

MICROWIRE/PLUS is a serial synchronous communications interface. The MICROWIRE/PLUS capability enables the device to interface with any of National Semiconductor's MICROWIRE peripherals (i.e. A/D converters, display drivers, E^2PROMs etc.) and with other microcontrollers which support the MICROWIRE interface. It consists of an 8-bit serial shift register (SIO) with serial data input (SI), serial data output (SO) and serial shift clock (SK). *Figure 14* shows a block diagram of the MICROWIRE/PLUS logic.



TL/DD/12520–17

FIGURE 14. MICROWIRE/PLUS Block Diagram

The shift clock can be selected from either an internal source or an external source. Operating the MICROWIRE/PLUS arrangement with the internal clock source is called the Master mode of operation. Similarly, operating the MICROWIRE/PLUS arrangement with an external shift clock is called the Slave mode of operation.

The CNTRL register is used to configure and control the MICROWIRE/PLUS mode. To use the MICROWIRE/PLUS, the MSEL bit in the CNTRL register is set to one. In the master mode, the SK clock rate is selected by the two bits, SL0 and SL1, in the CNTRL register. Table V details the different clock rates that may be selected.

TABLE IV. WATCHDOG Service Actions

Key Data	Window Data	Clock Monitor	Action
Match	Match	Match	Valid Service: Restart Service Window
Don't Care	Mismatch	Don't Care	Error: Generate WATCHDOG Output
Mismatch	Don't Care	Don't Care	Error: Generate WATCHDOG Output
Don't Care	Don't Care	Mismatch	Error: Generate WATCHDOG Output

TABLE V. MICROWIRE/PLUS Master Mode Clock Select

SL1	SL0	SK
0	0	$2 \times t_c$
0	1	$4 \times t_c$
1	x	$8 \times t_c$

Where t_c is the instruction cycle clock

MICROWIRE/PLUS (Continued)

MICROWIRE/PLUS OPERATION

Setting the BUSY bit in the PSW register causes the MICROWIRE/PLUS to start shifting the data. It gets reset when eight data bits have been shifted. The user may reset the BUSY bit by software to allow less than 8 bits to shift. If enabled, an interrupt is generated when eight data bits have been shifted. The device may enter the MICROWIRE/PLUS mode either as a Master or as a Slave. *Figure 15* shows how two microcontroller devices and several peripherals may be interconnected using the MICROWIRE/PLUS arrangements.

Warning:

The SIO register should only be loaded when the SK clock is low. Loading the SIO register while the SK clock is high will result in undefined data in the SIO register. SK clock is normally low when not shifting.

Setting the BUSY flag when the input SK clock is high in the MICROWIRE/PLUS slave mode may cause the current SK clock for the SIO shift register to be narrow. For safety, the BUSY flag should only be set when the input SK clock is low.

MICROWIRE/PLUS Master Mode Operation

In the MICROWIRE/PLUS Master mode of operation the shift clock (SK) is generated internally by the device. The MICROWIRE Master always initiates all data exchanges. The MSEL bit in the CNTRL register must be set to enable the SO and SK functions onto the G Port. The SO and SK pins must also be selected as outputs by setting appropriate bits in the Port G configuration register. Table VI summarizes the bit settings required for Master mode of operation.

MICROWIRE/PLUS Slave Mode Operation

In the MICROWIRE/PLUS Slave mode of operation the SK clock is generated by an external source. Setting the MSEL bit in the CNTRL register enables the SO and SK functions onto the G Port. The SK pin must be selected as an input and the SO pin is selected as an output pin by setting and resetting the appropriate bits in the Port G configuration register. Table VI summarizes the settings required to enter the Slave mode of operation.

The user must set the BUSY flag immediately upon entering the Slave mode. This will ensure that all data bits sent by the Master will be shifted properly. After eight clock pulses the BUSY flag will be cleared and the sequence may be repeated.

Alternate SK Phase Operation

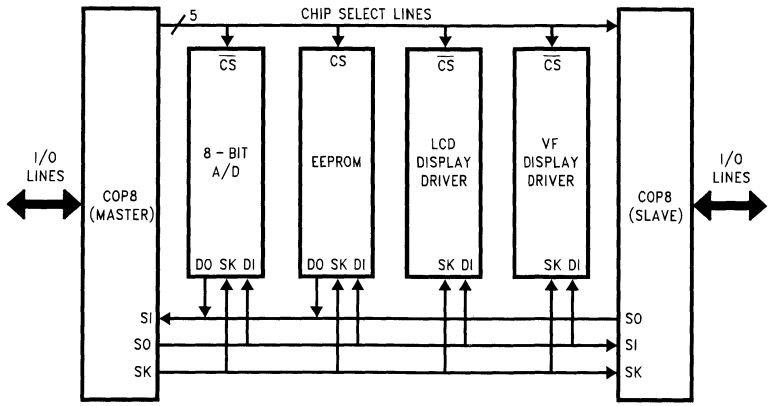
The device allows either the normal SK clock or an alternate phase SK clock to shift data in and out of the SIO register. In both the modes the SK is normally low. In the normal mode data is shifted in on the rising edge of the SK clock and the data is shifted out on the falling edge of the SK clock. The SIO register is shifted on each falling edge of the SK clock. In the alternate SK phase operation, data is shifted in on the falling edge of the SK clock and shifted out on the rising edge of the SK clock.

A control flag, SKSEL, allows either the normal SK clock or the alternate SK clock to be selected. Resetting SKSEL causes the MICROWIRE/PLUS logic to be clocked from the normal SK signal. Setting the SKSEL flag selects the alternate SK clock. The SKSEL is mapped into the G6 configuration bit. The SKSEL flag will power up in the reset condition, selecting the normal SK signal.

TABLE VI

This table assumes that the control flag MSEL is set.

G4 (SO) Config. Bit	G5 (SK) Config. Bit	G4 Fun.	G5 Fun.	Operation
1	1	SO	Int. SK	MICROWIRE/PLUS Master
0	1	TRI-STATE	Int. SK	MICROWIRE/PLUS Master
1	0	SO	Ext. SK	MICROWIRE/PLUS Slave
0	0	TRI-STATE	Ext. SK	MICROWIRE/PLUS Slave



TL/DD/12520–18

FIGURE 15. MICROWIRE/PLUS Application

Memory Map

All RAM, ports and registers (except A and PC) are mapped into data memory address space.

Address S/ADD REG	Contents
0000 to 006F	On-Chip RAM bytes (112 bytes)
0070 to 007F	Unused RAM Address Space (Reads As All Ones)
xx80 to xxAF	Unused RAM Address Space (Reads Undefined Data)
xxB0	Timer T3 Lower Byte
xxB1	Timer T3 Upper Byte
xxB2	Timer T3 Autoload Register T3RA Lower Byte
xxB3	Timer T3 Autoload Register T3RA Upper Byte
xxB4	Timer T3 Autoload Register T3RB Lower Byte
xxB5	Timer T3 Autoload Register T3RB Upper Byte
xxB6	Timer T3 Control Register
xxB7	Comparator Select Register (CMPSL)
xxB8 to xxBF	Reserved
xxC0	Timer T2 Lower Byte
xxC1	Timer T2 Upper Byte
xxC2	Timer T2 Autoload Register T2RA Lower Byte
xxC3	Timer T2 Autoload Register T2RA Upper Byte
xxC4	Timer T2 Autoload Register T2RB Lower Byte
xxC5	Timer T2 Autoload Register T2RB Upper Byte
xxC6	Timer T2 Control Register
xxC7	WATCHDOG Service Register (Reg:WDSVR)
xxC8	MIWU Edge Select Register (Reg:WKEDG)
xxC9	MIWU Enable Register (Reg:WKEN)
xxCA	MIWU Pending Register (Reg:WKPND)
xxCB	Reserved
xxCC	Reserved
xxCD to xxCF	Reserved

Address S/ADD REG	Contents
xxD0	Port L Data Register
xxD1	Port L Configuration Register
xxD2	Port L Input Pins (Read Only)
xxD3	Reserved for Port L
xxD4	Port G Data Register
xxD5	Port G Configuration Register
xxD6	Port G Input Pins (Read Only)
xxD7	Port I Input Pins (Read Only)
xxD8	Port C Data Register
xxD9	Port C Configuration Register
xxDA	Port C Input Pins (Read Only)
xxDB	Reserved for Port C
xxDC	Port D
xxDD to xxDF	Reserved
xxE0 to xxE5	Reserved
xxE6	Timer T1 Autoload Register T1RB Lower Byte
xxE7	Timer T1 Autoload Register T1RB Upper Byte
xxE8	ICNTRL Register
xxE9	MICROWIRE/PLUS Shift Register
xxEA	Timer T1 Lower Byte
xxEB	Timer T1 Upper Byte
xxEC	Timer T1 Autoload Register T1RA Lower Byte
xxED	Timer T1 Autoload Register T1RA Upper Byte
xxEE	CNTRL Control Register
xxEF	PSW Register
xxF0 to FB	On-Chip RAM Mapped as Registers
xxFC	X Register
xxFD	SP Register
xxFE	B Register
xxFF	S Register
0100 to 017F	On-Chip 128 RAM Bytes

Reading memory locations 0070H–007FH (Segment 0) will return all ones. Reading unused memory locations 0080H–00AFH (Segment 0) will return undefined data. Reading memory locations from other unused Segments (i.e., Segment 2, Segment 3, ... etc.) will return all ones.

Addressing Modes

There are ten addressing modes, six for operand addressing and four for transfer of control.

OPERAND ADDRESSING MODES

Register Indirect

This is the "normal" addressing mode. The operand is the data memory addressed by the B pointer or X pointer.

Register Indirect (with auto post increment or decrement of pointer)

This addressing mode is used with the LD and X instructions. The operand is the data memory addressed by the B pointer or X pointer. This is a register indirect mode that automatically post increments or decrements the B or X register after executing the instruction.

Direct

The instruction contains an 8-bit address field that directly points to the data memory for the operand.

Immediate

The instruction contains an 8-bit immediate field as the operand.

Short Immediate

This addressing mode is used with the Load B Immediate instruction. The instruction contains a 4-bit immediate field as the operand.

Indirect

This addressing mode is used with the LAID instruction. The contents of the accumulator are used as a partial address (lower 8 bits of PC) for accessing a data operand from the program memory.

TRANSFER OF CONTROL ADDRESSING MODES

Relative

This mode is used for the JP instruction, with the instruction field being added to the program counter to get the new program location. JP has a range from −31 to +32 to allow a 1-byte relative jump (JP + 1 is implemented by a NOP instruction). There are no "pages" when using JP, since all 15 bits of PC are used.

Absolute

This mode is used with the JMP and JSR instructions, with the instruction field of 12 bits replacing the lower 12 bits of the program counter (PC). This allows jumping to any location in the current 4k program memory segment.

Absolute Long

This mode is used with the JMPL and JSRL instructions, with the instruction field of 15 bits replacing the entire 15 bits of the program counter (PC). This allows jumping to any location up to 32k in the program memory space.

Indirect

This mode is used with the JID instruction. The contents of the accumulator are used as a partial address (lower 8 bits of PC) for accessing a location in the program memory. The contents of this program memory location serve as a partial address (lower 8 bits of PC) for the jump to the next instruction.

The VIS is a special case of the Indirect Transfer of Control addressing mode, where the double byte vector associated with the interrupt is transferred from adjacent addresses in the program memory into the program counter (PC) in order to jump to the associated interrupt service routine.

Instruction Set

Register and Symbol Definition

Registers	
A	8-Bit Accumulator Register
B	8-Bit Address Register
X	8-Bit Address Register
S	8-Bit Segment Register
SP	8-Bit Stack Pointer Register
PC	15-Bit Program Counter Register
PU	Upper 7 Bits of PC
PL	Lower 8 Bits of PC
C	1 Bit of PSW Register for Carry
HC	1 Bit of PSW Register for Half Carry
GIE	1 Bit of PSW Register for Global Interrupt Enable
VU	Interrupt Vector Upper Byte
VL	Interrupt Vector Lower Byte

Symbols	
[B]	Memory Indirectly Addressed by B Register
[X]	Memory Indirectly Addressed by X Register
MD	Direct Addressed Memory
Mem	Direct Addressed Memory or [B]
Meml	Direct Addressed Memory or [B] or Immediate Data
Imm	8-Bit Immediate Data
Reg	Register Memory: Addresses F0 to FF (Includes B, X and SP)
Bit	Bit Number (0 to 7)
←	Loaded with
↔	Exchanged with

Instruction Set (Continued)

INSTRUCTION SET

ADD	A,Meml	ADD	A ← A + Meml
ADC	A,Meml	ADD with Carry	A ← A + Meml + C, C ← Carry HC ← Half Carry
SUBC	A,Meml	Subtract with Carry	A ← A − $\overline{\text{Meml}}$ + C, C ← Carry HC ← Half Carry
AND	A,Meml	Logical AND	A ← A and Meml
ANDSZ	A,Imm	Logical AND Immed., Skip if Zero	Skip next if (A and Imm) = 0
OR	A,Meml	Logical OR	A ← A or Meml
XOR	A,Meml	Logical EXclusive OR	A ← A xor Meml
IFEQ	MD,Imm	IF EQual	Compare MD and Imm, Do next if MD = Imm
IFEQ	A,Meml	IF EQual	Compare A and Meml, Do next if A = Meml
IFNE	A,Meml	IF Not Equal	Compare A and Meml, Do next if A ≠ Meml
IFGT	A,Meml	IF Greater Than	Compare A and Meml, Do next if A > Meml
IFBNE	#	If B Not Equal	Do next if lower 4 bits of B ≠ Imm
DRSZ	Reg	Decrement Reg., Skip if Zero	Reg ← Reg − 1, Skip if Reg = 0
SBIT	#,Mem	Set BIT	1 to bit, Mem (bit = 0 to 7 immediate)
RBIT	#,Mem	Reset BIT	0 to bit, Mem
IFBIT	#,Mem	IF BIT	If bit in A or Mem is true do next instruction
RPND		Reset PeNDing Flag	Reset Software Interrupt Pending Flag
X	A,Mem	EXchange A with Memory	A ⟷ Mem
X	A,[X]	EXchange A with Memory [X]	A ⟷ [X]
LD	A,Meml	LoaD A with Memory	A ← Meml
LD	A,[X]	LoaD A with Memory [X]	A ← [X]
LD	B,Imm	LoaD B with Immed.	B ← Imm
LD	Mem,Imm	LoaD Memory Immed	Mem ← Imm
LD	Reg,Imm	LoaD Register Memory Immed.	Reg ← Imm
X	A, [B ±]	EXchange A with Memory [B]	A ⟷ [B], (B ← B ± 1)
X	A, [X ±]	EXchange A with Memory [X]	A ⟷ [X], (X ← ± 1)
LD	A, [B±]	LoaD A with Memory [B]	A ← [B], (B ← B ± 1)
LD	A, [X±]	LoaD A with Memory [X]	A ← [X], (X ← X ± 1)
LD	[B±],Imm	LoaD Memory [B] Immed.	[B] ← Imm, (B ← B ± 1)
CLR	A	CLeaR A	A ← 0
INC	A	INCrement A	A ← A + 1
DEC	A	DECrement A	A ← A − 1
LAID		Load A InDirect from ROM	A ← ROM (PU,A)
DCOR	A	Decimal CORrect A	A ← BCD correction of A (follows ADC, SUBC)
RRC	A	Rotate A Right thru C	C → A7 → . . . → A0 → C
RLC	A	Rotate A Left thru C	C ← A7 ← . . . ← A0 ← C
SWAP	A	SWAP nibbles of A	A7 . . . A4 ⟷ A3 . . . A0
SC		Set C	C ← 1, HC ← 1
RC		Reset C	C ← 0, HC ← 0
IFC		IF C	IF C is true, do next instruction
IFNC		IF Not C	If C is not true, do next instruction
POP	A	POP the stack into A	SP ← SP + 1, A ← [SP]
PUSH	A	PUSH A onto the stack	[SP] ← A, SP ← SP − 1
VIS		Vector to Interrupt Service Routine	PU ← [VU], PL ← [VL]
JMPL	Addr.	Jump absolute Long	PC ← ii (ii = 15 bits, 0 to 32k)
JMP	Addr.	Jump absolute	PC9 . . . 0 ← i (i = 12 bits)
JP	Disp.	Jump relative short	PC ← PC + r (r is −31 to +32, except 1)
JSRL	Addr	Jump SubRoutine Long	[SP] ← PL, [SP−1] ← PU,SP−2, PC ← ii
JSR	Addr	Jump SubRoutine	[SP] ← PL, [SP−1] ← PU,SP−2, PC9 . . . 0 ← i
JID		Jump InDirect	PL ← ROM (PU,A)
RET		RETurn from subroutine	SP + 2, PL ← [SP], PU ← [SP−1]
RETSK		RETurn and SKip	SP + 2, PL ← [SP],PU ← [SP−1]
RETI		RETurn from Interrupt	SP + 2, PL ← [SP],PU ← [SP−1],GIE ← 1
INTR		Generate an Interrupt	[SP] ← PL, [SP−1] ← PU, SP−2, PC ← 0FF
NOP		No OPeration	PC ← PC + 1

Instruction Execution Time

Most instructions are single byte (with immediate addressing mode instructions taking two bytes).

Most single byte instructions take one cycle time to execute.

Skipped instructions require x number of cycles to be skipped, where x equals the number of bytes in the skipped instruction opcode.

See the BYTES and CYCLES per INSTRUCTION table for details.

Bytes and Cycles per Instruction

The following table shows the number of bytes and cycles for each instruction in the format of byte/cycle.

Arithmetic and Logic Instructions

	[B]	Direct	Immed.
ADD	1/1	3/4	2/2
ADC	1/1	3/4	2/2
SUBC	1/1	3/4	2/2
AND	1/1	3/4	2/2
OR	1/1	3/4	2/2
XOR	1/1	3/4	2/2
IFEQ	1/1	3/4	2/2
IFNE	1/1	3/4	2/2
IFGT	1/1	3/4	2/2
IFBNE	1/1		
DRSZ		1/3	
SBIT	1/1	3/4	
RBIT	1/1	3/4	
IFBIT	1/1	3/4	

RPND	1/1

Instructions Using A & C

CLRA	1/1
INCA	1/1
DECA	1/1
LAID	1/3
DCOR	1/1
RRCA	1/1
RLCA	1/1
SWAPA	1/1
SC	1/1
RC	1/1
IFC	1/1
IFNC	1/1
PUSHA	1/3
POPA	1/3
ANDSZ	2/2

Transfer of Control Instructions

JMPL	3/4
JMP	2/3
JP	1/3
JSRL	3/5
JSR	2/5
JID	1/3
VIS	1/5
RET	1/5
RETSK	1/5
RETI	1/5
INTR	1/7
NOP	1/1

Memory Transfer Instructions

	Register Indirect		Direct	Immed.	Register Indirect Auto Incr. & Decr.		
	[B]	[X]			[B+, B−]	[X+, X−]	
X A,*	1/1	1/3	2/3		1/2	1/3	
LD A,*	1/1	1/3	2/3	2/2	1/2	1/3	
LD B, Imm				1/1			(IF B < 16)
LD B, Imm				2/2			(IF B > 15)
LD Mem, Imm	2/2		3/3		2/2		
LD Reg, Imm			2/3				
IFEQ MD, Imm			3/3				

* = > Memory location addressed by B or X or directly.

Opcode Table

Upper Nibble																Lower Nibble
F	E	D	C	B	A	9	8	7	6	5	4	3	2	1	0	
JP−15	JP−31	LD 0F0, # i	DRSZ 0F0	RRCA	RC	ADC A,#i	ADC A,[B]	IFBIT 0,[B]	ANDSZ A, #i	LD B,#0F	IFBNE 0	JSR x000−x0FF	JMP x000−x0FF	JP+17	INTR	0
JP−14	JP−30	LD 0F1, # i	DRSZ 0F1	*	SC	SUBC A, #i	SUB A,[B]	IFBIT 1,[B]	*	LD B,#0E	IFBNE 1	JSR x100−x1FF	JMP x100−x1FF	JP+18	JP+2	1
JP−13	JP−29	LD 0F2, # i	DRSZ 0F2	X A,[X+]	X A,[B+]	IFEQ A,#i	IFEQ A,[B]	IFBIT 2,[B]	*	LD B,#0D	IFBNE 2	JSR x200−x2FF	JMP x200−x2FF	JP+19	JP+3	2
JP−12	JP−28	LD 0F3, # i	DRSZ 0F3	X A,[X−]	X A,[B−]	IFGT A,#i	IFGT A,[B]	IFBIT 3,[B]	*	LD B,#0C	IFBNE 3	JSR x300−x3FF	JMP x300−x3FF	JP+20	JP+4	3
JP−11	JP−27	LD 0F4, # i	DRSZ 0F4	VIS	LAID	ADD A,#i	ADD A,[B]	IFBIT 4,[B]	CLRA	LD B,#0B	IFBNE 4	JSR x400−x4FF	JMP x400−x4FF	JP+21	JP+5	4
JP−10	JP−26	LD 0F5, # i	DRSZ 0F5	RPND	JID	AND A,#i	AND A,[B]	IFBIT 5,[B]	SWAPA	LD B,#0A	IFBNE 5	JSR x500−x5FF	JMP x500−x5FF	JP+22	JP+6	5
JP−9	JP−25	LD 0F6, # i	DRSZ 0F6	X A,[X]	X A,[B]	XOR A,#i	XOR A,[B]	IFBIT 6,[B]	DCORA	LD B,#09	IFBNE 6	JSR x600−x6FF	JMP x600−x6FF	JP+23	JP+7	6
JP−8	JP−24	LD 0F7, # i	DRSZ 0F7	*	*	OR A,#i	OR A,[B]	IFBIT 7,[B]	PUSHA	LD B,#08	IFBNE 7	JSR x700−x7FF	JMP x700−x7FF	JP+24	JP+8	7
JP−7	JP−23	LD 0F8, # i	DRSZ 0F8	NOP	RLCA	LD A,#i	IFC	SBIT 0,[B]	RBIT 0,[B]	LD B,#07	IFBNE 8	JSR x800−x8FF	JMP x800−x8FF	JP+25	JP+9	8
JP−6	JP−22	LD 0F9, # i	DRSZ 0F9	IFNE A,[B]	IFEQ Md,#i	IFNE A,#i	IFNC	SBIT 1,[B]	RBIT 1,[B]	LD B,#06	IFBNE 9	JSR x900−x9FF	JMP x900−x9FF	JP+26	JP+10	9
JP−5	JP−21	LD 0FA, # i	DRSZ 0FA	LD A,[X+]	LD A,[B+]	LD [B+],#i	INCA	SBIT 2,[B]	RBIT 2,[B]	LD B,#05	IFBNE 0A	JSR xA00−xAFF	JMP xA00−xAFF	JP+27	JP+11	A
JP−4	JP−20	LD 0FB, # i	DRSZ 0FB	LD A,[X−]	LD A,[B−]	LD [B−],#i	DECA	SBIT 3,[B]	RBIT 3,[B]	LD B,#04	IFBNE 0B	JSR xB00−xBFF	JMP xB00−xBFF	JP+28	JP+12	B
JP−3	JP−19	LD 0FC, # i	DRSZ 0FC	LD Md,#i	JMPL	X A,Md	POPA	SBIT 4,[B]	RBIT 4,[B]	LD B,#03	IFBNE 0C	JSR xC00−xCFF	JMP xC00−xCFF	JP+29	JP+13	C
JP−2	JP−18	LD 0FD, # i	DRSZ 0FD	DIR	JSRL	LD A,Md	RETSK	SBIT 5,[B]	RBIT 5,[B]	LD B,#02	IFBNE 0D	JSR xD00−xDFF	JMP xD00−xDFF	JP+30	JP+14	D
JP−1	JP−17	LD 0FE, # i	DRSZ 0FE	LD A,[X]	LD A,[B]	LD [B],#i	RET	SBIT 6,[B]	RBIT 6,[B]	LD B,#01	IFBNE 0E	JSR xE00−xEFF	JMP xE00−xEFF	JP+31	JP+15	E
JP−0	JP−16	LD 0FF, # i	DRSZ 0FF	*	*	LD B,#i	RETI	SBIT 7,[B]	RBIT 7,[B]	LD B,#00	IFBNE 0F	JSR xF00−xFFF	JMP xF00−xFFF	JP+32	JP+16	F

Where,

i is the immediate data

Md is a directly addressed memory location

* is an unused opcode

Note: The opcode 60 Hex is also the opcode for IFBIT #i,A

Development Support

SUMMARY

- iceMASTER™: IM-COP8/400—Full feature in-circuit emulation for all COP8 products. A full set of COP8 Basic and Feature Family device and package specific probes are available.
- COP8 Debug Module: Moderate cost in-circuit emulation and development programming unit.
- COP8 Evaluation and Programming Unit: EPU-COP888GG—low cost In-circuit simulation and development programming unit.
- Assembler: COP8-DEV-IBMA. A DOS installable cross development Assembler, Linker, Librarian and Utility Software Development Tool Kit.
- C Compiler: COP8C. A DOS installable cross development Software Tool Kit.
- OTP/EPROM Programmer Support: Covering needs from engineering prototype, pilot production to full production environments.

iceMASTER (IM) IN-CIRCUIT EMULATION

The iceMASTER IM-COP8/400 is a full feature, PC based, in-circuit emulation tool developed and marketed by MetaLink Corporation to support the whole COP8 family of products. National is a resale vendor for these products.

See *Figure 16* for configuration.

The iceMASTER IM-COP8/400 with its device specific COP8 Probe provides a rich feature set for developing, testing and maintaining product:

- Real-time in-circuit emulation; full 2.4V–5.5V operation range, full DC-10 MHz clock. Chip options are programmable or jumper selectable.
- Direct connection to application board by package compatible socket or surface mount assembly.
- Full 32 kbytes of loadable programming space that overlays (replaces) the on-chip ROM or EPROM. On-chip RAM and I/O blocks are used directly or recreated on the probe as necessary.
- Full 4k frame synchronous trace memory. Address, instruction, and 8 unspecified, circuit connectable trace lines. Display can be HLL source (e.g., C source), assembly or mixed.
- A full 64k hardware configurable break, trace on, trace off control, and pass count increment events.
- Tool set integrated interactive symbolic debugger—supports both assembler (COFF) and C Compiler (.COD) linked object formats.
- Real time performance profiling analysis; selectable bucket definition.
- Watch windows, content updated automatically at each execution break.
- Instruction by instruction memory/register changes displayed on source window when in single step operation.
- Single base unit and debugger software reconfigurable to support the entire COP8 family; only the probe personality needs to change. Debugger software is processor customized, and reconfigured from a master model file.
- Processor specific symbolic display of registers and bit level assignments, configured from master model file.
- Halt/Idle mode notification.
- On-line HELP customized to specific processor using master model file.
- Includes a copy of COP8-DEV-IBMA assembler and linker SDK.

IM Order Information

Base Unit	
IM-COP8/400-1	iceMASTER Base Unit, 110V Power Supply
IM-COP8/400-2	iceMASTER Base Unit, 220V Power Supply
iceMASTER Probe	
MHW-884EK28DWPC	28 DIP
MHW-888EK40DWPC	40 DIP
MHW-888EK44PWPC	44 PLCC
28 DIP to 28 SO Adapter	
MHW-SOIC28	28 SO

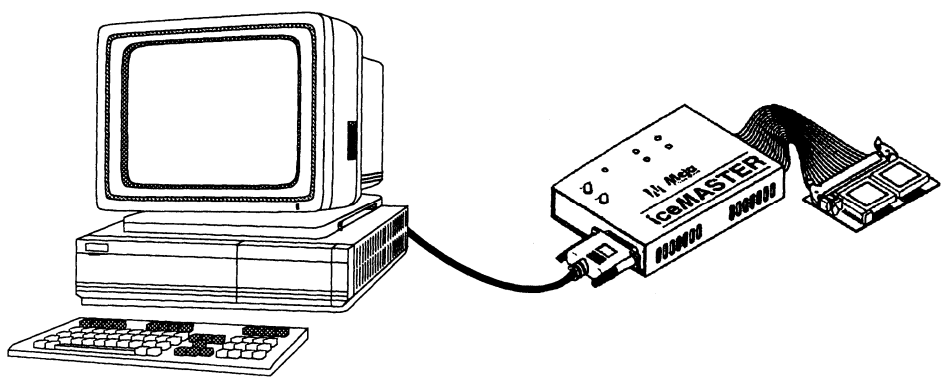


FIGURE 16. COP8 iceMASTER Environment

TL/DD/12520–19

Development Support (Continued)

iceMASTER DEBUG MODULE (DM)

The iceMASTER Debug Module is a PC based, combination in-circuit emulation tool and COP8 based OTP/EPROM programming tool developed and marketed by MetaLink Corporation to support the whole COP8 family of products. National is a resale vendor for these products.

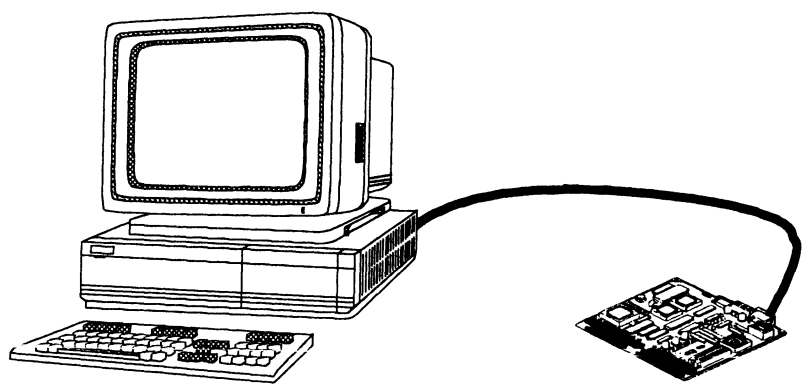
See *Figure 17* for configuration.

The iceMASTER Debug Module is a moderate cost development tool. It has the capability of in-circuit emulation for a specific COP8 microcontroller and in addition serves as a programming tool for COP8 OTP and EPROM product families. Summary of features is as follows:

- Real-time in-circuit emulation; full operating voltage range operation, full DC-10 MHz clock.
- All processor I/O pins can be cabled to an application development board with package compatible cable to socket and surface mount assembly.
- Full 32 kbytes of loadable programming space that overlays (replaces) the on-chip ROM or EPROM. On-chip RAM and I/O blocks are used directly or recreated as necessary.
- 100 frames of synchronous trace memory. The display can be HLL source (C source), assembly or mixed. The most recent history prior to a break is available in the trace memory.
- Configured break points; uses INTR instruction which is modestly intrusive.
- Software—only supported features are selectable.
- Tool set integrated interactive symbolic debugger—supports both assembler (COFF) and C Compiler (.COD) SDK linked object formats.
- Instruction by instruction memory/register changes displayed when in single step operation.
- Debugger software is processor customized, and reconfigured from a master model file.
- Processor specific symbolic display of registers and bit level assignments, configured from master model file.
- Halt/Idle mode notification.
- Programming menu supports full product line of programmable OTP and EPROM COP8 products. Program data is taken directly from the overlay RAM.
- Programming of 44 PLCC and 68 PLCC parts requires external programming adapters.
- Includes wall mount power supply.
- On-board V_{PP} generator from 5V input or connection to external supply supported. Requires V_{PP} level adjustment per the family programming specification (correct level is provided on an on-screen pop-down display).
- On-line HELP customized to specific processor using master model file.
- Includes a copy of COP8-DEV-IBMA assembler and linker SDK.

DM Order Information

Debug Model Unit	
COP8-DM/888EK	
Cable Adapters	
DM-COP8/28D	28 DIP
DM-COP8/40D	40 DIP
DM-COP8/44P	44 PLCC
28 DIP to 28 SO Adapter	
DM-COP8/28D-SO	28 SO



TL/DD/12520–20

FIGURE 17. COP8-DM Environment

Development Support (Continued)

COP8 ASSEMBLER/LINKER SOFTWARE DEVELOPMENT TOOL KIT

National Semiconductor offers a relocatable COP8 macro cross assembler, linker, librarian and utility software development tool kit. Features are summarized as follows:

- Basic and Feature Family instruction set by "device" type.
- Nested macro capability.
- Extensive set of assembler directives.
- Supported on PC/DOS platform.
- Generates National standard COFF output files.
- Integrated Linker and Librarian.
- Integrated utilities to generate ROM code file outputs.
- DUMPCOFF utility.

This product is integrated as a part of MetaLink tools as a development kit, fully supported by the MetaLink debugger. It may be ordered separately or it is bundled with the MetaLink products at no additional cost.

Order Information

Assembler SDK	
COP8-DEV-IBMA	Assembler SDK on installable 3.5" PC/DOS Floppy Disk Drive format. Periodic upgrades and most recent version is available on National's BBS and Internet.

COP8 C COMPILER

A C Compiler is developed and marketed by Byte Craft Limited. The COP8C compiler is a fully integrated development tool specifically designed to support the compact embedded configuration of the COP8 family of products.

Features are summarized as follows:

- ANSI C with some restrictions and extensions that optimize development for the COP8 embedded application.
- BITS data type extension. Register declaration #pragma with direct bit level definitions.
- C language support for interrupt routines.
- Expert system, rule based code generation and optimization.
- Performs consistency checks against the architectural definitions of the target COP8 device.
- Generates program memory code.
- Supports linking of compiled object or COP8 assembled object formats.
- Global optimization of linked code.
- Symbolic debug load format fully sourced level supported by the MetaLink debugger.

OTP/EMULATOR SUPPORT

The COP87L88EK/COP87L84EK devices provide emulation and OTP support for the COP888EK/COP884EK mask programmable devices.

Approved List

Manufacturer	North America	Europe	Asia
BP Microsystems	(800) 225-2102 (713) 688-4600 Fax: (713) 688-0920	+49-8152-4183 +49-8856-932616	+852-234-16611 +852-2710-8121
Data I/O	(800) 426-1045 (206) 881-6444 Fax: (206) 882-1043	+44-0734-440011	Call North America
HI-LO	(510) 623-8860	Call Asia	+886-2-764-0215 Fax: +886-2-756-6403
ICE Technology	(800) 624-8949 (919) 430-7915	+44-1226-767404 Fax: 0-1226-370-434	
MetaLink	(800) 638-2423 (602) 926-0797 Fax: (602) 693-0681	+49-80 9156 96-0 Fax: +49-80 9123 86	+852-737-1800
Systems General	(408) 263-6667	+41-1-9450300	+886-2-917-3005 Fax: +886-2-911-1283
Needhams	(916) 924-8037 Fax: (916) 924-8065		

Development Support (Continued)

OTP Ordering Information

Device Number	Clock Option	Package	Emulates
COP87L88EKV-XE	Crystal/HALT En	44 PLCC	COP888EK
COP87L88EKN-XE	Crystal/HALT En	40 DIP	COP888EK
COP87L84EKN-XE	Crystal/HALT En	28 DIP	COP884EK
COP87L84EKM-XE	Crystal/HALT En	28 SO	COP884EK

*Check with the local sales office about the availability.

INDUSTRY WIDE OTP/EPROM PROGRAMMING SUPPORT

Programming support, in addition to the MetaLink development tools, is provided by a full range of independent approved vendors to meet the needs from the engineering laboratory to full production.

AVAILABLE LITERATURE

For more information, please see the COP8 Basic Family User's Manual, Literature Number 620895, COP8 Feature Family User's Manual, Literature Number 620897 and National's Family of 8-bit Microcontrollers COP8 Selection Guide, Literature Number 630009.

DIAL-A-HELPER SERVICE

Dial-A-Helper is a service provided by the Microcontroller Applications group. The Dial-A-Helper is an Electronic Information System that may be accessed as a Bulletin Board System (BBS) via data modem, as an FTP site on the Internet via standard FTP client application or as an FTP site on the Internet using a standard Internet browser such as Netscape or Mosaic.

The Dial-A-Helper system provides access to an automated information storage and retrieval system. The system capabilities include a MESSAGE SECTION (electronic mail, when accessed as a BBS) for communications to and from the Microcontroller Applications Group and a FILE SECTION which consists of several file areas where valuable application software and utilities could be found.

DIAL-A-HELPER BBS via a Standard Modem

Modem: CANADA/U.S.: (800) NSC-MICRO
(800) 672-6427
EUROPE: (+49) 0-8141-351332
Baud: 14.4k
Set-Up: Length: 8-Bit
Parity: None
Stop Bit: 1
Operation: 24 Hours, 7 Days

DIAL-A-HELPER via FTP

ftp nscmicro.nsc.com
user: anonymous
password: username@yourhost.site.domain

DIAL-A-HELPER via a WorldWide Web Browser

ftp://nscmicro.nsc.com

National Semiconductor on the WorldWide Web

See us on the WorldWide Web at: http://www.natsemi.com

CUSTOMER RESPONSE CENTER

Complete product information and technical support is available from National's customer response centers.

CANADA/US:	Tel:	(800) 272-9959
	email:	support @tevm2.nsc.com
EUROPE:	email:	europe.support@nsc.com
	Deutsch Tel:	+49 (0) 180-530 85 85
	English Tel:	+49 (0) 180-532 78 32
	Français Tel:	+49 (0) 180-532 93 58
	Italiano Tel:	+49 (0) 180-534 16 80
JAPAN:	Tel:	+81-043-299-2309
S.E. ASIA:	Beijing Tel:	(+86) 10-6856-8601
	Shanghai Tel:	(+86) 21-6415-4092
	Hong Kong Tel:	(+852) 2737-1600
	Korea Tel:	(+82) 2-3771-6909
	Malaysia Tel:	(+60-4) 644-9061
	Singapore Tel:	(+65) 255-2226
	Taiwan Tel:	+886-2-521-3288
AUSTRALIA:	Tel:	(+61) 3-9558-9999
INDIA:	Tel:	(+91) 80-559-9467

National Semiconductor

PRELIMINARY

COP87L88EG/COP87L84EG 8-Bit One-Time Programmable (OTP) Microcontroller with UART and Three Multi-Function Timers

General Description

The COP87L88EG/COP87L84EG OTP microcontrollers are members of the COP8™ feature family using an 8-bit core architecture. It is pin and software compatible to the mask ROM COP888EG/COP884EG product family.

(Continued)

Key Features

- Full duplex UART
- Two analog comparators
- Three 16-bit timers, each with two 16-bit registers supporting:
  - — Processor independent PWM mode
  - — External event counter mode
  - — Input capture mode
- 8 kbytes on-board EPROM with security feature
- 256 bytes on-board RAM

Additional Peripheral Features

- Idle timer
- Multi-Input Wake Up (MIWU) with optional interrupts (8)
- WATCHDOG™ and clock monitor logic
- MICROWIRE/PLUS™ serial I/O

I/O Features

- Memory mapped I/O
- Software selectable I/O options (TRI-STATE® output, push-pull output, weak pull-up input, high impedance input
- Schmitt trigger inputs on ports G and L
- Packages:
  - — 44 PLCC with 40 I/O pins
  - — 40 DIP with 36 I/O pins
  - — 28 DIP with 24 I/O pins
  - — 28 SO with 24 I/O pins (contact local sales office for availability)

CPU/Instruction Set Features

- 1 μs instruction cycle time
- Fourteen multi-source vectored interrupts servicing
  - — External interrupt
  - — Idle timer T0
  - — Two timers (each with 2 interrupts)
  - — MICROWIRE/PLUS
  - — Multi-Input Wake Up
  - — Software trap
  - — UART (2)
  - — Default VIS (default interrupt)
- Versatile instruction set with true bit manipulation
- 8-bit Stack Pointer SP (stack in RAM)
- Two 8-bit register indirect data memory pointers (B and X)

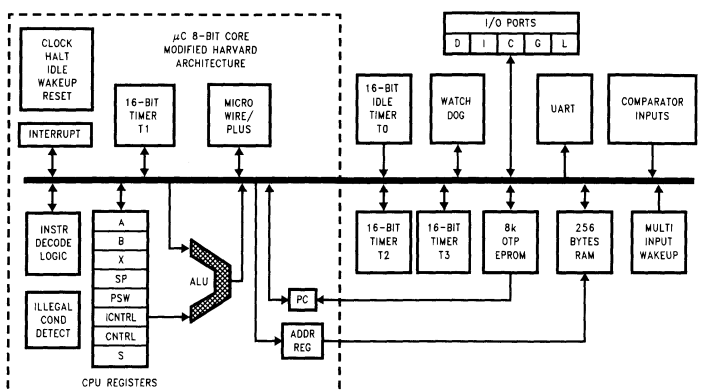
Fully Static CMOS

- Two power saving modes: HALT and IDLE
- Single supply operation: 2.7V–5.5V
- Temperature range: −40°C to +85°C

Development Support

- Emulation device for the COP888EG/COP884EG, COP888CG/COP884CG and COP888CS/COP884CS
- Real time emulation and full program debug offered by MetaLink Development System

Block Diagram



TL/DD12525–20

FIGURE 1. Block Diagram

General Description (Continued)

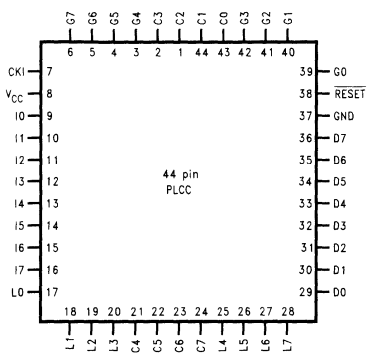
The device is a fully static part, fabricated using double-metal silicon gate microCMOS technology. Features include an 8-bit memory mapped architecture, MICROWIRE/PLUS™ serial I/O, three 16-bit timer/counters supporting three modes (Processor Independent PWM generation, External Event counter, and Input Capture mode capabilities), full duplex UART and two comparators. Each I/O pin has software selectable configurations. The devices operates over a voltage range of 2.7V to 5.5V. High throughput is achieved with an efficient, regular instruction set operating at a maximum of 1 μs per instruction rate.

The following table shows the differences between the various devices.

	ROM (Bytes)	RAM (Bytes)	Timers	# of Comparators
COP87L88EG/ COP87L84EG	8k	256	T0, T1, T2, T3	2
COP888EG/ COP884EG	8k	256	T0, T1, T2, T3	2
COP888CG/ COP884CG	4k	192	T0, T1, T2, T3	2
COP888CS/ COP884CS	4k	192	T0, T1	1

Connection Diagrams

Plastic Chip Carrier

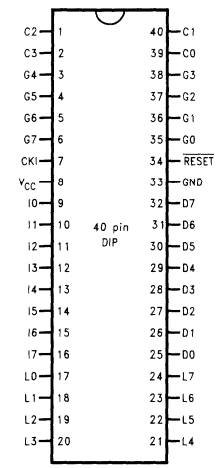


TL/DD12525–1

Top View

Order Number COP87L88EGV-XE
See NS Package Number V44A

Dual-In-Line Package

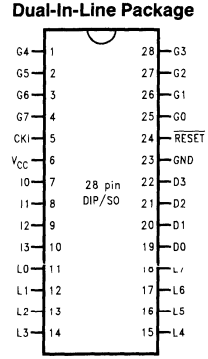


TL/DD12525–2

Top View

Order Number COP87L88EGN-XE
See NS Package Number N40A

Dual-In-Line Package



TL/DD12525–3

Note: -X Crystal Oscillator
-E Halt Enable

Top View

Order Number COP87L84EGN-XE
See NS Package Number M28B or N28B

FIGURE 2. COP87L88EG/COP87L84EG Connection Diagrams

Connection Diagrams (Continued)

Pinouts for 28-, 40- and 44-Pin Packages

Port	Type	Alt. Fun	Alt. Fun	28-Pin Pkg.	40-Pin Pkg.	44-Pin Pkg.
L0	I/O	MIWU		11	17	17
L1	I/O	MIWU	CKX	12	18	18
L2	I/O	MIWU	TDX	13	19	19
L3	I/O	MIWU	RDX	14	20	20
L4	I/O	MIWU	T2A	15	21	25
L5	I/O	MIWU	T2B	16	22	26
L6	I/O	MIWU	T3A	17	23	27
L7	I/O	MIWU	T3B	18	24	28
G0	I/O	INT		25	35	39
G1	WDOUT			26	36	40
G2	I/O	T1B		27	37	41
G3	I/O	T1A		28	38	42
G4	I/O	SO		1	3	3
G5	I/O	SK		2	4	4
G6	I	SI		3	5	5
G7	I/CKO	HALT Restart		4	6	6
D0	O			19	25	29
D1	O			20	26	30
D2	O			21	27	31
D3	O			22	28	32
I0	I			7	9	9
I1	I	COMP1IN−		8	10	10
I2	I	COMP1IN+		9	11	11
I3	I	COMP1OUT		10	12	12
I4	I	COMP2IN−			13	13
I5	I	COMP2IN+			14	14
I6	I	COMP2OUT			15	15
I7	I				16	16
D4	O				29	33
D5	O				30	34
D6	O				31	35
D7	O				32	36
C0	I/O				39	43
C1	I/O				40	44
C2	I/O				1	1
C3	I/O				2	2
C4	I/O					21
C5	I/O					22
C6	I/O					23
C7	I/O					24
V_{CC}				6	8	8
GND				23	33	37
CKI				5	7	7
$\overline{\text{RESET}}$				24	34	38

Absolute Maximum Ratings (Note)

If Military/Aerospace specified devices are required, please contact the National Semiconductor Sales Office/Distributors for availability and specifications.

Supply Voltage (V_{CC})	7V
Voltage at Any Pin	−0.3V to V_{CC} + 0.3V
Total Current into V_{CC} Pin (Source)	100 mA
Total Current out of GND Pin (Sink)	110 mA
Storage Temperature Range	−65°C to +140°C

Note: *Absolute maximum ratings indicate limits beyond which damage to the device may occur. DC and AC electrical specifications are not ensured when operating the device at absolute maximum ratings.*

DC Electrical Characteristics $-40°C \le T_A \le +85°C$ unless otherwise specified

Parameter	Conditions	Min	Typ	Max	Units
Operating Voltage		2.7		5.5	V
Power Supply Ripple (Note 1)	Peak-to-Peak			0.1 V_{CC}	V
Supply Current (Note 2)					
CKI = 10 MHz	V_{CC} = 5.5V, t_c = 1 μs			16.5	mA
CKI = 4 MHz	V_{CC} = 4.0V, t_c = 2.5 μs			6.5	mA
HALT Current (Note 3)	V_{CC} = 5.5V, CKI = 0 MHz			12	μA
	V_{CC} = 4.0V, CKI = 0 MHz			8	μA
IDLE Current					
CKI = 10 MHz	V_{CC} = 5.5V, t_c = 1 μs			3.5	mA
CKI = 1 MHz	V_{CC} = 4.0V, t_c = 10 μs			0.7	mA
Input Levels					
$\overline{RESET}$					
Logic High		0.8 V_{CC}			V
Logic Low				0.2 V_{CC}	V
CKI (External and Crystal Osc. Modes)					
Logic High		0.7 V_{CC}			V
Logic Low				0.2 V_{CC}	V
All Other Inputs					
Logic High		0.7 V_{CC}			V
Logic Low				0.2 V_{CC}	V
Hi-Z Input Leakage	V_{CC} = 5.5V	−2		+2	μA
Input Pullup Current	V_{CC} = 5.5V	40		250	μA
G and L Port Input Hysteresis			0.05 V_{CC}	0.35 V_{CC}	V
Output Current Levels					
D Outputs					
Source	V_{CC} = 4.5V, V_{OH} = 3.3V	0.4			mA
Sink (Note 4)	V_{CC} = 4.5V, V_{OL} = 1V	10			mA
All Others					
Source (Weak Pull-Up Mode)	V_{CC} = 4.5V, V_{OH} = 2.7V	10		100	μA
Source (Push-Pull Mode)	V_{CC} = 4.5V, V_{OH} = 3.3V	0.4			mA
Sink (Push-Pull Mode)	V_{CC} = 4.5V, V_{OL} = 0.4V	1.6			mA
TRI-STATE Leakage	V_{CC} = 5.5V	−2		+2	μA
Allowable Sink/Source Current per Pin					
D Outputs (Sink)				15	mA
All others				3	mA
Maximum Input Current without Latchup (Note 5)	T_A = 25°C			±100	mA
RAM Retention Voltage, V_r	500 ns Rise and Fall Time (Min)	2			V
Input Capacitance				7	pF
Load Capacitance on D2				1000	pF

Note 1: Rate of voltage change must be less then 0.5 V/ms.

Note 2: Supply current is measured after running 2000 cycles with a square wave CKI input, CKO open, inputs at rails and outputs open.

Note 3: The HALT mode will stop CKI from oscillating in the RC and the Crystal configurations by bringing CKI high. Test conditions: All inputs tied to V_{CC}, L and G ports in the TRI-STATE mode and tied to ground, all outputs low and tied to ground. The clock monitor is disabled.

Note 4: The user must guarantee that D2 pin does not source more than 10 mA during RESET. If D2 sources more than 10 mA during reset, the device will go into programming mode.

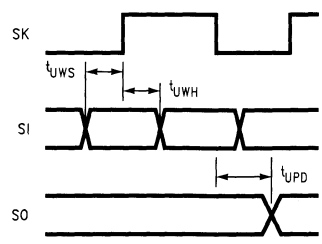
Note 5: Pins G6 and $\overline{RESET}$ are designed with a high voltage input network for factory testing. These pins allow input voltages greater than V_{CC} and the pins will have sink current to V_{CC} when biased at voltages greater than V_{CC} (the pins do not have source current when biased at a voltage below V_{CC}). The effective resistance to V_{CC} is 750Ω (typical). These two pins will not latch up. The voltage at the pins must be limited to less than 14V.

AC Electrical Characteristics $-40°C \le T_A \le +85°C$ unless otherwise specified

Parameter	Conditions	Min	Typ	Max	Units
Instruction Cycle Time (t_c)					
Crystal, Resonator,		1		DC	μs
R/C Oscillator		3		DC	μs
Inputs					
t_{SETUP}		200			ns
t_{HOLD}		60			ns
Output Propagation Delay	$R_L = 2.2k, C_L = 100$ pF				
t_{PD1}, t_{PD0}					
SO, SK				0.7	μs
All Others				1	μs
MICROWIRE™ Setup Time (t_{UWS})		20			ns
MICROWIRE Hold Time (t_{UWH})		56			ns
MICROWIRE Output Propagation Delay (t_{UPD})				220	ns
Input Pulse Width					
Interrupt Input High Time		1			t_c
Interrupt Input Low Time		1			t_c
Timer Input High Time		1			t_c
Timer Input Low Time		1			t_c
Reset Pulse Width		1			μs

Comparators AC and DC Characteristics $V_{CC} = 5V, T_A = 25°C$

Parameter	Conditions	Min	Typ	Max	Units
Input Offset Voltage	$0.4V \le V_{IN} \le V_{CC} - 1.5V$		±10	±25	mV
Input Common Mode Voltage Range		0.4		$V_{CC} - 1.5$	V
Low Level Output Current	$V_{OL} = 0.4V$	1.6			mA
High Level Output Current	$V_{OH} = 4.6V$	1.6			mA
DC Supply Current Per Comparator (When Enabled)				250	μA
Response Time	TBD mV Step, TBD mV Overdrive, 100 pF Load		1		μs



TL/DD12525-4

FIGURE 3. MICROWIRE/PLUS Timing

Pin Descriptions

V_{CC} and GND are the power supply pins.

CKI is the clock input. This can come from an R/C generated oscillator, or a crystal oscillator (in conjunction with CKO). See Oscillator Description section.

$\overline{\text{RESET}}$ is the master reset input. See Reset Description section.

The device contains three bidirectional 8-bit I/O ports (C, G and L), where each individual bit may be independently configured as an input (Schmitt trigger inputs on ports L and G), output or TRI-STATE under program control. Three data memory address locations are allocated for each of these I/O ports. Each I/O port has two associated 8-bit memory mapped registers, the CONFIGURATION register and the output DATA register. A memory mapped address is also reserved for the input pins of each I/O port. (See the memory map for the various addresses associated with the I/O ports.) *Figure 4* shows the I/O port configurations. The DATA and CONFIGURATION registers allow for each port bit to be individually configured under software control as shown below:

CONFIGURATION Register	DATA Register	Port Set-Up
0	0	Hi-Z Input (TRI-STATE Output)
0	1	Input with Weak Pull-Up
1	0	Push-Pull Zero Output
1	1	Push-Pull One Output

PORT L is an 8-bit I/O port. All L-pins have Schmitt triggers on the inputs.

Port L supports Multi-Input Wake Up (MIWU) on all eight pins. L1 is used for the UART external clock. L2 and L3 are used for the UART transmit and receive. L4 and L5 are used for the timer input functions T2A and T2B. L6 and L7 are used for the timer input functions T3A and T3B.

Port L has the following alternate features:

- L0 MIWU
- L1 MIWU or CKX
- L2 MIWU or TDX
- L3 MIWU or RDX
- L4 MIWU or T2A
- L5 MIWU or T2B
- L6 MIWU or T3A
- L7 MIWU or T3B

Port G is an 8-bit port with 5 I/O pins (G0, G2–G5), an input pin (G6), and two dedicated output pins (G1 and G7). Pins G0 and G2–G6 all have Schmitt Triggers on their inputs. Pin G1 serves as the dedicated WDOUT WATCHDOG output, while pin G7 is either input or output depending on the oscillator mask option selected. With the crystal oscillator option selected, G7 serves as the dedicated output pin for the CKO clock output. With the single-pin R/C oscillator mask option selected, G7 serves as a general purpose input pin but is also used to bring the device out of HALT mode with a low to high transition on G7. There are two registers associated with the G Port, a data register and a configuration register. Therefore, each of the 5 I/O bits (G0, G2–G5) can be individually configured under software control.

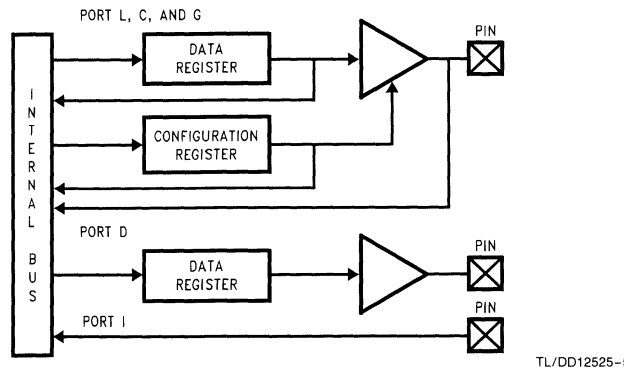


FIGURE 4. I/O Port Configurations

Pin Descriptions (Continued)

Since G6 is an input only pin and G7 is the dedicated CKO clock output pin (crystal clock option) or general purpose input (R/C clock option), the associated bits in the data and configuration registers for G6 and G7 are used for special purpose functions as outlined below. Reading the G6 and G7 data bits will return zeros.

Note that the chip will be placed in the HALT mode by writing a "1" to bit 7 of the Port G Data Register. Similarly the chip will be placed in the IDLE mode by writing a "1" to bit 6 of the Port G Data Register.

Writing a "1" to bit 6 of the Port G Configuration Register enables the MICROWIRE/PLUS to operate with the alternate phase of the SK clock. The G7 configuration bit, if set high, enables the clock start up delay after HALT when the R/C clock configuration is used.

	Config Reg.	Data Reg.
G7	CLKDLY	HALT
G6	Alternate SK	IDLE

Port G has the following alternate features:

- G0 INTR (External Interrupt Input)
- G2 T1B (Timer T1 Capture Input)
- G3 T1A (Timer T1 I/O)
- G4 SO (MICROWIRE Serial Data Output)
- G5 SK (MICROWIRE Serial Clock)
- G6 SI (MICROWIRE Serial Data Input)

Port G has the following dedicated functions:

- G1 WDOUT WATCHDOG and/or Clock Monitor dedicated output
- G7 CKO Oscillator dedicated output or general purpose input

Port C is an 8-bit I/O port. The 40-pin device does not have a full complement of Port C pins. The unavailable pins are not terminated. A read operation for these unterminated pins will return unpredictable values.

PORT I is an eight-bit Hi-Z input port. The 28-pin device does not have a full complement of Port I pins. The unavailable pins are not terminated i.e., they are floating. A read operation for these unterminated pins will return unpredictable values. The user must ensure that the software takes this into account by either masking or restricting the accesses to bit operations. The unterminated Port I pins will draw power only when addressed. The I port leakage may be higher in 28-pin devices.

Port I1–I3 are used for Comparator 1. Port I4–I6 are used for Comparator 2.

The Port I has the following alternate features.

- I1 COMP1 − IN (Comparator 1 Negative Input)
- I2 COMP1 + IN (Comparator 1 Positive Input)
- I3 COMP1OUT (Comparator 1 Output)
- I4 COMP2 − IN (Comparator 2 Negative Input)
- I5 COMP2 + IN (Comparator 2 Positive Input)
- I6 COMP2OUT (Comparator 2 Output)

Port D is a recreated 8-bit output port that is preset high when $\overline{\text{RESET}}$ goes low. D port recreation is one clock cycle behind normal port timing. The user can tie two or more D port outputs (except D2) together in order to get a higher drive.

Functional Description

The architecture of the device is modified Harvard architecture. With the Harvard architecture, the control store program memory (ROM) is separated from the data store memory (RAM). Both ROM and RAM have their own separate addressing space with separate address buses. The architecture, though based on Harvard architecture, permits transfer of data from ROM to RAM.

CPU REGISTERS

The CPU can do an 8-bit addition, subtraction, logical or shift operation in one instruction (t_C) cycle time.

There are six CPU registers:

A is the 8-bit Accumulator Register

PC is the 15-bit Program Counter Register

PU is the upper 7 bits of the program counter (PC)
PL is the lower 8 bits of the program counter (PC)

B is an 8-bit RAM address pointer, which can be optionally post auto incremented or decremented.

X is an 8-bit alternate RAM address pointer, which can be optionally post auto incremented or decremented.

SP is the 8-bit stack pointer, which points to the subroutine/interrupt stack (in RAM). The SP is initialized to RAM address 06F with reset.

S is the 8-bit Data Segment Address Register used to extend the lower half of the address range (00 to 7F) into 256 data segments of 128 bytes each.

All the CPU registers are memory mapped with the exception of the Accumulator (A) and the Program Counter (PC).

PROGRAM MEMORY

The program memory consists of 8 kbytes of OTP EPROM. These bytes may hold program instructions or constant data (data tables for the LAID instruction, jump vectors for the JID instruction, and interrupt vectors for the VIS instruction). The program memory is addressed by the 15-bit program counter (PC). All interrupts in the devices vector to program memory location 0FF Hex.

The device can be configured to inhibit external reads of the program memory. This is done by programming the Security Byte. See the SECURITY FEATURE section for more details.

SECURITY FEATURE

The program memory array has an associate Security Byte that is located outside of the program address range. This byte can be addressed only from programming mode by a programmer tool.

Security is an optional feature and can only be asserted after the memory array has been programmed and verified. A secured part will read all 00(hex) by a programmer. The part will fail Blank Check and will fail Verify operations. A Read operaiton will fill the programmer's memory with 00(hex). The Security Byte itself is always readable with value of 00(hex) if unsecure and FF(hex) if secure.

DATA MEMORY

The data memory address space includes the on-chip RAM and data registers, the I/O registers (Configuration, Data and Pin), the control registers, the MICROWIRE/PLUS SIO shift register, and the various registers, and counters associated with the timers (with the exception of the IDLE timer). Data memory is addressed directly by the instruction or indirectly by the B, X, SP pointers and S register.

Functional Description (Continued)

The data memory consists of 256 bytes of RAM. Sixteen bytes of RAM are mapped as "registers" at addresses 0F0 to 0FF Hex. These registers can be loaded immediately, and also decremented and tested with the DRSZ (decrement register and skip if zero) instruction. The memory pointer registers X, SP, B and S are memory mapped into this space at address locations 0FC to 0FF Hex respectively, with the other registers being available for general usage.

The instruction set permits any bit in memory to be set, reset or tested. All I/O and registers (except A and PC) are memory mapped; therefore, I/O bits and register bits can be directly and individually set, reset and tested. The accumulator (A) bits can also be directly and individually tested.

Data Memory Segment RAM Extension

Data memory address 0FF is used as a memory mapped location for the Data Segment Address Register (S).

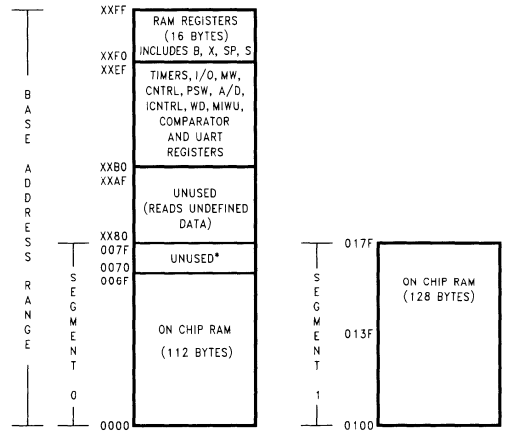
The data store memory is either addressed directly by a single byte address within the instruction, or indirectly relative to the reference of the B, X, or SP pointers (each contains a single-byte address). This single-byte address allows an addressing range of 256 locations from 00 to FF hex. The upper bit of this single-byte address divides the data store memory into two separate sections as outlined previously. With the exception of the RAM register memory from address locations 00F0 to 00FF, all RAM memory is memory mapped with the upper bit of the single-byte address being equal to zero. This allows the upper bit of the single-byte address to determine whether or not the base address range (from 0000 to 00FF) is extended. If this upper bit equals one (representing address range 0080 to 00FF), then address extension does not take place. Alternatively, if this upper bit equals zero, then the data segment extension register S is used to extend the base address range (from 0000 to 007F) from XX00 to XX7F, where XX represents the 8 bits from the S register. Thus the 128-byte data segment extensions are located from addresses 0100 to 017F for data segment 1, 0200 to 027F for data segment 2, etc., up to FF00 to FF7F for data segment 255. The base address range from 0000 to 007F represents data segment 0.

Figure 5 illustrates how the S register data memory extension is used in extending the lower half of the base address range (00 to 7F hex) into 256 data segments of 128 bytes each, with a total addressing range of 32 kbytes from XX00 to XX7F. This organization allows a total of 256 data segments of 128 bytes each with an additional upper base segment of 128 bytes. Furthermore, all addressing modes are available for all data segments. The S register must be changed under program control to move from one data segment (128 bytes) to another. However, the upper base segment (containing the 16 memory registers, I/O registers, control registers, etc.) is always available regardless of the contents of the S register, since the upper base segment (address range 0080 to 00FF) is independent of data segment extension.

The instructions that utilize the stack pointer (SP) always reference the stack as part of the base segment (Segment 0), regardless of the contents of the S register. The S register is not changed by these instructions. Consequently, the stack (used with subroutine linkage and interrupts) is always located in the base segment. The stack pointer will be initialized to point at data memory location 006F as a result of reset.

The 128 bytes of RAM contained in the base segment are split between the lower and upper base segments. The first 116 bytes of RAM are resident from address 0000 to 006F in the lower base segment, while the remaining 16 bytes of RAM represent the 16 data memory registers located at addresses 00F0 to 00FF of the upper base segment. No RAM is located at the upper sixteen addresses (0070 to 007F) of the lower base segment.

Additional RAM beyond these initial 128 bytes, however, will always be memory mapped in groups of 128 bytes (or less) at the data segment address extensions (XX00 to XX7F) of the lower base segment. The additional 128 bytes of RAM are memory mapped at address locations 0100 to 017F hex.



TL/DD12525-6

*Reads as all ones.

FIGURE 5. RAM Organization

Reset

The $\overline{\text{RESET}}$ input when pulled low initializes the microcontroller. Initialization will occur whenever the $\overline{\text{RESET}}$ input is pulled low. Upon initialization, the data and configuration registers for ports L, G and C are cleared, resulting in these Ports being initialized to the TRI-STATE mode. Pin G1 of the G Port is an exception (as noted below) since pin G1 is dedicated as the WATCHDOG and/or Clock Monitor error output pin. Port D is set high. The PC, PSW, ICNTRL, CNTRL, T2CNTRL and T3CNTRL control registers are cleared. The UART registers PSR, ENU (except that TBMT bit is set), ENUR and ENUI are cleared. The Comparator Select Register is cleared. The S register is initialized to zero. The Multi-Input Wake Up registers WKEN, WKEDG and WKPND are cleared. The stack pointer, SP, is initialized to 6F Hex.

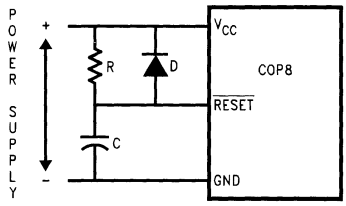
The device comes out of reset with both the WATCHDOG logic and the Clock Monitor detector armed, with the WATCHDOG service window bits set and the Clock Monitor bit set. The WATCHDOG and Clock Monitor circuits are inhibited during reset. The WATCHDOG service window bits being initialized high default to the maximum WATCHDOG service window of 64k t_C clock cycles. The Clock Monitor bit being initialized high will cause a Clock Monitor error following reset if the clock has not reached the minimum specified frequency at the termination of reset. A Clock Monitor error

Reset (Continued)

will cause an active low error output on pin G1. This error output will continue until 16 t_c–32 t_c clock cycles following the clock frequency reaching the minimum specified value, at which time the G1 output will enter the TRI-STATE mode.

The external RC network shown in *Figure 6* should be used to ensure that the $\overline{\text{RESET}}$ pin is held low until the power supply to the chip stabilizes.

Note: Continual state of reset will cause the device to draw excessive current.



TL/DD/12525–7

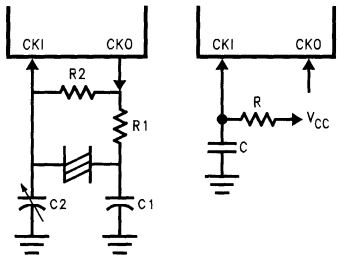
RC > 5 × Power Supply Rise Time

FIGURE 6. Recommended Reset Circuit

Oscillator Circuits

The chip can be driven by a clock input on the CKI input pin which can be between DC and 10 MHz. The CKO output clock is on pin G7 (crystal configuration). The CKI input frequency is divided down by 10 to produce the instruction cycle clock ($1/t_c$).

Figure 7 shows the Crystal and R/C diagrams.



TL/DD12525–8

FIGURE 7. Crystal and R/C Oscillator Diagrams

CRYSTAL OSCILLATOR

CKI and CKO can be connected to make a closed loop crystal (or resonator) controlled oscillator.

Table I shows the component values required for various standard crystal values.

TABLE I. Crystal Oscillator Configuration, T_A = 25°C

R1 (kΩ)	R2 (MΩ)	C1 (pF)	C2 (pF)	CKI Freq (MHz)	Conditions
0	1	30	30–36	10	V_{CC} = 5V
0	1	30	30–36	4	V_{CC} = 5V
0	1	200	100–150	0.455	V_{CC} = 5V

R/C OSCILLATOR

By selecting CKI as a single pin oscillator input, a single pin R/C oscillator circuit can be connected to it. CKO is available as a general purpose input, and/or HALT restart pin.

Table II shows the variation in the oscillator frequencies as functions of the component (R and C) values.

TABLE II. R/C Oscillator Configuration, T_A = 25°C

R (kΩ)	C (pF)	CKI Freq (MHz)	Instr. Cycle (μs)	Conditions
3.3	82	2.2–2.7	3.7–4.6	V_{CC} = 5V
5.6	100	1.1–1.3	7.4–9.0	V_{CC} = 5V
6.8	100	0.9–1.1	8.8–10.8	V_{CC} = 5V

Note: 3k ≤ R ≤ 200k
50 pF ≤ C ≤ 200 pF

Control Registers

CNTRL Register (Address X'00EE)

The Timer1 (T1) and MICROWIRE/PLUS control register contains the following bits:

SL1 & SL0	Select the MICROWIRE/PLUS clock divide by (00 = 2, 01 = 4, 1x = 8)
IEDG	External interrupt edge polarity select (0 = Rising edge, 1 = Falling edge)
MSEL	Selects G5 and G4 as MICROWIRE/PLUS signals SK and SO respectively
T1C0	Timer T1 Start/Stop control in timer modes 1 and 2 Timer T1 Underflow Interrupt Pending Flag in timer mode 3
T1C1	Timer T1 mode control bit
T1C2	Timer T1 mode control bit
T1C3	Timer T1 mode control bit

T1C3	T1C2	T1C1	T1C0	MSEL	IEDG	SL1	SL0

Bit 7 Bit 0

Control Registers (Continued)

PSW Register (Address X'00EF)

The PSW register contains the following select bits:

- GIE Global interrupt enable (enables interrupts)
- EXEN Enable external interrupt
- BUSY MICROWIRE/PLUS busy shifting flag
- EXPND External interrupt pending
- T1ENA Timer T1 Interrupt Enable for Timer Underflow or T1A Input capture edge
- T1PNDA Timer T1 Interrupt Pending Flag (Autoreload RA in mode 1, T1 Underflow in Mode 2, T1A capture edge in mode 3)
- C Carry Flag
- HC Half Carry Flag

HC	C	T1PNDA	T1ENA	EXPND	BUSY	EXEN	GIE

Bit 7 Bit 0

The Half-Carry bit is also affected by all the instructions that affect the Carry flag. The SC (Set Carry) and RC (Reset Carry) instructions will respectively set or clear both the carry flags. In addition to the SC and RC instructions, ADC, SUBC, RRC and RLC instructions affect the carry and Half Carry flags.

ICNTRL Register (Address X'00E8)

The ICNTRL register contains the following bits:

- T1ENB Timer T1 Interrupt Enable for T1B Input capture edge
- T1PNDB Timer T1 Interrupt Pending Flag for T1B capture edge
- WEN Enable MICROWIRE/PLUS interrupt
- WPND MICROWIRE/PLUS interrupt pending
- T0EN Timer T0 Interrupt Enable (Bit 12 toggle)
- T0PND Timer T0 Interrupt pending
- LPEN L Port Interrupt Enable (Multi-Input Wake Up/ Interrupt)

  Bit 7 could be used as a flag

Unused	LPEN	T0PND	T0EN	WPND	WEN	T1PNDB	T1ENB

Bit 7 Bit 0

T2CNTRL Register (Address X'00C6)

The T2CNTRL register contains the following bits:

- T2ENB Timer T2 Interrupt Enable for T2B Input capture edge
- T2PNDB Timer T2 Interrupt Pending Flag for T2B capture edge
- T2ENA Timer T2 Interrupt Enable for Timer Underflow or T2A Input capture edge
- T2PNDA Timer T2 Interrupt Pending Flag (Autoreload RA in mode 1, T2 Underflow in mode 2, T2A capture edge in mode 3)
- T2C0 Timer T2 Start/Stop control in timer modes 1 and 2 Timer T2 Underflow Interrupt Pending Flag in timer mode 3
- T2C1 Timer T2 mode control bit
- T2C2 Timer T2 mode control bit
- T2C3 Timer T2 mode control bit

T2C3	T2C2	T2C1	T2C0	T2PNDA	T2ENA	T2PNDB	T2ENB

Bit 7 Bit 0

T3CNTRL Register (Address X'00B6)

The T3CNTRL register contains the following bits:

- T3ENB Timer T3 Interrupt Enable for T3B
- T3PNDB Timer T3 Interrupt Pending Flag for T3B pin (T3B capture edge)
- T3ENA Timer T3 Interrupt Enable for Timer Underflow or T3A pin
- T3PNDA Timer T3 Interrupt Pending Flag (Autoload RA in mode 1, T3 Underflow in mode 2, T3a capture edge in mode 3)
- T3C0 Timer T3 Start/Stop control in timer modes 1 and 2

  Timer T3 Underflow Interrupt Pending Flag in timer mode 3
- T3C1 Timer T3 mode control bit
- T3C2 Timer T3 mode control bit
- T3C3 Timer T3 mode control bit

T3C3	T3C2	T3C1	T3C0	T3PNDA	T3ENA	T3PNDB	T3ENB

Bit 7 Bit 0

Timers

The device contains a very versatile set of timers (T0, T1, T2, T3). All timers and associated autoreload/capture registers power up containing random data.

TIMER T0 (IDLE TIMER)

The devices support applications that require maintaining real time and low power with the IDLE mode. This IDLE mode support is furnished by the IDLE timer T0, which is a 16-bit timer. The Timer T0 runs continuously at the fixed rate of the instruction cycle clock, t_C. The user cannot read or write to the IDLE Timer T0, which is a count down timer.

The Timer T0 supports the following functions:

Exit out of the Idle Mode (See Idle Mode description)
WATCHDOG logic (See WATCHDOG description)
Start up delay out of the HALT mode

The IDLE Timer T0 can generate an interrupt when the thirteenth bit toggles. This toggle is latched into the T0PND pending flag, and will occur every 4 ms at the maximum clock frequency (t_C = 1 μs). A control flag T0EN allows the interrupt from the thirteenth bit of Timer T0 to be enabled or disabled. Setting T0EN will enable the interrupt, while resetting it will disable the interrupt.

Timers (Continued)

TIMER T1, TIMER T2 AND TIMER T3

The devices have a set of three powerful timer/counter blocks, T1, T2 and T3. The associated features and functioning of a timer block are described by referring to the timer block Tx. Since the three timer blocks, T1, T2 and T3 are identical, all comments are equally applicable to any of the three timer blocks.

Each timer block consists of a 16-bit timer, Tx, and two supporting 16-bit autoreload/capture registers, RxA and RxB. Each timer block has two pins associated with it, TxA and TxB. The pin TxA supports I/O required by the timer block, while the pin TxB is an input to the timer block. The powerful and flexible timer block allows the device to easily perform all timer functions with minimal software overhead. The timer block has three operating modes: Processor Independent PWM mode, External Event Counter mode, and Input Capture mode.

The control bits TxC3, TxC2, and TxC1 allow selection of the different modes of operation.

Mode 1. Processor Independent PWM Mode

As the name suggests, this mode allows the device to generate a PWM signal with very minimal user intervention. The user only has to define the parameters of the PWM signal (ON time and OFF time). Once begun, the timer block will continuously generate the PWM signal completely independent of the microcontroller. The user software services the timer block only when the PWM parameters require updating.

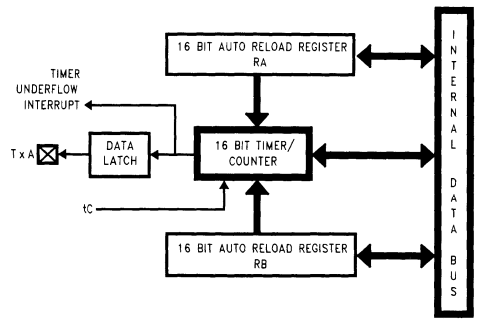
In this mode the timer Tx counts down at a fixed rate of t_C. Upon every underflow the timer is alternately reloaded with the contents of supporting registers, RxA and RxB. The very first underflow of the timer causes the timer to reload from the register RxA. Subsequent underflows cause the timer to be reloaded from the registers alternately beginning with the register RxB.

The Tx Timer control bits, TxC3, TxC2 and TxC1 set up the timer for PWM mode operation.

Figure 8 shows a block diagram of the timer in PWM mode. The underflows can be programmed to toggle the TxA output pin. The underflows can also be programmed to generate interrupts.

Underflows from the timer are alternately latched into two pending flags, TxPNDA and TxPNDB. The user must reset these pending flags under software control. Two control enable flags, TxENA and TxENB, allow the interrupts from the timer underflow to be enabled or disabled. Setting the timer enable flag TxENA will cause an interrupt when a timer underflow causes the RxA register to be reloaded into the timer. Setting the timer enable flag TxENB will cause an interrupt when a timer underflow causes the RxB register to be reloaded into the timer. Resetting the timer enable flags will disable the associated interrupts.

Either or both of the timer underflow interrupts may be enabled. This gives the user the flexibility of interrupting once per PWM period on either the rising or falling edge of the PWM output. Alternatively, the user may choose to interrupt on both edges of the PWM output.



TL/DD12525–9

FIGURE 8. Timer in PWM Mode

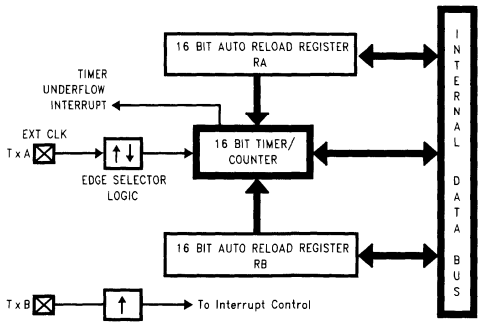
Mode 2. External Event Counter Mode

This mode is quite similar to the processor independent PWM mode described above. The main difference is that the timer, Tx, is clocked by the input signal from the TxA pin. The Tx timer control bits, TxC3, TxC2 and TxC1 allow the timer to be clocked either on a positive or negative edge from the TxA pin. Underflows from the timer are latched into the TxPNDA pending flag. Setting the TxENA control flag will cause an interrupt when the timer underflows.

In this mode the input pin TxB can be used as an independent positive edge sensitive interrupt input if the TxENB control flag is set. The occurrence of a positive edge on the TxB input pin is latched into the TxPNDB flag.

Figure 9 shows a block diagram of the timer in External Event Counter mode.

Note: The PWM output is not available in this mode since the TxA pin is being used as the counter input clock.



TL/DD12525–10

FIGURE 9. Timer in External Event Counter Mode

Mode 3. Input Capture Mode

The device can precisely measure external frequencies or time external events by placing the timer block, Tx, in the input capture mode.

In this mode, the timer Tx is constantly running at the fixed t_C rate. The two registers, RxA and RxB, act as capture registers. Each register acts in conjunction with a pin. The register RxA acts in conjunction with the TxA pin and the register RxB acts in conjunction with the TxB pin.

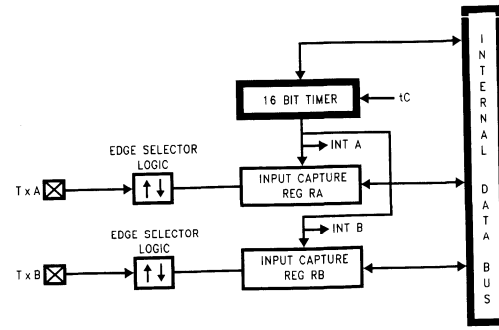
Timers (Continued)

The timer value gets copied over into the register when a trigger event occurs on its corresponding pin. Control bits, TxC3, TxC2 and TxC1, allow the trigger events to be specified either as a positive or a negative edge. The trigger condition for each input pin can be specified independently.

The trigger conditions can also be programmed to generate interrupts. The occurrence of the specified trigger condition on the TxA and TxB pins will be respectively latched into the pending flags, TxPNDA and TxPNDB. The control flag TxENA allows the interrupt on TxA to be either enabled or disabled. Setting the TxENA flag enables interrupts to be generated when the selected trigger condition occurs on the TxA pin. Similarly, the flag TxENB controls the interrupts from the TxB pin.

Underflows from the timer can also be programmed to generate interrupts. Underflows are latched into the timer TxC0 pending flag (the TxC0 control bit serves as the timer underflow interrupt pending flag in the Input Capture mode). Consequently, the TxC0 control bit should be reset when entering the Input Capture mode. The timer underflow interrupt is enabled with the TxENA control flag. When a TxA interrupt occurs in the Input Capture mode, the user must check both the TxPNDA and TxC0 pending flags in order to determine whether a TxA input capture or a timer underflow (or both) caused the interrupt.

Figure 10 shows a block diagram of the timer in Input Capture mode.



TL/DD12525–11

FIGURE 10. Timer in Input Capture Mode

TIMER CONTROL FLAGS

The timers T1, T2 and T3 have indentical control structures. The control bits and their functions are summarized below.

TxC0	Timer Start/Stop control in Modes 1 and 2 (Processor Independent PWM and External Event Counter), where 1 = Start, 0 = Stop Timer Underflow Interrupt Pending Flag in Mode 3 (Input Capture)
TxPNDA	Timer Interrupt Pending Flag
TxPNDB	Timer Interrupt Pending Flag
TxENA	Timer Interrupt Enable Flag
TxENB	Timer Interrupt Enable Flag 1 = Timer Interrupt Enabled 0 = Timer Interrupt Disabled
TxC3	Timer mode control
TxC2	Timer mode control
TxC1	Timer mode control

Timers (Continued)

The timer mode control bits (TxC3, TxC2 and TxC1) are detailed below:

TxC3	TxC2	TxC1	Timer Mode	Interrupt A Source	Interrupt B Source	Timer Counts On
0	0	0	MODE 2 (External Event Counter)	Timer Underflow	Pos. TxB Edge	TxA Pos. Edge
0	0	1	MODE 2 (External Event Counter)	Timer Underflow	Pos. TxB Edge	TxA Neg. Edge
1	0	1	MODE 1 (PWM) TxA Toggle	Autoreload RA	Autoreload RB	t_C
1	0	0	MODE 1 (PWM) No TxA Toggle	Autoreload RA	Autoreload RB	t_C
0	1	0	MODE 3 (Capture) Captures: TxA Pos. Edge TxB Pos. Edge	Pos. TxA Edge or Timer Underflow	Pos. TxB Edge	t_C
1	1	0	MODE 3 (Capture) Captures: TxA Pos. Edge TxB Neg. Edge	Pos. TxA Edge or Timer Underflow	Neg. TxB Edge	t_C
0	1	1	MODE 3 (Capture) Captures: TxA Neg. Edge TxB Pos. Edge	Neg. TxA Edge or Timer Underflow	Pos. TxB Edge	t_C
1	1	1	MODE 3 (Capture) Captures: TxA Neg. Edge TxB Neg. Edge	Neg. TxA Edge or Timer Underflow	Neg. TxB Edge	t_C

Power Save Modes

The devices offer the user two power save modes of operation: HALT and IDLE. In the HALT mode, all microcontroller activities are stopped. In the IDLE mode, the on-board oscillator circuitry the WATCHDOG logic, the Clock Monitor and timer T0 are active but all other microcontroller activities are stopped. In either mode, all on-board RAM, registers, I/O states, and timers (with the exception of T0) are unaltered.

HALT MODE

The devices can be placed in the HALT mode by writing a "1" to the HALT flag (G7 data bit). All microcontroller activities, including the clock and timers, are stopped. The WATCHDOG logic on the device is disabled during the HALT mode. However, the clock monitor circuitry if enabled remains active and will cause the WATCHDOG output pin (WDOUT) to go low. If the HALT mode is used and the user does not want to activate the WDOUT pin, the Clock Monitor should be disabled after the device comes out of reset (resetting the Clock Monitor control bit with the first write to the WDSVR register). In the HALT mode, the power requirements of the device are minimal and the applied voltage (V_{CC}) may be decreased to V_r (V_r = 2.0V) without altering the state of the machine.

The devices support three different ways of exiting the HALT mode. The first method of exiting the HALT mode is with the Multi-Input Wake Up feature on the L port. The second method is with a low to high transition on the CKO (G7) pin. This method precludes the use of the crystal clock configuration (since CKO becomes a dedicated output), and so may be used with an RC clock configuration. The third method of exiting the HALT mode is by pulling the $\overline{\text{RESET}}$ pin low.

Since a crystal or ceramic resonator may be selected as the oscillator, the Wake Up signal is not allowed to start the chip running immediately since crystal oscillators and ceramic resonators have a delayed start up time to reach full amplitude and frequency stability. The IDLE timer is used to generate a fixed delay to ensure that the oscillator has indeed stabilized before allowing instruction execution. In this case, upon detecting a valid Wake Up signal, only the oscillator circuitry is enabled. The IDLE timer is loaded with a value of 256 and is clocked with the t_C instruction cycle clock. The t_C clock is derived by dividing the oscillator clock down by a factor of 10. The Schmitt trigger following the CKI inverter on the chip ensures that the IDLE timer is clocked only when the oscillator has a sufficiently large amplitude to meet the Schmitt trigger specifications. This Schmitt trigger is not part of the oscillator closed loop. The startup timeout from the IDLE timer enables the clock signals to be routed to the rest of the chip.

If an RC clock option is being used, the fixed delay is introduced optionally. A control bit, CLKDLY, mapped as configuration bit G7, controls whether the delay is to be introduced or not. The delay is included if CLKDLY is set, and excluded if CLKDLY is reset. The CLKDLY bit is cleared on reset.

Power Save Modes (Continued)

The WATCHDOG detector circuit is inhibited during the HALT mode. However, the clock monitor circuit if enabled remains active during HALT mode in order to ensure a clock monitor error if the device inadvertently enters the HALT mode as a result of a runaway program or power glitch.

IDLE MODE

The device is placed in the IDLE mode by writing a "1" to the IDLE flag (G6 data bit). In this mode, all activities, except the associated on-board oscillator circuitry, the WATCHDOG logic, the clock monitor and the IDLE Timer T0, are stopped. The power supply requirements of the micro-controller in this mode of operation are typically around 30% of normal power requirement of the microcontroller.

As with the HALT mode, the device can be returned to normal operation with a reset, or with a Multi-Input Wake Up from the L Port. Alternately, the microcontroller resumes normal operation from the IDLE mode when the thirteenth bit (representing 4.096 ms at internal clock frequency of 1 MHz, $t_c = 1\ \mu s$) of the IDLE Timer toggles.

This toggle condition of the thirteenth bit of the IDLE Timer T0 is latched into the T0PND pending flag.

The user has the option of being interrupted with a transition on the thirteenth bit of the IDLE Timer T0. The interrupt can be enabled or disabled via the T0EN control bit. Setting the T0EN flag enables the interrupt and vice versa.

The user can enter the IDLE mode with the Timer T0 interrupt enabled. In this case, when the T0PND bit gets set, the device will first execute the Timer T0 interrupt service routine and then return to the instruction following the "Enter Idle Mode" instruction.

Alternatively, the user can enter the IDLE mode with the IDLE Timer T0 interrupt disabled. In this case, the device will resume normal operation with the instruction immediately following the "Enter IDLE Mode" instruction.

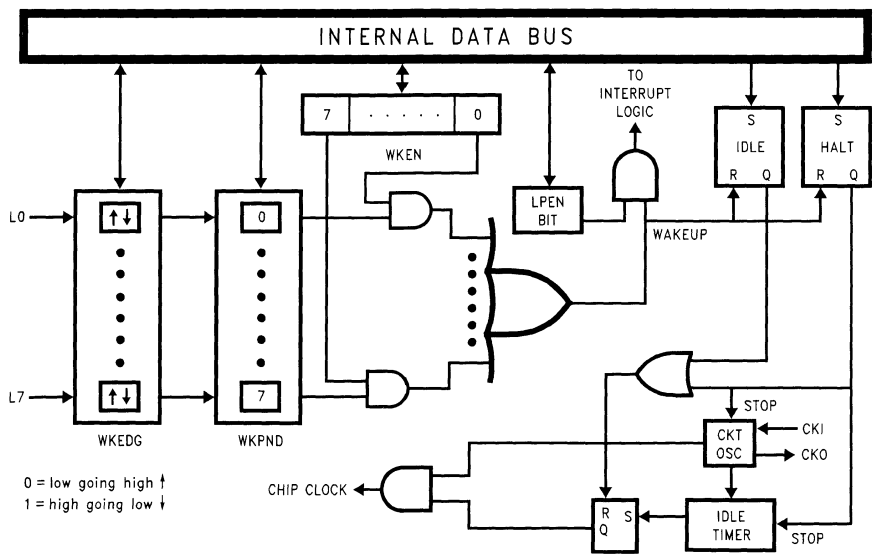
Note: It is necessary to program two NOP instructions following both the set HALT mode and set IDLE mode instructions. These NOP instructions are necessary to allow clock resynchronization following the HALT or IDLE modes.

Due to the on-board 8k EPROM with port recreation logic, the HALT/IDLE current is much higher compared to the equivalent masked port.

Multi-Input Wake Up

The Multi-Input Wake Up feature is ued to return (Wake Up) the device from either the HALT or IDLE modes. Alternately Multi-Input Wake Up/Interrupt feature may also be used to generate up to 8 edge selectable external interrupts.

Figure 11 shows the Multi-Input Wake Up logic. The Multi-Input Wake Up feature utilizes the L Port. The user selects which particular L port bit (or combination of L Port bits) will cause the device to exit the HALT or IDLE modes. The selection is done through the Reg: WKEN. The Reg: WKEN



TL/DD12525–12

FIGURE 11. Multi-Input Wake Up Logic

Multi-Input Wake Up (Continued)

is an 8-bit read/write register, which contains a control bit for every L port bit. Setting a particular WKEN bit enables a Wake Up from the associated L port pin.

The user can select whether the trigger condition on the selected L Port pin is going to be either a positive edge (low to high transition) or a negative edge (high to low transition). This selection is made via the Reg: WKEDG, which is an 8-bit control register with a bit assigned to each L Port pin. Setting the control bit will select the trigger condition to be a negative edge on that particular L Port pin. Resetting the bit selects the trigger condition to be a positive edge. Changing an edge select entails several steps in order to avoid a pseudo Wake Up condition as a result of the edge change. First, the associated WKEN bit should be reset, followed by the edge select change in WKEDG. Next, the associated WKPND bit should be cleared, followed by the associated WKEN bit being re-enabled.

An example may serve to clarify this procedure. Suppose we wish to change the edge select from positive (low going high) to negative (high going low) for L Port bit 5, where bit 5 has previously been enabled for an input interrupt. The program would be as follows:

```
RBIT  5, WKEN
SBIT  5, WKEDG
RBIT  5, WKPND
SBIT  5, WKEN
```

If the L port bits have been used as outputs and then changed to inputs with Multi-Input Wake Up/Interrupt, a safety procedure should also be followed to avoid inherited pseudo wakeup conditions. After the selected L port bits have been changed from output to input but before the associated WKEN bits are enabled, the associated edge select bits in WKEDG should be set or reset for the desired edge selects, followed by the associated WKPND bits being cleared.

This same procedure should be used following reset, since the L port inputs are left floating as a result of reset.

The occurrence of the selected trigger condition for Multi-Input Wake Up is latched into a pending register called WKPND. The respective bits of the WKPND register will be set on the occurrence of the selected trigger edge on the corresponding Port L pin. The user has the responsibility of clearing these pending flags. Since WKPND is a pending register for the occurrence of selected Wake Up conditions, the device will not enter the HALT mode if any Wake Up bit is both enabled and pending. Consequently, the user has the responsibility of clearing the pending flags before attempting to enter the HALT mode.

WKEN, WKPND and WKEDG are all read/write registers, and are cleared at reset.

PORT L INTERRUPTS

Port L provides the user with an additional eight fully selectable, edge sensitive interrupts which are all vectored into the same service subroutine.

The interrupt from Port L shares logic with the wake up circuitry. The register WKEN allows interrupts from Port L to be individually enabled or disabled. The register WKEDG specifies the trigger condition to be either a positive or a negative edge. Finally, the register WKPND latches in the pending trigger conditions.

The GIE (Global Interrupt Enable) bit enables the interrupt function.

A control flag, LPEN, functions as a global interrupt enable for Port L interrupts. Setting the LPEN flag will enable interrupts and vice versa. A separate global pending flag is not needed since the register WKPND is adequate.

Since Port L is also used for waking the device out of the HALT or IDLE modes, the user can elect to exit the HALT or IDLE modes either with or without the interrupt enabled. If he elects to disable the interrupt, then the device will restart execution from the instruction immediately following the instruction that placed the microcontroller in the HALT or IDLE modes. In the other case, the device will first execute the interrupt service routine and then revert to normal operation.

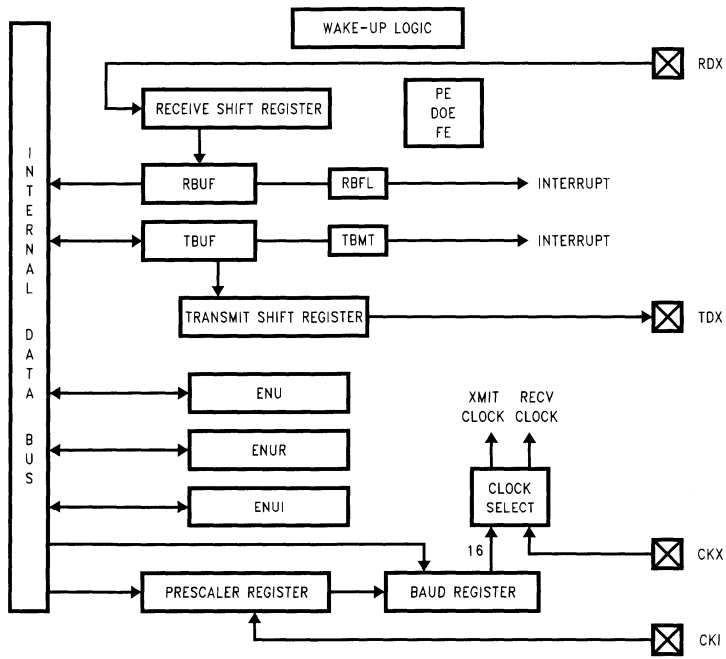
The Wake Up signal will not start the chip running immediately since crystal oscillators or ceramic resonators have a finite start up time. The IDLE Timer (T0) generates a fixed delay to ensure that the oscillator has indeed stabilized before allowing the device to execute instructions. In this case, upon detecting a valid Wake Up signal, only the oscillator circuitry and the IDLE Timer T0 are enabled. The IDLE Timer is loaded with a value of 256 and is clocked from the t_C instruction cycle clock. The t_C clock is derived by dividing down the oscillator clock by a factor of 10. A Schmitt trigger following the CKI on-chip inverter ensures that the IDLE timer is clocked only when the oscillator has a sufficiently large amplitude to meet the Schmitt trigger specifications. This Schmitt trigger is not part of the oscillator closed loop. The startup timeout from the IDLE timer enables the clock signals to be routed to the rest of the chip.

If the RC clock option is used, the fixed delay is under software control. A control flag, CLKDLY, in the G7 configuration bit allows the clock start up delay to be optionally inserted. Setting CLKDLY flag high will cause clock start up delay to be inserted and resetting it will exclude the clock start up delay. The CLKDLY flag is cleared during reset, so the clock start up delay is not present following reset with the RC clock options.

UART

The device contains a full-duplex software programmable UART. The UART *(Figure 12)* consists of a transmit shift register, a receiver shift register and seven addressable registers, as follows: a transmit buffer register (TBUF), a receiver buffer register (RBUF), a UART control and status register (ENU), a UART receive control and status register (ENUR), a UART interrupt and clock source register (ENUI), a prescaler select register (PSR) and baud (BAUD) register. The ENU register contains flags for transmit and receive functions; this register also determines the length of the data frame (7, 8 or 9 bits), the value of the ninth bit in transmission, and parity selection bits. The ENUR register flags framing, data overrun and parity errors while the UART is receiving.

Other functions of the ENUR register include saving the ninth bit received in the data frame, enabling or disabling the UART's attention mode of operation and providing additional receiver/transmitter status information via RCVG and XMTG bits. The determination of an internal or external clock source is done by the ENUI register, as well as selecting the number of stop bits and enabling or disabling transmit and receive interrupts. A control flag in this register can also select the UART mode of operation: asynchronous or synchronous.



TL/DD12525–13

FIGURE 12. UART Block Diagram

UART (Continued)

UART CONTROL AND STATUS REGISTERS

The operation of the UART is programmed through three registers: ENU, ENUR and ENUI. The function of the individual bits in these registers is as follows:

ENU-UART Control and Status Register (Address at 0BA)

PEN	PSEL1	XBIT9/ PSEL0	CHL1	CHL0	ERR	RBFL	TBMT
0RW	0RW	0RW	0RW	0RW	0R	0R	1R

Bit 7 Bit 0

ENUR-UART Receive Control and Status Register (Address at 0BB)

DOE	FE	PE	SPARE	RBIT9	ATTN	XMTG	RCVG
0RD	0RD	0RD	0RW*	0R	0RW	0R	0R

Bit7 Bit0

ENUI-UART Interrupt and Clock Source Register (Address at 0BC)

STP2	STP78	ETDX	SSEL	XRCLK	XTCLK	ERI	ETI
0RW	0RW	0RW	0RW	0RW	0RW	0RW	0RW

Bit7 Bit0

*Bit is not used.

0 Bit is cleared on reset.

1 Bit is set to one on reset.

R Bit is read-only; it cannot be written by software.

RW Bit is read/write.

D Bit is cleared on read; when read by software as a one, it is cleared automatically. Writing to the bit does not affect its state.

DESCRIPTION OF UART REGISTER BITS

ENU—UART CONTROL AND STATUS REGISTER

TBMT: This bit is set when the UART transfers a byte of data from the TBUF register into the TSFT register for transmission. It is automatically reset when software writes into the TBUF register.

RBFL: This bit is set when the UART has received a complete character and has copied it into the RBUF register. It is automatically reset when software reads the character from RBUF.

ERR: This bit is a global UART error flag which gets set if any or a combination of the errors (DOE, FE, PE) occur.

CHL1, CHL0: These bits select the character frame format. Parity is not included and is generated/verified by hardware.

CHL1 = 0, CHL0 = 0 The frame contains eight data bits.
CHL1 = 0, CHL0 = 1 The frame contains seven data bits.
CHL1 = 1, CHL0 = 0 The frame contains nine data bits.
CHL1 = 1, CHL0 = 1 Loopback Mode selected. Transmitter output internally looped back to receiver input. Nine bit framing format is used.

XBIT9/PSEL0: Programs the ninth bit for transmission when the UART is operating with nine data bits per frame. For seven or eight data bits per frame, this bit in conjunction with PSEL1 selects parity.

PSEL1, PSEL0: Parity select bits.

PSEL1 = 0, PSEL0 = 0 Odd Parity (if Parity enabled)
PSEL1 = 0, PSEL0 = 1 Odd Parity (if Parity enabled)
PSEL1 = 1, PSEL0 = 0 Mark(1) (if Parity enabled)
PSEL1 = 1, PSEL1 = 1 Space(0) (if Parity enabled)

PEN: This bit enables/disables Parity (7- and 8-bit modes only).

PEN = 0 Parity disabled.
PEN = 1 Parity enabled.

ENUR—UART RECEIVE CONTROL AND STATUS REGISTER

RCVG: This bit is set high whenever a framing error occurs and goes low when RDX goes high.

XMTG: This bit is set to indicate that the UART is transmitting. It gets reset at the end of the last frame (end of last Stop bit).

ATTN: ATTENTION Mode is enabled while this bit is set. This bit is cleared automatically on receiving a character with data bit nine set.

RBIT9: Contains the ninth data bit received when the UART is operating with nine data bits per frame.

SPARE: Reserved for future use.

PE: Flags a Parity Error.

PE = 0 Indicates no Parity Error has been detected since the last time the ENUR register was read.
PE = 1 Indicates the occurrence of a Parity Error.

FE: Flags a Framing Error.

FE = 0 Indicates no Framing Error has been detected since the last time the ENUR register was read.
FE = 1 Indicates the occurrence of a Framing Error.

DOE: Flags a Data Overrun Error.

DOE = 0 Indicates no Data Overrun Error has been detected since the last time the ENUR register was read.
DOE = 1 Indicates the occurrence of a Data Overrun Error.

ENUI—UART INTERRUPT AND CLOCK SOURCE REGISTER

ETI: This bit enables/disables interrupt from the transmitter section.

ETI = 0 Interrupt from the transmitter is disabled.
ETI = 1 Interrupt from the transmitter is enabled.

ERI: This bit enables/disables interrupt from the receiver section.

ERI = 0 Interrupt from the receiver is disabled.
ERI = 1 Interrupt from the receiver is enabled.

XTCLK: This bit selects the clock source for the transmitter section.

XTCLK = 0 The clock source is selected through the PSR and BAUD registers.
XTCLK = 1 Signal on CKX (L1) pin is used as the clock.

XRCLK: This bit selects the clock source for the receiver section.

XRCLK = 0 The clock source is selected through the PSR and BAUD registers.
XRCLK = 1 Signal on CKX (L1) pin is used as the clock.

SSEL: UART mode select.

SSEL = 0 Asynchronous Mode.
SSEL = 1 Synchronous Mode.

UART (Continued)

ETDX: TDX (UART Transmit Pin) is the alternate function assigned to Port L pin L2; it is selected by setting ETDX bit. To simulate line break generation, software should reset ETDX bit and output logic zero to TDX pin through Port L data and configuration registers.

STP78: This bit is set to program the last Stop bit to be 7/8th of a bit in length.

STP2: This bit programs the number of Stop bits to be transmitted.

STP2 = 0 One Stop bit transmitted.
STP2 = 1 Two Stop bits transmitted.

Associated I/O Pins

Data is transmitted on the TDX pin and received on the RDX pin. TDX is the alternate function assigned to Port L pin L2; it is selected by setting ETDX (in the ENUI register) to one. RDX is an inherent function of Port L pin L3, requiring no setup.

The baud rate clock for the UART can be generated on-chip, or can be taken from an external source. Port L pin L1 (CKX) is the external clock I/O pin. The CKX pin can be either an input or an output, as determined by Port L Configuration and Data registers (Bit 1). As an input, it accepts a clock signal which may be selected to drive the transmitter and/or receiver. As an output, it presents the internal Baud Rate Generator output.

UART Operation

The UART has two modes of operation: asynchronous mode and synchronous mode.

ASYNCHRONOUS MODE

This mode is selected by resetting the SSEL (in the ENUI register) bit to zero. The input frequency to the UART is 16 times the baud rate.

The TSFT and TBUF registers double-buffer data for transmission. While TSFT is shifting out the current character on the TDX pin, the TBUF register may be loaded by software with the next byte to be transmitted. When TSFT finishes transmitting the current character the contents of TBUF are transferred to the TSFT register and the Transmit Buffer Empty Flag (TBMT in the ENU register) is set. The TBMT flag is automatically reset by the UART when software loads a new character into the TBUF register. There is also the XMTG bit which is set to indicate that the UART is transmitting. This bit gets reset at the end of the last frame (end of last Stop bit). TBUF is a read/write register.

The RSFT and RBUF registers double-buffer data being received. The UART receiver continually monitors the signal on the RDX pin for a low level to detect the beginning of a Start bit. Upon sensing this low level, it waits for half a bit time and samples again. If the RDX pin is still low, the receiver considers this to be a valid Start bit, and the remaining bits in the character frame are each sampled a single time, at the mid-bit position. Serial data input on the RDX pin is shifted into the RSFT register. Upon receiving the complete character, the contents of the RSFT register are copied into the RBUF register and the Received Buffer Full Flag (RBFL) is set. RBFL is automatically reset when software reads the character from the RBUF register. RBUF is a read only register. There is also the RCVG bit which is set high when a framing error occurs and goes low once RDX goes high. TBMT, XMTG, RBFL and RCVG are read only bits.

SYNCHRONOUS MODE

In this mode data is transferred synchronously with the clock. Data is transmitted on the rising edge and received on the falling edge of the synchronous clock.

This mode is selected by setting SSEL bit in the ENUI register. The input frequency to the UART is the same as the baud rate.

When an external clock input is selected at the CKX pin, data transmit and receive are performed synchronously with this clock through TDX/RDX pins.

If data transmit and receive are selected with the CKX pin as clock output, the device generates the synchronous clock output at the CKX pin. The internal baud rate generator is used to produce the synchronous clock. Data transmit and receive are performed synchronously with this clock.

FRAMING FORMATS

The UART supports several serial framing formats *(Figure 13)*. The format is selected using control bits in the ENU, ENUR and ENUI registers.

The first format (1, 1a, 1b, 1c) for data transmission (CHL0 = 1, CHL1 = 0) consists of Start bit, seven Data bits (excluding parity) and 7/8, one or two Stop bits. In applications using parity, the parity bit is generated and verified by hardware.

The second format (CHL0 = 0, CHL1 = 0) consists of one Start bit, eight Data bits (excluding parity) and 7/8, one or two Stop bits. Parity bit is generated and verified by hardware.

The third format for transmission (CHL0 = 0, CHL1 = 1) consists of one Start bit, nine Data bits and 7/8, one or two Stop bits. This format also supports the UART "ATTENTION" feature. When operating in this format, all eight bits of TBUF and RBUF are used for data. The ninth data bit is transmitted and received using two bits in the ENU and ENUR registers, called XBIT9 and RBIT9. RBIT9 is a read only bit. Parity is not generated or verified in this mode.

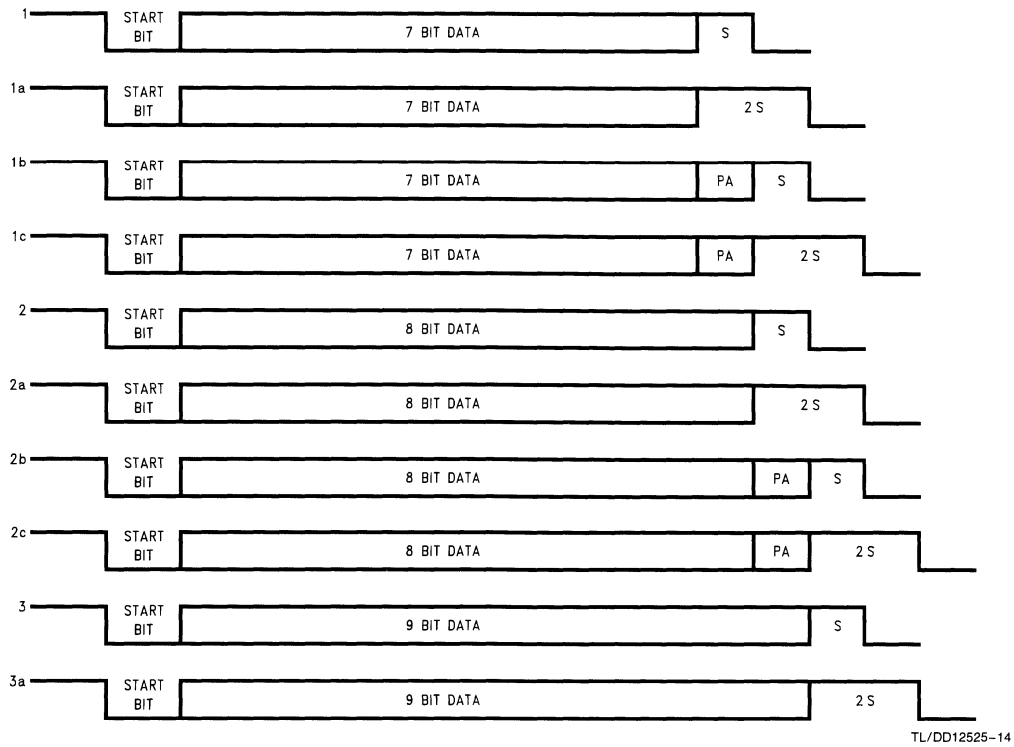
For any of the above framing formats, the last Stop bit can be programmed to be 7/8th of a bit in length. If two Stop bits are selected and the 7/8th bit is set (selected), the second Stop bit will be 7/8th of a bit in length.

The parity is enabled/disabled by PEN bit located in the ENU register. Parity is selected for 7- and 8-bit modes only. If parity is enabled (PEN = 1), the parity selection is then performed by PSEL0 and PSEL1 bits located in the ENU register.

Note that the XBIT9/PSEL0 bit located in the ENU register serves two mutually exclusive functions. This bit programs the ninth bit for transmission when the UART is operating with nine data bits per frame. There is no parity selection in this framing format. For other framing formats XBIT9 is not needed and the bit is PSEL0 used in conjunction with PSEL1 to select parity.

The frame formats for the receiver differ from the transmitter in the number of Stop bits required. The receiver only requires one Stop bit in a frame, regardless of the setting of the Stop bit selection bits in the control register. Note that an implicit assumption is made for full duplex UART operation that the framing formats are the same for the transmitter and receiver.

UART Operation (Continued)



TL/DD12525–14

FIGURE 13. Framing Formats

UART INTERRUPTS

The UART is capable of generating interrupts. Interrupts are generated on Receive Buffer Full and Transmit Buffer Empty. Both interrupts have individual interrupt vectors. Two bytes of program memory space are reserved for each interrupt vector. The two vectors are located at addresses 0xEC to 0xEF Hex in the program memory space. The interrupts can be individually enabled or disabled using Enable Transmit Interrupt (ETI) and Enable Receive Interrupt (ERI) bits in the ENUI register.

The interrupt from the transmitter is set pending, and remains pending, as long as both the TBMT and ETI bits are set. To remove this interrupt, software must either clear the ETI bit or write to the TBUF register (thus clearing the TBMT bit).

The interrupt from the receiver is set pending, and remains pending, as long as both the RBFL and ERI bits are set. To remove this interrupt, software must either clear the ERI bit or read from the RBUF register (thus clearing the RBFL bit).

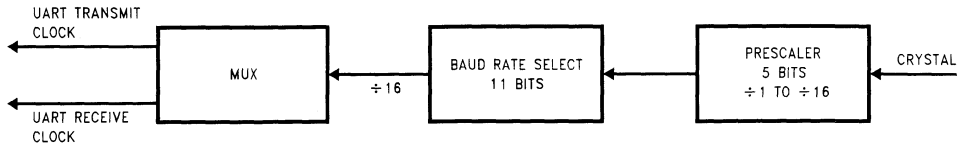
Baud Clock Generation

The clock inputs to the transmitter and receiver sections of the UART can be individually selected to come either from an external source at the CKX pin (port L, pin L1) or from a source selected in the PSR and BAUD registers. Internally, the basic baud clock is created from the oscillator frequency through a two-stage divider chain consisting of a 1–16 (increments of 0.5) prescaler and an 11-bit binary counter. *(Figure 14)* The divide factors are specified through two read/write registers shown in *Figure 15.* Note that the 11-bit Baud Rate Divisor spills over into the Prescaler Select Register (PSR). PSR is cleared upon reset.

As shown in Table III, a Prescaler Factor of 0 corresponds to NO CLOCK. NO CLOCK condition is the UART power down mode where the UART clock is turned off for power saving purpose. The user must also turn the UART clock off when a different baud rate is chosen.

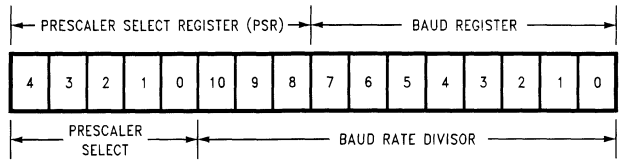
The correspondences between the 5-bit Prescaler Select and Prescaler factors are shown in Table III. There are many ways to calculate the two divisor factors, but one particularly effective method would be to achieve a 1.8432 MHz frequency coming out of the first stage. The 1.8432 MHz prescaler output is then used to drive the software programmable baud rate counter to create a x16 clock for the following baud rates: 110, 134.5, 150, 300, 600, 1200, 1800, 2400, 3600, 4800, 7200, 9600, 19200 and 38400 (Table IV). Other baud rates may be created by using appropriate divisors. The x16 clock is then divided by 16 to provide the rate for the serial shift registers of the transmitter and receiver.

Baud Clock Generation (Continued)



TL/DD12525-15

FIGURE 14. UART BAUD Clock Generation



TL/DD12525-16

FIGURE 15. UART BAUD Clock Divisor Registers

TABLE III. Prescaler Factors

Prescaler Select	Prescaler Factor	Prescaler Select	Prescaler Factor
00000	NO CLOCK	10000	8.5
00001	1	10001	9
00010	1.5	10010	9.5
00011	2	10011	10
00100	2.5	10100	10.5
00101	3	10101	11
00110	3.5	10110	11.5
00111	4	10111	12
01000	4.5	11000	12.5
01001	5	11001	13
01010	5.5	11010	13.5
01011	6	11011	14
01100	6.5	11100	14.5
01101	7	11101	15
01110	7.5	11110	15.5
01111	8	11111	16

TABLE IV. Baud Rate Divisors (1.8432 MHz Prescaler Output)

Baud Rate	Baud Rate Divisor − 1 (N-1)
110 (110.03)	1046
134.5 (134.58)	855
150	767
300	383
600	191
1200	95
1800	63
2400	47
3600	31
4800	23
7200	15
9600	11
19200	5
38400	2

Note: The entries in Table IV assume a prescaler output of 1.8432 MHz. In the asynchronous mode the baud rate could be as high as 625k.

As an example, considering the Asynchronous Mode and a CKI clock of 4.608 MHz, the prescaler factor selected is:

$$4.608/1.8432 = 2.5$$

The 2.5 entry is available in Table III. The 1.8432 MHz prescaler output is then used with proper Baud Rate Divisor (Table II) to obtain different baud rates. For a baud rate of 19200 e.g., the entry in Table IV is 5.

$N - 1 = 5$ ($N - 1$ is the value from Table IV)

$N = 6$ (N is the Baud Rate Divisor)

Baud Rate $= 1.8432\ \text{MHz}/(16 \times 6) = 19200$

The divide by 16 is performed because in the asynchronous mode, the input frequency to the UART is 16 times the baud rate. The equation to calculate baud rates is given below.

The actual Baud Rate may be found from:

$$BR = Fc/(16 \times N \times P)$$

Baud Clock Generation (Continued)

Where:

BR is the Baud Rate

Fc is the CKI frequency

N is the Baud Rate Divisor (Table IV).

P is the Prescaler Divide Factor selected by the value in the Prescaler Select Register (Table III)

Note: In the Synchronous Mode, the divisor 16 is replaced by two.

Example:

Asynchronous Mode:

Crystal Frequency = 5 MHz

Desired baud rate = 9600

Using the above equation N × P can be calculated first.

$$N \times P = (5 \times 10^6)/(16 \times 9600) = 32.552$$

Now 32.552 is divided by each Prescaler Factor (Table III) to obtain a value closest to an integer. This factor happens to be 6.5 (P = 6.5).

$$N = 32.552/6.5 = 5.008 \ (N = 5)$$

The programmed value (from Table IV) should be 4 (N − 1).

Using the above values calculated for N and P:

$$BR = (5 \times 10^6)/(16 \times 5 \times 6.5) = 9615.384$$

$$\% \text{ error} = (9615.385 - 9600)/9600 = 0.16$$

Effect of HALT/IDLE

The UART logic is reinitialized when either the HALT or IDLE modes are entered. This reinitialization sets the TBMT flag and resets all read only bits in the UART control and status registers. Read/Write bits remain unchanged. The Transmit Buffer (TBUF) is not affected, but the Transmit Shift register (TSFT) bits are set to one. The receiver registers RBUF and RSFT are not affected.

The device will exit from the HALT/IDLE modes when the Start bit of a character is detected at the RDX (L3) pin. This feature is obtained by using the Multi-Input Wake Up scheme provided on the device.

Before entering the HALT or IDLE modes the user program must select the Wake Up source to be on the RDX pin. This selection is done by setting bit 3 of WKEN (Wake Up Enable) register. The Wake Up trigger condition is then selected to be high to low transition. This is done via the WKEDG register (Bit 3 is one.)

If the device is halted and crystal oscillator is used, the Wake Up signal will not start the chip running immediately because of the finite start up time requirement of the crystal oscillator. The idle timer (T0) generates a fixed (256 t_C) delay to ensure that the oscillator has indeed stabilized before allowing the device to execute code. The user has to consider this delay when data transfer is expected immediately after exiting the HALT mode.

Diagnostic

Bits CHARL0 and CHARL1 in the ENU register provide a loopback feature for diagnostic testing of the UART. When these bits are set to one, the following occur: The receiver input pin (RDX) is internally connected to the transmitter output pin (TDX); the output of the Transmitter Shift Register is "looped back" into the Receive Shift Register input. In this mode, data that is transmitted is immediately received. This feature allows the processor to verify the transmit and receive data paths of the UART.

Note that the framing format for this mode is the nine bit format; one Start bit, nine data bits, and 7/8, one or two Stop bits. Parity is not generated or verified in this mode.

Attention Mode

The UART Receiver section supports an alternate mode of operation, referred to as ATTENTION Mode. This mode of operation is selected by the ATTN bit in the ENUR register. The data format for transmission must also be selected as having nine Data bits and either 7/8, one or two Stop bits.

The ATTENTION mode of operation is intended for use in networking the device with other processors. Typically in such environments the messages consists of device addresses, indicating which of several destinations should receive them, and the actual data. This Mode supports a scheme in which addresses are flagged by having the ninth bit of the data field set to a 1. If the ninth bit is reset to a zero the byte is a Data byte.

While in ATTENTION mode, the UART monitors the communication flow, but ignores all characters until an address character is received. Upon receiving an address character, the UART signals that the character is ready by setting the RBFL flag, which in turn interrupts the processor if UART Receiver interrupts are enabled. The ATTN bit is also cleared automatically at this point, so that data characters as well as address characters are recognized. Software examines the contents of the RBUF and responds by deciding either to accept the subsequent data stream (by leaving the ATTN bit reset) or to wait until the next address character is seen (by setting the ATTN bit again).

Operation of the UART Transmitter is not affected by selection of this Mode. The value of the ninth bit to be transmitted is programmed by setting XBIT9 appropriately. The value of the ninth bit received is obtained by reading RBIT9. Since this bit is located in ENUR register where the error flags reside, a bit operation on it will reset the error flags.

Comparators

The devices contain two differential comparators, each with a pair of inputs (positive and negative) and an output. Ports I1–I3 and I4–I6 are used for the comparators. The following is the Port I assignment:

- I1 Comparator1 negative input
- I2 Comparator1 positive input
- I3 Comparator1 output
- I4 Comparator2 negative input
- I5 Comparator2 positive input
- I6 Comparator2 output

A Comparator Select Register (CMPSL) is used to enable the comparators, read the outputs of the comparators internally, and enable the outputs of the comparators to the pins. Two control bits (enable and output enable) and one result bit are associated with each comparator. The comparator result bits (CMP1RD and CMP2RD) are read only bits which will read as zero if the associated comparator is not enabled. The Comparator Select Register is cleared with reset, resulting in the comparators being disabled. The comparators should also be disabled before entering either the HALT or IDLE modes in order to save power. The configuration of the CMPSL register is as follows:

Comparators (Continued)

CMPSL REGISTER (ADDRESS X'00B7)

The CMPSL register contains the following bits:

CMP1EN Enable comparator 1

CMP1RD Comparator 1 result (this is a read only bit, which will read as 0 if the comparator is not enabled)

CMP10E Selects pin I3 as comparator 1 output provided that CMPIEN is set to enable the comparator

CMP2EN Enable comparator 2

CMP2RD Comparator 2 result (this is a read only bit, which will read as 0 if the comparator is not enabled)

CMP20E Selects pin I6 as comparator 2 output provided that CMP2EN is set to enable the comparator

Unused	CMP20E	CMP2RD	CMP2EN	CMP10E	CMP1RD	CMP1EN	Unused

Bit 7 Bit 0

Note that the two unused bits of CMPSL may be used as software flags.

Comparator outputs have the same spec as Ports L and G except that the rise and fall times are symmetrical.

Interrupts

The devices support a vectored interrupt scheme. It supports a total of fourteen interrupt sources. The following table lists all the possible device interrupt sources, their arbitration ranking and the memory locations reserved for the interrupt vector for each source.

Two bytes of program memory space are reserved for each interrupt source. All interrupt sources except the software interrupt are maskable. Each of the maskable interrupts have an Enable bit and a Pending bit. A maskable interrupt is active if its associated enable and pending bits are set. If GIE = 1 and an interrupt is active, then the processor will be interrupted as soon as it is ready to start executing an instruction except if the above conditions happen during the Software Trap service routine. This exception is described in the Software Trap sub-section.

The interruption process is accomplished with the INTR instruction (opcode 00), which is jammed inside the Instruction Register and replaces the opcode about to be executed. The following steps are performed for every interrupt:

1. The GIE (Global Interrupt Enable) bit is reset.
2. The address of the instruction about to be executed is pushed into the stack.
3. The PC (Program Counter) branches to address 00FF. This procedure takes 7 t_C cycles to execute.

Arbitration Ranking	Source	Description	Vector Address Hi-Low Byte
(1) Highest	Software	INTR Instruction	0yFE–0yFF
(2)	Reserved	for Future Use	0yFC–0yFD
(3)	External	Pin G0 Edge	0yFA–0yFB
(4)	Timer T0	Underflow	0yF8–0yF9
(5)	Timer T1	T1A/Underflow	0yF6–0yF7
(6)	Timer T1	T1B	0yF4–0yF5
(7)	MICROWIRE/PLUS	BUSY Goes Low	0yF2–0yF3
(8)	Reserved	for Future Use	0yF0–0yF1
(9)	UART	Receive	0yEE–0yEF
(10)	UART	Transmit	0yEC–0yED
(11)	Timer T2	T2A/Underflow	0yEA–0yEB
(12)	Timer T2	T2B	0yE8–0yE9
(13)	Timer T3	T3A/Underflow	0yE6–0yE7
(14)	Timer T3	T3B	0yE4–0yE5
(15)	Port L/Wake Up	Port L Edge	0yE2–0yE3
(16) Lowest	Default	VIS Instr. Execution without Any Interrupts	0yE0–0yE1

y is VIS page, y ≠ 0.

Interrupts (Continued)

At this time, since GIE = 0, other maskable interrupts are disabled. The user is now free to do whatever context switching is required by saving the context of the machine in the stack with PUSH instructions. The user would then program a VIS (Vector Interrupt Select) instruction in order to branch to the interrupt service routine of the highest priority interrupt enabled and pending at the time of the VIS. Note that this is not necessarily the interrupt that caused the branch to address location 00FF Hex prior to the context switching.

Thus, if an interrupt with a higher rank than the one which caused the interruption becomes active before the decision of which interrupt to service is made by the VIS, then the interrupt with the higher rank will override any lower ones and will be acknowledged. The lower priority interrupt(s) are still pending, however, and will cause another interrupt immediately following the completion of the interrupt service routine associated with the higher priority interrupt just serviced. This lower priority interrupt will occur immediately following the RETI (Return from Interrupt) instruction at the end of the interrupt service routine just completed.

Inside the interrupt service routine, the associated pending bit has to be cleared by software. The RETI (Return from Interrupt) instruction at the end of the interrupt service routine will set the GIE (Global Interrupt Enable) bit, allowing the processor to be interrupted again if another interrupt is active and pending.

The VIS instruction looks at all the active interrupts at the time it is executed and performs an indirect jump to the beginning of the service routine of the one with the highest rank.

The addresses of the different interrupt service routines, called vectors, are chosen by the user and stored in ROM in a table starting at 01E0 (assuming that VIS is located between 00FF and 01DF). The vectors are 15-bit wide and therefore occupy 2 ROM locations.

VIS and the vector table must be located in the same 256-byte block (0y00 to 0yFF) except if VIS is located at the last address of a block. In this case, the table must be in the next block. The vector table cannot be inserted in the first 256-byte block (y ≠ 0).

The vector of the maskable interrupt with the lowest rank is located at 0yE0 (Hi-Order byte) and 0yE1 (Lo-Order byte) and so forth in increasing rank number. The vector of the maskable interrupt with the highest rank is located at 0yFA (Hi-Order byte) and 0yFB (Lo-Order byte).

The Software Trap has the highest rank and its vector is located at 0yFE and 0yFF.

If, by accident, a VIS gets executed and no interrupt is active, then the PC (Program Counter) will branch to a vector located at 0yE0–0yE1.

WARNING

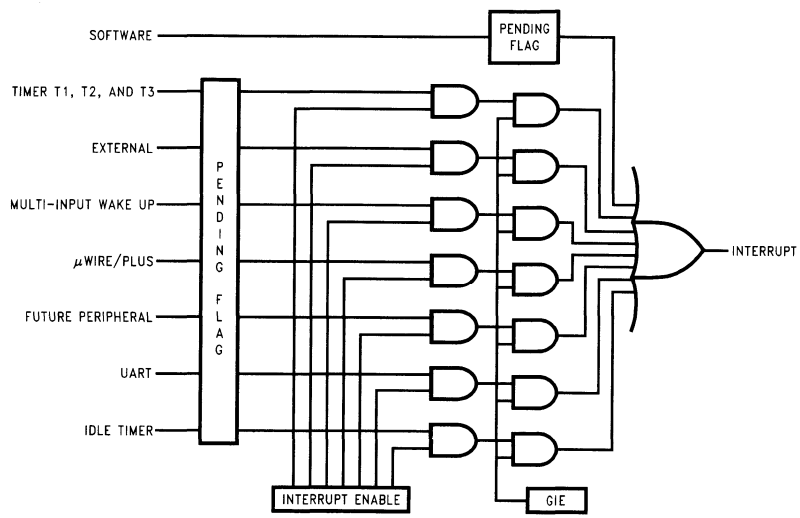
A Default VIS interrupt handler routine must be present. As a minimum, this handler should confirm that the GIE bit is cleared (this indicates that the interrupt sequence has been taken), take care of any required housekeeping, restore context and return. Some sort of Warm Restart procedure should be implemented. These events can occur without any error on the part of the system designer or programmer.

Note: There is always the possibility of an interrupt occurring during an instruction which is attempting to reset the GIE bit or any other interrupt enable bit. If this occurs when a single cycle instruction is being used to reset the interrupt enable bit, the interrupt enable bit will be reset but an interrupt may still occur. This is because interrupt processing is started at the same time as the interrupt bit is being reset. To avoid this scenario, the user should always use a two, three, or four cycle instruction to reset interrupt enable bits.

Figure 16 shows the Interrupt block diagram.

SOFTWARE TRAP

The Software Trap (ST) is a special kind of non-maskable interrupt which occurs when the INTR instruction (used to acknowledge interrupts) is fetched from ROM and placed inside the instruction register. This may happen when the PC is pointing beyond the available ROM address space or when the stack is over-popped.



TL/DD12525–17

FIGURE 16. Interrupt Block Diagram

Interrupts (Continued)

When an ST occurs, the user can re-initialize the stack pointer and do a recovery procedure (similar to reset, but not necessarily containing all of the same initialization procedures) before restarting.

The occurrence of an ST is latched into the ST pending bit. The GIE bit is not affected and the ST pending bit **(not accessible by the user)** is used to inhibit other interrupts and to direct the program to the ST service routine with the VIS instruction. The RPND instruction is used to clear the software interrupt pending bit. This pending bit is also cleared on reset.

The ST has the highest rank among all interrupts.

Nothing (except another ST) can interrupt an ST being serviced.

WATCHDOG

The devices contain a WATCHDOG and clock monitor. The WATCHDOG is designed to detect the user program getting stuck in infinite loops resulting in loss of program control or "runaway" programs. The Clock Monitor is used to detect the absence of a clock or a very slow clock below a specified rate on the CKI pin.

The WATCHDOG consists of two independent logic blocks: WD UPPER and WD LOWER. WD UPPER establishes the upper limit on the service window and WD LOWER defines the lower limit of the service window.

Servicing the WATCHDOG consists of writing a specific value to a WATCHDOG Service Register named WDSVR which is memory mapped in the RAM. This value is composed of three fields, consisting of a 2-bit Window Select, a 5-bit Key Data field, and the 1-bit Clock Monitor Select field. Table V shows the WDSVR register.

The lower limit of the service window is fixed at 2048 instruction cycles. Bits 7 and 6 of the WDSVR register allow the user to pick an upper limit of the service window.

Table VI shows the four possible combinations of lower and upper limits for the WATCHDOG service window. This flexibility in choosing the WATCHDOG service window prevents any undue burden on the user software.

Bits 5, 4, 3, 2 and 1 of the WDSVR register represent the 5-bit Key Data field. The key data is fixed at 01100. Bit 0 of the WDSVR Register is the Clock Monitor Select bit.

TABLE V. WATCHDOG Service Register (WDSVR)

Window Select		Key Data					Clock Monitor
X	X	0	1	1	0	0	Y
7	6	5	4	3	2	1	0

TABLE VI. WATCHDOG Service Window Select

WDSVR Bit 7	WDSVR Bit 6	Service Window (Lower-Upper Limits)
0	0	2k–8k t_C Cycles
0	1	2k–16k t_C Cycles
1	0	2k–32k t_C Cycles
1	1	2k–64k t_C Cycles

Clock Monitor

The Clock Monitor aboard the device can be selected or deselected under program control. The Clock Monitor is guaranteed not to reject the clock if the instruction cycle clock ($1/t_C$) is greater or equal to 10 kHz. This equates to a clock input rate on CKI of greater or equal to 100 kHz.

WATCHDOG Operation

The WATCHDOG and Clock Monitor are disabled during reset. The device comes out of reset with the WATCHDOG armed, the WATCHDOG Window Select bits (bits 6, 7 of the WDSVR Register) set, and the Clock Monitor bit (bit 0 of the WDSVR Register) enabled. Thus, a Clock Monitor error will occur after coming out of reset, if the instruction cycle clock frequency has not reached a minimum specified value, including the case where the oscillator fails to start.

The WDSVR register can be written to only once after reset and the key data (bits 5 through 1 of the WDSVR Register) must match to be a valid write. This write to the WDSVR register involves two irrevocable choices: (i) the selection of the WATCHDOG service window (ii) enabling or disabling of the Clock Monitor. Hence, the first write to WDSVR Register involves selecting or deselecting the Clock Monitor, select the WATCHDOG service window and match the WATCHDOG key data. Subsequent writes to the WDSVR register will compare the value being written by the user to the WATCHDOG service window value and the key data (bits 7 through 1) in the WDSVR Register. Table VII shows the sequence of events that can occur.

The user must service the WATCHDOG at least once before the upper limit of the service window expires. The WATCHDOG may not be serviced more than once in every lower limit of the service window. The user may service the WATCHDOG as many times as wished in the time period between the lower and upper limits of the service window. The first write to the WDSVR Register is also counted as a WATCHDOG service.

The WATCHDOG has an output pin associated with it. This is the WDOUT pin, on pin 1 of the port G. WDOUT is active low. The WDOUT pin is in the high impedance state in the inactive state. Upon triggering the WATCHDOG, the logic will pull the WDOUT (G1) pin low for an additional 16 t_C–32 t_C cycles after the signal level on WDOUT pin goes below the lower Schmitt trigger threshold. After this delay, the device will stop forcing the WDOUT output low.

TABLE VII. WATCHDOG Service Actions

Key Data	Window Data	Clock Monitor	Action
Match	Match	Match	Valid Service: Restart Service Window
Don't Care	Mismatch	Don't Care	Error: Generate WATCHDOG Output
Mismatch	Don't Care	Don't Care	Error: Generate WATCHDOG Output
Don't Care	Don't Care	Mismatch	Error: Generate WATCHDOG Output

WATCHDOG Operation (Continued)

The WATCHDOG service window will restart when the WDOUT pin goes high. It is recommended that the user tie the WDOUT pin back to V_{CC} through a resistor in order to pull WDOUT high.

A WATCHDOG service while the WDOUT signal is active will be ignored. The state of the WDOUT pin is not guaranteed on reset, but if it powers up low then the WATCHDOG will time out and WDOUT will enter high impedance state.

The Clock Monitor forces the G1 pin low upon detecting a clock frequency error. The Clock Monitor error will continue until the clock frequency has reached the minimum specified value, after which the G1 output will enter the high impedance TRI-STATE mode following 16 t_C–32 t_C clock cycles. The Clock Monitor generates a continual Clock Monitor error if the oscillator fails to start, or fails to reach the minimum specified frequency. The specification for the Clock Monitor is as follows:

$1/t_C > 10$ kHz—No clock rejection.

$1/t_C < 10$ Hz—Guaranteed clock rejection.

WATCHDOG AND CLOCK MONITOR SUMMARY

The following salient points regarding the WATCHDOG and CLOCK MONITOR should be noted:

- Both the WATCHDOG and CLOCK MONITOR detector circuits are inhibited during RESET.
- Following RESET, the WATCHDOG and CLOCK MONITOR are both enabled, with the WATCHDOG having the maximum service window selected.
- The WATCHDOG service window and CLOCK MONITOR enable/disable option can only be changed once, during the initial WATCHDOG service following RESET.
- The initial WATCHDOG service must match the key data value in the WATCHDOG Service register WDSVR in order to avoid a WATCHDOG error.
- Subsequent WATCHDOG services must match all three data fields in WDSVR in order to avoid WATCHDOG errors.
- The correct key data value cannot be read from the WATCHDOG Service register WDSVR. Any attempt to read this key data value of 01100 from WDSVR will read as key data value of all 0's.
- The WATCHDOG detector circuit is inhibited during both the HALT and IDLE modes.
- The CLOCK MONITOR detector circuit is active during both the HALT and IDLE modes. Consequently, the COP888 inadvertently entering the HALT mode will be detected as a CLOCK MONITOR error (provided that the CLOCK MONITOR enable option has been selected by the program).
- With the single-pin R/C oscillator mask option selected and the CLKDLY bit reset, the WATCHDOG service window will resume following HALT mode from where it left off before entering the HALT mode.
- With the crystal oscillator mask option selected, or with the single-pin R/C oscillator mask option selected and the CLKDLY bit set, the WATCHDOG service window will be set to its selected value from WDSVR following HALT. Consequently, the WATCHDOG should not be serviced for at least 2048 instruction cycles following HALT, but must be serviced within the selected window to avoid a WATCHDOG error.
- The IDLE timer T0 is not initialized with RESET.
- The user can sync in to the IDLE counter cycle with an IDLE counter (T0) interrupt or by monitoring the T0PND flag. The T0PND flag is set whenever the thirteenth bit of the IDLE counter toggles (every 4096 instruction cycles). The user is responsible for resetting the T0PND flag.
- A hardware WATCHDOG service occurs just as the device exits the IDLE mode. Consequently, the WATCHDOG should not be serviced for at least 2048 instruction cycles following IDLE, but must be serviced within the selected window to avoid a WATCHDOG error.
- Following RESET, the initial WATCHDOG service (where the service window and the CLOCK MONITOR enable/disable must be selected) may be programmed anywhere within the maximum service window (65,536 instruction cycles) initialized by RESET. Note that this initial WATCHDOG service may be programmed within the initial 2048 instruction cycles without causing a WATCHDOG error.

Detection of Illegal Conditions

The device can detect various illegal conditions resulting from coding errors, transient noise, power supply voltage drops, runaway programs, etc.

Reading of undefined ROM gets zeros. The opcode for software interrupt is zero. If the program fetches instructions from undefined ROM, this will force a software interrupt, thus signaling that an illegal condition has occurred.

The subroutine stack grows down for each call (jump to subroutine), interrupt, or PUSH, and grows up for each return or POP. The stack pointer is initialized to RAM location 06F Hex during reset. Consequently, if there are more returns than calls, the stack pointer will point to addresses 070 and 071 Hex (which are undefined RAM). Undefined RAM from addresses 070 to 07F (Segment 0), 140 to 17F (Segment 1), and all other segments (i.e., Segments 3 . . . etc.) is read as all 1's, which in turn will cause the program to return to address 7FFF Hex. This is an undefined ROM location and the instruction fetched (all 0's) from this location will generate a software interrupt signaling an illegal condition.

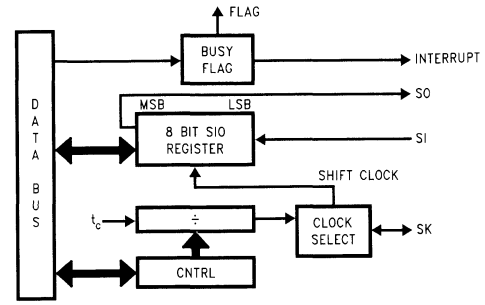
Thus, the chip can detect the following illegal conditions:

1. Executing from undefined ROM
2. Over "POP"ing the stack by having more returns than calls.

When the software interrupt occurs, the user can re-initialize the stack pointer and do a recovery procedure before restarting (this recovery program is probably similar to that following reset, but might not contain the same program initialization procedures). The recovery program should reset the software interrupt pending bit using the RPND instruction.

MICROWIRE/PLUS

MICROWIRE/PLUS is a serial synchronous communications interface. The MICROWIRE/PLUS capability enables the device to interface with any of National Semiconductor's MICROWIRE peripherals (i.e. A/D converters, display drivers, E^2PROMs etc.) and with other microcontrollers which support the MICROWIRE interface. It consists of an 8-bit serial shift register (SIO) with serial data input (SI), serial data output (SO) and serial shift clock (SK). *Figure 17* shows a block diagram of the MICROWIRE/PLUS logic.



TL/DD12525–18

FIGURE 17. MICROWIRE/PLUS Block Diagram

The shift clock can be selected from either an internal source or an external source. Operating the MICROWIRE/PLUS arrangement with the internal clock source is called the Master mode of operation. Similarly, operating the MICROWIRE/PLUS arrangement with an external shift clock is called the Slave mode of operation.

The CNTRL register is used to configure and control the MICROWIRE/PLUS mode. To use the MICROWIRE/PLUS, the MSEL bit in the CNTRL register is set to one. In the master mode, the SK clock rate is selected by the two bits, SL0 and SL1, in the CNTRL register. Table VIII details the different clock rates that may be selected.

TABLE VIII. MICROWIRE/PLUS Master Mode Clock Select

SL1	SL0	SK
0	0	$2 \times t_c$
0	1	$4 \times t_c$
1	x	$8 \times t_c$

Where t_c is the instruction cycle clock

MICROWIRE/PLUS (Continued)

MICROWIRE/PLUS OPERATION

Setting the BUSY bit in the PSW register causes the MICROWIRE/PLUS to start shifting the data. It gets reset when eight data bits have been shifted. The user may reset the BUSY bit by software to allow less than 8 bits to shift. If enabled, an interrupt is generated when eight data bits have been shifted. The device may enter the MICROWIRE/PLUS mode either as a Master or as a Slave. *Figure 18* shows how two devices, microcontrollers and several peripherals may be interconnected using the MICROWIRE/PLUS arrangements.

Warning

The SIO register should only be loaded when the SK clock is low. Loading the SIO register while the SK clock is high will result in undefined data in the SIO register. SK clock is normally low when not shifting.

Setting the BUSY flag when the input SK clock is high in the MICROWIRE/PLUS slave mode may cause the current SK clock for the SIO shift register to be narrow. For safety, the BUSY flag should only be set when the input SK clock is low.

MICROWIRE/PLUS Master Mode Operation

In the MICROWIRE/PLUS Master mode of operation the shift clock (SK) is generated internally by the device. The MICROWIRE Master always initiates all data exchanges. The MSEL bit in the CNTRL register must be set to enable the SO and SK functions onto the G Port. The SO and SK pins must also be selected as outputs by setting appropriate bits in the Port G configuration register. Table IX summarizes the bit settings required for Master mode of operation.

MICROWIRE/PLUS Slave Mode Operation

In the MICROWIRE/PLUS Slave mode of operation the SK clock is generated by an external source. Setting the MSEL bit in the CNTRL register enables the SO and SK functions onto the G Port. The SK pin must be selected as an input and the SO pin is selected as an output pin by setting and resetting the appropriate bit in the Port G configuration register. Table IX summarizes the settings required to enter the Slave mode of operation.

The user must set the BUSY flag immediately upon entering the Slave mode. This will ensure that all data bits sent by the Master will be shifted properly. After eight clock pulses the BUSY flag will be cleared and the sequence may be repeated.

Alternate SK Phase Operation

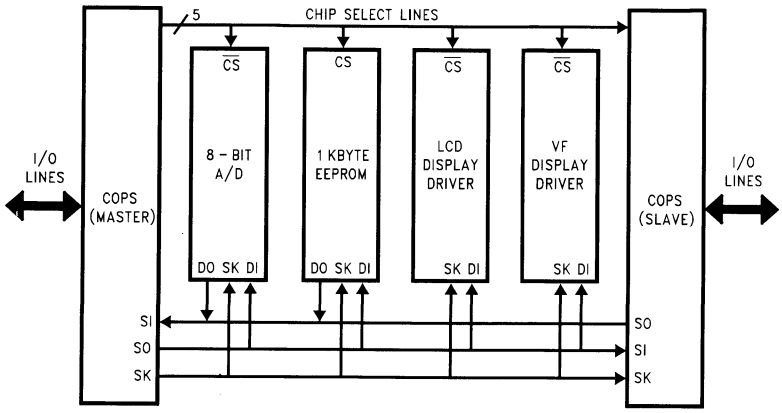
The device allows either the normal SK clock or an alternate phase SK clock to shift data in and out of the SIO register. In both the modes the SK is normally low. In the normal mode data is shifted in on the rising edge of the SK clock and the data is shifted out on the falling edge of the SK clock. The SIO register is shifted on each falling edge of the SK clock. In the alternate SK phase operation, data is shifted in on the falling edge of the SK clock and shifted out on the rising edge of the SK clock.

A control flag, SKSEL, allows either the normal SK clock or the alternate SK clock to be selected. Resetting SKSEL causes the MICROWIRE/PLUS logic to be clocked from the normal SK signal. Setting the SKSEL flag selects the alternate SK clock. The SKSEL is mapped into the G6 configuration bit. The SKSEL flag will power up in the reset condition, selecting the normal SK signal.

TABLE IX. MICROWIRE/PLUS Mode Selection

G4 (SO) Config. Bit	G5 (SK) Config. Bit	G4 Fun.	G5 Fun.	Operation
1	1	SO	Int. SK	MICROWIRE/PLUS Master
0	1	TRI-STATE	Int. SK	MICROWIRE/PLUS Master
1	0	SO	Ext. SK	MICROWIRE/PLUS Slave
0	0	TRI-STATE	Ext. SK	MICROWIRE/PLUS Slave

Note: This table assumes that the control flag MSEL is set.



TL/DD12525–19

FIGURE 18. MICROWIRE/PLUS Application

Memory Map

All RAM, ports and registers (except A and PC) are mapped into data memory address space.

Address S/ADD REG	Contents
0000 to 006F	On-Chip RAM bytes (112 bytes)
0070 to 007F	Unused RAM Address Space (Reads As All Ones)
xx80 to xxAF	Unused RAM Address Space (Reads Undefined Data)
xxB0	Timer T3 Lower Byte
xxB1	Timer T3 Upper Byte
xxB2	Timer T3 Autoload Register T3RA Lower Byte
xxB3	Timer T3 Autoload Register T3RA Upper Byte
xxB4	Timer T3 Autoload Register T3RB Lower Byte
xxB5	Timer T3 Autoload Register T3RB Upper Byte
xxB6	Timer T3 Control Register
xxB7	Comparator Select Register (CMPSL)
xxB8	UART Transmit Buffer (TBUF)
xxB9	UART Receive Buffer (RBUF)
xxBA	UART Control and Status Register (ENU)
xxBB	UART Receive Control and Status Register (ENUR)
xxBC	UART Interrupt and Clock Source Register (ENUI)
xxBD	UART Baud Register (BAUD)
xxBE	UART Prescale Select Register (PSR)
xxBF	Reserved for UART
xxC0	Timer T2 Lower Byte
xxC1	Timer T2 Upper Byte
xxC2	Timer T2 Autoload Register T2RA Lower Byte
xxC3	Timer T2 Autoload Register T2RA Upper Byte
xxC4	Timer T2 Autoload Register T2RB Lower Byte
xxC5	Timer T2 Autoload Register T2RB Upper Byte
xxC6	Timer T2 Control Register
xxC7	WATCHDOG Service Register (Reg:WDSVR)
xxC8	MIWU Edge Select Register (Reg:WKEDG)
xxC9	MIWU Enable Register (Reg:WKEN)
xxCA	MIWU Pending Register (Reg:WKPND)
xxCB	Reserved
xxCC	Reserved
xxCD to xxCF	Reserved

Address S/ADD REG	Contents
xxD0	Port L Data Register
xxD1	Port L Configuration Register
xxD2	Port L Input Pins (Read Only)
xxD3	Reserved for Port L
xxD4	Port G Data Register
xxD5	Port G Configuration Register
xxD6	Port G Input Pins (Read Only)
xxD7	Port I Input Pins (Read Only)
xxD8	Port C Data Register
xxD9	Port C Configuration Register
xxDA	Port C Input Pins (Read Only)
xxDB	Reserved for Port C
xxDC	Port D
xxDD to DF	Reserved for Port D
xxE0 to xxE5	Reserved for EE Control Registers
xxE6	Timer T1 Autoload Register T1RB Lower Byte
xxE7	Timer T1 Autoload Register T1RB Upper Byte
xxE8	ICNTRL Register
xxE9	MICROWIRE/PLUS Shift Register
xxEA	Timer T1 Lower Byte
xxEB	Timer T1 Upper Byte
xxEC	Timer T1 Autoload Register T1RA Lower Byte
xxED	Timer T1 Autoload Register T1RA Upper Byte
xxEE	CNTRL Control Register
xxEF	PSW Register
xxF0 to FB	On-Chip RAM Mapped as Registers
xxFC	X Register
xxFD	SP Register
xxFE	B Register
xxFF	S Register
0100–017F	On-Chip 128 RAM Bytes

Note: Reading memory locations 0070H–007FH (Segment 0) will return all ones. Reading unused memory locations 0080H–00AFH (Segment 0) will return undefined data. Reading memory locations from other Segments (i.e., Segment 2, Segment 3, ... etc.) will return all ones.

Addressing Modes

There are ten addressing modes, six for operand addressing and four for transfer of control.

OPERAND ADDRESSING MODES

Register Indirect

This is the "normal" addressing mode. The operand is the data memory addressed by the B pointer or X pointer.

Register Indirect (with auto post increment or decrement of pointer)

This addressing mode is used with the LD and X instructions. The operand is the data memory addressed by the B pointer or X pointer. This is a register indirect mode that automatically post increments or decrements the B or X register after executing the instruction.

Direct

The instruction contains an 8-bit address field that directly points to the data memory for the operand.

Immediate

The instruction contains an 8-bit immediate field as the operand.

Short Immediate

This addressing mode is used with the Load B Immediate instruction. The instruction contains a 4-bit immediate field as the operand.

Indirect

This addressing mode is used with the LAID instruction. The contents of the accumulator are used as a partial address (lower 8 bits of PC) for accessing a data operand from the program memory.

TRANSFER OF CONTROL ADDRESSING MODES

Relative

This mode is used for the JP instruction, with the instruction field being added to the program counter to get the new program location. JP has a range from −31 to +32 to allow a 1-byte relative jump (JP + 1 is implemented by a NOP instruction). There are no "pages" when using JP, since all 15 bits of PC are used.

Absolute

This mode is used with the JMP and JSR instructions, with the instruction field of 12 bits replacing the lower 12 bits of the program counter (PC). This allows jumping to any location in the current 4k program memory segment.

Absolute Long

This mode is used with the JMPL and JSRL instructions, with the instruction field of 15 bits replacing the entire 15 bits of the program counter (PC). This allows jumping to any location up to 32k in the program memory space.

Indirect

This mode is used with the JID instruction. The contents of the accumulator are used as a partial address (lower 8 bits of PC) for accessing a location in the program memory. The contents of this program memory location serve as a partial address (lower 8 bits of PC) for the jump to the next instruction.

Note: The VIS is a special case of the Indirect Transfer of Control addressing mode, where the double byte vector associated with the interrupt is transferred from adjacent addresses in the program memory into the program counter (PC) in order to jump to the associated interrupt service routine.

Instruction Set

Register and Symbol Definition

Registers	
A	8-Bit Accumulator Register
B	8-Bit Address Register
X	8-Bit Address Register
SP	8-Bit Stack Pointer Register
PC	15-Bit Program Counter Register
PU	Upper 7 Bits of PC
PL	Lower 8 Bits of PC
C	1 Bit of PSW Register for Carry
HC	1 Bit of PSW Register for Half Carry
GIE	1 Bit of PSW Register for Global Interrupt Enable
VU	Interrupt Vector Upper Byte
VL	Interrupt Vector Lower Byte

Symbols	
[B]	Memory Indirectly Addressed by B Register
[X]	Memory Indirectly Addressed by X Register
MD	Direct Addressed Memory
Mem	Direct Addressed Memory or [B]
Meml	Direct Addressed Memory or [B] or Immediate Data
Imm	8-Bit Immediate Data
Reg	Register Memory: Addresses F0 to FF (Includes B, X and SP)
Bit	Bit Number (0 to 7)
←	Loaded with
↔	Exchanged with

Instruction Set (Continued)

INSTRUCTION SET

ADD	A,Meml	ADD	A ← A + Meml
ADC	A,Meml	ADD with Carry	A ← A + Meml + C, C ← Carry, HC ← Half Carry
SUBC	A,Meml	Subtract with Carry	A ← A − Meml + C, C ← Carry, HC ← Half Carry
AND	A,Meml	Logical AND	A ← A and Meml
ANDSZ	A,Imm	Logical AND Immed., Skip if Zero	Skip next if (A and Imm) = 0
OR	A,Meml	Logical OR	A ← A or Meml
XOR	A,Meml	Logical EXclusive OR	A ← A xor Meml
IFEQ	MD,Imm	IF EQual	Compare MD and Imm, Do next if MD = Imm
IFEQ	A,Meml	IF EQual	Compare A and Meml, Do next if A = Meml
IFNE	A,Meml	IF Not Equal	Compare A and Meml, Do next if A ≠ Meml
IFGT	A,Meml	IF Greater Than	Compare A and Meml, Do next if A > Meml
IFBNE	#	If B Not Equal	Do next if lower 4 bits of B ≠ Imm
DRSZ	Reg	Decrement Reg., Skip if Zero	Reg ← Reg − 1, Skip if Reg = 0
SBIT	#,Mem	Set BIT	1 to bit, Mem (bit = 0 to 7 immediate)
RBIT	#,Mem	Reset BIT	0 to bit, Mem
IFBIT	#,Mem	IF BIT	If bit in A or Mem is true do next instruction
RPND		Reset PeNDing Flag	Reset Software Interrupt Pending Flag
X	A,Mem	EXchange A with Memory	A ⟷ Mem
X	A,[X]	EXchange A with Memory [X]	A ⟷ [X]
LD	A,Meml	LoaD A with Memory	A ← Meml
LD	A,[X]	LoaD A with Memory [X]	A ← [X]
LD	B,Imm	LoaD B with Immed.	B ← Imm
LD	Mem,Imm	LoaD Memory Immed.	Mem ← Imm
LD	Reg,Imm	LoaD Register Memory Immed.	Reg ← Imm
X	A, [B ±]	EXchange A with Memory [B]	A ⟷ [B], (B ← B ±1)
X	A, [X ±]	EXchange A with Memory [X]	A ⟷ [X], (X ← ±1)
LD	A, [B±]	LoaD A with Memory [B]	A ← [B], (B ← B ±1)
LD	A, [X±]	LoaD A with Memory [X]	A ← [X], (X ← X±1)
LD	[B±],Imm	LoaD Memory [B] Immed.	[B] ← Imm, (B ← B±1)
CLR	A	CLeaR A	A ← 0
INC	A	INCrement A	A ← A + 1
DEC	A	DECrementA	A ← A − 1
LAID		Load A InDirect from ROM	A ← ROM (PU,A)
DCOR	A	Decimal CORrect A	A ← BCD correction of A (follows ADC, SUBC)
RRC	A	Rotate A Right thru C	C → A7 → . . . → A0 → C
RLC	A	Rotate A Left thru C	C ← A7 ← . . . ← A0 ← C
SWAP	A	SWAP nibbles of A	A7 . . . A4 ⟷ A3 . . . A0
SC		Set C	C ← 1, HC ← 1
RC		Reset C	C ← 0, HC ← 0
IFC		IF C	IF C is true, do next instruction
IFNC		IF Not C	If C is not true, do next instruction
POP	A	POP the stack into A	SP ← SP + 1, A ← [SP]
PUSH	A	PUSH A onto the stack	[SP] ← A, SP ← SP − 1
VIS		Vector to Interrupt Service Routine	PU ← [VU], PL ← [VL]
JMPL	Addr.	Jump absolute Long	PC ← ii (ii = 15 bits, 0k to 32k)
JMP	Addr.	Jump absolute	PC9 . . . 0 ← i (i = 12 bits)
JP	Disp.	Jump relative short	PC ← PC + r (r is −31 to +32, except 1)
JSRL	Addr.	Jump SubRoutine Long	[SP] ← PL, [SP−1] ← PU,SP−2, PC ← ii
JSR	Add.	Jump SubRoutine	[SP] ← PL, [SP−1] ← PU,SP−2, PC9 . . . 0 ← i
JID		Jump InDirect	PL ← ROM (PU,A)
RET		RETurn from subroutine	SP + 2, PL ← [SP], PU ← [SP−1]
RETSK		RETurn and SKip	SP + 2, PL ← [SP],PU ← [SP−1]
RETI		RETurn from Interrupt	SP + 2, PL ← [SP],PU ← [SP−1],GIE ← 1
INTR		Generate an Interrupt	[SP] ← PL, [SP−1] ← PU, SP−2, PC ← 0FF
NOP		No OPeration	PC ← PC + 1

Instruction Execution Time

Most instructions are single byte (with immediate addressing mode instructions taking two bytes).

Most single byte instructions take one cycle time to execute.

Skipped instructions require x number of cycles to be skipped, where x equals the number of bytes in the skipped instruction opcode.

See the BYTES and CYCLES per INSTRUCTION table for details.

Bytes and Cycles per Instruction

The following table shows the number of bytes and cycles for each instruction in the format of byte/cycle.

Logic and Arithmetic Instructions

	[B]	Direct	Immed.
ADD	1/1	3/4	2/2
ADC	1/1	3/4	2/2
SUBC	1/1	3/4	2/2
AND	1/1	3/4	2/2
OR	1/1	3/4	2/2
XOR	1/1	3/4	2/2
IFEQ	1/1	3/4	2/2
IFGT	1/1	3/4	2/2
IFBNE	1/1		
DRSZ		1/3	
SBIT	1/1	3/4	
RBIT	1/1	3/4	
IFBIT	1/1	3/4	

RPND	1/1

Instructions Using A and C

CLRA	1/1
INCA	1/1
DECA	1/1
LAID	1/3
DCORA	1/1
RRCA	1/1
RLCA	1/1
SWAPA	1/1
SC	1/1
RC	1/1
IFC	1/1
IFNC	1/1
PUSHA	1/3
POPA	1/3
ANDSZ	2/2

Transfer of Control Instructions

JMPL	3/4
JMP	2/3
JP	1/3
JSRL	3/5
JSR	2/5
JID	1/3
VIS	1/5
RET	1/5
RETSK	1/5
RETI	1/5
INTR	1/7
NOP	1/1

Memory Transfer Instructions

	Register Indirect		Direct	Immed.	Register Indirect Auto Incr. and Decr.		
	[B]	[X]			[B+, B−]	[X+, X−]	
X A,*	1/1	1/3	2/3		1/2	1/3	
LD A,*	1/1	1/3	2/3	2/2	1/2	1/3	
LD B, Imm				1/1			(IF B < 16)
LD B, Imm				2/3			(IF B > 15)
LD Mem, Imm	2/2	2/2	3/3		2/2		
LD Reg, Imm			2/3				
IFEQ MD, Imm			3/3				

* = > Memory location addressed by B or X or directly.

COP87L88EG/COP87L84EG Opcode Table

UPPER NIBBLE																	LOWER NIBBLE
F	E	D	C	B	A	9	8	7	6	5	4	3	2	1	0		
JP −15	JP −31	LD 0F0, #i	DRSZ 0F0	RRCA	RC	ADC A, #i	ADC A,[B]	IFBIT 0,[B]	ANDSZ A, #i	LD B,#0F	IFBNE 0	JSR x000−x0FF	JMP x000−x0FF	JP + 17	INTR	0	
JP −14	JP −30	LD 0F1, #i	DRSZ 0F1	*	SC	SUBC A, #i	SUB A,[B]	IFBIT 1,[B]	*	LD B,#0E	IFBNE 1	JSR x100−x1FF	JMP x100−x1FF	JP + 18	JP + 2	1	
JP −13	JP −29	LD 0F2, #i	DRSZ 0F2	X A, [X+]	X A, [B+]	IFEQ A, #i	IFEQ A,[B]	IFBIT 2,[B]	*	LD B,#0D	IFBNE 2	JSR x200−x2FF	JMP x200−x2FF	JP + 19	JP + 3	2	
JP −12	JP −28	LD 0F3, #i	DRSZ 0F3	X A, [X−]	X A, [B−]	IFGT A, #i	IFGT A,[B]	IFBIT 3,[B]	*	LD B,#0C	IFBNE 3	JSR x300−x3FF	JMP x300−x3FF	JP + 20	JP + 4	3	
JP −11	JP −27	LD 0F4, #i	DRSZ 0F4	VIS	LAID	ADD A, #i	ADD A,[B]	IFBIT 4,[B]	CLRA	LD B,#0B	IFBNE 4	JSR x400−x4FF	JMP x400−x4FF	JP + 21	JP + 5	4	
JP −10	JP −26	LD 0F5, #i	DRSZ 0F5	RPND	JID	AND A, #i	AND A,[B]	IFBIT 5,[B]	SWAPA	LD B,#0A	IFBNE 5	JSR x500−x5FF	JMP x500−x5FF	JP + 22	JP + 6	5	
JP −9	JP −25	LD 0F6, #i	DRSZ 0F6	X A,[X]	X A,[B]	XOR A, #i	XOR A,[B]	IFBIT 6,[B]	DCORA	LD B,#09	IFBNE 6	JSR x600−x6FF	JMP x600−x6FF	JP + 23	JP + 7	6	
JP −8	JP −24	LD 0F7, #i	DRSZ 0F7	*	*	OR A,#i	OR A,[B]	IFBIT 7,[B]	PUSHA	LD B,#08	IFBNE 7	JSR x700−x7FF	JMP x700−x7FF	JP + 24	JP + 8	7	
JP −7	JP −23	LD 0F8, #i	DRSZ 0F8	NOP	RLCA	LD A,#i	IFC	SBIT 0,[B]	RBIT 0,[B]	LD B,#07	IFBNE 8	JSR x800−x8FF	JMP x800−x8FF	JP + 25	JP + 9	8	
JP −6	JP −22	LD 0F9, #i	DRSZ 0F9	IFNE A,[B]	IFEQ Md,#i	IFNE A,#i	IFNC	SBIT 1,[B]	RBIT 1,[B]	LD B,#06	IFBNE 9	JSR x900−x9FF	JMP x900−x9FF	JP + 26	JP + 10	9	
JP −5	JP −21	LD 0FA, #i	DRSZ 0FA	LD A, [X+]	LD A, [B+]	LD [B+], #i	INCA	SBIT 2,[B]	RBIT 2,[B]	LD B,#05	IFBNE 0A	JSR xA00−xAFF	JMP xA00−xAFF	JP + 27	JP + 11	A	
JP −4	JP −20	LD 0FB, #i	DRSZ 0FB	LD A, [X−]	LD A, [B−]	LD [B−], #i	DECA	SBIT 3,[B]	RBIT 3,[B]	LD B,#04	IFBNE 0B	JSR xB00−xBFF	JMP xB00−xBFF	JP + 28	JP + 12	B	
JP −3	JP −19	LD 0FC, #i	DRSZ 0FC	LD Md,#i	JMPL	X A,Md	POPA	SBIT 4,[B]	RBIT 4,[B]	LD B,#03	IFBNE 0C	JSR xC00−xCFF	JMP xC00−xCFF	JP + 29	JP + 13	C	
JP −2	JP −18	LD 0FD, #i	DRSZ 0FD	DIR	JSRL	LD A,Md	RETSK	SBIT 5,[B]	RBIT 5,[B]	LD B,#02	IFBNE 0D	JSR xD00−xDFF	JMP xD00−xDFF	JP + 30	JP + 14	D	
JP −1	JP −17	LD 0FE, #i	DRSZ 0FE	LD A,[X]	LD A,[B]	LD [B],#i	RET	SBIT 6,[B]	RBIT 6,[B]	LD B,#01	IFBNE 0E	JSR xE00−xEFF	JMP xE00−xEFF	JP + 31	JP + 15	E	
JP −0	JP −16	LD 0FF, #i	DRSZ 0FF	*	*	LD B,#i	RETI	SBIT 7,[B]	RBIT 7,[B]	LD B,#00	IFBNE 0F	JSR xF00−xFFF	JMP xF00−xFFF	JP + 32	JP + 16	F	

where,

i is the immediate data

Md is a directly addressed memory location

* is an unused opcode

Note: The opcode 60 Hex is also the opcode for IFBIT #i,A

Development Support

SUMMARY

- iceMASTER™: IM-COP8/400—Full feature in-circuit emulation for all COP8 products. A full set of COP8 Basic and Feature Family device and package specific probes are available.
- COP8 Debug Module: Moderate cost in-circuit emulation and development programming unit.
- COP8 Evaluation and Programming Unit: EPU-COP888GG—low cost in-circuit simulation and development programming unit.
- Assembler: COP8-DEV-IBMA. A DOS installable cross development Assembler, Linker, Librarian and Utility Software Development Tool Kit.
- C Compiler: COP8C. A DOS installable cross development Software Tool Kit.
- OTP/EPROM Programmer Support: Covering needs from engineering prototype, pilot production to full production environments.

iceMASTER (IM) IN-CIRCUIT EMULATION

The iceMASTER IM-COP8/400 is a full feature, PC based, in-circuit emulation tool developed and marketed by Meta-Link Corporation to support the whole COP8 family of products. National is a resale vendor for these products.

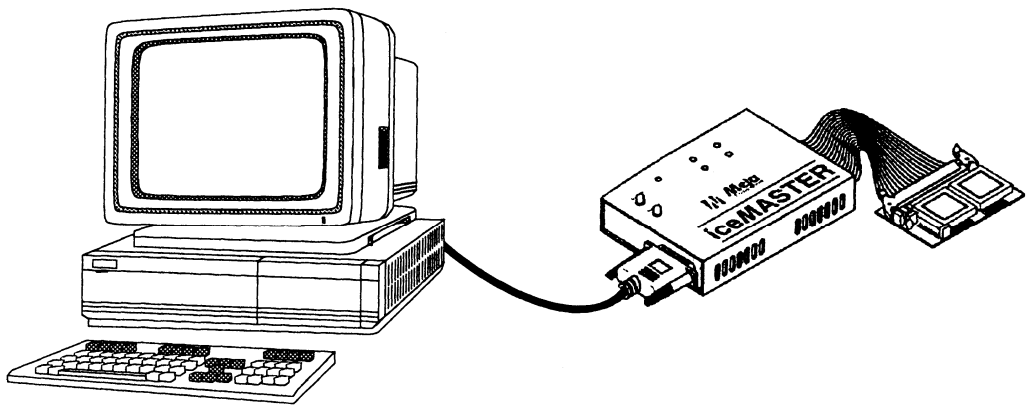
See *Figure 19* for configuration.

The iceMASTER IM-COP8/400 with its device specific COP8 Probe provides a rich feature set for developing, testing and maintaining product:

- Real-time in-circuit emulation; full 2.4V–5.5V operation range, full DC-10 MHz clock. Chip options are programmable or jumper selectable.
- Direct connection to application board by package compatible socket or surface mount assembly.
- Full 32 kbytes of loadable programming space that overlays (replaces) the on-chip ROM or EPROM. On-chip RAM and I/O blocks are used directly or recreated on the probe as necessary.
- Full 4k frame synchronous trace memory. Address, instruction, and 8 unspecified, circuit connectable trace lines. Display can be HLL source (e.g., C source), assembly or mixed.
- A full 64k hardware configurable break, trace on, trace off control, and pass count increment events.
- Tool set integrated interactive symbolic debugger—supports both assembler (COFF) and C Compiler (.COD) linked object formats.
- Real time performance profiling analysis; selectable bucket definition.
- Watch windows, content updated automatically at each execution break.
- Instruction by instruction memory/register changes displayed on source window when in single step operation.
- Single base unit and debugger software reconfigurable to support the entire COP8 family; only the probe personality needs to change. Debugger software is processor customized, and reconfigured from a master model file.
- Processor specific symbolic display of registers and bit level assignments, configured from master model file.
- Halt/Idle mode notification.
- On-line HELP customized to specific processor using master model file.
- Includes a copy of COP8-DEV-IBMA assembler and linker SDK.

IM Order Information

Base Unit	
IM-COP8/400-1	iceMASTER Base Unit, 110V Power Supply
IM-COP8/400-2	iceMASTER Base Unit, 220V Power Supply
iceMASTER Probe	
MHW-884EG28DWPC	28 DIP
MHW-888EG40DWPC	40 DIP
MHW-888EG44PWPC	44 PLCC
28 DIP to 28 SO Adapter	
MHW-SOIC28	28 SO



TL/DD/12525–21

FIGURE 19. COP8 iceMASTER Environment

Development Support (Continued)

iceMASTER DEBUG MODULE (DM)

The iceMASTER Debug Module is a PC based, combination in-circuit emulation tool and COP8 based OTP/EPROM programming tool developed and marketed by MetaLink Corporation to support the whole COP8 family of products. National is a resale vendor for these products.

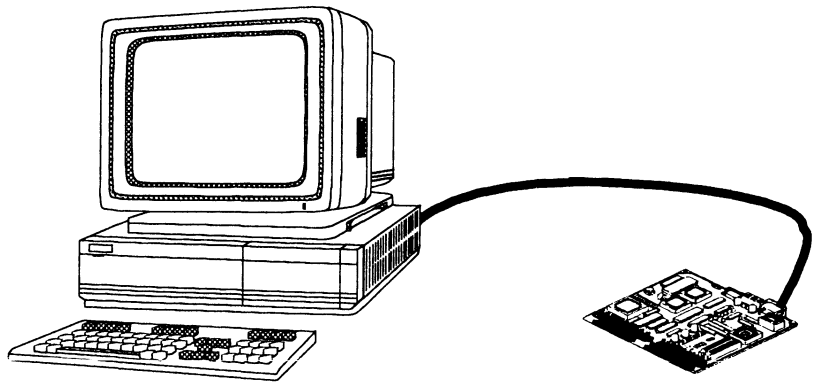
See *Figure 20* for configuration.

The iceMASTER Debug Module is a moderate cost development tool. It has the capability of in-circuit emulation for a specific COP8 microcontroller and in addition serves as a programming tool for COP8 OTP and EPROM product families. Summary of features is as follows:

- Real-time in-circuit emulation; full operating voltage range operation, full DC-10 MHz clock.
- All processor I/O pins can be cabled to an application development board with package compatible cable to socket and surface mount assembly.
- Full 32 kbytes of loadable programming space that overlays (replaces) the on-chip ROM or EPROM. On-chip RAM and I/O blocks are used directly or recreated as necessary.
- 100 frames of synchronous trace memory. The display can be HLL source (C source), assembly or mixed. The most recent history prior to a break is available in the trace memory.
- Configured break points; uses INTR instruction which is modestly intrusive.
- Software—only supported features are selectable.
- Tool set integrated interactive symbolic debugger—supports both assembler (COFF) and C Compiler (.COD) SDK linked object formats.
- Instruction by instruction memory/register changes displayed when in single step operation.
- Debugger software is processor customized, and reconfigured from a master model file.
- Processor specific symbolic display of registers and bit level assignments, configured from master model file.
- Halt/Idle mode notification.
- Programming menu supports full product line of programmable OTP and EPROM COP8 products. Program data is taken directly from the overlay RAM.
- Programming of 44 PLCC and 68 PLCC parts requires external programming adapters.
- Includes wallmount power supply.
- On-board V_{PP} generator from 5V input or connection to external supply supported. Rquires V_{PP} level adjustment per the family programming specification (correct level is provided on an on-screen pop-down display).
- On-line HELP customized to specific processor using master model file.
- Includes a copy of COP8-DEV-IBMA assembler and linker SDK.

DM Order Information

Debug Module Unit	
COP8-DM/888EG	
Cable Adapters	
DM-COP8/28D	28 DIP
DM-COP8/40D	40 DIP
DM-COP8/44P	44 PLCC
28 DIP to 28 SO Adapter	
DM-COP8/28D-SO	28 SO



TL/DD/12525–22

FIGURE 20. COP8-DM Environment

Development Support (Continued)

iceMASTER EVALUATION PROGRAMMING UNIT (EPU)

The iceMASTER EPU-COP888GG is a PC based, in-circuit simulation tool to support the feature family COP8 products.

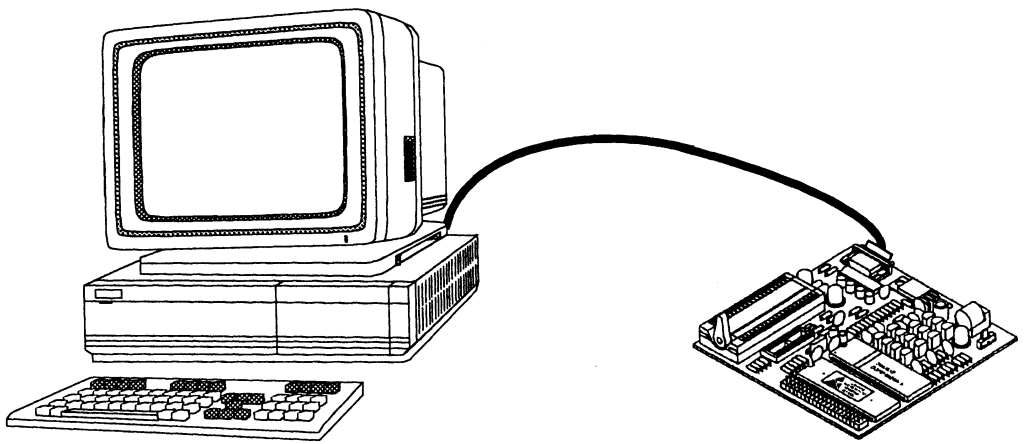
See *Figure 21* for configuration.

The simulation capability is a very low cost means of evaluating the general COP8 architecture. In addition, the EPU has programming capability, with added adapters, for programming the whole COP8 product family of OTP and EPROM products. The product includes the following features:

- Non-real-time in-circuit simulation. Program overlay memory is PC resident; instructions are downloaded over RS-232 as executed. Approximate performance is 20 kHz.
- Includes a 40-pin DIP cable adapter. Other target packages are not supported. All processor I/O pins are cabled to the application development environment.
- Full 32 kbytes of loadable programming space that overlays (replaces) the on-chip ROM or EPROM. On-chip RAM and I/O blocks are used directly or recreated as necessary.
- On-chip timer and WATCHDOG execution are not well synchronized to the instruction simulation.
- 100 frames of synchronous trace memory. The display can be HLL source (e.g., C source), assembly or mixed. The most recent history prior to a break is available in the trace memory.
- Up to eight software configured break points; uses INTR instruction which is modestly intrusive.
- Common look-feel debugger software across all MetaLink products—only supported features are selectable.
- Tool set integrated interactive symbolic debugger—supports both assembler (COFF) and C Compiler (.COD) SDK linked object formats.
- Instruction by instruction memory/register changes displayed when in single step operation.
- Processor specific symbolic display of registers and bit level assignments, configured from master model file.
- Halt/Idle mode notification. Restart requires special handling.
- Programming menu supports full product line of programmable OTP and EPROM COP8 products. Only a 40 ZIF socket is available on the EPU unit. Adapters are available for other part package configurations.
- Integral wall mount power supply provides 5V and develops the required V_{PP} to program parts.
- Includes a copy of COP8-DEV-IBMA assembler, linker SDK.

EPU Order Information

Evaluation Programming Unit	
EPU-COP888GG	Evaluation Programming Unit with debugger and programmer control software with 40 ZIF programming socket.
General Programming Adapters	
COP8-PGMA-DS44P	28 and 20 DIP and SOIC plus 44 PLCC adapter



TL/DD/12525–23

FIGURE 21. EPU-COP8 Tool Environment

Development Support (Continued)

COP8 ASSEMBLER/LINKER SOFTWARE DEVELOPMENT TOOL KIT

National Semiconductor offers a relocateable COP8 macro cross assembler, linker, librarian and utility software development tool kit. Features are summarized as follows:

- Basic and Feature Family instruction set by "device" type.
- Nested macro capability.
- Extensive set of assembler directives.
- Supported on PC/DOS platform.
- Generates National standard COFF output files.
- Integrated Linker and Librarian.
- Integrated utilities to generate ROM code file outputs.
- DUMPCOFF utility.

This product is integrated as a part of MetaLink tools as a development kit, fully supported by the MetaLink debugger. It may be ordered separately or it is bundled with the MetaLink products at no additional cost.

Order Information

Assembler SDK	
COP8-DEV-IBMA	Assembler SDK on installable 3.5" PC/DOS Floppy Disk Drive format. Periodic upgrades and most recent version is available on National's BBS and Internet.

COP8 C COMPILER

A C Compiler is developed and marketed by Byte Craft Limited. The COP8C compiler is a fully integrated development tool specifically designed to support the compact embedded configuration of the COP8 family of products.

Features are summarized as follows:

- ANSI C with some restrictions and extensions that optimize development for the COP8 embedded application.
- BITS data type extension. Register declaration #pragma with direct bit level definitions.
- C language support for interrupt routines.
- Expert system, rule based code generation and optimization.
- Performs consistency checks against the architectural definitions of the target COP8 device.
- Generates program memory code.
- Supports linking of compiled object or COP8 assembled object formats.
- Global optimization of linked code.
- Symbolic debug load format fully sourced level supported by the MetaLink debugger.

OTP/EMULATOR SUPPORT

The COP87L88EG/COP87L84EG devices provide emulation and OTP support for the COP888EG/COP884EG devices.

Approved List

Manufacturer	North America	Europe	Asia
BP Microsystems	(800) 225-2102 (713) 688-4600 Fax: (713) 688-0920	+49-8152-4183 +49-8856-932616	+852-234-16611 +852-2710-8121
Data I/O	(800) 426-1045 (206) 881-6444 Fax: (206) 882-1043	+44-0734-440011	Call North America
HI-LO	(510) 623-8860	Call Asia	+886-2-764-0215 Fax: +886-2-756-6403
ICE Technology	(800) 624-8949 (919) 430-7915	+44-1226-767404 Fax: 0-1226-370-434	
MetaLink	(800) 638-2423 (602) 926-0797 Fax: (602) 693-0681	+49-80 9156 96-0 Fax: +49-80 9123 86	+852-737-1800
Systems General	(408) 263-6667	+41-1-9450300	+886-2-917-3005 Fax: +886-2-911-1283
Needhams	(916) 924-8037 Fax: (916) 924-8065		

Development Support (Continued)

COP87L88EG/COP87L84EG Ordering Information

Device Number	Clock Option	Package	Emulates
COP87L88EGV-XE	Crystal/HALT En	44 PLCC	COP888EG COP888CG COP888CS
COP87L88EGN-XE	Crystal/HALT En	40 DIP	COP888EG COP888CG COP888CS
COP87L84EGN-XE	Crystal/HALT En	28 DIP	COP884EG COP884CG COP884CS
COP87L84EGM-CE	Crystal/HALT En	28 SO	COP884EG COP884CG COP884CS

INDUSTRY WIDE OTP/EPROM PROGRAMMING SUPPORT

Programming support, in addition to the MetaLink development tools, is provided by a full range of independent approved vendors to meet the needs from the engineering laboratory to full production.

AVAILABLE LITERATURE

For more information, please see the COP8 Basic Family User's Manual, Literature Number 620895, COP8 Feature Family User's Manual, Literature Number 620897 and National's Family of 8-bit Microcontrollers COP8 Selection Guide, Literature Number 630009.

DIAL-A-HELPER SERVICE

Dial-A-Helper is a service provided by the Microcontroller Applications group. The Dial-A-Helper is an Electronic Information System that may be accessed as a Bulletin Board System (BBS) via data modem, as an FTP site on the Internet via standard FTP client application or as an FTP site on the Internet using a standard Internet browser such as Netscape or Mosaic.

The Dial-A-Helper system provides access to an automated information storage and retrieval system . The system capabilities include a MESSAGE SECTION (electronic mail, when accessed as a BBS) for communications to and from the Microcontroller Applications Group and a FILE SECTION which consists of several file areas where valuable application software and utilities could be found.

DIAL-A-HELPER BBS via Standard Modem

Modem: CANADA/U.S.: (800) NSC-MICRO
(800) 672-6427
EUROPE: (+49) 0-8141-351332
Baud: 14.4k
Set-Up: Length: 8-Bit
Parity: None
Stop Bit: 1
Operation: 24 Hours, 7 Days

DIAL-A-HELPER via FTP

ftp nscmicro.nsc.com
user: anonymous
password: username@yourhost.site.domain

DIAL-A-HELPER via WorldWide Web Browser

ftp://nscmicro.nsc.com

National Semiconductor on the WorldWide Web

See us on the WorldWide Web at: http://www.natsemi.com

CUSTOMER RESPONSE CENTER

Complete product information and technical support is available from National's customer response centers.

CANADA/U.S.:	Tel:	(800)272-9959
	email:	support@tevm2.nsc.com
EUROPE:	email:	europe.support@nsc.com
	Deutsch Tel:	+49 (0) 180-530 85 85
	English Tel:	+49 (0) 180-532 78 32
	Français Tel:	+49 (0) 180-532 93 58
	Italiano Tel:	+49 (0) 180-534 16 80
JAPAN:	Tel:	+81-043-299-2309
S.E. ASIA:	Beijing Tel:	(+86) 10-6856-8601
	Shanghai Tel:	(+86) 21-6415-4092
	Hong Kong Tel:	(+852) 2737-1600
	Korea Tel:	(+82) 2-3771-6909
	Malaysia Tel:	(+60-4) 644-9061
	Singapore Tel:	(+65) 255-2226
	Taiwan Tel:	+886-2-521-3288
AUSTRALIA:	Tel:	(+61) 3-9558-9999
INDIA:	Tel:	(+91) 80-559-9467

National Semiconductor

PRELIMINARY

COP87L88FH/COP87L84FH 8-Bit Microcontrollers with UART, Three Multi-Function Timers and Multiply/Divide Block

General Description

The COP87L88FH/COP87L84FH OTP microcontrollers are members of the COP8™ feature family using an 8-bit core architecture. They are pin and software compatible to the mask ROM COP888FH product family. (Continued)

Key Features

- Multiply/Divide Functions
- Full duplex UART
- Three 16-bit timers, each with two 16-bit registers supporting:
  - — Processor Independent PWM mode
  - — External Event counter mode
  - — Input Capture mode
- 12 kbytes on-board OTP EPROM with security features
- 512 bytes on-board RAM

Additional Peripheral Features

- Idle Timer
- Multi-Input Wakeup (MIWU) with optional interrupts (8)
- Two analog comparators
- WATCHDOG™ and Clock Monitor logic
- MICROWIRE/PLUS™ serial I/O

I/O Features

- Memory mapped I/O
- Software selectable I/O options (TRI-STATE®, Push-Pull, Weak Pull-Up, and High Impedance)
- Schmitt trigger inputs on ports G and L
- Packages:
  - — 40 DIP with 36 I/O pins
  - — 44 PLCC with 40 I/O pins
  - — 28 SO with 24 I/O pins
  - — 28 DIP with 24 I/O pins

CPU/Instruction Set Features

- 1 μs instruction cycle time
- Fourteen multi-source vectored interrupts servicing
  - — External Interrupt
  - — Idle Timer T0
  - — Three Timers (Each with 2 Interrupts)
  - — MICROWIRE/PLUS
  - — Multi-Input Wake Up
  - — Software Trap
  - — UART (2)
  - — Default VIS (default interrupt)
- Versatile and easy to use instruction set
- 8-bit Stack Pointer (SP)—stack in RAM
- Two 8-bit Register Indirect Data Memory Pointers (B and X)

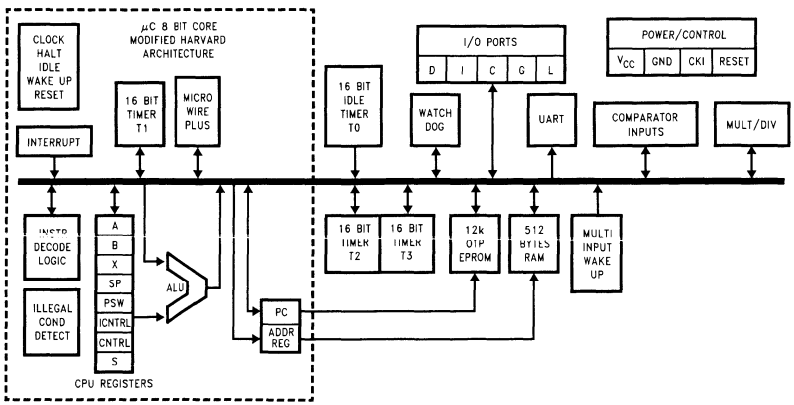
Fully Static CMOS

- Low current drain (typically < 5 μA)
- Two power saving modes: HALT and IDLE
- Single supply operation: 2.7V–5.5V
- Temperature range: −40°C to +85°C

Development Support

- Emulation device for COP888FH
- Real time emulation and full program debug offered by MetaLink Development System

Block Diagram



TL/DD/12863–1

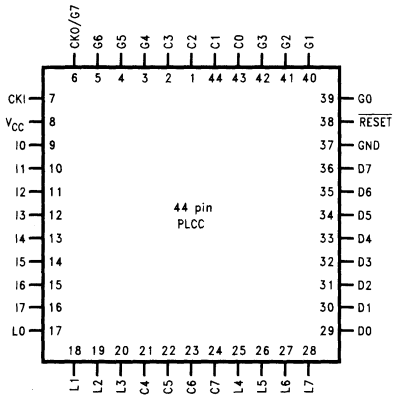
FIGURE 1. COP888FH Block Diagram

General Description (Continued)

They are fully static parts, fabricated using double-metal silicon gate microCMOS technology. Features include an 8-bit memory mapped architecture, MICROWIRE/PLUS serial I/O, three 16-bit timer/counters supporting three modes (Processor Independent PWM generation, External Event counter, and Input Capture mode capabilities), full duplex UART, two comparators, and two power savings modes (HALT and IDLE), both with a multi-sourced wakeup/interrupt capability. This multi-sourced interrupt capability may also be used independent of the HALT or IDLE modes. Each I/O pin has software selectable configurations. The device operates over a voltage range of 2.5V to 5.5V. High throughput is achieved with an efficient, regular instruction set operating at a maximum of 1 μs per instruction rate.

Connection Diagrams

Plastic Chip Carrier

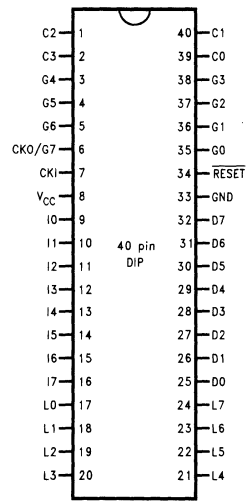


TL/DD/12863-2

Top View

Order Number COP87L88FHV-XE
See NS Plastic Chip Package Number V44A

Dual-In-Line Package

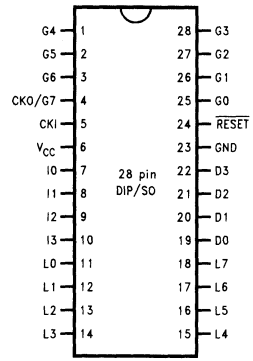


TL/DD/12863-3

Top View

Order Number COP87L88FHN-XE
See NS Molded Package Number N40A

Dual-In-Line Package



TL/DD/12863-4

Top View

Order Number COP87L84FHM-XE or COP87L84FHN-XE
See NS Molded Package Number M28B or N28B

Note: -X Crystal Oscillator
-E Halt Enable

FIGURE 2. Connection Diagrams

Connection Diagrams (Continued)

Pinouts for 28-, 40- and 44-Pin Packages

Port	Type	Alt. Fun	Alt. Fun	28-Pin SO/DIP	40-Pin DIP	44-Pin PLCC
L0	I/O	MIWU		11	17	17
L1	I/O	MIWU	CKX	12	18	18
L2	I/O	MIWU	TDX	13	19	19
L3	I/O	MIWU	RDX	14	20	20
L4	I/O	MIWU	T2A	15	21	25
L5	I/O	MIWU	T2B	16	22	26
L6	I/O	MIWU	T3A	17	23	27
L7	I/O	MIWU	T3B	18	24	28
G0	I/O	INT		25	35	39
G1	WDOUT			26	36	40
G2	I/O	T1B		27	37	41
G3	I/O	T1A		28	38	42
G4	I/O	SO		1	3	3
G5	I/O	SK		2	4	4
G6	I	SI		3	5	5
G7	I/CKO	HALT Restart		4	6	6
D0	O			19	25	29
D1	O			20	26	30
D2	O			21	27	31
D3	O			22	28	32
D4	O				29	33
D5	O				30	34
D6	O				31	35
D7	O				32	36
I0	I			7	9	9
I1	I	COMP1IN−		8	10	10
I2	I	COMP1IN+		9	11	11
I3	I	COMP1OUT		10	12	12
I4	I	COMP2IN−			13	13
I5	I	COMP2IN+			14	14
I6	I	COMP2OUT			15	15
I7	I				16	16
C0	I/O				39	43
C1	I/O				40	44
C2	I/O				1	1
C3	I/O				2	2
C4	I/O					21
C5	I/O					22
C6	I/O					23
C7	I/O					24
V_{CC}				6	8	8
GND				23	33	37
CKI				5	7	7
$\overline{\text{RESET}}$				24	34	38

Absolute Maximum Ratings

If Military/Aerospace specified devices are required, please contact the National Semiconductor Sales Office/Distributors for availability and specifications.

Supply Voltage (V_{CC})	7V
Voltage at Any Pin	−0.3V to V_{CC} + 0.3V
Total Current into V_{CC} Pin (Source)	100 mA
Total Current out of GND Pin (Sink)	110 mA
Storage Temperature Range	−65°C to +140°C

Note: *Absolute maximum ratings indicate limits beyond which damage to the device may occur. DC and AC electrical specifications are not ensured when operating the device at absolute maximum ratings.*

DC Electrical Characteristics $-40°C \le T_A \le +85°C$ unless otherwise specified

Parameter	Conditions	Min	Typ	Max	Units
Operating Voltage		2.7		5.5	V
Power Supply Ripple (Note 1)	Peak-to-Peak			0.1 V_{CC}	V
Supply Current (Note 2)					
CKI = 10 MHz	V_{CC} = 5.5V, t_c = 1 μs			12.5	mA
CKI = 4 MHz	V_{CC} = 5.5V, t_c = 2.5 μs			5.5	mA
HALT Current (Note 3)	V_{CC} = 5.5V, CKI = 0 MHz		<5	10	μA
	V_{CC} = 4.0V, CKI = 0 MHz		<3	6	μA
IDLE Current					
CKI = 10 MHz	V_{CC} = 5.5V, t_c = 1 μs			3.5	mA
CKI = 4 MHz	V_{CC} = 5.5V, t_c = 2.5 μs			2.5	mA
Input Levels					
$\overline{\text{RESET}}$					
Logic High		0.8 V_{CC}			V
Logic Low				0.2 V_{CC}	V
CKI (All Other Inputs)					
Logic High		0.7 V_{CC}			V
Logic Low				0.2 V_{CC}	V
Hi-Z Input Leakage	V_{CC} = 5.5V, V_{IN} = 0V	−2		+2	μA
Input Pullup Current	V_{CC} = 5.5V, V_{IN} = 0V	−40		−250	μA
G and L Port Input Hysteresis	(Note 5)			0.35 V_{CC}	V
Output Current Levels					
D Outputs					
Source	V_{CC} = 4.5V, V_{OH} = 3.3V	−0.4			mA
	V_{CC} = 2.7V, V_{OH} = 1.8V	−0.2			mA
Sink	V_{CC} = 4.5V, V_{OL} = 1V	10			mA
	V_{CC} = 2.7V, V_{OL} = 0.4V	2.0			mA
All Others					
Source (Weak Pull-Up Mode)	V_{CC} = 4.5V, V_{OH} = 2.7V	−10		−100	μA
	V_{CC} = 2.7V, V_{OH} = 1.8V	−2.5		−33	μA
Source (Push-Pull Mode)	V_{CC} = 4.5V, V_{OH} = 3.3V	−0.4			mA
	V_{CC} = 2.7V, V_{OH} = 1.8V	−0.2			mA
Sink (Push-Pull Mode)	V_{CC} = 4.5V, V_{OL} = 0.4V	1.6			mA
	V_{CC} = 2.7V, V_{OL} = 0.4V	0.7			mA
TRI-STATE Leakage	V_{CC} = 5.5V	−2		+2	μA
Allowable Sink/Source Current per Pin					
D Outputs (Sink)				15	mA
All others				3	mA
Maximum Input Current without Latchup (Notes 4, 5)	Room Temp			±200	mA
RAM Retention Voltage, V_r	500 ns Rise and Fall Time (Min)	2			V
Input Capacitance	(Note 5)			7	pF
Load Capacitance on D2	(Note 5)			1000	pF

AC Electrical Characteristics $-40°C \leq T_A \leq +85°C$ unless otherwise specified

Parameter	Conditions	Min	Typ	Max	Units
Instruction Cycle Time (t_c)					
Crystal Resonator or External	$2.7V \leq V_{CC} \leq 4.5V$	2.5		DC	μs
	$4.5V \leq V_{CC} \leq 5.5V$	1.0		DC	μs
R/C Oscillator	$2.7V \leq V_{CC} < 4.5V$	7.5		DC	μs
	$4.5V \leq V_{CC} \leq 5.5V$	3.0		DC	μs
Inputs					
t_{SETUP}	$4.5V \leq V_{CC} \leq 5.5V$	200			ns
	$2.7V \leq V_{CC} < 4.5V$	500			ns
t_{HOLD}	$4.5V \leq V_{CC} \leq 5.5V$	60			ns
	$2.7V \leq V_{CC} < 4.5V$	150			ns
Output Propagation Delay	$R_L = 2.2k, C_L = 100$ pF				
t_{PD1}, t_{PD0}					
SO, SK	$4.5V \leq V_{CC} \leq 5.5V$			0.7	μs
	$2.7V \leq V_{CC} < 4.5V$			1.75	μs
All Others	$4.5V \leq V_{CC} \leq 5.5V$			1	μs
	$2.7V \leq V_{CC} < 4.5V$			2.5	μs
MICROWIRE Setup Time (t_{UWS}) (Note 5)	$V_{CC} \geq 4.5V$	20			ns
MICROWIRE Hold Time (t_{UWH}) (Note 5)	$V_{CC} \geq 4.5V$	56			ns
MICROWIRE Output Propagation Delay (t_{UPD})	$V_{CC} \geq 4.5V$			220	ns
Input Pulse Width (Note 6)					
Interrupt Input High Time		1			t_c
Interrupt Input Low Time		1			t_c
Timer 1, 2, 3 Input High Time		1			t_c
Timer 1, 2, 3 Input Low Time		1			t_c
Reset Pulse Width		1			μs

Note 1: Maximum rate of voltage change must be less than 0.5V/ms.

Note 2: Supply current is measured after running 2000 cycles with a square wave CKI input, CKO open, inputs at rails and outputs open.

Note 3: The HALT mode will stop CKI from oscillating in the RC and the Crystal configurations. Measurement of I_{DD} HALT is done with device neither sourcing or sinking current; with L, C, and G0–G5 programmed as low outputs and not driving a load; all outputs programmed low and not driving a load; all inputs tied to V_{CC}; clock monitor and comparators disabled. Parameter refers to HALT mode entered via setting bit 7 of the G Port data register. Part will pull up CKI during HALT in crystal clock mode.

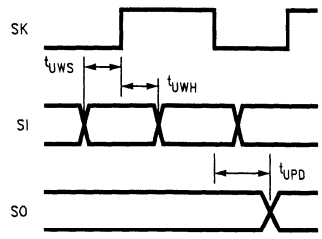
Note 4: Pins G6 and $\overline{\text{RESET}}$ are designed with a high voltage input network. These pins allow input voltages greater than V_{CC} and the pins will have sink current to V_{CC} when biased at voltages greater than V_{CC} (the pins do not have source current when biased at a voltage below V_{CC}). The effective resistance to V_{CC} is 750Ω (typical). These two pins will not latch up. The voltage at the pins must be limited to less than 14V. **WARNING: Voltages in excess of 14V will cause damage to the pins. This warning excludes ESD transients.**

Note 5: Parameter characterized but not tested.

Note 6: t_c = Instruction cycle time.

Comparators AC and DC Characteristics $V_{CC} = 5V, -40°C \leq T_A \leq +85°C$

Parameter	Conditions	Min	Typ	Max	Units
Input Offset Voltage	$0.4V \leq V_{IN} \leq V_{CC} - 1.5V$		±10	±25	mV
Input Common Mode Voltage Range		0.4		$V_{CC} - 1.5$	V
Low Level Output Current	$V_{OL} = 0.4V$	1.6			mA
High Level Output Current	$V_{OH} = 4.6V$	1.6			mA
DC Supply Current Per Comparator (When Enabled)				250	μA
Response Time	100 mV Overdrive, 100 pF Load			1	μs



TL/DD/12863–5

FIGURE 3. MICROWIRE/PLUS Timing

Pin Descriptions

V_{CC} and GND are the power supply pins. All V_{CC} and GND pins must be connected.

CKI is the clock input. This can come from an R/C generated oscillator, or a crystal oscillator (in conjunction with CKO). See Oscillator Description section.

$\overline{\text{RESET}}$ is the master reset input. See Reset Description section.

The device contains three bidirectional 8-bit I/O ports (C, G and L), where each individual bit may be independently configured as an input (Schmitt trigger inputs on ports L and G), output or TRI-STATE under program control. Three data memory address locations are allocated for each of these I/O ports. Each I/O port has two associated 8-bit memory mapped registers, the CONFIGURATION register and the output DATA register. A memory mapped address is also reserved for the input pins of each I/O port. (See the memory map for the various addresses associated with the I/O ports.) *Figure 4* shows the I/O port configurations. The DATA and CONFIGURATION registers allow for each port bit to be individually configured under software control as shown below:

CONFIGURATION Register	DATA Register	Port Set-Up
0	0	Hi-Z Input (TRI-STATE Output)
0	1	Input with Weak Pull-Up
1	0	Push-Pull Zero Output
1	1	Push-Pull One Output

PORT L is an 8-bit I/O port. All L-pins have Schmitt triggers on the inputs.

The Port L supports Multi-Input Wake Up on all eight pins. L1 is used for the UART external clock. L2 and L3 are used for the UART transmit and receive. L4 and L5 are used for the timer input functions T2A and T2B. L6 and L7 are used for the timer input functions T3A and T3B.

The Port L has the following alternate features:

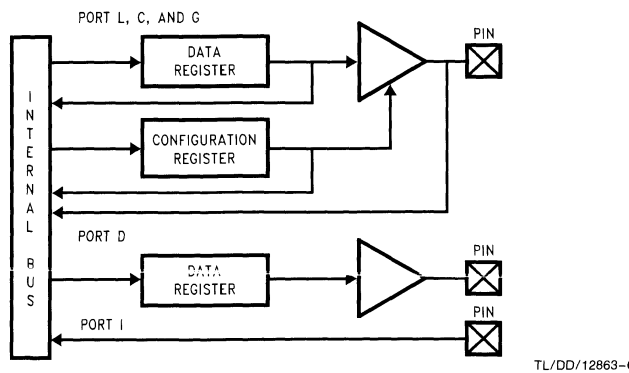
- L0 MIWU
- L1 MIWU or CKX
- L2 MIWU or TDX
- L3 MIWU or RDX
- L4 MIWU or T2A
- L5 MIWU or T2B
- L6 MIWU or T3A
- L7 MIWU or T3B

Port G is an 8-bit port with 5 I/O pins (G0, G2–G5), an input pin (G6), and two dedicated output pins (G1 and G7). Pins G0 and G2–G6 all have Schmitt Triggers on their inputs. Pin G1 serves as the dedicated WDOUT WATCHDOG output, while pin G7 is either input or output depending on the oscillator mask option selected. With the crystal oscillator option selected, G7 serves as the dedicated output pin for the CKO clock output. With the single-pin R/C oscillator mask option selected, G7 serves as a general purpose input pin but is also used to bring the device out of HALT mode with a low to high transition on G7. There are two registers associated with the G Port, a data register and a configuration register. Therefore, each of the 5 I/O bits (G0, G2–G5) can be individually configured under software control.

Since G6 is an input only pin and G7 is the dedicated CKO clock output pin (crystal clock option) or general purpose input (R/C clock option), the associated bits in the data and configuration registers for G6 and G7 are used for special purpose functions as outlined below. Reading the G6 and G7 data bits will return zeros.

Note that the chip will be placed in the HALT mode by writing a "1" to bit 7 of the Port G Data Register. Similarly the chip will be placed in the IDLE mode by writing a "1" to bit 6 of the Port G Data Register.

Writing a "1" to bit 6 of the Port G Configuration Register enables the MICROWIRE/PLUS to operate with the alternate phase of the SK clock. The G7 configuration bit, if set high, enables the clock start up delay after HALT when the R/C clock configuration is used.



TL/DD/12863–6

FIGURE 4. I/O Port Configurations

Pin Descriptions (Continued)

	Config Reg.	Data Reg.
G7	CLKDLY	HALT
G6	Alternate SK	IDLE

Port G has the following alternate features:

- G0 INTR (External Interrupt Input)
- G2 T1B (Timer T1 Capture Input)
- G3 T1A (Timer T1 I/O)
- G4 SO (MICROWIRE™ Serial Data Output)
- G5 SK (MICROWIRE Serial Clock)
- G6 SI (MICROWIRE Serial Data Input)

Port G has the following dedicated functions:

- G1 WDOUT WATCHDOG and/or Clock Monitor dedicated output
- G7 CKO Oscillator dedicated output or general purpose input

Port C is an 8-bit I/O port. The 40-pin device does not have a full complement of Port C pins. The unavailable pins are not terminated. A read operation for these unterminated pins will return unpredicatable values.

PORT I is an eight-bit Hi-Z input port. The 28-pin device does not have a full complement of Port I pins. The unavailable pins are not terminated i.e., they are floating. A read operation for these unterminated pins will return unpredictable values. The user must ensure that the software takes this into account by either masking or restricting the accesses to bit operations. The unterminated Port I pins will draw power only when addressed.

Port I1–I3 are used for Comparator 1. Port I4–I6 are used for Comparator 2.

The Port I has the following alternate features.

- I1 COMP1 − IN (Comparator 1 Negative Input)
- I2 COMP1 + IN (Comparator 1 Positive Input)
- I3 COMP1OUT (Comparator 1 Output)
- I4 COMP2 − IN (Comparator 2 Negative Input)
- I5 COMP2 + IN (Comparator 2 Positive Input)
- I6 COMP2OUT (Comparator 2 Output)

Port D is an 8-bit output port that is preset high when $\overline{\text{RESET}}$ goes low. The user can tie two or more D port outputs (except D2) together in order to get a higher drive.

Note: Care must be exercised with the D2 pin operation. At RESET, the external loads on this pin must ensure that the output voltages stay above 0.8 V_{CC} to prevent the chip from entering special modes. Also keep the external loading on D2 to less than 1000 pF.

Functional Description

The architecture of the device is modified Harvard architecture. With the Harvard architecture, the control store program memory (ROM) is separated from the data store memory (RAM). Both ROM and RAM have their own separate addressing space with separate address buses. The architecture, though based on Harvard architecture, permits transfer of data from ROM to RAM.

CPU REGISTERS

The CPU can do an 8-bit addition, subtraction, logical or shift operation in one instruction (t_C) cycle time.

There are six CPU registers:

A is the 8-bit Accumulator Register

PC is the 15-bit Program Counter Register

PU is the upper 7 bits of the program counter (PC)
PL is the lower 8 bits of the program counter (PC)

B is an 8-bit RAM address pointer, which can be optionally post auto incremented or decremented.

X is an 8-bit alternate RAM address pointer, which can be optionally post auto incremented or decremented.

SP is the 8-bit stack pointer, which points to the subroutine/interrupt stack (in RAM). The SP is initialized to RAM address 06F with reset.

S is the 8-bit Data Segment Address Register used to extend the lower half of the address range (00 to 7F) into 256 data segments of 128 bytes each.

All the CPU registers are memory mapped with the exception of the Accumulator (A) and the Program Counter (PC).

PROGRAM MEMORY

The program memory consists of 12288 bytes of OTP EPROM. These bytes may hold program instructions or constant data (data tables for the LAID instruction, jump vectors for the JID instruction, and interrupt vectors for the VIS instruction). The program memory is addressed by the 15-bit program counter (PC). All interrupts in the devices vector to program memory location 0FF Hex.

The device can be configured to inhibit external reads of the program memory. This is done by programming the Security Byte.

SECURITY FEATURE

The program memory array has an associate Security Byte that is located outside of the program address range. This byte can be addressed only from programming mode by a programmer tool.

Security is an optional feature and can only be asserted after the memory array has been programmed and verified. A secured part will read all 00(hex) by a programmer. The part will fail Blank Check and will fail Verify operations. A Read operation will fill the programmer's memory with 00(hex). The Security Byte itself is always readable with value of 00(hex) if unsecure and FF(hex) if secure.

DATA MEMORY

The data memory address space includes the on-chip RAM and data registers, the I/O registers (Configuration, Data and Pin), the control registers, the MICROWIRE/PLUS SIO shift register, and the various registers, and counters associated with the timers (with the exception of the IDLE timer). Data memory is addressed directly by the instruction or indirectly by the B, X, SP pointers and S register.

The data memory consists of 512 bytes of RAM. Sixteen bytes of RAM are mapped as "registers" at addresses 0F0 to 0FF Hex. These registers can be loaded immediately, and also decremented and tested with the DRSZ (decrement register and skip if zero) instruction. The memory pointer registers X, SP, B and S are memory mapped into this space at address locations 0FC to 0FF Hex respectively, with the other registers being available for general usage.

The instruction set permits any bit in memory to be set, reset or tested. All I/O and registers (except A and PC) are memory mapped; therefore, I/O bits and register bits can be directly and individually set, reset and tested. The accumulator (A) bits can also be directly and individually tested.

Note: RAM contents are undefined upon power-up.

Data Memory Segment RAM Extension

Data memory address 0FF is used as a memory mapped location for the Data Segment Address Register (S).

The data store memory is either addressed directly by a single byte address within the instruction, or indirectly relative to the reference of the B, X, or SP pointers (each contains a single-byte address). This single-byte address allows an addressing range of 256 locations from 00 to FF hex. The upper bit of this single-byte address divides the data store memory into two separate sections as outlined previously. With the exception of the RAM register memory from address locations 00F0 to 00FF, all RAM memory is memory mapped with the upper bit of the single-byte address being equal to zero. This allows the upper bit of the single-byte address to determine whether or not the base address range (from 0000 to 00FF) is extended. If this upper bit equals one (representing address range 0080 to 00FF), then address extension does not take place. Alternatively, if this upper bit equals zero, then the data segment extension register S is used to extend the base address range (from 0000 to 007F) from XX00 to XX7F, where XX represents the 8 bits from the S register. Thus the 128-byte data segment extensions are located from addresses 0100 to 017F for data segment 1, 0200 to 027F for data segment 2, etc., up to FF00 to FF7F for data segment 255. The base address range from 0000 to 007F represents data segment 0.

Figure 5 illustrates how the S register data memory extension is used in extending the lower half of the base address range (00 to 7F hex) into 256 data segments of 128 bytes each, with a total addressing range of 32 kbytes from XX00 to XX7F. This organization allows a total of 256 data segments of 128 bytes each with an additional upper base segment of 128 bytes. Furthermore, all addressing modes are available for all data segments. The S register must be changed under program control to move from one data segment (128 bytes) to another. However, the upper base segment (containing the 16 memory registers, I/O registers, control registers, etc.) is always available regardless of the contents of the S register, since the upper base segment (address range 0080 to 00FF) is independent of data segment extension.

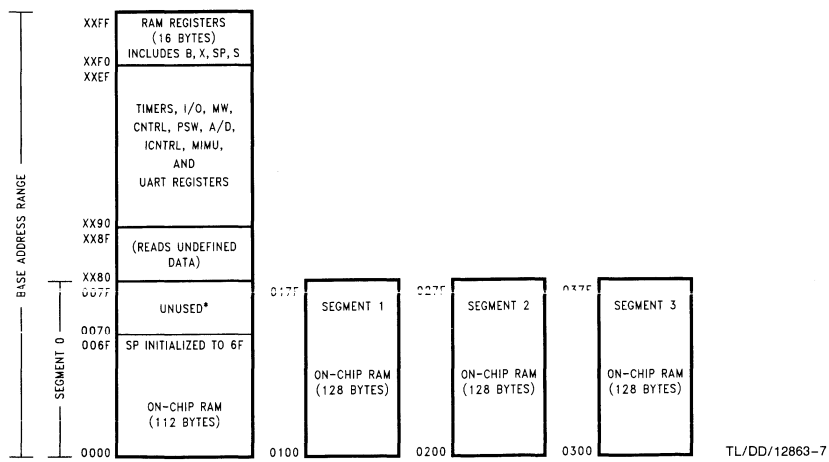
The instructions that utilize the stack pointer (SP) always reference the stack as part of the base segment (Segment 0), regardless of the contents of the S register. The S register is not changed by these instructions. Consequently, the stack (used with subroutine linkage and interrupts) is always located in the base segment. The stack pointer will be initialized to point at data memory location 006F as a result of reset.

The 128 bytes of RAM contained in the base segment are split between the lower and upper base segments. The first 112 bytes of RAM are resident from address 0000 to 006F in the lower base segment, while the remaining 16 bytes of RAM represent the 16 data memory registers located at addresses 00F0 to 00FF of the upper base segment. No RAM is located at the upper sixteen addresses (0070 to 007F) of the lower base segment.

Additional RAM beyond these initial 128 bytes, however, will always be memory mapped in groups of 128 bytes (or less) at the data segment address extensions (XX00 to XX7F) of the lower base segment. The additional 128 bytes of RAM are memory mapped at address locations 0100 to 017F hex.

Reset

The $\overline{\text{RESET}}$ input when pulled low initializes the microcontroller. Initialization will occur whenever the $\overline{\text{RESET}}$ input is pulled low. Upon initialization, the data and configuration registers for ports L, G and C are cleared, resulting in these Ports being initialized to the TRI-STATE mode. Pin G1 of the G Port is an exception (as noted below) since pin G1 is dedicated as the WATCHDOG and/or Clock Monitor error output pin. Port D is set high. The PC, PSW, ICNTRL, CNTRL, T2CNTRL and T3CNTRL control registers are cleared. The UART registers PSR, ENU (except that TBMT bit is set), ENUR and ENUI are cleared. The Comparator Select Register is cleared. The S register is initialized to zero. The Multi-Input Wakeup registers WKEN, WKEDG and WKPND are cleared. (Wakeup register WKPND is unknown.) The stack pointer, SP, is initialized to 6F Hex.



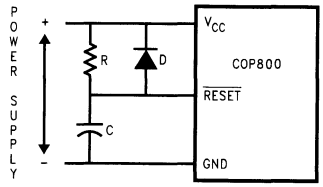
*Reads as all ones.

FIGURE 5. RAM Organization

Reset (Continued)

The device comes out of reset with both the WATCHDOG logic and the Clock Monitor detector armed, with the WATCHDOG service window bits set and the Clock Monitor bit set. The WATCHDOG and Clock Monitor circuits are inhibited during reset. The WATCHDOG service window bits being initialized high default to the maximum WATCHDOG service window of 64k t_C clock cycles. The Clock Monitor bit being initialized high will cause a Clock Monitor error following reset if the clock has not reached the minimum specified frequency at the termination of reset. A Clock Monitor error will cause an active low error output on pin G1. This error output will continue until 16 t_C–32 t_C clock cycles following the clock frequency reaching the minimum specified value, at which time the G1 output will enter the TRI-STATE mode.

The external RC network shown in *Figure 6* should be used to ensure that the $\overline{\text{RESET}}$ pin is held low until the power supply to the chip stabilizes.



TL/DD/12863–8

RC > 5 × Power Supply Rise Time

FIGURE 6. Recommended Reset Circuit

Oscillator Circuits

The chip can be driven by a clock input on the CKI input pin which can be between DC and 10 MHz. The CKO output clock is on pin G7 (crystal configuration). The CKI input frequency is divided down by 10 to produce the instruction cycle clock (t_c).

Figure 7 shows the Crystal and R/C oscillator diagrams.

CRYSTAL OSCILLATOR

CKI and CKO can be connected to make a closed loop crystal (or resonator) controlled oscillator.

Table A shows the component values required for various standard crystal values.

R/C OSCILLATOR

By selecting CKI as a single pin oscillator input, a single pin R/C oscillator circuit can be connected to it. CKO is available as a general purpose input, and/or HALT restart input.

Table B shows the variation in the oscillator frequencies as functions of the component (R and C) values.

EXTERNAL OSCILLATOR

CKI can be driven by an external clock signal. CKO is available as a general purpose input and/or HALT restart control.

TABLE A. Crystal Oscillator Configuration, T_A = 25°C

R1 (kΩ)	R2 (MΩ)	C1 (pF)	C2 (pF)	CKI Freq (MHz)	Conditions
0	1	30	30–36	10	V_{CC} = 5V
0	1	30	30–36	4	V_{CC} = 5V
0	1	200	100–150	0.455	V_{CC} = 5V

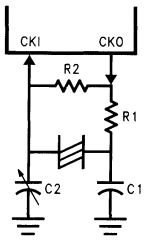
TABLE B. RC Oscillator Configuration, T_A = 25°C

R (kΩ)	C (pF)	CKI Freq (MHz)	Instr. Cycle (μs)	Conditions
3.3	82	2.2 to 2.7	3.7 to 4.6	V_{CC} = 5V
5.6	100	1.1 to 1.3	7.4 to 9.0	V_{CC} = 5V
6.8	100	0.9 to 1.1	8.8 to 10.8	V_{CC} = 5V

Note: 3k ≤ R ≤ 200k

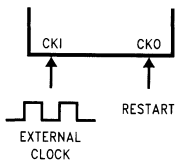
50 pF ≤ C ≤ 200 pF

Crystal Oscillator



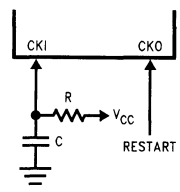
TL/DD/12863–9

External Oscillator



TL/DD/12863–10

R/C Oscillator



TL/DD/12863–11

FIGURE 7. Crystal, R/C and External Oscillator Diagrams

Control Registers

CNTRL Register (Address X'00EE)

The Timer1 (T1) and MICROWIRE/PLUS control register contains the following bits:

- SL1 & SL0 Select the MICROWIRE/PLUS clock divide by (00 = 2, 01 = 4, 1x = 8)
- IEDG External interrupt edge polarity select (0 = Rising edge, 1 = Falling edge)
- MSEL Selects G5 and G4 as MICROWIRE/PLUS signals SK and SO respectively
- T1C0 Timer T1 Start/Stop control in timer modes 1 and 2
  Timer T1 Underflow Interrupt Pending Flag in timer mode 3
- T1C1 Timer T1 mode control bit
- T1C2 Timer T1 mode control bit
- T1C3 Timer T1 mode control bit

T1C3	T1C2	T1C1	T1C0	MSEL	IEDG	SL1	SL0

Bit 7 Bit 0

PSW Register (Address X'00EF)

The PSW register contains the following select bits:

- GIE Global interrupt enable (enables interrupts)
- EXEN Enable external interrupt
- BUSY MICROWIRE/PLUS busy shifting flag
- EXPND External interrupt pending
- T1ENA Timer T1 Interrupt Enable for Timer Underflow or T1A Input capture edge
- T1PNDA Timer T1 Interrupt Pending Flag (Autoreload RA in mode 1, T1 Underflow in Mode 2, T1A capture edge in mode 3)
- C Carry Flag
- HC Half Carry Flag

HC	C	T1PNDA	T1ENA	EXPND	BUSY	EXEN	GIE

Bit 7 Bit 0

The Half-Carry bit is also affected by all the instructions that affect the Carry flag. The SC (Set Carry) and RC (Reset Carry) instructions will respectively set or clear both the carry flags. In addition to the SC and RC instructions, ADC, SUBC, RRC and RLC instructions affect the carry and Half Carry flags.

ICNTRL Register (Address X'00E8)

The ICNTRL register contains the following bits:

- T1ENB Timer T1 Interrupt Enable for T1B Input capture edge
- T1PNDB Timer T1 Interrupt Pending Flag for T1B capture edge
- μWEN Enable MICROWIRE/PLUS interrupt
- μWPND MICROWIRE/PLUS interrupt pending
- T0EN Timer T0 Interrupt Enable (Bit 12 toggle)
- T0PND Timer T0 Interrupt pending
- LPEN L Port Interrupt Enable (Multi-Input Wakeup/Interrupt)
  Bit 7 could be used as a flag

Unused	LPEN	T0PND	T0EN	μWPND	μWEN	T1PNDB	T1ENB

Bit 7 Bit 0

T2CNTRL Register (Address X'00C6)

The T2CNTRL register contains the following bits:

- T2ENB Timer T2 Interrupt Enable for T2B Input capture edge
- T2PNDB Timer T2 Interrupt Pending Flag for T2B capture edge
- T2ENA Timer T2 Interrupt Enable for Timer Underflow or T2A Input capture edge
- T2PNDA Timer T2 Interrupt Pending Flag (Autoreload RA in mode 1, T2 Underflow in mode 2, T2A capture edge in mode 3)
- T2C0 Timer T2 Start/Stop control in timer modes 1 and 2 Timer T2 Underflow Interrupt Pending Flag in timer mode 3
- T2C1 Timer T2 mode control bit
- T2C2 Timer T2 mode control bit
- T2C3 Timer T2 mode control bit

T2C3	T2C2	T2C1	T2C0	T2PNDA	T2ENA	T2PNDB	T2ENB

Bit 7 Bit 0

T3CNTRL Register (Address X'00B6)

The T3CNTRL register contains the following bits:

- T3ENB Timer T3 Interrupt Enable for T3B Input capture edge
- T3PNDB Timer T3 Interrupt Pending Flag for T3B pin (T3B capture edge)
- T3ENA Timer T3 Interrupt Enable for Timer Underflow or T3A Input capture edge
- T3PNDA Timer T3 Interrupt Pending Flag (Autoload RA in mode 1, T3 Underflow in mode 2, T3A capture edge in mode 3)
- T3C0 Timer T3 Start/Stop control in timer modes 1 and 2
  Timer T3 Underflow Interrupt Pending Flag in timer mode 3
- T3C1 Timer T3 mode control bit
- T3C2 Timer T3 mode control bit
- T3C3 Timer T3 mode control bit

T3C3	T3C2	T3C1	T3C0	T3PNDA	T3ENA	T3PNDB	T3ENB

Bit 7 Bit 0

Timers

The device contains a very versatile set of timers (T0, T1, T2, T3). All timers and associated autoreload/capture registers power up containing random data.

TIMER T0 (IDLE TIMER)

The device supports applications that require maintaining real time and low power with the IDLE mode. This IDLE

Timers (Continued)

mode support is furnished by the IDLE timer T0, which is a 16-bit timer. The Timer T0 runs continuously at the fixed rate of the instruction cycle clock, t_C. The user cannot read or write to the IDLE Timer T0, which is a count down timer.

The Timer T0 supports the following functions:

- Exit out of the Idle Mode (See Idle Mode description)
- WATCHDOG logic (See WATCHDOG description)
- Start up delay out of the HALT mode

The IDLE Timer T0 can generate an interrupt when the thirteenth bit toggles. This toggle is latched into the T0PND pending flag, and will occur every 4 ms at the maximum clock frequency (t_C = 1 μs). A control flag T0EN allows the interrupt from the thirteenth bit of Timer T0 to be enabled or disabled. Setting T0EN will enable the interrupt, while resetting it will disable the interrupt.

TIMER T1, TIMER T2 AND TIMER T3

The device has a set of three powerful timer/counter blocks, T1, T2 and T3. The associated features and functioning of a timer block are described by referring to the timer block Tx. Since the three timer blocks, T1, T2 and T3 are identical, all comments are equally applicable to any of the three timer blocks.

Each timer block consists of a 16-bit timer, Tx, and two supporting 16-bit autoreload/capture registers, RxA and RxB. Each timer block has two pins associated with it, TxA and TxB. The pin TxA supports I/O required by the timer block, while the pin TxB is an input to the timer block. The powerful and flexible timer block allows the device to easily perform all timer functions with minimal software overhead. The timer block has three operating modes: Processor Independent PWM mode, External Event Counter mode, and Input Capture mode.

The control bits TxC3, TxC2, and TxC1 allow selection of the different modes of operation.

Mode 1. Processor Independent PWM Mode

As the name suggests, this mode allows the device to generate a PWM signal with very minimal user intervention. The user only has to define the parameters of the PWM signal (ON time and OFF time). Once begun, the timer block will continuously generate the PWM signal completely independent of the microcontroller. The user software services the timer block only when the PWM parameters require updating.

In this mode the timer Tx counts down at a fixed rate of t_C. Upon every underflow the timer is alternately reloaded with the contents of supporting registers, RxA and RxB. The very first underflow of the timer causes the timer to reload from the register RxA. Subsequent underflows cause the timer to be reloaded from the registers alternately beginning with the register RxB.

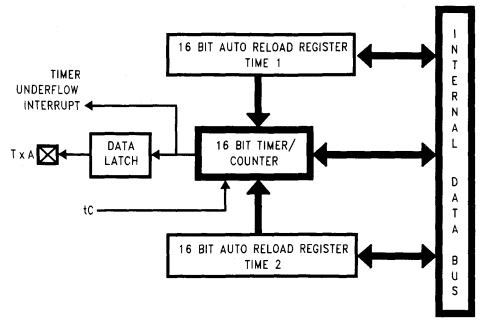
The Tx Timer control bits, TxC3, TxC2 and TxC1 set up the timer for PWM mode operation.

Figure 8 shows a block diagram of the timer in PWM mode.

The underflows can be programmed to toggle the TxA output pin. The underflows can also be programmed to generate interrupts.

Underflows from the timer are alternately latched into two pending flags, TxPNDA and TxPNDB. The user must reset these pending flags under software control. Two control enable flags, TxENA and TxENB, allow the interrupts from the timer underflow to be enabled or disabled. Setting the timer enable flag TxENA will cause an interrupt when a timer underflow causes the RxA register to be reloaded into the timer. Setting the timer enable flag TxENB will cause an interrupt when a timer underflow causes the RxB register to be reloaded into the timer. Resetting the timer enable flags will disable the associated interrupts.

Either or both of the timer underflow interrupts may be enabled. This gives the user the flexibility of interrupting once per PWM period on either the rising or falling edge of the PWM output. Alternatively, the user may choose to interrupt on both edges of the PWM output.



TL/DD/12863–12

FIGURE 8. Timer in PWM Mode

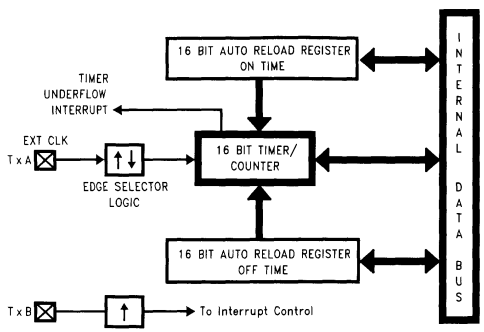
Mode 2. External Event Counter Mode

This mode is quite similar to the processor independent PWM mode described above. The main difference is that the timer, Tx, is clocked by the input signal from the TxA pin. The Tx timer control bits, TxC3, TxC2 and TxC1 allow the timer to be clocked either on a positive or negative edge from the TxA pin. Underflows from the timer are latched into the TxPNDA pending flag. Setting the TxENA control flag will cause an interrupt when the timer underflows.

In this mode the input pin TxB can be used as an independent positive edge sensitive interrupt input if the TxENB control flag is set. The occurrence of a positive edge on the TxB input pin is latched into the TxPNDB flag.

Figure 9 shows a block diagram of the timer in External Event Counter mode.

Note: The PWM output is not available in this mode since the TxA pin is being used as the counter input clock.



TL/DD/12863–13

FIGURE 9. Timer in External Event Counter Mode

Timers (Continued)

Mode 3. Input Capture Mode

The device can precisely measure external frequencies or time external events by placing the timer block, Tx, in the input capture mode.

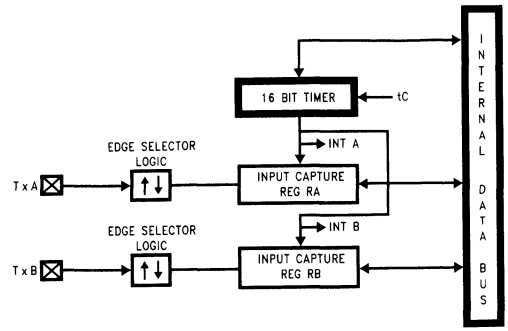
In this mode, the timer Tx is constantly running at the fixed t_C rate. The two registers, RxA and RxB, act as capture registers. Each register acts in conjunction with a pin. The register RxA acts in conjunction with the TxA pin and the register RxB acts in conjunction with the TxB pin.

The timer value gets copied over into the register when a trigger event occurs on its corresponding pin. Control bits, TxC3, TxC2 and TxC1, allow the trigger events to be specified either as a positive or a negative edge. The trigger condition for each input pin can be specified independently.

The trigger conditions can also be programmed to generate interrupts. The occurrence of the specified trigger condition on the TxA and TxB pins will be respectively latched into the pending flags, TxPNDA and TxPNDB. The control flag TxENA allows the interrupt on TxA to be either enabled or disabled. Setting the TxENA flag enables interrupts to be generated when the selected trigger condition occurs on the TxA pin. Similarly, the flag TxENB controls the interrupts from the TxB pin.

Underflows from the timer can also be programmed to generate interrupts. Underflows are latched into the timer TxC0 pending flag (the TxC0 control bit serves as the timer underflow interrupt pending flag in the Input Capture mode). Consequently, the TxC0 control bit should be reset when entering the Input Capture mode. The timer underflow interrupt is enabled with the TxENA control flag. When a TxA interrupt occurs in the Input Capture mode, the user must check both the TxPNDA and TxC0 pending flags in order to determine whether a TxA input capture or a timer underflow (or both) caused the interrupt.

Figure 10 shows a block diagram of the timer in Input Capture mode.



TL/DD/12863–14

FIGURE 10. Timer in Input Capture Mode

TIMER CONTROL FLAGS

The timers T1, T2 and T3 have indentical control structures. The control bits and their functions are summarized below.

TxC0	Timer Start/Stop control in Modes 1 and 2 (Processor Independent PWM and External Event Counter), where 1 = Start, 0 = Stop Timer Underflow Interrupt Pending Flag in Mode 3 (Input Capture)
TxPNDA	Timer Interrupt Pending Flag
TxPNDB	Timer Interrupt Pending Flag
TxENA	Timer Interrupt Enable Flag
TxENB	Timer Interrupt Enable Flag 1 = Timer Interrupt Enabled 0 = Timer Interrupt Disabled
TxC3	Timer mode control
TxC2	Timer mode control
TxC1	Timer mode control

Timers (Continued)

The timer mode control bits (TxC3, TxC2 and TxC1) are detailed below:

TxC3	TxC2	TxC1	Timer Mode	Interrupt A Source	Interrupt B Source	Timer Counts On
0	0	0	MODE 2 (External Event Counter)	Timer Underflow	Pos. TxB Edge	TxA Pos. Edge
0	0	1	MODE 2 (External Event Counter)	Timer Underflow	Pos. TxB Edge	TxA Neg. Edge
1	0	1	MODE 1 (PWM) TxA Toggle	Autoreload RA	Autoreload RB	t_C
1	0	0	MODE 1 (PWM) No TxA Toggle	Autoreload RA	Autoreload RB	t_C
0	1	0	MODE 3 (Capture) Captures: TxA Pos. Edge TxB Pos. Edge	Pos. TxA Edge or Timer Underflow	Pos. TxB Edge	t_C
1	1	0	MODE 3 (Capture) Captures: TxA Pos. Edge TxB Neg. Edge	Pos. TxA Edge or Timer Underflow	Neg. TxB Edge	t_C
0	1	1	MODE 3 (Capture) Captures: TxA Neg. Edge TxB Pos. Edge	Neg. TxA Edge or Timer Underflow	Pos. TxB Edge	t_C
1	1	1	MODE 3 (Capture) Captures: TxA Neg. Edge TxB Neg. Edge	Neg. TxA Edge or Timer Underflow	Neg. TxB Edge	t_C

Power Save Modes

The device offers the user two power save modes of operation: HALT and IDLE. In the HALT mode, all microcontroller activities are stopped. In the IDLE mode, the on-board oscillator circuitry and timer T0 are active but all other microcontroller activities are stopped. In either mode, all on-board RAM, registers, I/O states, and timers (with the exception of T0) are unaltered.

HALT MODE

The device can be placed in the HALT mode by writing a "1" to the HALT flag (G7 data bit). All microcontroller activities, including the clock and timers, are stopped. The WATCHDOG logic on the device is disabled during the HALT mode. However, the clock monitor circuitry if enabled remains active and will cause the WATCHDOG output pin (WDOUT) to go low. If the HALT mode is used and the user does not want to activate the WDOUT pin, the Clock Monitor should be disabled after the device comes out of reset (resetting the Clock Monitor control bit with the first write to the WDSVR register). In the HALT mode, the power requirements of the device are minimal and the applied voltage (V_{CC}) may be decreased to V_r (V_r = 2.0V) without altering the state of the machine.

The device supports three different ways of exiting the HALT mode. The first method of exiting the HALT mode is with the Multi-Input Wakeup feature on the L port. The second method is with a low to high transition on the CKO (G7) pin. This method precludes the use of the crystal clock configuration (since CKO becomes a dedicated output), and so may be used with an RC clock configuration. The third method of exiting the HALT mode is by pulling the $\overline{\text{RESET}}$ pin low.

Since a crystal or ceramic resonator may be selected as the oscillator, the Wakeup signal is not allowed to start the chip running immediately since crystal oscillators and ceramic resonators have a delayed start up time to reach full amplitude and frequency stability. The IDLE timer is used to generate a fixed delay to ensure that the oscillator has indeed stabilized before allowing instruction execution. In this case, upon detecting a valid Wakeup signal, only the oscillator circuitry is enabled. The IDLE timer is loaded with a value of 256 and is clocked with the t_C instruction cycle clock. The t_C clock is derived by dividing the oscillator clock down by a factor of 10. The Schmitt trigger following the CKI inverter on the chip ensures that the IDLE timer is clocked only when the oscillator has a sufficiently large amplitude to meet the Schmitt trigger specifications. This Schmitt trigger is not part of the oscillator closed loop. The startup timeout from the IDLE timer enables the clock signals to be routed to the rest of the chip.

If an RC clock option is being used, the fixed delay is introduced optionally. A control bit, CLKDLY, mapped as configuration bit G7, controls whether the delay is to be introduced or not. The delay is included if CLKDLY is set, and excluded if CLKDLY is reset. The CLKDLY bit is cleared on reset.

Power Save Modes (Continued)

The device has two mask options associated with the HALT mode. The first mask option enables the HALT mode feature, while the second mask option disables the HALT mode. With the HALT mode enable mask option, the device will enter and exit the HALT mode as described above. With the HALT disable mask option, the device cannot be placed in the HALT mode (writing a "1" to the HALT flag will have no effect, the HALT flag will remain "0").

IDLE MODE

The device is placed in the IDLE mode by writing a "1" to the IDLE flag (G6 data bit). In this mode, all activities, except the associated on-board oscillator circuitry, and the IDLE Timer T0, are stopped.

The power supply requirements of the microcontroller in this mode of operation are typically around 30% of normal power requirement of the microcontroller.

As with the HALT mode, the device can be returned to normal operation with a reset, or with a Multi-Input Wakeup from the L Port. Alternately, the microcontroller resumes normal operation from the IDLE mode when the thirteenth bit (representing 4.096 ms at internal clock frequency of 1 MHz, t_C = 1 μs) of the IDLE Timer toggles.

This toggle condition of the thirteenth bit of the IDLE Timer T0 is latched into the T0PND pending flag.

The user has the option of being interrupted with a transition on the thirteenth bit of the IDLE Timer T0. The interrupt can be enabled or disabled via the T0EN control bit. Setting the T0EN flag enables the interrupt and vice versa.

The user can enter the IDLE mode with the Timer T0 interrupt enabled. In this case, when the T0PND bit gets set, the device will first execute the Timer T0 interrupt service routine and then return to the instruction following the "Enter Idle Mode" instruction.

Alternatively, the user can enter the IDLE mode with the IDLE Timer T0 interrupt disabled. In this case, the device will resume normal operation with the instruction immediately following the "Enter IDLE Mode" instruction.

Note: It is necessary to program two NOP instructions following both the set HALT mode and set IDLE mode instructions. These NOP instructions are necessary to allow clock resynchronization following the HALT or IDLE modes.

Multi-Input Wakeup

The Multi-Input Wakeup feature is ued to return (wakeup) the device from either the HALT or IDLE modes. Alternately Multi-Input Wakeup/Interrupt feature may also be used to generate up to 8 edge selectable external interrupts.

Figure 11 shows the Multi-Input Wakeup logic.

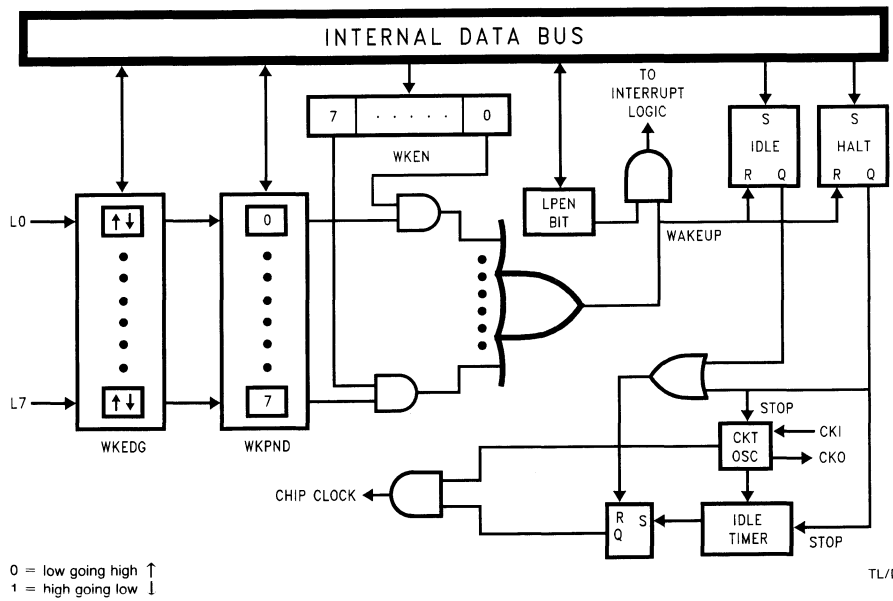


FIGURE 11. Multi-Input Wake Up Logic

Multi-Input Wakeup (Continued)

The Multi-Input Wakeup feature utilizes the L Port. The user selects which particular L port bit (or combination of L Port bits) will cause the device to exit the HALT or IDLE modes. The selection is done through the Reg: WKEN. The Reg: WKEN is an 8-bit read/write register, which contains a control bit for every L port bit. Setting a particular WKEN bit enables a Wakeup from the associated L port pin.

The user can select whether the trigger condition on the selected L Port pin is going to be either a positive edge (low to high transition) or a negative edge (high to low transition). This selection is made via the Reg: WKEDG, which is an 8-bit control register with a bit assigned to each L Port pin. Setting the control bit will select the trigger condition to be a negative edge on that particular L Port pin. Resetting the bit selects the trigger condition to be a positive edge. Changing an edge select entails several steps in order to avoid a pseudo Wakeup condition as a result of the edge change. First, the associated WKEN bit should be reset, followed by the edge select change in WKEDG. Next, the associated WKPND bit should be cleared, followed by the associated WKEN bit being re-enabled.

An example may serve to clarify this procedure. Suppose we wish to change the edge select from positive (low going high) to negative (high going low) for L Port bit 5, where bit 5 has previously been enabled for an input interrupt. The program would be as follows:

```
RBIT  5, WKEN
SBIT  5, WKEDG
RBIT  5, WKPND
SBIT  5, WKEN
```

If the L port bits have been used as outputs and then changed to inputs with Multi-Input Wakeup/Interrupt, a safety procedure should also be followed to avoid inherited pseudo wakeup conditions. After the selected L port bits have been changed from output to input but before the associated WKEN bits are enabled, the associated edge select bits in WKEDG should be set or reset for the desired edge selects, followed by the associated WKPND bits being cleared.

This same procedure should be used following reset, since the L port inputs are left floating as a result of reset.

The occurrence of the selected trigger condition for Multi-Input Wakeup is latched into a pending register called WKPND. The respective bits of the WKPND register will be set on the occurrence of the selected trigger edge on the corresponding Port L pin. The user has the responsibility of clearing these pending flags. Since WKPND is a pending register for the occurrence of selected wakeup conditions, the device will not enter the HALT mode if any Wakeup bit is both enabled and pending. Consequently, the user has the responsibility of clearing the pending flags before attempting to enter the HALT mode.

WKEN, WKPND and WKEDG are all read/write registers, and are cleared at reset.

PORT L INTERRUPTS

Port L provides the user with an additional eight fully selectable, edge sensitive interrupts which are all vectored into the same service subroutine.

The interrupt from Port L shares logic with the wake up circuitry. The register WKEN allows interrupts from Port L to be individually enabled or disabled. The register WKEDG specifies the trigger condition to be either a positive or a negative edge. Finally, the register WKPND latches in the pending trigger conditions.

The GIE (Global Interrupt Enable) bit enables the interrupt function.

A control flag, LPEN, functions as a global interrupt enable for Port L interrupts. Setting the LPEN flag will enable interrupts and vice versa. A separate global pending flag is not needed since the register WKPND is adequate.

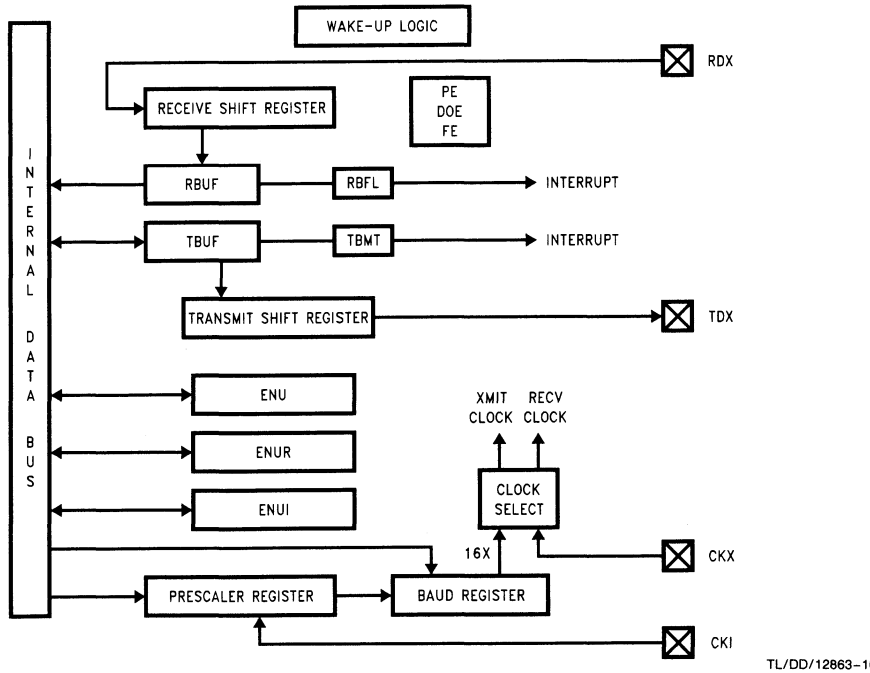
Since Port L is also used for waking the device out of the HALT or IDLE modes, the user can elect to exit the HALT or IDLE modes either with or without the interrupt enabled. If he elects to disable the interrupt, then the device will restart execution from the instruction immediately following the instruction that placed the microcontroller in the HALT or IDLE modes. In the other case, the device will first execute the interrupt service routine and then revert to normal operation. (See HALT mode for clock option wakeup information.)

Note: There is always the possibility of an interrupt occurring during an instruction which is attempting to reset the GIE bit or any other interrupt enable bit. If this occurs when a single cycle instruction is being used to reset the interrupt enable bit, the interrupt enable bit will be reset but an interrupt may still occur. This is because interrupt processing is started at the same time as the interrupt bit is being reset. To avoid this scenario, the user should always use a two, three, or four cycle instruction to reset interrupt enable bits.

UART

The device contains a full-duplex software programmable UART. The UART *(Figure 12)* consists of a transmit shift register, a receiver shift register and seven addressable registers, as follows: a transmit buffer register (TBUF), a receiver buffer register (RBUF), a UART control and status register (ENU), a UART receive control and status register (ENUR), a UART interrupt and clock source register (ENUI), a prescaler select register (PSR) and baud (BAUD) register. The ENU register contains flags for transmit and receive functions; this register also determines the length of the data frame (7, 8 or 9 bits), the value of the ninth bit in transmission, and parity selection bits. The ENUR register flags framming, data overrun and parity errors while the UART is receiving.

Other functions of the ENUR register include saving the ninth bit received in the data frame, enabling or disabling the UART's attention mode of operation and providing additional receiver/transmitter status information via RCVG and XMTG bits. The determination of an internal or external clock source is done by the ENUI register, as well as selecting the number of stop bits and enabling or disabling transmit and receive interrupts. A control flag in this register can also select the UART mode of operation: asynchronous or synchronous.



TL/DD/12863–16

FIGURE 12. UART Block Diagram

UART (Continued)

UART CONTROL AND STATUS REGISTERS

The operation of the UART is programmed through three registers: ENU, ENUR and ENUI. The function of the individual bits in these registers is as follows:

ENU-UART Control and Status Register (Address at 0BA)

PEN	PSEL1	XBIT9/ PSEL0	CHL1	CHL0	ERR	RBFL	TBMT
0RW	0RW	0RW	0RW	0RW	0R	0R	1R

Bit 7 Bit 0

ENUR-UART Receive Control and Status Register (Address at 0BB)

DOE	FE	PE	SPARE	RBIT9	ATTN	XMTG	RCVG
0RD	0RD	0RD	0RW*	0R	0RW	0R	0R

Bit 7 Bit 0

ENUI-UART Interrupt and Clock Source Register (Address at 0BC)

STP2	STP78	ETDX	SSEL	XRCLK	XTCLK	ERI	ETI
0RW	0RW	0RW	0RW	0RW	0RW	0RW	0RW

Bit 7 Bit 0

*Bit is not used.

0 Bit is cleared on reset.

1 Bit is set to one on reset.

R Bit is read-only; it cannot be written by software.

RW Bit is read/write.

D Bit is cleared on read; when read by software as a one, it is cleared automatically. Writing to the bit does not affect its state.

DESCRIPTION OF UART REGISTER BITS

ENU—UART CONTROL AND STATUS REGISTER

TBMT: This bit is set when the UART transfers a byte of data from the TBUF register into the TSFT register for transmission. It is automatically reset when software writes into the TBUF register.

RBFL: This bit is set when the UART has received a complete character and has copied it into the RBUF register. It is automatically reset when software reads the character from RBUF.

ERR: This bit is a global UART error flag which gets set if any or a combination of the errors (DOE, FE, PE) occur.

CHL1, CHL0: These bits select the character frame format. Parity is not included and is generated/verified by hardware.

CHL1 = 0, CHL0 = 0 The frame contains eight data bits.
CHL1 = 0, CHL0 = 1 The frame contains seven data bits.
CHL1 = 1, CHL0 = 0 The frame contains nine data bits.
CHL1 = 1, CHL0 = 1 Loopback Mode selected. Transmitter output internally looped back to receiver input. Nine bit framing format is used.

XBIT9/PSEL0: Programs the ninth bit for transmission when the UART is operating with nine data bits per frame. For seven or eight data bits per frame, this bit in conjunction with PSEL1 selects parity.

PSEL1, PSEL0: Parity select bits.

PSEL1 = 0, PSEL0 = 0 Odd Parity (if Parity enabled)
PSEL1 = 0, PSEL0 = 1 Even Parity (if Parity enabled)
PSEL1 = 1, PSEL0 = 0 Mark(1) (if Parity enabled)
PSEL1 = 1, PSEL0 = 1 Space(0) (if Parity enabled)

PEN: This bit enables/disables Parity (7- and 8-bit modes only).

PEN = 0 Parity disabled.
PEN = 1 Parity enabled.

ENUR—UART RECEIVE CONTROL AND STATUS REGISTER

RCVG: This bit is set high whenever a framing error occurs and goes low when RDX goes high.

XMTG: This bit is set to indicate that the UART is transmitting. It gets reset at the end of the last frame (end of last Stop bit).

ATTN: ATTENTION Mode is enabled while this bit is set. This bit is cleared automatically on receiving a character with data bit nine set.

RBIT9: Contains the ninth data bit received when the UART is operating with nine data bits per frame.

SPARE: Reserved for future use.

PE: Flags a Parity Error.

PE = 0 Indicates no Parity Error has been detected since the last time the ENUR register was read.
PE = 1 Indicates the occurrence of a Parity Error.

FE: Flags a Framing Error.

FE = 0 Indicates no Framing Error has been detected since the last time the ENUR register was read.
FE = 1 Indicates the occurrence of a Framing Error.

DOE: Flags a Data Overrun Error.

DOE = 0 Indicates no Data Overrun Error has been detected since the last time the ENUR register was read.
DOE = 1 Indicates the occurrence of a Data Overrun Error.

ENUI—UART INTERRUPT AND CLOCK SOURCE REGISTER

ETI: This bit enables/disables interrupt from the transmitter section.

ETI = 0 Interrupt from the transmitter is disabled.
ETI = 1 Interrupt from the transmitter is enabled.

ERI: This bit enables/disables interrupt from the receiver section.

ERI = 0 Interrupt from the receiver is disabled.
ERI = 1 Interrupt from the receiver is enabled.

XTCLK: This bit selects the clock source for the transmitter-section.

XTCLK = 0 The clock source is selected through the PSR and BAUD registers.
XTCLK = 1 Signal on CKX (L1) pin is used as the clock.

XRCLK: This bit selects the clock source for the receiver section.

XRCLK = 0 The clock source is selected through the PSR and BAUD registers.
XRCLK = 1 Signal on CKX (L1) pin is used as the clock.

SSEL: UART mode select.

SSEL = 0 Asynchronous Mode.
SSEL = 1 Synchronous Mode.

UART (Continued)

ETDX: TDX (UART Transmit Pin) is the alternate function assigned to Port L pin L2; it is selected by setting ETDX bit. To simulate line break generation, software should reset ETDX bit and output logic zero to TDX pin through Port L data and configuration registers.

STP78: This bit is set to program the last Stop bit to be 7/8th of a bit in length.

STP2: This bit programs the number of Stop bits to be transmitted.

STP2 = 0 One Stop bit transmitted.
STP2 = 1 Two Stop bits transmitted.

Associated I/O Pins

Data is transmitted on the TDX pin and received on the RDX pin. TDX is the alternate function assigned to Port L pin L2; it is selected by setting ETDX (in the ENUI register) to one. RDX is an inherent function of Port L pin L3, requiring no setup.

The baud rate clock for the UART can be generated on-chip, or can be taken from an external source. Port L pin L1 (CKX) is the external clock I/O pin. The CKX pin can be either an input or an output, as determined by Port L Configuration and Data registers (Bit 1). As an input, it accepts a clock signal which may be selected to drive the transmitter and/or receiver. As an output, it presents the internal Baud Rate Generator output.

UART Operation

The UART has two modes of operation: asynchronous mode and synchronous mode.

ASYNCHRONOUS MODE

This mode is selected by resetting the SSEL (in the ENUI register) bit to zero. The input frequency to the UART is 16 times the baud rate.

The TSFT and TBUF registers double-buffer data for transmission. While TSFT is shifting out the current character on the TDX pin, the TBUF register may be loaded by software with the next byte to be transmitted. When TSFT finishes transmitting the current character the contents of TBUF are transferred to the TSFT register and the Transmit Buffer Empty Flag (TBMT in the ENU register) is set. The TBMT flag is automatically reset by the UART when software loads a new character into the TBUF register. There is also the XMTG bit which is set to indicate that the UART is transmitting. This bit gets reset at the end of the last frame (end of last Stop bit). TBUF is a read/write register.

The RSFT and RBUF registers double-buffer data being received. The UART receiver continually monitors the signal on the RDX pin for a low level to detect the beginning of a Start bit. Upon sensing this low level, it waits for half a bit time and samples again. If the RDX pin is still low, the receiver considers this to be a valid Start bit, and the remaining bits in the character frame are each sampled a single time, at the mid-bit position. Serial data input on the RDX pin is shifted into the RSFT register. Upon receiving the complete character, the contents of the RSFT register are copied into the RBUF register and the Received Buffer Full Flag (RBFL) is set. RBFL is automatically reset when software reads the character from the RBUF register. RBUF is a read only register. There is also the RCVG bit which is set high when a framing error occurs and goes low once RDX goes high. TBMT, XMTG, RBFL and RCVG are read only bits.

SYNCHRONOUS MODE

In this mode data is transferred synchronously with the clock. Data is transmitted on the rising edge and received on the falling edge of the synchronous clock.

This mode is selected by setting SSEL bit in the ENUI register. The input frequency to the UART is the same as the baud rate.

When an external clock input is selected at the CKX pin, data transmit and receive are performed synchronously with this clock through TDX/RDX pins.

If data transmit and receive are selected with the CKX pin as clock output, the device generates the synchronous clock output at the CKX pin. The internal baud rate generator is used to produce the synchronous clock. Data transmit and receive are performed synchronously with this clock.

FRAMING FORMATS

The UART supports several serial framing formats *(Figure 13).* The format is selected using control bits in the ENU, ENUR and ENUI registers.

The first format (1, 1a, 1b, 1c) for data transmission (CHL0 = 1, CHL1 = 0) consists of Start bit, seven Data bits (excluding parity) and 7/8, one or two Stop bits. In applications using parity, the parity bit is generated and verified by hardware.

The second format (CHL0 = 0, CHL1 = 0) consists of one Start bit, eight Data bits (excluding parity) and 7/8, one or two Stop bits. Parity bit is generated and verified by hardware.

The third format for transmission (CHL0 = 0, CHL1 = 1) consists of one Start bit, nine Data bits and 7/8, one or two Stop bits. This format also supports the UART "ATTENTION" feature. When operating in this format, all eight bits of TBUF and RBUF are used for data. The ninth data bit is transmitted and received using two bits in the ENU and ENUR registers, called XBIT9 and RBIT9. RBIT9 is a read only bit. Parity is not generated or verified in this mode.

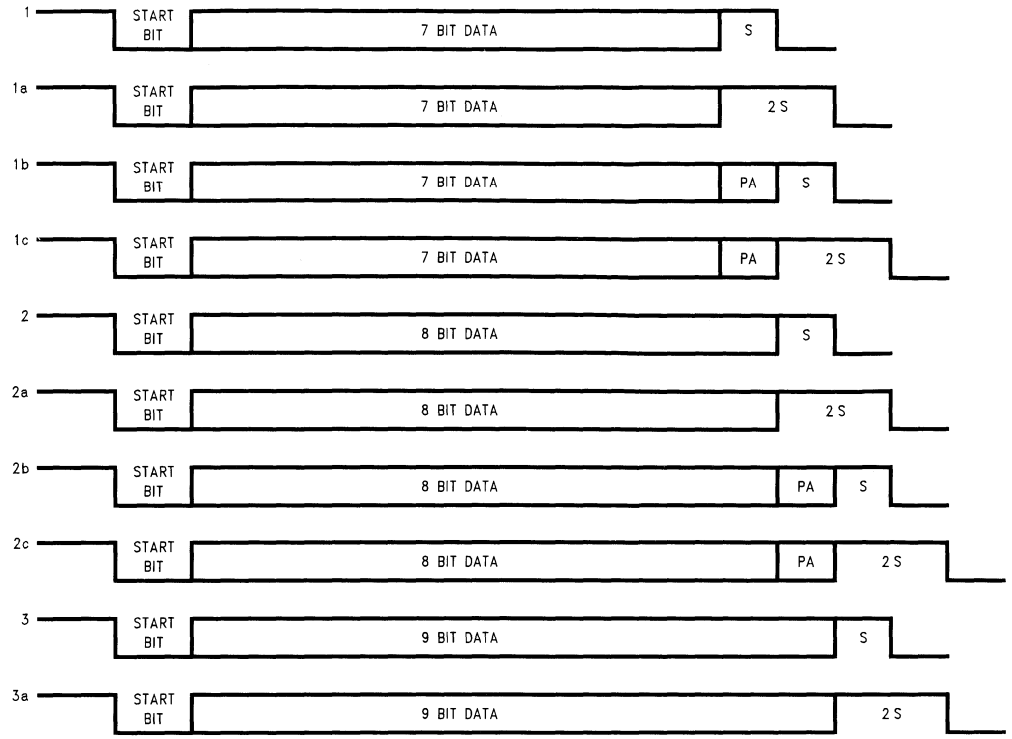
For any of the above framing formats, the last Stop bit can be programmed to be 7/8th of a bit in length. If two Stop bits are selected and the 7/8th bit is set (selected), the second Stop bit will be 7/8th of a bit in length.

The parity is enabled/disabled by PEN bit located in the ENU register. Parity is selected for 7- and 8-bit modes only. If parity is enabled (PEN = 1), the parity selection is then performed by PSEL0 and PSEL1 bits located in the ENU register.

Note that the XBIT9/PSEL0 bit located in the ENU register serves two mutually exclusive functions. This bit programs the ninth bit for transmission when the UART is operating with nine data bits per frame. There is no parity selection in this framing format. For other framing formats XBIT9 is not needed and the bit is PSEL0 used in conjunction with PSEL1 to select parity.

The frame formats for the receiver differ from the transmitter in the number of Stop bits required. The receiver only requires one Stop bit in a frame, regardless of the setting of the Stop bit selection bits in the control register. Note that an implicit assumption is made for full duplex UART operation that the framing formats are the same for the transmitter and receiver.

UART Operation (Continued)



TL/DD/12863–17

FIGURE 13. Framing Formats

UART INTERRUPTS

The UART is capable of generating interrupts. Interrupts are generated on Receive Buffer Full and Transmit Buffer Empty. Both interrupts have individual interrupt vectors. Two bytes of program memory space are reserved for each interrupt vector. The two vectors are located at addresses 0xEC to 0xEF Hex in the program memory space. The interrupts can be individually enabled or disabled using Enable Transmit Interrupt (ETI) and Enable Receive Interrupt (ERI) bits in the ENUI register.

The interrupt from the Transmitter is set pending, and remains pending, as long as both the TBMT and ETI bits are set. To remove this interrupt, software must either clear the ETI bit or write to the TBUF register (thus clearing the TBMT bit).

The interrupt from the receiver is set pending, and remains pending, as long as both the RBFL and ERI bits are set. To remove this interrupt, software must either clear the ERI bit or read from the RBUF register (thus clearing the RBFL bit).

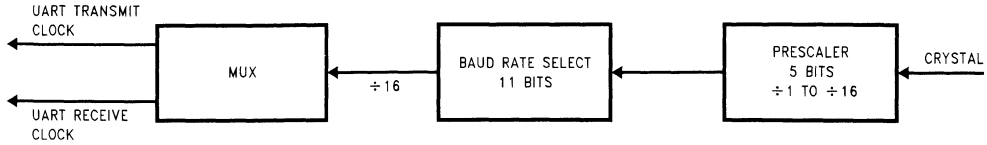
Baud Clock Generation

The clock inputs to the transmitter and receiver sections of the UART can be individually selected to come either from an external source at the CKX pin (port L, pin L1) or from a source selected in the PSR and BAUD registers. Internally, the basic baud clock is created from the oscillator frequency through a two-stage divider chain consisting of a 1–16 (increments of 0.5) prescaler and an 11-bit binary counter. *(Figure 14)* The divide factors are specified through two read/write registers shown in *Figure 15.* Note that the 11-bit Baud Rate Divisor spills over into the Prescaler Select Register (PSR). PSR is cleared upon reset.

As shown in Table I, a Prescaler Factor of 0 corresponds to NO CLOCK. NO CLOCK condition is the UART power down mode where the UART clock is turned off for power saving purpose. The user must also turn the UART clock off when a different baud rate is chosen.

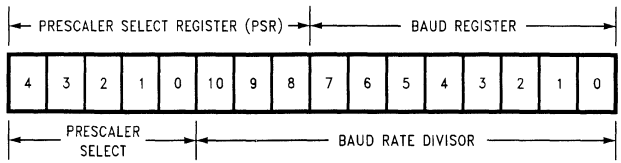
The correspondences between the 5-bit Prescaler Select and Prescaler factors are shown in Table I. Therer are many ways to calculate the two divisor factors, but one particularly effective method would be to achieve a 1.8432 MHz frequency coming out of the first stage. The 1.8432 MHz prescaler output is then used to drive the software programmable baud rate counter to create a 16x clock for the following baud rates: 110, 134.5, 150, 300, 600, 1200, 1800, 2400, 3600, 4800, 7200, 9600, 19200 and 38400 (Table II). Other baud rates may be created by using appropriate divisors. The 16x clock is then divided by 16 to provide the rate for the serial shift registers of the transmitter and receiver.

Baud Clock Generation (Continued)



TL/DD/12863-18

FIGURE 14. UART BAUD Clock Generation



TL/DD/12863-19

FIGURE 15. UART BAUD Clock Divisor Registers

TABLE I. Prescaler Factors

Prescaler Select	Prescaler Factor
00000	NO CLOCK
00001	1
00010	1.5
00011	2
00100	2.5
00101	3
00110	3.5
00111	4
01000	4.5
01001	5
01010	5.5
01011	6
01100	6.5
01101	7
01110	7.5
01111	8
10000	8.5
10001	9
10010	9.5
10011	10
10100	10.5
10101	11
10110	11.5
10111	12
11000	12.5
11001	13
11010	13.5
11011	14
11100	14.5
11101	15
11110	15.5
11111	16

TABLE II. Baud Rate Divisors (1.8432 MHz Prescaler Output)

Baud Rate	Baud Rate Divisor − 1 (N-1)
110 (110.03)	1046
134.5 (134.58)	855
150	767
300	383
600	191
1200	95
1800	63
2400	47
3600	31
4800	23
7200	15
9600	11
19200	5
38400	2

The entries in Table II assume a prescaler output of 1.8432 MHz. In the asynchronous mode the baud rate could be as high as 625k.

As an example, considering the Asynchronous Mode and a CKI clock of 4.608 MHz, the prescaler factor selected is:

$$4.608/1.8432 = 2.5$$

The 2.5 entry is available in Table I. The 1.8432 MHz prescaler output is then used with proper Baud Rate Divisor (Table II) to obtain different baud rates. For a baud rate of 19200 e.g., the entry in Table II is 5.

$N - 1 = 5$ ($N - 1$ is the value from Table II)

$N = 6$ (N is the Baud Rate Divisor)

Baud Rate = 1.8432 MHz/(16 × 6) = 19200

The divide by 16 is performed because in the asynchronous mode, the input frequency to the UART is 16 times the baud rate. The equation to calculate baud rates is given below.

The actual Baud Rate may be found from:

$$BR = Fc/(16 \times N \times P)$$

Baud Clock Generation (Continued)

Where:

BR is the Baud Rate

Fc is the CKI frequency

N is the Baud Rate Divisor (Table II).

P is the Prescaler Divide Factor selected by the value in the Prescaler Select Register (Table I)

Note: In the Synchronous Mode, the divisor 16 is replaced by two.

Example:

Asynchronous Mode:

Crystal Frequency = 5 MHz

Desired baud rate = 9600

Using the above equation N × P can be calculated first.

$$N \times P = (5 \times 10^6)/(16 \times 9600) = 32.552$$

Now 32.552 is divided by each Prescaler Factor (Table II) to obtain a value closest to an integer. This factor happens to be 6.5 (P = 6.5).

$$N = 32.552/6.5 = 5.008 \ (N = 5)$$

The programmed value (from Table II) should be 4 (N − 1).

Using the above values calculated for N and P:

$$BR = (5 \times 10^6)/(16 \times 5 \times 6.5) = 9615.384$$

$$\% \text{ error} = (9615.385 - 9600)/9600 = 0.16$$

Effect of HALT/IDLE

The UART logic is reinitialized when either the HALT or IDLE modes are entered. This reinitialization sets the TBMT flag and resets all read only bits in the UART control and status registers. Read/Write bits remain unchanged. The Transmit Buffer (TBUF) is not affected, but the Transmit Shift register (TSFT) bits are set to one. The receiver registers RBUF and RSFT are not affected.

The device will exit from the HALT/IDLE modes when the Start bit of a character is detected at the RDX (L3) pin. This feature is obtained by using the Multi-Input Wakeup scheme provided on the device.

Before entering the HALT or IDLE modes the user program must select the Wakeup source to be on the RDX pin. This selection is done by setting bit 3 of WKEN (Wakeup Enable) register. The Wakeup trigger condition is then selected to be high to low transition. This is done via the WKEDG register (Bit 3 is one.)

If the device is halted and crystal oscillator is used, the Wakeup signal will not start the chip running immediately because of the finite start up time requirement of the crystal oscillator. The idle timer (T0) generates a fixed (256 t_C) delay to ensure that the oscillator has indeed stabilized before allowing the device to execute code. The user has to consider this delay when data transfer is expected immediately after exiting the HALT mode.

Diagnostic

Bits CHARL0 and CHARL1 in the ENU register provide a loopback feature for diagnostic testing of the UART. When these bits are set to one, the following occur: The receiver input pin (RDX) is internally connected to the transmitter output pin (TDX); the output of the Transmitter Shift Register is "looped back" into the Receive Shift Register input. In this mode, data that is transmitted is immediately received. This feature allows the processor to verify the transmit and receive data paths of the UART.

Note that the framing format for this mode is the nine bit format; one Start bit, nine data bits, and 7/8, one or two Stop bits. Parity is not generated or verified in this mode.

Attention Mode

The UART Receiver section supports an alternate mode of operation, referred to as ATTENTION Mode. This mode of operation is selected by the ATTN bit in the ENUR register. The data format for transmission must also be selected as having nine Data bits and either 7/8, one or two Stop bits.

The ATTENTION mode of operation is intended for use in networking the device with other processors. Typically in such environments the messages consists of device addresses, indicating which of several destinations should receive them, and the actual data. This Mode supports a scheme in which addresses are flagged by having the ninth bit of the data field set to a 1. If the ninth bit is reset to a zero the byte is a Data byte.

While in ATTENTION mode, the UART monitors the communication flow, but ignores all characters until an address character is received. Upon receiving an address character, the UART signals that the character is ready by setting the RBFL flag, which in turn interrupts the processor if UART Receiver interrupts are enabled. The ATTN bit is also cleared automatically at this point, so that data characters as well as address characters are recognized. Software examines the contents of the RBUF and responds by deciding either to accept the subsequent data stream (by leaving the ATTN bit reset) or to wait until the next address character is seen (by setting the ATTN bit again).

Operation of the UART Transmitter is not affected by selection of this Mode. The value of the ninth bit to be transmitted is programmed by setting XBIT9 appropriately. The value of the ninth bit received is obtained by reading RBIT9. Since this bit is located in ENUR register where the error flags reside, a bit operation on it will reset the error flags.

Comparators

The device contains two differential comparators, each with a pair of inputs (positive and negative) and an output. Ports I1–I3 and I4–I6 are used for the comparators. The following is the Port I assignment:

- I1 Comparator1 negative input
- I2 Comparator1 positive input
- I3 Comparator1 output
- I4 Comparator2 negative input
- I5 Comparator2 positive input
- I6 Comparator2 output

A Comparator Select Register (CMPSL) is used to enable the comparators, read the outputs of the comparators internally, and enable the outputs of the comparators to the pins. Two control bits (enable and output enable) and one result bit are associated with each comparator. The comparator result bits (CMP1RD and CMP2RD) are read only bits which will read as zero if the associated comparator is not enabled. The Comparator Select Register is cleared with

Comparators (Continued)

reset, resulting in the comparators being disabled. The comparators should also be disabled before entering either the HALT or IDLE modes in order to save power. The configuration of the CMPSL register is as follows:

CMPSL REGISTER (ADDRESS X'00B7)

The CMPSL register contains the following bits:

CMP1EN Enable comparator 1

CMP1RD Comparator 1 result (this is a read only bit, which will read as 0 if the comparator is not enabled)

CMP10E Selects pin I3 as comparator 1 output provided that CMPIEN is set to enable the comparator

CMP2EN Enable comparator 2

CMP2RD Comparator 2 result (this is a read only bit, which will read as 0 if the comparator is not enabled)

CMP20E Selects pin I6 as comparator 2 output provided that CMP2EN is set to enable the comparator

Unused	CMP20E	CMP2RD	CMP2EN	CMP10E	CMP1RD	CMP1EN	Unused
Bit 7							Bit 0

Note that the two unused bits of CMPSL may be used as software flags.

Comparator outputs have the same spec as Ports L and G except that the rise and fall times are symmetrical.

MULTIPLY/DIVIDE

This device contains a multiply/divide block. This block supports a 1 byte x 2 bytes (3 bytes result) multiply or a 3 bytes/2 bytes (2 bytes result) divide operation. The multiply or divide operation is executed by setting control bits located in the multiply/divide control register. The multiply or divide operands must be placed into the appropriate memory mapped locations before the operation is initiated.

CONTROL REGISTER BITS

The Multiply/Divide control register (MDCR) is located at address xx9D. It has the following bit assignments:

MULT Start Multiplication Operation (1 = start)

DIV Start Division Operation (1 = start)

DIVOVF Division Overflow (if the result of a division is greater than 16 bits or the user attempted to divide by zero; 1 = error)

Rsvd	Rsvd	Rsvd	Rsvd	Rsvd	DIV OVF	DIV	MULT
Bit 7							Bit 0

After the appropriate MDR registers are loaded, the MULT and DIV start bits are set by the user to start a multiply or divide operation. The division operation has priority, if both bits are set simultaneously. The MULT and DIV bits are BOTH automatically cleared by hardware at the end of a divide or multiply operation. Each division operation causes the DIVOVF flag to be set/reset as appropriate. The DIVOVF flag is cleared following a multiplication operation. DIVOVF is a read-only bit. The MULT and DIV bits are read/writable. Bits 3–7 in MDCR should not be used, as the MULT and DIV operations will change their values.

MULTIPLY/DIVIDE OPERATION

For the multiply operation, the multiplicand is placed at addresses xx9B and xx9C. The multiplier is placed at address xx99. For the divide operation, the dividend is placed at addresses xx98 to xx9A and the divisor is placed at addresses xx9B to xx9C. In both operations, all operands are interpreted as unsigned values. The divide or multiply operation is started by setting the appropriate MDCR bit. If both the MULT and DIV bits are set, the microcontroller performs a divide operation. (The user is not required to read or clear the DIVOVF error bit prior to beginning a new multiply/divide operation. This bit is ignored during subsequent operations. However, the next divide operation will overwrite the error flag as appropriate, and the next multiply operation will clear it.)

TABLE III. Multiply/Divide Registers

Register Name (Address)	Multiplication Assignment		Division Assignment	
	Before Operation	After Operation	Before Operation	After Operation
MDR1 (xx98)	Unused	Unchanged	Low Byte of Dividend	Low Byte of Result
MDR2 (xx99)	Multiplier	Low Byte of Result	Middle Byte of Dividend	High Byte of Result
MDR3 (xx9A)		Middle Byte of Result	High Byte of Dividend	Undefined
MDR4 (xx9B)	Low Byte of Multiplicand	High Byte of Result	Low Byte of Divisor	Low Byte of Divisor
MDR5 (xx9C)	High Byte of Multiplicand	Unchanged	High Byte of Divisor	High Byte of Divisor

Comparators (Continued)

The multiply operation requires 1 instruction cycle to complete. The divide operation requires 2 instruction cycles to complete. A divide by zero or a division which produces an overflow requires only 1 instruction cycle to execute. The MDR1 through MDR5 registers and the MDCR register can not be read from or written to during a multiply or divide operation. Any attempt to write in to these registers will be ignored. Any attempt to read these registers will return undefined data.

The result of a multiply is placed in addresses xx99–xx9B. The result of a divide is placed in a ddresses xx98–xx99. If a division by zero is attempted or if the resulting quotient of a divide operation is more than 16 bits long, then the DIVOVF bit is set in the multiply/divide control register. The dividend and the divisor are left unchanged. The divide operation always causes the DIVOVF flag to be set or reset as appropriate. The DIVOVF flag is cleared following a multiply operation.

RESET STATE

A reset signal applied to the device during normal operation has the following affects:

MDCR is cleared, and any operation in progress is stopped.

MDR1 through MDR5 are undefined.

Interrupts

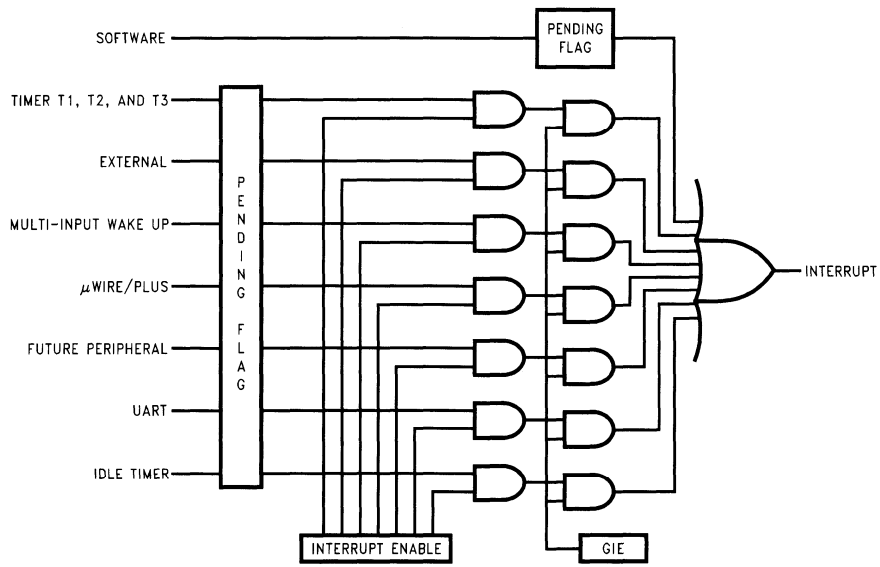
The device supports a vectored interrupt scheme. It supports a total of fourteen interrupt sources. The following table lists all the possible interrupt sources, their arbitration ranking and the memory locations reserved for the interrupt vector for each source.

Two bytes of program memory space are reserved for each interrupt source. All interrupt sources except the software interrupt are maskable. Each of the maskable interrupts have an Enable bit and a Pending bit. A maskable interrupt is active if its associated enable and pending bits are set. If GIE = 1 and an interrupt is active, then the processor will be interrupted as soon as it is ready to start executing an instruction except if the above conditions happen during the Software Trap service routine. This exception is described in the Software Trap sub-section.

The interruption process is accomplished with the INTR instruction (opcode 00), which is jammed inside the Instruction Register and replaces the opcode about to be executed. The following steps are performed for every interrupt:

1. The GIE (Global Interrupt Enable) bit is reset.
2. The address of the instruction about to be executed is pushed into the stack.
3. The PC (Program Counter) branches to address 00FF. This procedure takes 7 t_C cycles to execute.

At this time, since GIE = 0, other maskable interrupts are disabled. The user is now free to do whatever context switching is required by saving the context of the machine in the stack with PUSH instructions. The user would then program a VIS (Vector Interrupt Select) instruction in order to branch to the interrupt service routine of the highest priority interrupt enabled and pending at the time of the VIS. Note that this is not necessarily the interrupt that caused the branch to address location 00FF Hex prior to the context switching.



TL/DD/12863–20

FIGURE 16. Interrupt Block Diagram

Interrupts (Continued)

Arbitration Ranking	Source	Description	Vector Address Hi-Low Byte
(1) Highest	Software	INTR Instruction	0yFE–0yFF
(2)	Reserved		0yFC–0yFD
(3)	External	Pin G0 Edge	0yFA–0yFB
(4)	Timer T0	Underflow	0yF8–0yF9
(5)	Timer T1	T1A/Underflow	0yF6–0yF7
(6)	Timer T1	T1B	0yF4–0yF5
(7)	MICROWIRE/PLUS	BUSY Goes Low	0yF2–0yF3
(8)	Reserved		0yF0–0yF1
(9)	UART	Receive	0yEE–0yEF
(10)	UART	Transmit	0yEC–0yED
(11)	Timer T2	T2A/Underflow	0yEA–0yEB
(12)	Timer T2	T2B	0yE8–0yE9
(13)	Timer T3	T3A/Underflow	0yE6–0yE7
(14)	Timer T3	T3B	0yE4–0yE5
(15)	Port L/Wakeup	Port L Edge	0yE2–0yE3
(16) Lowest	Default VIS	Reserved	0yE0–0yE1

*y is a variable which represents the VIS block. VIS and the vector table must be located in the same 256-byte block except if VIS is located at the last address of a block. In this case, the table must be in the next block.

Thus, if an interrupt with a higher rank than the one which caused the interruption becomes active before the decision of which interrupt to service is made by the VIS, then the interrupt with the higher rank will override any lower ones and will be acknowledged. The lower priority interrupt(s) are still pending, however, and will cause another interrupt immediately following the completion of the interrupt service routine associated with the higher priority interrupt just serviced. This lower priority interrupt will occur immediately following the RETI (Return from Interrupt) instruction at the end of the interrupt service routine just completed.

Inside the interrupt service routine, the associated pending bit has to be cleared by software. The RETI (Return from Interrupt) instruction at the end of the interrupt service routine will set the GIE (Global Interrupt Enable) bit, allowing the processor to be interrupted again if another interrupt is active and pending.

The VIS instruction looks at all the active interrupts at the time it is executed and performs an indirect jump to the beginning of the service routine of the one with the highest rank.

The addresses of the different interrupt service routines, called vectors, are chosen by the user and stored in ROM in a table starting at 01E0 (assuming that VIS is located between 00FF and 01DF). The vectors are 15-bit wide and therefore occupy 2 ROM locations.

VIS and the vector table must be located in the same 256-byte block (0y00 to 0yFF) except if VIS is located at the last address of a block. In this case, the table must be in the next block. The vector table cannot be inserted in the first 256-byte block (y $\neq$ 0).

The vector of the maskable interrupt with the lowest rank is located at 0yE0 (Hi-Order byte) and 0yE1 (Lo-Order byte) and so forth in increasing rank number. The vector of the maskable interrupt with the highest rank is located at 0yFA (Hi-Order byte) and 0yFB (Lo-Order byte).

The Software Trap has the highest rank and its vector is located at 0yFE and 0yFF.

If, by accident, a VIS gets executed and no interrupt is active, then the PC (Program Counter) will branch to a vector located at 0yE0–0yE1.

WARNING

A Default VIS interrupt handler routine must be present. As a minimum, this handler should confirm that the GIE bit is cleared (this indicates that the interrupt sequence has been taken), take care of any required housekeeping, restore context and return. Some sort of Warm Restart procedure should be implemented. These events can occur without any error on the part of the system designer or programmer.

Note: There is always the possibility of an interrupt occurring during an instruction which is attempting to reset the GIE bit or any other interrupt enable bit. If this occurs when a single cycle instruction is being used to reset the interrupt enable bit, the interrupt enable bit will be reset but an interrupt may still occur. This is because interrupt processing is started at the same time as the interrupt bit is being reset. To avoid this scenario, the user should always use a two, three, or four cycle instruction to reset interrupt enable bits.

Figure 16 shows the Interrupt block diagram.

Interrupts (Continued)

SOFTWARE TRAP

The Software Trap (ST) is a special kind of non-maskable interrupt which occurs when the INTR instruction (used to acknowledge interrupts) is fetched from ROM and placed inside the instruction register. This may happen when the PC is pointing beyond the available ROM address space or when the stack is over-popped.

When an ST occurs, the user can re-initialize the stack pointer and do a recovery procedure (similar to reset, but not necessarily containing all of the same initialization procedures) before restarting.

The occurrence of an ST is latched into the ST pending bit. The GIE bit is not affected and the ST pending bit **(not accessible by the user)** is used to inhibit other interrupts and to direct the program to the ST service routine with the VIS instruction. The RPND instruction is used to clear the software interrupt pending bit. This pending bit is also cleared on reset.

The ST has the highest rank among all interrupts.

Nothing (except another ST) can interrupt an ST being serviced.

WATCHDOG

The device contains a WATCHDOG and clock monitor. The WATCHDOG is designed to detect the user program getting stuck in infinite loops resulting in loss of program control or "runaway" programs. The Clock Monitor is used to detect the absence of a clock or a very slow clock below a specified rate on the CKI pin.

The WATCHDOG consists of two independent logic blocks: WD UPPER and WD LOWER. WD UPPER establishes the upper limit on the service window and WD LOWER defines the lower limit of the service window.

Servicing the WATCHDOG consists of writing a specific value to a WATCHDOG Service Register named WDSVR which is memory mapped in the RAM. This value is composed of three fields, consisting of a 2-bit Window Select, a 5-bit Key Data field, and the 1-bit Clock Monitor Select field. Table III shows the WDSVR register.

The lower limit of the service window is fixed at 2048 instruction cycles. Bits 7 and 6 of the WDSVR register allow the user to pick an upper limit of the service window.

Table V shows the four possible combinations of lower and upper limits for the WATCHDOG service window. This flexibility in choosing the WATCHDOG service window prevents any undue burden on the user software.

Bits 5, 4, 3, 2 and 1 of the WDSVR register represent the 5-bit Key Data field. The key data is fixed at 01100. Bit 0 of the WDSVR Register is the Clock Monitor Select bit.

TABLE IV. WATCHDOG Service Register (WDSVR)

Window Select		Key Data					Clock Monitor
X	X	0	1	1	0	0	Y
7	6	5	4	3	2	1	0

TABLE V. WATCHDOG Service Window Select

WDSVR Bit 7	WDSVR Bit 6	Service Window (Lower-Upper Limits)
0	0	2k–8k t_C Cycles
0	1	2k–16k t_C Cycles
1	0	2k–32k t_C Cycles
1	1	2k–64k t_C Cycles

Clock Monitor

The Clock Monitor aboard the device can be selected or deselected under program control. The Clock Monitor is guaranteed not to reject the clock if the instruction cycle clock ($1/t_C$) is greater or equal to 10 kHz. This equates to a clock input rate on CKI of greater or equal to 100 kHz.

WATCHDOG Operation

The WATCHDOG and Clock Monitor are disabled during reset. The device comes out of reset with the WATCHDOG armed, the WATCHDOG Window Select bits (bits 6, 7 of the WDSVR Register) set, and the Clock Monitor bit (bit 0 of the WDSVR Register) enabled. Thus, a Clock Monitor error will occur after coming out of reset, if the instruction cycle clock frequency has not reached a minimum specified value, including the case where the oscillator fails to start.

The WDSVR register can be written to only once after reset and the key data (bits 5 through 1 of the WDSVR Register) must match to be a valid write. This write to the WDSVR register involves two irrevocable choices: (i) the selection of the WATCHDOG service window (ii) enabling or disabling of the Clock Monitor. Hence, the first write to WDSVR Register involves selecting or deselecting the Clock Monitor, select the WATCHDOG service window and match the WATCHDOG key data. Subsequent writes to the WDSVR register will compare the value being written by the user to the WATCHDOG service window value and the key data (bits 7 through 1) in the WDSVR Register. Table VI shows the sequence of events that can occur.

The user must service the WATCHDOG at least once before the upper limit of the service window expires. The WATCHDOG may not be serviced more than once in every lower limit of the service window. The user may service the WATCHDOG as many times as wished in the time period between the lower and upper limits of the service window. The first write to the WDSVR Register is also counted as a WATCHDOG service.

The WATCHDOG has an output pin associated with it. This is the WDOUT pin, on pin 1 of the port G. WDOUT is active low. The WDOUT pin is in the high impedance state in the inactive state. Upon triggering the WATCHDOG, the logic will pull the WDOUT (G1) pin low for an additional 16 t_C–32 t_C cycles after the signal level on WDOUT pin goes below the lower Schmitt trigger threshold. After this delay, the device will stop forcing the WDOUT output low.

The WATCHDOG service window will restart when the WDOUT pin goes high. It is recommended that the user tie the WDOUT pin back to V_{CC} through a resistor in order to pull WDOUT high.

A WATCHDOG service while the WDOUT signal is active will be ignored. The state of the WDOUT pin is not guaranteed on reset, but if it powers up low then the WATCHDOG will time out and WDOUT will enter high impedance state.

WATCHDOG Operation (Continued)

The Clock Monitor forces the G1 pin low upon detecting a clock frequency error. The Clock Monitor error will continue until the clock frequency has reached the minimum specified value, after which the G1 output will enter the high impedance TRI-STATE mode following 16 t_C–32 t_C clock cycles. The Clock Monitor generates a continual Clock Monitor error if the oscillator fails to start, or fails to reach the minimum specified frequency. The specification for the Clock Monitor is as follows:

$1/t_C > 10$ kHz—No clock rejection.

$1/t_C < 10$ Hz—Guaranteed clock rejection.

WATCHDOG AND CLOCK MONITOR SUMMARY

The following salient points regarding the WATCHDOG and CLOCK MONITOR should be noted:

- Both the WATCHDOG and CLOCK MONITOR detector circuits are inhibited during RESET.
- Following RESET, the WATCHDOG and CLOCK MONITOR are both enabled, with the WATCHDOG having he maximum service window selected.
- The WATCHDOG service window and CLOCK MONITOR enable/disable option can only be changed once, during the initial WATCHDOG service following RESET.
- The initial WATCHDOG service must match the key data value in the WATCHDOG Service register WDSVR in order to avoid a WATCHDOG error.
- Subsequent WATCHDOG services must match all three data fields in WDSVR in order to avoid WATCHDOG errors.
- The correct key data value cannot be read from the WATCHDOG Service register WDSVR. Any attempt to read this key data value of 01100 from WDSVR will read as key data value of all 0's.
- The WATCHDOG detector circuit is inhibited during both the HALT and IDLE modes.
- The CLOCK MONITOR detector circuit is active during both the HALT and IDLE modes. Consequently, the device inadvertently entering the HALT mode will be detected as a CLOCK MONITOR error (provided that the CLOCK MONITOR enable option has been selected by the program).
- With the single-pin R/C oscillator mask option selected and the CLKDLY bit reset, the WATCHDOG service window will resume following HALT mode from where it left off before entering the HALT mode.
- With the crystal oscillator mask option selected, or with the single-pin R/C oscillator mask option selected and the CLKDLY bit set, the WATCHDOG service window will be set to its selected value from WDSVR following HALT. Consequently, the WATCHDOG should not be serviced for at least 2048 instruction cycles following HALT, but must be serviced within the selected window to avoid a WATCHDOG error.
- The IDLE timer T0 is not initialized with RESET.
- The user can sync in to the IDLE counter cycle with an IDLE counter (T0) interrupt or by monitoring the T0PND flag. The T0PND flag is set whenever the thirteenth bit of the IDLE counter toggles (every 4096 instruction cycles). The user is responsible for resetting the T0PND flag.
- A hardware WATCHDOG service occurs just as the device exits the IDLE mode. Consequently, the WATCHDOG should not be serviced for at least 2048 instruction cycles following IDLE, but must be serviced within the selected window to avoid a WATCHDOG error.
- Following RESET, the initial WATCHDOG service (where the service window and the CLOCK MONITOR enable/disable must be selected) may be programmed anywhere within the maximum service window (65,536 instruction cycles) initialized by RESET. Note that this initial WATCHDOG service may be programmed within the initial 2048 instruction cycles without causing a WATCHDOG error.

Detection of Illegal Conditions

The device can detect various illegal conditions resulting from coding errors, transient noise, power supply voltage drops, runaway programs, etc.

Reading of undefined ROM gets zeros. The opcode for software interrupt is zero. If the program fetches instructions from undefined ROM, this will force a software interrupt, thus signaling that an illegal condition has occurred.

The subroutine stack grows down for each call (jump to subroutine), interrupt, or PUSH, and grows up for each return or POP. The stack pointer is initialized to RAM location 06F Hex during reset. Consequently, if there are more returns than calls, the stack pointer will point to addresses 070 and 071 Hex (which are undefined RAM). Undefined RAM from addresses 070 to 07F (Segment 0), 140 to 17F (Segment 1), and all other segments (i.e., Segments 2 . . . etc.) is read as all 1's, which in turn will cause the program to return to address 7FFF Hex. This is an undefined ROM location and the instruction fetched (all 0's) from this location will generate a software interrupt signaling an illegal condition.

Thus, the chip can detect the following illegal conditions:

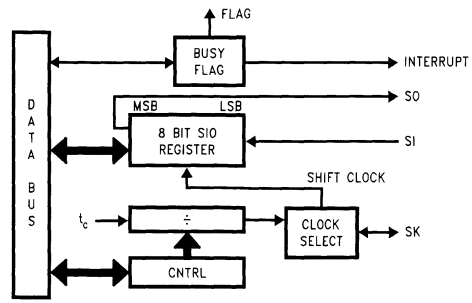
a. Executing from undefined ROM

b. Over "POP"ing the stack by having more returns than calls.

When the software interrupt occurs, the user can re-initialize the stack pointer and do a recovery procedure before restarting (this recovery program is probably similar to that following reset, but might not contain the same program initialization procedures). The recovery program should reset the software interrupt pending bit using the RPND instruction.

MICROWIRE/PLUS

MICROWIRE/PLUS is a serial synchronous communications interface. The MICROWIRE/PLUS capability enables the device to interface with any of National Semiconductor's MICROWIRE peripherals (i.e. A/D converters, display drivers, E^2PROMs etc.) and with other microcontrollers which support the MICROWIRE interface. It consists of an 8-bit serial shift register (SIO) with serial data input (SI), serial data output (SO) and serial shift clock (SK). *Figure 17* shows a block diagram of the MICROWIRE/PLUS logic.



TL/DD/12863–21

FIGURE 17. MICROWIRE/PLUS Block Diagram

The shift clock can be selected from either an internal source or an external source. Operating the MICROWIRE/PLUS arrangement with the internal clock source is called the Master mode of operation. Similarly, operating the MICROWIRE/PLUS arrangement with an external shift clock is called the Slave mode of operation.

The CNTRL register is used to configure and control the MICROWIRE/PLUS mode. To use the MICROWIRE/PLUS, the MSEL bit in the CNTRL register is set to one. In the master mode, the SK clock rate is selected by the two bits, SL0 and SL1, in the CNTRL register. Table VII details the different clock rates that may be selected.

TABLE VI. WATCHDOG Service Actions

Key Data	Window Data	Clock Monitor	Action
Match	Match	Match	Valid Service: Restart Service Window
Don't Care	Mismatch	Don't Care	Error: Generate WATCHDOG Output
Mismatch	Don't Care	Don't Care	Error: Generate WATCHDOG Output
Don't Care	Don't Care	Mismatch	Error: Generate WATCHDOG Output

TABLE VII. MICROWIRE/PLUS Master Mode Clock Select

SL1	SL0	SK
0	0	$2 \times t_c$
0	1	$4 \times t_c$
1	x	$8 \times t_c$

Where t_c is the instruction cycle clock

MICROWIRE/PLUS (Continued)

MICROWIRE/PLUS OPERATION

Setting the BUSY bit in the PSW register causes the MICROWIRE/PLUS to start shifting the data. It gets reset when eight data bits have been shifted. The user may reset the BUSY bit by software to allow less than 8 bits to shift. If enabled, an interrupt is generated when eight data bits have been shifted. The device may enter the MICROWIRE/PLUS mode either as a Master or as a Slave. *Figure 18* shows how two devices, microcontrollers and several peripherals may be interconnected using the MICROWIRE/PLUS arrangements.

Warning:

The SIO register should only be loaded when the SK clock is low. Loading the SIO register while the SK clock is high will result in undefined data in the SIO register. SK clock is normally low when not shifting.

Setting the BUSY flag when the input SK clock is high in the MICROWIRE/PLUS slave mode may cause the current SK clock for the SIO shift register to be narrow. For safety, the BUSY flag should only be set when the input SK clock is low.

MICROWIRE/PLUS Master Mode Operation

In the MICROWIRE/PLUS Master mode of operation the shift clock (SK) is generated internally by the device. The MICROWIRE Master always initiates all data exchanges. The MSEL bit in the CNTRL register must be set to enable the SO and SK functions onto the G Port. The SO and SK pins must also be selected as outputs by setting appropriate bits in the Port G configuration register. Table VII summarizes the bit settings required for Master mode of operation.

MICROWIRE/PLUS Slave Mode Operation

In the MICROWIRE/PLUS Slave mode of operation the SK clock is generated by an external source. Setting the MSEL bit in the CNTRL register enables the SO and SK functions onto the G Port. The SK pin must be selected as an input and the SO pin is selected as an output pin by setting and resetting the appropriate bits in the Port G configuration register. Table VII summarizes the settings required to enter the Slave mode of operation.

The user must set the BUSY flag immediately upon entering the Slave mode. This will ensure that all data bits sent by the Master will be shifted properly. After eight clock pulses the BUSY flag will be cleared and the sequence may be repeated.

Alternate SK Phase Operation

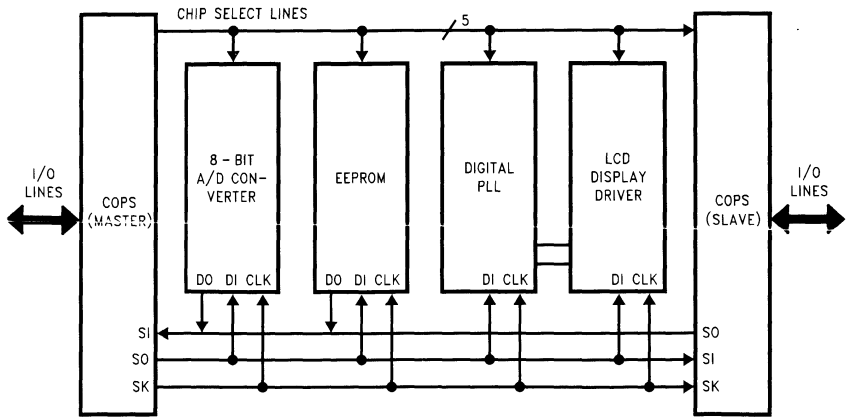
The device allows either the normal SK clock or an alternate phase SK clock to shift data in and out of the SIO register. In both the modes the SK is normally low. In the normal mode data is shifted in on the rising edge of the SK clock and the data is shifted out on the falling edge of the SK clock. The SIO register is shifted on each falling edge of the SK clock. In the alternate SK phase operation, data is shifted in on the falling edge of the SK clock and shifted out on the rising edge of the SK clock.

A control flag, SKSEL, allows either the normal SK clock or the alternate SK clock to be selected. Resetting SKSEL causes the MICROWIRE/PLUS logic to be clocked from the normal SK signal. Setting the SKSEL flag selects the alternate SK clock. The SKSEL is mapped into the G6 configuration bit. The SKSEL flag will power up in the reset condition, selecting the normal SK signal.

TABLE VIII

This table assumes that the control flag MSEL is set.

G4 (SO) Config. Bit	G5 (SK) Config. Bit	G4 Fun.	G5 Fun.	Operation
1	1	SO	Int. SK	MICROWIRE/PLUS Master
0	1	TRI-STATE	Int. SK	MICROWIRE/PLUS Master
1	0	SO	Ext. SK	MICROWIRE/PLUS Slave
0	0	TRI-STATE	Ext. SK	MICROWIRE/PLUS Slave



TL/DD/12863–22

FIGURE 18. MICROWIRE/PLUS Application

Memory Map

All RAM, ports and registers (except A and PC) are mapped into data memory address space.

Address S/ADD REG	Contents
0000 to 006F	On-Chip RAM bytes (112 bytes)
0070 to 007F	Unused RAM Address Space (Reads As All Ones)
xx80 to xxAF	Unused RAM Address Space (Reads Undefined Data)
0098	Dividend or Result Byte (MDR1)
0099	Dividend/Multiplier or Result Byte (MDR2)
009A	Dividend/Result Byte or Undefined (MDR3)
009B	Dividend/Multiplicand or Result Byte (MDR4)
009C	Divisor or Multiplicand Byte (MDR5) Multiply/Divide Control Register (MDCR)
xxB0	Timer T3 Lower Byte
XXB1	Timer T3 Upper Byte
xxB2	Timer T3 Autoload Register T3RA Lower Byte
xxB3	Timer T3 Autoload Register T3RA Upper Byte
xxB4	Timer T3 Autoload Register T3RB Lower Byte
xxB5	Timer T3 Autoload Register T3RB Upper Byte
xxB6	Timer T3 Control Register
xxB7	Comparator Select Register (CMPSL)
xxB8	UART Transmit Buffer (TBUF)
xxB9	UART Receive Buffer (RBUF)
xxBA	UART Control and Status Register (ENU)
xxBB	UART Receive Control and Status Register (ENUR)
xxBC	UART Interrupt and Clock Source Register (ENUI)
xxBD	UART Baud Register (BAUD)
xxBE	UART Prescale Select Register (PSR)
xxBF	Reserved for UART
xxC0	Timer T2 Lower Byte
xxC1	Timer T2 Upper Byte
xxC2	Timer T2 Autoload Register T2RA Lower Byte
xxC3	Timer T2 Autoload Register T2RA Upper Byte
xxC4	Timer T2 Autoload Register T2RB Lower Byte
xxC5	Timer T2 Autoload Register T2RB Upper Byte
xxC6	Timer T2 Control Register
xxC7	WATCHDOG Service Register (Reg:WDSVR)
xxC8	MIWU Edge Select Register (Reg:WKEDG)
xxC9	MIWU Enable Register (Reg:WKEN)
xxCA	MIWU Pending Register (Reg:WKPND)
xxCB	Reserved
xxCC	Reserved
xxCD to xxCF	Reserved

Address S/ADD REG	Contents
xxD0	Port L Data Register
xxD1	Port L Configuration Register
xxD2	Port L Input Pins (Read Only)
xxD3	Reserved for Port L
xxD4	Port G Data Register
xxD5	Port G Configuration Register
xxD6	Port G Input Pins (Read Only)
xxD7	Port I Input Pins (Read Only)
xxD8	Port C Data Register
xxD9	Port C Configuration Register
xxDA	Port C Input Pins (Read Only)
xxDB	Reserved for Port C
xxDC	Port D
xxDD to DF	Reserved for Port D
xxE0 to xxE5	Reserved for EE Control Registers
xxE6	Timer T1 Autoload Register T1RB Lower Byte
xxE7	Timer T1 Autoload Register T1RB Upper Byte
xxE8	ICNTRL Register
xxE9	MICROWIRE/PLUS Shift Register
xxEA	Timer T1 Lower Byte
xxEB	Timer T1 Upper Byte
xxEC	Timer T1 Autoload Register T1RA Lower Byte
xxED	Timer T1 Autoload Register T1RA Upper Byte
xxEE	CNTRL Control Register
xxEF	PSW Register
xxF0 to FB	On-Chip RAM Mapped as Registers
xxFC	X Register
xxFD	SP Register
xxFE	B Register
xxFF	S Register
0100 to 017F	On-Chip 128 RAM Bytes
0200 to 027F	On-Chip 128 RAM Bytes
0300 to 037F	On-Chip 128 RAM Bytes

Reading memory locations 0070H–007FH (Segment 0) will return all ones. Reading unused memory locations 0080H–00AFH (Segment 0) will return undefined data. Reading memory locations from other unused Segments (i.e., Segment 4, Segment 5, ... etc.) will return all ones.

Addressing Modes

There are ten addressing modes, six for operand addressing and four for transfer of control.

OPERAND ADDRESSING MODES

Register Indirect

This is the "normal" addressing mode. The operand is the data memory addressed by the B pointer or X pointer.

Register Indirect (with auto post increment or decrement of pointer)

This addressing mode is used with the LD and X instructions. The operand is the data memory addressed by the B pointer or X pointer. This is a register indirect mode that automatically post increments or decrements the B or X register after executing the instruction.

Direct

The instruction contains an 8-bit address field that directly points to the data memory for the operand.

Immediate

The instruction contains an 8-bit immediate field as the operand.

Short Immediate

This addressing mode is used with the Load B Immediate instruction. The instruction contains a 4-bit immediate field as the operand.

Indirect

This addressing mode is used with the LAID instruction. The contents of the accumulator are used as a partial address (lower 8 bits of PC) for accessing a data operand from the program memory.

TRANSFER OF CONTROL ADDRESSING MODES

Relative

This mode is used for the JP instruction, with the instruction field being added to the program counter to get the new program location. JP has a range from −31 to +32 to allow a 1-byte relative jump (JP + 1 is implemented by a NOP instruction). There are no "pages" when using JP, since all 15 bits of PC are used.

Absolute

This mode is used with the JMP and JSR instructions, with the instruction field of 12 bits replacing the lower 12 bits of the program counter (PC). This allows jumping to any location in the current 4k program memory segment.

Absolute Long

This mode is used with the JMPL and JSRL instructions, with the instruction field of 15 bits replacing the entire 15 bits of the program counter (PC). This allows jumping to any location up to 32k program memory space.

Indirect

This mode is used with the JID instruction. The contents of the accumulator are used as a partial address (lower 8 bits of PC) for accessing a location in the program memory. The contents of this program memory location serve as a partial address (lower 8 bits of PC) for the jump to the next instruction.

Note: The VIS is a special case of the Indirect Transfer of Control addressing mode, where the double byte vector associated with the interrupt is transferred from adjacent addresses in the program memory into the program counter (PC) in order to jump to the associated interrupt service routine.

Instruction Set

Register and Symbol Definition

Registers	
A	8-Bit Accumulator Register
B	8-Bit Address Register
X	8-Bit Address Register
SP	8-Bit Stack Pointer Register
PC	15-Bit Program Counter Register
PU	Upper 7 Bits of PC
PL	Lower 8 Bits of PC
C	1 Bit of PSW Register for Carry
HC	1 Bit of PSW Register for Half Carry
GIE	1 Bit of PSW Register for Global Interrupt Enable
VU	Interrupt Vector Upper Byte
VL	Interrupt Vector Lower Byte

Symbols	
[B]	Memory Indirectly Addressed by B Register
[X]	Memory Indirectly Addressed by X Register
MD	Direct Addressed Memory
Mem	Direct Addressed Memory or [B]
Meml	Direct Addressed Memory or [B] or Immediate Data
Imm	8-Bit Immediate Data
Reg	Register Memory: Addresses F0 to FF (Includes B, X and SP)
Bit	Bit Number (0 to 7)
←	Loaded with
↔	Exchanged with

Instruction Set (Continued)

INSTRUCTION SET

ADD	A,Meml	ADD	A ← A + Meml
ADC	A,Meml	ADD with Carry	A ← A + Meml + C, C ← Carry HC ← Half Carry
SUBC	A,Meml	Subtract with Carry	A ← A − $\overline{\text{Meml}}$ + C, C ← Carry HC ← Half Carry
AND	A,Meml	Logical AND	A ← A and Meml
ANDSZ	A,Imm	Logical AND Immed., Skip if Zero	Skip next if (A and Imm) = 0
OR	A,Meml	Logical OR	A ← A or Meml
XOR	A,Meml	Logical EXclusive OR	A ← A xor Meml
IFEQ	MD,Imm	IF EQual	Compare MD and Imm, Do next if MD = Imm
IFEQ	A,Meml	IF EQual	Compare A and Meml, Do next if A = Meml
IFNE	A,Meml	IF Not Equal	Compare A and Meml, Do next if A ≠ Meml
IFGT	A,Meml	IF Greater Than	Compare A and Meml, Do next if A > Meml
IFBNE	#	If B Not Equal	Do next if lower 4 bits of B ≠ Imm
DRSZ	Reg	Decrement Reg., Skip if Zero	Reg ← Reg − 1, Skip if Reg = 0
SBIT	#,Mem	Set BIT	1 to bit, Mem (bit = 0 to 7 immediate)
RBIT	#,Mem	Reset BIT	0 to bit, Mem
IFBIT	#,Mem	IF BIT	If bit in A or Mem is true do next instruction
RPND		Reset PeNDing Flag	Reset Software Interrupt Pending Flag
X	A,Mem	EXchange A with Memory	A ⟷ Mem
X	A,[X]	EXchange A with Memory [X]	A ⟷ [X]
LD	A,Meml	LoaD A with Memory	A ← Meml
LD	A,[X]	LoaD A with Memory [X]	A ← [X]
LD	B,Imm	LoaD B with Immed.	B ← Imm
LD	Mem,Imm	LoaD Memory Immed	Mem ← Imm
LD	Reg,Imm	LoaD Register Memory Immed.	Reg ← Imm
X	A, [B ±]	EXchange A with Memory [B]	A ⟷ [B], (B ← B ±1)
X	A, [X ±]	EXchange A with Memory [X]	A ⟷ [X], (X ← X ±1)
LD	A, [B±]	LoaD A with Memory [B]	A ← [B], (B ← B ±1)
LD	A, [X±]	LoaD A with Memory [X]	A ← [X], (X ← X±1)
LD	[B±],Imm	LoaD Memory [B] Immed.	[B] ← Imm, (B ← B±1)
CLR	A	CLeaR A	A ← 0
INC	A	INCrement A	A ← A + 1
DEC	A	DECrementA	A ← A − 1
LAID		Load A InDirect from ROM	A ← ROM (PU,A)
DCOR	A	Decimal CORrect A	A ← BCD correction of A (follows ADC, SUBC)
RRC	A	Rotate A Right thru C	C → A7 → . . . → A0 → C
RLC	A	Rotate A Left thru C	C ← A7 ← . . . ← A0 ← C
SWAP	A	SWAP nibbles of A	A7 . . . A4 ⟷ A3 . . . A0
SC		Set C	C ← 1, HC ← 1
RC		Reset C	C ← 0, HC ← 0
IFC		IF C	IF C is true, do next instruction
IFNC		IF Not C	If C is not true, do next instruction
POP	A	POP the stack into A	SP ← SP + 1, A ← [SP]
PUSH	A	PUSH A onto the stack	[SP] ← A, SP ← SP − 1
VIS		Vector to Interrupt Service Routine	PU ← [VU], PL ← [VL]
JMPL	Addr.	Jump absolute Long	PC ← ii (ii = 15 bits, 0 to 32k)
JMP	Addr.	Jump absolute	PC9 . . . 0 ← i (i = 12 bits)
JP	Disp.	Jump relative short	PC ← PC + r (r is −31 to +32, except 1)
JSRL	Addr.	Jump SubRoutine Long	[SP] ← PL, [SP−1] ← PU,SP−2, PC ← ii
JSR	Addr	Jump SubRoutine	[SP] ← PL, [SP−1] ← PU,SP−2, PC9 . . . 0 ← i
JID		Jump InDirect	PL ← ROM (PU,A)
RET		RETurn from subroutine	SP + 2, PL ← [SP], PU ← [SP−1]
RETSK		RETurn and SKip	SP + 2, PL ← [SP],PU ← [SP−1], Skip Next Instruction
RETI		RETurn from Interrupt	SP + 2, PL ← [SP],PU ← [SP−1],GIE ← 1
INTR		Generate an Interrupt	[SP] ← PL, [SP−1] ← PU, SP−2, PC ← 0FF
NOP		No OPeration	PC ← PC + 1

Instruction Execution Time

Most instructions are single byte (with immediate addressing mode instructions taking two bytes).

Most single byte instructions take one cycle time to execute.

Skipped instructions require x number of cycles to be skipped, where x equals the number of bytes in the skipped instruction opcode.

See the BYTES and CYCLES per INSTRUCTION table for details.

Bytes and Cycles per Instruction

The following table shows the number of bytes and cycles for each instruction in the format of byte/cycle.

Arithmetic and Logic Instructions

	[B]	Direct	Immed.
ADD	1/1	3/4	2/2
ADC	1/1	3/4	2/2
SUBC	1/1	3/4	2/2
AND	1/1	3/4	2/2
OR	1/1	3/4	2/2
XOR	1/1	3/4	2/2
IFEQ	1/1	3/4	2/2
IFNE	1/1	3/4	2/2
IFGT	1/1	3/4	2/2
IFBNE	1/1		
DRSZ		1/3	
SBIT	1/1	3/4	
RBIT	1/1	3/4	
IFBIT	1/1	3/4	

RPND	1/1

Instructions Using A & C

CLRA	1/1
INCA	1/1
DECA	1/1
LAID	1/3
DCOR	1/1
RRCA	1/1
RLCA	1/1
SWAPA	1/1
SC	1/1
RC	1/1
IFC	1/1
IFNC	1/1
PUSHA	1/3
POPA	1/3
ANDSZ	2/2

Transfer of Control Instructions

JMPL	3/4
JMP	2/3
JP	1/3
JSRL	3/5
JSR	2/5
JID	1/3
VIS	1/5
RET	1/5
RETSK	1/5
RETI	1/5
INTR	1/7
NOP	1/1

Memory Transfer Instructions

	Register Indirect		Direct	Immed.	Register Indirect Auto Incr. & Decr.		
	[B]	[X]			[B+, B−]	[X+, X−]	
X A,*	1/1	1/3	2/3		1/2	1/3	
LD A,*	1/1	1/3	2/3	2/2	1/2	1/3	
LD B, Imm				1/1			(IF B < 16)
LD B, Imm				2/2			(IF B > 15)
LD Mem, Imm	2/2		3/3		2/2		
LD Reg, Imm			2/3				
IFEQ MD, Imm			3/3				

* = > Memory location addressed by B or X or directly.

Opcode Table

Upper Nibble Along X-Axis

Lower Nibble Along Y-Axis

F	E	D	C	B	A	9	8	
JP −15	JP −31	LD 0F0, # i	DRSZ 0F0	RRCA	RC	ADC A,#i	ADC A,[B]	0
JP −14	JP −30	LD 0F1, # i	DRSZ 0F1	*	SC	SUBC A, #i	SUB A,[B]	1
JP −13	JP −29	LD 0F2, # i	DRSZ 0F2	X A, [X+]	X A,[B+]	IFEQ A,#i	IFEQ A,[B]	2
JP −12	JP −28	LD 0F3, # i	DRSZ 0F3	X A, [X−]	X A,[B−]	IFGT A,#i	IFGT A,[B]	3
JP −11	JP −27	LD 0F4, # i	DRSZ 0F4	VIS	LAID	ADD A,#i	ADD A,[B]	4
JP −10	JP −26	LD 0F5, # i	DRSZ 0F5	RPND	JID	AND A,#i	AND A,[B]	5
JP −9	JP −25	LD 0F6, # i	DRSZ 0F6	X A,[X]	X A,[B]	XOR A,#i	XOR A,[B]	6
JP −8	JP −24	LD 0F7, # i	DRSZ 0F7	*	*	OR A,#i	OR A,[B]	7
JP −7	JP −23	LD 0F8, # i	DRSZ 0F8	NOP	RLCA	LD A,#i	IFC	8
JP −6	JP −22	LD 0F9, # i	DRSZ 0F9	IFNE A,[B]	IFEQ Md,#i	IFNE A,#i	IFNC	9
JP −5	JP −21	LD 0FA, # i	DRSZ 0FA	LD A,[X+]	LD A,[B+]	LD [B+],#i	INCA	A
JP −4	JP −20	LD 0FB, # i	DRSZ 0FB	LD A,[X−]	LD A,[B−]	LD [B−],#i	DECA	B
JP −3	JP −19	LD 0FC, # i	DRSZ 0FC	LD Md,#i	JMPL	X A,Md	POPA	C
JP −2	JP −18	LD 0FD, # i	DRSZ 0FD	DIR	JSRL	LD A,Md	RETSK	D
JP −1	JP −17	LD 0FE, # i	DRSZ 0FE	LD A,[X]	LD A,[B]	LD [B],#i	RET	E
JP −0	JP −16	LD 0FF, # i	DRSZ 0FF	*	*	LD B,#i	RETI	F

Opcode Table (Continued)

Upper Nibble Along X-Axis

Lower Nibble Along Y-Axis

7	6	5	4	3	2	1	0	
IFBIT 0,[B]	ANDSZ A, #i	LD B,#0F	IFBNE 0	JSR x000–x0FF	JMP x000–x0FF	JP +17	INTR	0
IFBIT 1,[B]	*	LD B,#0E	IFBNE 1	JSR x100–x1FF	JMP x100–x1FF	JP +18	JP + 2	1
IFBIT 2,[B]	*	LD B,#0D	IFBNE 2	JSR x200–x2FF	JMP x200–x2FF	JP +19	JP + 3	2
IFBIT 3,[B]	*	LD B,#0C	IFBNE 3	JSR x300–x3FF	JMP x300–x3FF	JP +20	JP + 4	3
IFBIT 4,[B]	CLRA	LD B,#0B	IFBNE 4	JSR x400–x4FF	JMP x400–x4FF	JP +21	JP + 5	4
IFBIT 5,[B]	SWAPA	LD B,#0A	IFBNE 5	JSR x500–x5FF	JMP x500–x5FF	JP +22	JP + 6	5
IFBIT 6,[B]	DCORA	LD B,#09	IFBNE 6	JSR x600–x6FF	JMP x600–x6FF	JP +23	JP + 7	6
IFBIT 7,[B]	PUSHA	LD B,#08	IFBNE 7	JSR x700–x7FF	JMP x700–x7FF	JP +24	JP + 8	7
SBIT 0,[B]	RBIT 0,[B]	LD B,#07	IFBNE 8	JSR x800–x8FF	JMP x800–x8FF	JP +25	JP + 9	8
SBIT 1,[B]	RBIT 1,[B]	LD B,#06	IFBNE 9	JSR x900–x9FF	JMP x900–x9FF	JP +26	JP + 10	9
SBIT 2,[B]	RBIT 2,[B]	LD B,#05	IFBNE 0A	JSR xA00–xAFF	JMP xA00–xAFF	JP +27	JP + 11	A
SBIT 3,[B]	RBIT 3,[B]	LD B,#04	IFBNE 0B	JSR xB00–xBFF	JMP xB00–xBFF	JP +28	JP + 12	B
SBIT 4,[B]	RBIT 4,[B]	LD B,#03	IFBNE 0C	JSR xC00–xCFF	JMP xC00–xCFF	JP +29	JP + 13	C
SBIT 5,[B]	RBIT 5,[B]	LD B,#02	IFBNE 0D	JSR xD00–xDFF	JMP xD00–xDFF	JP +30	JP + 14	D
SBIT 6,[B]	RBIT 6,[B]	LD B,#01	IFBNE 0E	JSR xE00–xEFF	JMP xE00–xEFF	JP +31	JP + 15	E
SBIT 7,[B]	RBIT 7,[B]	LD B,#00	IFBNE 0F	JSR xF00–xFFF	JMP xF00–xFFF	JP +32	JP + 16	F

Where,

i is the immediate data

Md is a directly addressed memory location

* is an unused opcode

Note: The opcode 60 Hex is also the opcode for IFBIT #i,A

Development Support

SUMMARY

- iceMASTER™: IM-COP8/400—Full feature in-circuit emulation for all COP8 products. A full set of COP8 Basic and Feature Family device and package specific probes are available.
- COP8 Debug Module: Moderate cost in-circuit emulation and development programming unit.
- COP8 Evaluation and Programming Unit: EPU-COP888GG—low cost In-circuit simulation and development programming unit.
- Assembler: COP-8-DEV-IBMA. A DOS installable cross development Assembler, Linker, Librarian and Utility Software Development Tool Kit.
- C Compiler: COP8C. A DOS installable cross development Software Tool Kit.
- OTP/EPROM Programmer Support: Covering needs from engineering prototype, pilot production to full production environments.

IceMASTER (IM) IN-CIRCUIT EMULATION

The iceMASTER IM-COP8/400 is a full feature, PC based, in-circuit emulation tool developed and marketed by Meta-Link Corporation to support the whole COP8 family of products. National is a resale vendor for these products.

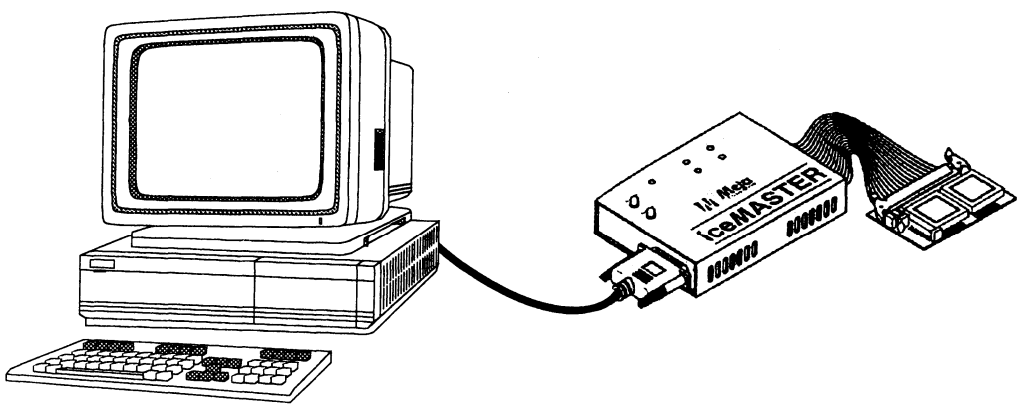
See *Figure 19* for configuration.

The iceMASTER IM-COP8/400 with its device specific COP8 Probe provides a rich feature set for developing, testing and maintaining product:

- Real-time in-circuit emulation; full 2.4V–5.5V operation range, full DC-10 MHz clock. Chip options are programmable or jumper selectable.
- Direct connection to application board by package compatible socket or surface mount assembly.
- Full 32k byte of loadable programming space that overlays (replaces) the on-chip ROM or EPROM. On-chip RAM and I/O blocks are used directly or recreated on the probe as necessary.
- Full 4k frame synchronous trace memory. Address, instruction, and 8 unspecified, circuit connectable trace lines. Display can be HLL source (e.g., C source), assembly or mixed.
- A full 64k hardware configurable break, trace on, trace off control, and pass count increment events.
- Tool set integrated interactive symbolic debugger—supports both assembler (COFF) and C Compiler (.COD) linked object formats.
- Real time performance profiling analysis; selectable bucket definition.
- Watch windows, content updated automatically at each execution break.
- Instruction by instruction memory/register changes displayed on source window when in single step operation.
- Single base unit and debugger software reconfigurable to support the entire COP8 family; only the probe personality needs to change. Debugger software is processor customized, and reconfigured from a master model file.
- Processor specific symbolic display of registers and bit level assignments, configured from master model file.
- Halt/Idle mode notification.
- On-line HELP customized to specific processor using master model file.
- Includes a copy of COP8-DEV-IBMA assembler and linker SDK.

IM Order Information

Base Unit	
IM-COP8/400-1	iceMASTER base unit, 110V power supply
IM-COP8/400-2	iceMASTER base unit, 220V power supply
iceMASTER Probe	
MHW-884FH28DWPC	28 DIP
MHW-888FH40DWPC	40 DIP
MHW-888FH44PWPC	44 PLCC
28 DIP to 28 SO Adapter	
MHW-SOIC28	28 SO



TL/DD/12863–23

FIGURE 19. COP8 iceMASTER Environment

Development Support (Continued)

IceMASTER DEBUG MODULE (DM)

The iceMASTER Debug Module is a PC based, combination in-circuit emulation tool and COP8 based OTP/EPROM programming tool developed and marketed by MetaLink Corporation to support the whole COP8 family of products. National is a resale vendor for these products.

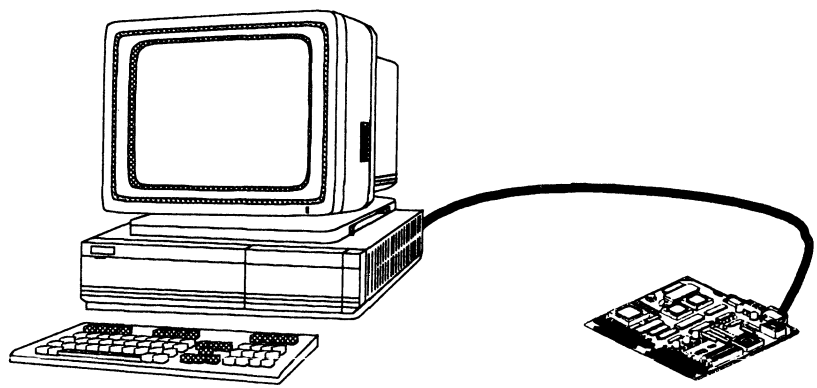
See *Figure 20* for configuration.

The iceMASTER Debug Module is a moderate cost development tool. It has the capability of in-circuit emulation for a specific COP8 microcontroller and in addition serves as a programming tool for COP8 OTP and EPROM product families. Summary of features is as follows:

- Real-time in-circuit emulation; full operating voltage range operation, full DC-10 MHz clock.
- All processor I/O pins can be cabled to an application development board with package compatible cable to socket and surface mount assembly.
- Full 32k byte of loadable programming space that overlays (replaces) the on-chip ROM or EPROM. On-chip RAM and I/O blocks are used directly or recreated as necessary.
- 100 frames of synchronous trace memory. The display can be HLL source (C source), assembly or mixed. The most recent history prior to a break is available in the trace memory.
- Configured break points; uses INTR instruction which is modestly intrusive.
- Software—only supported features are selectable.
- Tool set integrated interactive symbolic debugger—supports both assembler (COFF) and C Compiler (.COD) SDK linked object formats.
- Instruction by instruction memory/register changes displayed when in single step operation.
- Debugger software is processor customized, and reconfigured from a master model file.
- Processor specific symbolic display of registers and bit level assignments, configured from master model file.
- Halt/Idle mode notification.
- Programming menu supports full product line of programmable OTP and EPROM COP8 products. Program data is taken directly from the overlay RAM.
- Programming of 44PLCC 68PLCC parts requires external programming adapters.
- Power supply. (Includes wallmount)
- On-board V_{PP} generator from 5V input or connection to external supply supported. Requires V_{PP} level adjustment per the family programming specification (correct level is provided on an on-screen pop-down display). On-line HELP customized to specific processor using master model file.
- Includes a copy of COP8-DEV-IBMA assembler and linker SDK.

IM Order Information

Debug Module Unit	
COP8-DM/888FH	
Cable Adapters	
DM-COP8/28D	28 DIP
DM-COP8/40D	40 DIP
DM-COP8/44P	44 PLCC
28 DIP to 28 SO Adapter	
DM-COP8/28D-SO	28 SO



TL/DD/12863–24

FIGURE 20. COP8-DM Environment

Development Support (Continued)

IceMASTER EMULATION PROGRAMMING UNIT (EPU)

The iceMASTER EPU-COP888GG is a PC based, in-circuit. simulation tool to support the feature family COP8 products.

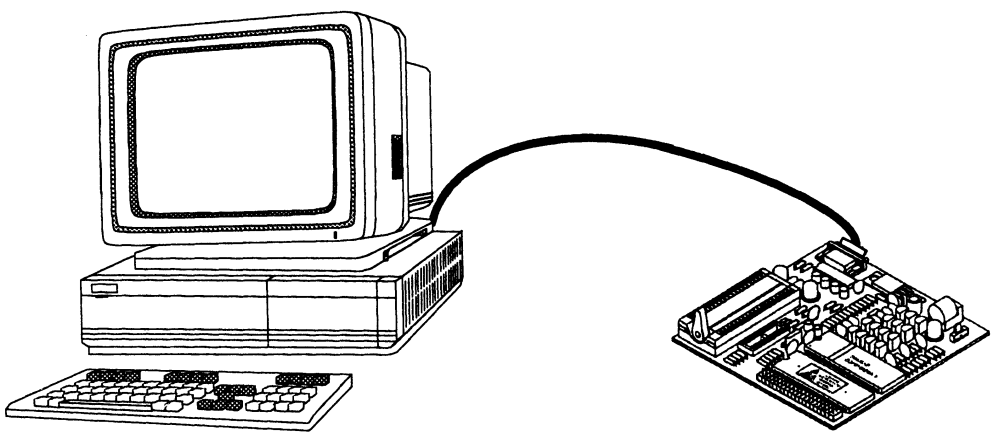
See *Figure 21* for configuration.

The simulation capability is a very low cost means of evaluating the general COP8 architecture. In addition, the EPU has programming capability, with added adapters, for programming the whole COP8 product family of OTP and EPROM products. The product includes the following features:

- Non-real-time in-circuit simulation. Program overlay memory is PC resident; instructions are downloaded over RS-232 as executed. Approximate performance is 20 kHz.
- Includes a 40-pin DIP cable adapter. Other target packages are not supported. All processor I/O pins are cabled to the application development environment.
- Full 32k byte of loadable programming space that overlays (replaces) the on-chip ROM or EPROM. On-chip RAM and I/O blocks are used directly or recreated as necessary.
- On-chip timer and WATCHDOG execution are not well synchronized to the instruction simulation.
- 100 frames of synchronous trace memory. The display can be HLL source (e.g., C source), assembly or mixed. The most recent history prior to a break is available in the trace memory.
- Up to eight software configured break points; uses INTR instruction which is modestly intrusive.
- Common look-feel debugger software across all MetaLink products—only supported features are selectable.
- Tool set integrated interactive symbolic debugger—supports both assembler (COFF) and C Compiler (.COD) SDK linked object formats.
- Instruction by instruction memory/register changes displayed when in single step operation.
- Processor specific symbolic display of registers and bit level assignments, configured from master model file.
- Halt/Idle mode notification. Restart requires special handling.
- Programming menu supports full product line of programmable OTP and EPROM COP8 products. Only a 40 ZIF socket is available on the EPU unit. Adapters are available for other part package configurations.
- Integral wall mount power supply provides 5V and develops the required V_{PP} to program parts.
- Includes a copy of COP8-DEV-IBMA assembler, linker SDK.

EPU Order Information

Evaluation Programming Unit	
EPU-COP888GG	Evaluation Programming Unit with debugger and programmer control software and 40 ZIF programming socket
General Programming Adapters	
COP8-PGMA-DS	28 & 20 DIP and SOIC adapter
COP8-PGMA-DS44P	28 & 20 DIP and SOIC plus 44 PLCC adapter



TL/DD/12863-25

FIGURE 21. EPU-COP8 Tool Environment

Development Support (Continued)

COP8 ASSEMBLER/LINKER SOFTWARE DEVELOPMENT TOOL KIT

National Semiconductor offers a relocateable COP8 macro cross assembler, linker, librarian and utility software development tool kit. Features are summarized as follows:

- Basic and Feature Family instruction set by "device" type.
- Nested macro capability.
- Extensive set of assembler directives.
- Supported on PC/DOS platform.
- Generates National standard COFF output files.
- Integrated Linker and Librarian.
- Integrated utilities to generate ROM code file outputs.
- DUMPCOFF utility.

This product is integrated as a part of MetaLink tools as a development kit, fully supported by the MetaLink debugger. It may be ordered separately or it is bundled with the MetaLink products at no additional cost.

Order Information

Assembler SDK	
COP8-DEV-IBMA	Assembler SDK on installable 3.5" PC/DOS Floppy Disk Drive format. Periodic upgrades and most recent version is available on National's BBS and Internet.

COP8 C COMPILER

A C Compiler is developed and marketed by Byte Craft Limited. The COP8C compiler is a fully integrated development tool specifically designed to support the compact embedded configuration of the COP8 family of products.

Features are summarized as follows:

- ANSI C with some restrictions and extensions that optimize development for the COP8 embedded application.
- BITS data type extension. Register declaration #pragma with direct bit level definitions.
- C language support for interrupt routines.
- Expert system, rule based code geration and optimization.
- Performs consistency checks against the architectural definitions of the target COP8 device.
- Generates program memory code.
- Supports linking of compiled object or COP8 assembled object formats.
- Global optimization of linked code.
- Symbolic debug load format fully source level supported by the MetaLink debugger.

Approved List

Manufacturer	North America	Europe	Asia
BP Microsystems	(800) 225-2102 (713) 688-4600 Fax: (713) 688-0920	+49-8152-4183 +49-8856-932616	+852-234-16611 +852-2710-8121
Data I/O	(800) 426-1045 (206) 881-6444 Fax: (206) 882-1043	+44-0734-440011	Call North America
HI-LO	(510) 623-8860	Call Asia	+886-2-764-0215 Fax: +886-2-756-6403
ICE Technology	(800) 624-8949 (919) 430-7915	+44-1226-767404 Fax: 0-1226-370-434	
MetaLink	(800) 638-2423 (602) 926-0797 Fax: (602) 693-0681	+49-80915696-0 Fax: +49-8091 2386	+852-737-1800
Systems General	(408) 263-6667	+41-1-9450300	+886-2-917-3005 Fax: +886-2-911-1283
Needhams	(916) 924-8037 Fax: (916) 924-8065		

Development Support (Continued)

SINGLE CHIP OTP/EMULATOR SUPPORT

The COP8 family is supported by single chip OTP emulators. For detailed information refer to the emulator specific datasheet and the emulator selection table below:

OTP Emulator Ordering Information

Device Number	Clock Option	Package	Emulates
COP87L84FHN-XE	Crystal	28N	COP884FH
COP87L84FHN-TE	External	28N	COP884FH
COP87L84FHV-XE	Crystal	28 DIP	COP884FH
COP87L84FHV-TE	External	28 DIP	COP884FH
COP87L88FHN-XE	Crystal	40N	COP888FH
COP87L88FHN-TE	External	40N	COP888FH
COP87L88FHV-XE	Crystal	44V	COP888FH
COP87L88FHV-TE	External	44V	COP888FH

INDUSTRY WIDE OTP/EPROM PROGRAMMING SUPPORT

Programming support, in addition to the MetaLink development tools, is provided by a full range of independent approved vendors to meet the needs from the engineering laboratory to full production.

AVAILABLE LITERATURE

For more information, please see the COP8 Basic Family User's Manual, Literature Number 620895, COP8 Feature Family User's Manual Literature Number 620897 and National's Family of 8-bit Microcontrollers COP8 Selection Guide, Literature Number 630009.

DIAL-A-HELPER SERVICE

Dial-A-Helper is a service provided by the Microcontroller Applications group. The Dial-A-Helper is an Electronic Information System that may be accessed as a Bulletin Board System (BBS) via data modem, as an FTP site on the Internet via standard FTP client application or as an FTP site on the Internet using a standard Internet browser such as Netscape or Mosaic.

The Dial-A-Helper system provides access to an automated information storage and retrieval system. The system capabilities include a MESSAGE SECTION (electronic mail, when accessed as a BBS) for communication to and from the Microcontroller Applications Group and a FILE SECTION which consists of several file areas where valuable application software and utilities could be found.

DIAL-A-HELPER BBS via a Standard Modem

Modem: CANADA/U.S.: (800) NSC-MICRO
(800) 672-6427
EUROPE: (+49) 0-8141-351332
Baud: 14.4k
Set-up: Length: 8-Bit
Parity: None
Stop Bit: 1
Operation: 24 Hrs., 7 Days

DIAL-A-HELPER via FTP

ftp nscmicro.nsc.com
user: anonymous
password: username@yourhost.site.domain

DIAL-A-HELPER via a WorldWide Web Browser

ftp://nscmicro.nsc.com

National Semiconductor on the WorldWide Web

See us on the WorldWide Web at: http://www.national.com

CUSTOMER RESPONSE CENTER

Complete product information and technical support is available from National's customer response centers.

CANADA/U.S.:	Tel:	(800) 272-9959
	email:	support @tevm2.nsc.com
EUROPE:	email:	europe.support@nsc.com
	Deutsch Tel:	+49 (0) 180-530 85 85
	English Tel:	+49 (0) 180-532 78 32
	Français Tel:	+49 (0) 180-532 93 58
	Italiano Tel:	+49 (0) 180-534 16 80
JAPAN:	Tel:	+81-043-299-2309
S.E. ASIA:	Beijing Tel:	(+86) 10-6856-8601
	Shanghai Tel:	(+86) 21-6415-4092
	Hong Kong Tel:	(+852) 2737-1600
	Korea Tel:	(+82) 2-3771-6909
	Malaysia Tel:	(+60-4) 644-9061
	Singapore Tel:	(+65) 255-2226
	Taiwan Tel:	+886-2-521-3288
AUSTRALIA:	Tel:	(+61) 3-9558-9999
INDIA:	Tel:	(+91) 80-559-9467

National Semiconductor

PRELIMINARY

COP87L88GG
8-Bit One-Time Programmable (OTP) Microcontroller with UART and Three Multi-Function Timers

General Description

The COP87L88GG OTP microcontroller is a member of the COP8™ feature family using an 8-bit core architecture. It is pin and software compatible to the mask ROM COP888GG product family.

(Continued)

Features

- Full duplex UART
- Three 16-bit timers, each with two 16-bit registers supporting:
  - — Processor independent PWM mode
  - — External event counter mode
  - — Input capture mode
- 16 kbytes on-board OTP EPROM with security feature
- 512 bytes on-board RAM

Additional Peripheral Features

- Idle Timer
- Multi-Input Wakeup (MIWU) with optional interrupts (8)
- WATCHDOG™ and clock monitor logic
- Two analog comparators
- MICROWIRE/PLUS™ serial I/O

I/O Features

- Memory mapped I/O
- Software selectable I/O options (TRI-STATE® output, push-pull output, weak pull-up input, high impedance input)
- Schmitt trigger inputs on ports G and L
- Packages:
  - — 40 DIP with 36 I/O pins
  - — 44 PLCC with 40 I/O pins

CPU/Instruction Set Features

- 1 μs instruction cycle time
- Fourteen multi-source vectored interrupts servicing
  - — External interrupt with selectable edge
  - — Idle Timer T0
  - — Three Timers (each with 2 interrupts)
  - — MICROWIRE/PLUS
  - — Multi-Input Wakeup
  - — Software trap
  - — UART (2)
  - — Default VIS (default interrupt)
- Versatile and easy to use instruction set
- 8-bit Stack Pointer SP—(stack in RAM)
- Two 8-bit register indirect data memory pointers (B and X)

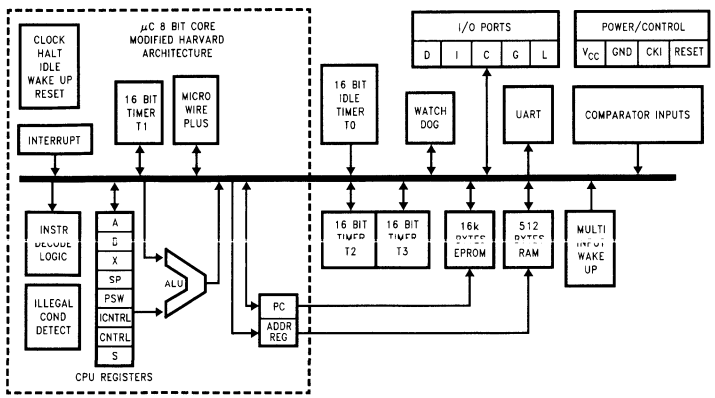
Fully Static CMOS

- Two power saving modes: HALT and IDLE
- Single supply operation: 2.7V–5.5V
- Temperature ranges: −40°C to +85°C

Development Support

- Emulation device for the COP888GG and COP888HG
- Real time emulation and full program debug offered by MetaLink Development System

Block Diagram



TL/DD/12532–1

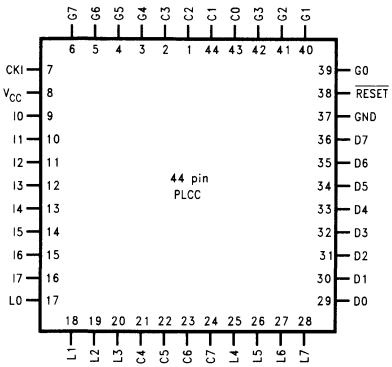
FIGURE 1. Block Diagram

General Description (Continued)

It is a fully static part, fabricated using double-metal silicon gate microCMOS technology. Features include an 8-bit memory mapped architecture, MICROWIRE/PLUS serial I/O, three 16-bit timer/counters supporting three modes (Processor Independent PWM generation, External Event counter, and Input Capture mode capabilities), full duplex UART, two comparators, and two power saving modes (HALT and IDLE), both with a multi-sourced wakeup/interrupt capability. This multi-sourced interrupt capability may also be used independent of the HALT or IDLE modes. Each I/O pin has software selectable configurations. The devices operate over a voltage range of 2.7V to 5.5V. High throughput is achieved with an efficient, regular instruction set operating at a maximum rate of 1 μs per instruction.

Connection Diagrams

Plastic Chip Carrier

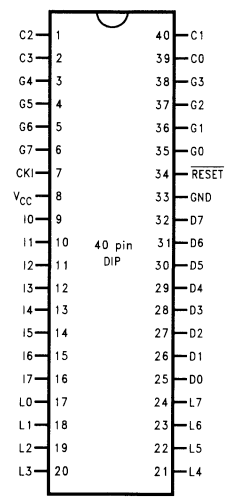


TL/DD/12532-2

Top View

Order Number COP87L88GGV-XE
See NS Package Number V44A

Dual-In-Line Package



TL/DD/12532-3

Top View

Order Number COP87L88GGN-XE
See NS Package Number N40A

Note: -X Crystal Oscillator
-E Halt Enable

FIGURE 2. Connection Diagrams

Connection Diagrams (Continued)

Pinouts for 40- and 44-Pin Packages

Port	Type	Alt. Fun	Alt. Fun	40-Pin Pack.	44-Pin Pack.
L0	I/O	MIWU		17	17
L1	I/O	MIWU	CKX	18	18
L2	I/O	MIWU	TDX	19	19
L3	I/O	MIWU	RDX	20	20
L4	I/O	MIWU	T2A	21	25
L5	I/O	MIWU	T2B	22	26
L6	I/O	MIWU	T3A	23	27
L7	I/O	MIWU	T3B	24	28
G0	I/O	INT		35	39
G1	WDOUT			36	40
G2	I/O	T1B		37	41
G3	I/O	T1A		38	42
G4	I/O	SO		3	3
G5	I/O	SK		4	4
G6	I	SI		5	5
G7	I/CKO	HALT Restart		6	6
D0	O			25	29
D1	O			26	30
D2	O			27	31
D3	O			28	32
I0	I			9	9
I1	I	COMP1IN−		10	10
I2	I	COMP1IN+		11	11
I3	I	COMP1OUT		12	12
I4	I	COMP2IN−		13	13
I5	I	COMP2IN+		14	14
I6	I	COMP2OUT		15	15
I7	I			16	16
D4	O			29	33
D5	O			30	34
D6	O			31	35
D7	O			32	36
C0	I/O			39	43
C1	I/O			40	44
C2	I/O			1	1
C3	I/O			2	2
C4	I/O				21
C5	I/O				22
C6	I/O				23
C7	I/O				24
V_{CC}				8	8
GND				33	37
CKI				7	7
$\overline{\text{RESET}}$				34	38

Absolute Maximum Ratings

If Military/Aerospace specified devices are required, please contact the National Semiconductor Sales Office/Distributors for availability and specifications.

Supply Voltage (V_{CC})	7V
Voltage at Any Pin	−0.3V to V_{CC} + 0.3V
Total Current into V_{CC} Pin (Source)	100 mA
Total Current out of GND Pin (Sink)	110 mA
Storage Temperature Range	−65°C to +140°C

Note: *Absolute maximum ratings indicate limits beyond which damage to the device may occur. DC and AC electrical specifications are not ensured when operating the device at absolute maximum ratings.*

DC Electrical Characteristics $-40°C \le T_A \le +85°C$ unless otherwise specified

Parameter	Conditions	Min	Typ	Max	Units
Operating Voltage		2.7		5.5	V
Power Supply Ripple (Note 1)	Peak-to-Peak			0.1 V_{CC}	V
Supply Current (Note 2)					
CKI = 10 MHz	V_{CC} = 5.5V, t_c = 1 μs			14	mA
CKI = 4 MHz	V_{CC} = 4.0V, t_c = 2.5 μs			4.5	mA
HALT Current (Note 3)	V_{CC} = 5.5V, CKI = 0 MHz			12	μA
	V_{CC} = 4.0V, CKI = 0 MHz			8	μA
IDLE Current (Note 2)					
CKI = 10 MHz	V_{CC} = 5.5V, t_c = 1 μs			3.5	mA
CKI = 1 MHz	V_{CC} = 4.0V, t_c = 10 μs			0.7	mA
Input Levels					
$\overline{\text{RESET}}$					
Logic High		0.8 V_{CC}			V
Logic Low				0.2 V_{CC}	V
CKI, (External and Crystal Osc. Modes)					
Logic High		0.7 V_{CC}			V
Logic Low				0.2 V_{CC}	V
All Other Inputs					
Logic High		0.7 V_{CC}			V
Logic Low				0.2 V_{CC}	V
Hi-Z Input Leakage	V_{CC} = 5.5V	−2		+2	μA
Input Pullup Current	V_{CC} = 5.5V	40		250	μA
G and L Port Input Hysteresis (Note 7)				0.35 V_{CC}	V
Output Current Levels					
D Outputs					
Source	V_{CC} = 4.5V, V_{OH} = 3.3V	0.4			mA
	V_{CC} = 2.7V, V_{OH} = 1.8V	0.2			mA
Sink (Note 4)	V_{CC} = 4.5V, V_{OL} = 1V	10			mA
	V_{CC} = 2.7V, V_{OL} = 0.4V	2.0			mA
All Others					
Source (Weak Pull-Up Mode)	V_{CC} = 4.5V, V_{OH} = 2.7V	10		100	μA
	V_{CC} = 2.7V, V_{OH} = 1.8V	2.5		33	μA
Source (Push-Pull Mode)	V_{CC} = 4.5V, V_{OH} = 3.3V	0.4			mA
	V_{CC} = 2.7V, V_{OH} = 1.8V	0.2			mA
Sink (Push-Pull Mode)	V_{CC} = 4.5V, V_{OL} = 0.4V	1.6			mA
	V_{CC} = 2.7V, V_{OL} = 0.4V	0.7			mA
TRI-STATE Leakage	V_{CC} = 5.5V	−2		+2	μA
Allowable Sink/Source Current per Pin (Note 6)					
D Outputs (Sink)				15	mA
All others				3	mA
Maximum Input Current without Latchup (Note 5)				±200	mA
RAM Retention Voltage, V_r	500 ns Rise and Fall Time (min)	2			V
Input Capacitance				7	pF
Load Capacitance on D2				1000	pF

AC Electrical Characteristics $-40°C \leq T_A \leq +85°C$ unless otherwise specified

Parameter	Conditions	Min	Typ	Max	Units
Instruction Cycle Time (t_c)					
Crystal, Resonator,	$2.7V \leq V_{CC} \leq 4.5V$	2.5		DC	μs
	$4.5V \leq V_{CC} \leq 5.5V$	1		DC	μs
R/C Oscillator	$2.7V \leq V_{CC} \leq 4.5V$	7.5		DC	μs
	$4.5V \leq V_{CC} \leq 5.5V$	3		DC	μs
Inputs					
t_{SETUP}	$4.5V \leq V_{CC} \leq 5.5V$	200			ns
	$2.7V \leq V_{CC} \leq 4.5V$	500			ns
t_{HOLD}	$4.5V \leq V_{CC} \leq 5.5V$	60			ns
	$2.7V \leq V_{CC} \leq 4.5V$	150			ns
Output Propagation Delay (Note 6)	$R_L = 2.2k, C_L = 100$ pF				
t_{PD1}, t_{PD0}					
SO, SK	$4.5V \leq V_{CC} \leq 5.5V$			0.7	μs
	$2.7V \leq V_{CC} \leq 4.5V$			1.75	μs
All Others	$4.5V \leq V_{CC} \leq 5.5V$			1.0	μs
	$2.7V \leq V_{CC} \leq 4.5V$			2.5	μs
MICROWIRE Setup Time (t_{UWS})	$V_{CC} \geq 4.5V$	20			ns
MICROWIRE Hold Time (t_{UWH})	$V_{CC} \geq 4.5V$	56			ns
MICROWIRE Output Propagation Delay (t_{UPD})	$V_{CC} \geq 4.5V$			220	ns
Input Pulse Width (Note 7)					
Interrupt Input High Time		1.0			t_c
Interrupt Input Low Time		1.0			t_c
Timer 1, 2, 3 Input High Time		1.0			t_c
Timer 1, 2, 3 Input Low Time		1.0			t_c
Reset Pulse Width		1.0			μs

t_c = Instruction Cycle Time

Note 1: Maximum rate of voltage change must be < 0.5 V/ms.

Note 2: Supply and IDLE currents are measured with CKI driven with a square wave Oscillator, CKO driven 180° out of phase with CKI, inputs connected to V_{CC} and outputs driven low but not connected to a load.

Note 3: The HALT mode will stop CKI from oscillating in the RC and the Crystal configurations by bringing CKI high. Test Conditions: All inputs tied to V_{CC}, L and G ports in the TRI-STATE mode and tied to ground, all outputs low and tied to ground. The clock monitor is disabled.

Note 4: The user must guarantee that D2 pin does not source more than 10 mA during RESET. If D2 sources more than 10 mA during reset, the device will go into programming mode.

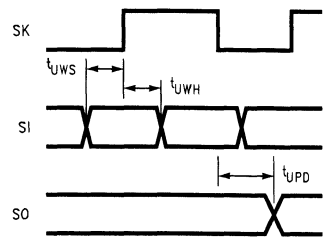
Note 5: Pins G6 and $\overline{\text{RESET}}$ are designed with a high voltage input network. These pins allow input voltages > V_{CC} and the pins will have sink current to V_{CC} when biased at voltages > V_{CC} (the pins do not have source current when biased at a voltage below V_{CC}). The effective resistance to V_{CC} is 750Ω (typical). These two pins will not latch up. The voltage at the pins must be limited to < 14V. **WARNING: Voltages in excess of 14V will cause damage to the pins. This warning excludes ESD transients.**

Note 6: The output propagation delay is referenced to the end of the instruction cycle where the output change occurs.

Note 7: Parameter characterized but not tested.

Comparators AC and DC Characteristics $V_{CC} = 5V, T_A = 25°C.$

Parameter	Conditions	Min	Typ	Max	Units
Input Offset Voltage	$0.4V \le V_{IN} \le V_{CC} - 1.5V$		±10	±25	mV
Input Common Mode Voltage Range		0.4		$V_{CC} - 1.5$	V
Low Level Output Current	$V_{OL} = 0.4V$	1.6			mA
High Level Output Current	$V_{OH} = 4.6V$	1.6			mA
DC Supply Current per Comparator (When Enabled)				250	μA
Response Time	100 pF Load		1		μs



TL/DD/12532–4

FIGURE 3. MICROWIRE/PLUS Timing

Pin Descriptions

V_{CC} and GND are the power supply pins. All V_{CC} and GND pins must be connected.

CKI is the clock input. This can come from an R/C generated oscillator, or a crystal oscillator (in conjunction with CKO). See Oscillator Description section.

$\overline{\text{RESET}}$ is the master reset input. See Reset Description section.

The device contains three bidirectional 8-bit I/O ports (C, G and L), where each individual bit may be independently configured as an input (Schmitt Trigger inputs on ports L and G), output or TRI-STATE under program control. Three data memory address locations are allocated for each of these I/O ports. Each I/O port has two associated 8-bit memory mapped registers, the CONFIGURATION register and the output DATA register. A memory mapped address is also reserved for the input pins of each I/O port. (See the memory map for the various addresses associated with the I/O ports.) *Figure 4* shows the I/O port configurations. The DATA and CONFIGURATION registers allow for each port bit to be individually configured under software control as shown below:

CONFIGURATION Register	DATA Register	Port Set-Up
0	0	Hi-Z Input (TRI-STATE Output)
0	1	Input with Weak Pull-Up
1	0	Push-Pull Zero Output
1	1	Push-Pull One Output

PORT L is an 8-bit I/O port. All L-pins have Schmitt triggers on the inputs.

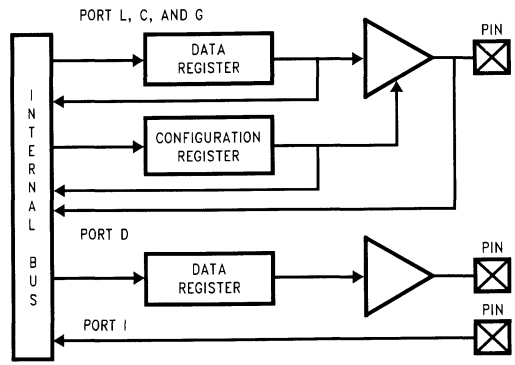
The Port L supports Multi-Input Wake Up on all eight pins. L1 is used for the UART external clock. L2 and L3 are used for the UART transmit and receive. L4 and L5 are used for the timer input functions T2A and T2B. L6 and L7 are used for the timer input functions T3A and T3B.

The Port L has the following alternate features:

- L0 MIWU
- L1 MIWU or CKX
- L2 MIWU or TDX
- L3 MIWU or RDX
- L4 MIWU or T2A
- L5 MIWU or T2B
- L6 MIWU or T3A
- L7 MIWU or T3B

Port G is an 8-bit port with 5 I/O pins (G0, G2–G5), an input pin (G6), and a dedicated output pin (G7). Pins G0 and G2–G6 all have Schmitt Triggers on their inputs. Pin G1 serves as the dedicated WDOUT WATCHDOG output, while pin G7 is either input or output depending on the oscillator mask option selected. With the crystal oscillator option selected, G7 serves as the dedicated output pin for the CKO clock output. With the single-pin R/C oscillator mask option selected, G7 serves as a general purpose input pin but is also used to bring the device out of HALT mode with a low to high transition on G7. There are two registers associated with the G Port, a data register and a configuration register. Therefore, each of the 5 I/O bits (G0, G2–G5) can be individually configured under software control.

Since G6 is an input only pin and G7 is the dedicated CKO clock output pin (crystal clock option) or general purpose input (R/C clock option), the associated bits in the data and configuration registers for G6 and G7 are used for special purpose functions as outlined on the next page. Reading the G6 and G7 data bits will return zeros.



TL/DD/12532–5

FIGURE 4. I/O Port Configurations

Pin Descriptions (Continued)

Note that the chip will be placed in the HALT mode by writing a "1" to bit 7 of the Port G Data Register. Similarly the chip will be placed in the IDLE mode by writing a "1" to bit 6 of the Port G Data Register.

Writing a "1" to bit 6 of the Port G Configuration Register enables the MICROWIRE/PLUS to operate with the alternate phase of the SK clock. The G7 configuration bit, if set high, enables the clock start up delay after HALT when the R/C clock configuration is used.

	Config Reg.	Data Reg.
G7	CLKDLY	HALT
G6	Alternate SK	IDLE

Port G has the following alternate features:

- G0 INTR (External Interrupt Input)
- G2 T1B (Timer T1 Capture Input)
- G3 T1A (Timer T1 I/O)
- G4 SO (MICROWIRE Serial Data Output)
- G5 SK (MICROWIRE Serial Clock)
- G6 SI (MICROWIRE Serial Data Input)

Port G has the following dedicated functions:

- G1 WDOUT WATCHDOG and/or Clock Monitor dedicated output
- G7 CKO Oscillator dedicated output or general purpose input

Port C is an 8-bit I/O port. The 40-pin device does not have a full complement of Port C pins. The unavailable pins are not terminated. A read operation for these unterminated pins will return unpredicatable values.

Port I is an eight-bit Hi-Z input port.

Port I1–I3 are used for Comparator 1. Port I4–I6 are used for Comparator 2.

The Port I has the following alternate features:

- I1 COMP1 − IN (Comparator 1 Negative Input)
- I2 COMP1 + IN (Comparator 1 Positive Input)
- I3 COMP1OUT (Comparator 1 Output)
- I4 COMP2 − IN (Comparator 2 Negative Input)
- I5 COMP2 + IN (Comparator 2 Positive Input)
- I6 COMP2OUT (Comparator 2 Output)

Port D is a recreated 8-bit output port that is preset high when $\overline{\text{RESET}}$ goes low. The user can tie two or more D port outputs (except D2) together in order to get a higher drive.

Functional Description

The architecture of the device is modified Harvard architecture. With the Harvard architecture, the control store program memory (ROM) is separated from the data store memory (RAM). Both ROM and RAM have their own separate addressing space with separate address buses. The architecture, though based on Harvard architecture, permits transfer of data from ROM to RAM.

CPU REGISTERS

The CPU can do an 8-bit addition, subtraction, logical or shift operation in one instruction (t_C) cycle time.

There are six CPU registers:

A is the 8-bit Accumulator Register

PC is the 15-bit Program Counter Register

PU is the upper 7 bits of the program counter (PC)
PL is the lower 8 bits of the program counter (PC)

B is an 8-bit RAM address pointer, which can be optionally post auto incremented or decremented.

X is an 8-bit alternate RAM address pointer, which can be optionally post auto incremented or decremented.

SP is the 8-bit stack pointer, which points to the subroutine/interrupt stack (in RAM). The SP is initialized to RAM address 06F with reset.

S is the 8-bit Data Segment Address Register used to extend the lower half of the address range (00 to 7F) into 256 data segments of 128 bytes each.

All the CPU registers are memory mapped with the exception of the Accumulator (A) and the Program Counter (PC).

PROGRAM MEMORY

The program memory consists of 16 kbytes of OTP EPROM. These bytes may hold program instructions or constant data (data tables for the LAID instruction, jump vectors for the JID instruction, and interrupt vectors for the VIS instruction). The program memory is addressed by the 15-bit program counter (PC). All interrupts in the devices vector to program memory location 0FF Hex.

The device can be configured to inhibit external reads of the program memory. This is done by programming the Security Byte.

SECURITY FEATURE

The program memory array has an associate Security Byte that is located outside of the program address range. This byte can be addressed only from programming mode by a programmer tool.

Security is an optional feature and can only be asserted after the memory array has been programmed and verified. A secured part will read all 00(hex) by a programmer. The part will fail Blank Check and will fail Verify operations. A Read operation will fill the programmer's memory with 00(hex). The Security Byte itself is always readable with value of 00(hex) if unsecure and FF(hex) if secure.

DATA MEMORY

The data memory address space includes the on-chip RAM and data registers, the I/O registers (Configuration, Data and Pin), the control registers, the MICROWIRE/PLUS SIO shift register, and the various registers, and counters associated with the timers (with the exception of the IDLE timer). Data memory is addressed directly by the instruction or indirectly by the B, X, SP pointers and S register.

The data memory consists of 512 bytes of RAM. Sixteen bytes of RAM are mapped as "registers" at addresses 0F0 to 0FF Hex. These registers can be loaded immediately, and also decremented and tested with the DRSZ (decrement register and skip if zero) instruction. The memory pointer registers X, SP, B and S are memory mapped into this space at address locations 0FC to 0FF Hex respectively, with the other registers being available for general usage.

The instruction set permits any bit in memory to be set, reset or tested. All I/O and registers (except A and PC) are memory mapped; therefore, I/O bits and register bits can be directly and individually set, reset and tested. The accumulator (A) bits can also be directly and individually tested.

Note: RAM contents are undefined upon power-up.

Data Memory Segment RAM Extension

Data memory address 0FF is used as a memory mapped location for the Data Segment Address Register (S).

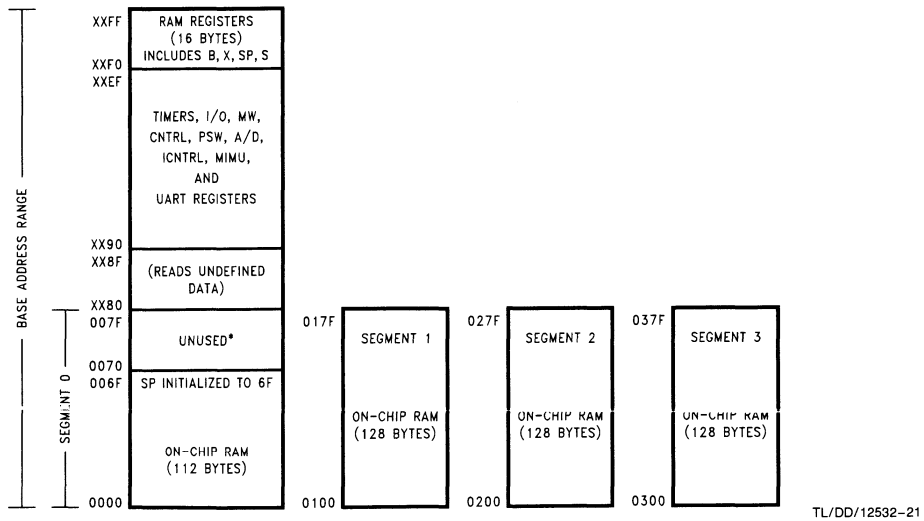
The data store memory is either addressed directly by a single byte address within the instruction, or indirectly relative to the reference of the B, X, or SP pointers (each contains a single-byte address). This single-byte address allows an addressing range of 256 locations from 00 to FF hex. The upper bit of this single-byte address divides the data store memory into two separate sections as outlined previously. With the exception of the RAM register memory from address locations 00F0 to 00FF, all RAM memory is memory mapped with the upper bit of the single-byte address being equal to zero. This allows the upper bit of the single-byte address to determine whether or not the base address range (from 0000 to 00FF) is extended. If this upper bit equals one (representing address range 0080 to 00FF), then address extension does not take place. Alternatively, if this upper bit equals zero, then the data segment extension register S is used to extend the base address range (from 0000 to 007F) from XX00 to XX7F, where XX represents the 8 bits from the S register. Thus the 128-byte data segment extensions are located from addresses 0100 to 017F for data segment 1, 0200 to 027F for data segment 2, etc., up to FF00 to FF7F for data segment 255. The base address range from 0000 to 007F represents data segment 0.

Figure 5 illustrates how the S register data memory extension is used in extending the lower half of the base address range (00 to 7F hex) into 256 data segments of 128 bytes each, with a total addressing range of 32 kbytes from XX00 to XX7F. This organization allows a total of 256 data segments of 128 bytes each with an additional upper base segment of 128 bytes. Furthermore, all addressing modes are available for all data segments. The S register must be changed under program control to move from one data segment (128 bytes) to another. However, the upper base segment (containing the 16 memory registers, I/O registers, control registers, etc.) is always available regardless of the contents of the S register, since the upper base segment (address range 0080 to 00FF) is independent of data segment extension.

The instructions that utilize the stack pointer (SP) always reference the stack as part of the base segment (Segment 0), regardless of the contents of the S register. The S register is not changed by these instructions. Consequently, the stack (used with subroutine linkage and interrupts) is always located in the base segment. The stack pointer will be initialized to point at data memory location 006F as a result of reset.

The 128 bytes of RAM contained in the base segment are split between the lower and upper base segments. The first 112 bytes of RAM are resident from address 0000 to 006F in the lower base segment, while the remaining 16 bytes of RAM represent the 16 data memory registers located at addresses 00F0 to 00FF of the upper base segment. No RAM is located at the upper sixteen addresses (0070 to 007F) of the lower base segment.

Additional RAM beyond these initial 128 bytes, however, will always be memory mapped in groups of 128 bytes (or less) at the data segment address extensions (XX00 to XX7F) of the lower base segment. The additional 128 bytes of RAM are memory mapped at address locations 0100 to 017F hex.



TL/DD/12532-21

*Reads as all ones.

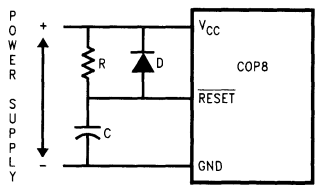
FIGURE 5. RAM Organization

Reset

The $\overline{\text{RESET}}$ input when pulled low initializes the microcontroller. Initialization will occur whenever the $\overline{\text{RESET}}$ input is pulled low. Upon initialization, the data and configuration registers for ports L, G and C are cleared, resulting in these Ports being initialized to the TRI-STATE mode. Pin G1 of the G Port is an exception (as noted below) since pin G1 is dedicated as the WATCHDOG and/or Clock Monitor error output pin. Port D is set high. The PC, PSW, ICNTRL, CNTRL, T2CNTRL and T3CNTRL control registers are cleared. The Comparator Select Register is cleared. The S register is initialized to zero. The Multi-Input Wakeup registers WKEN and WKEDG are cleared. Wakeup register WKPND is unknown. The stack pointer, SP, is initialized to 6F hex.

The device comes out of reset with both the WATCHDOG logic and the Clock Monitor detector armed, with the WATCHDOG service window bits set and the Clock Monitor bit set. The WATCHDOG and Clock Monitor circuits are inhibited during reset. The WATCHDOG service window bits being initialized high default to the maximum WATCHDOG service window of 64k t_C clock cycles. The Clock Monitor bit being initialized high will cause a Clock Monitor error following reset if the clock has not reached the minimum specified frequency at the termination of reset. A Clock Monitor error will cause an active low error output on pin G1. This error output will continue until 16 t_C–32 t_C clock cycles following the clock frequency reaching the minimum specified value, at which time the G1 output will enter the TRI-STATE mode.

The external RC network shown in *Figure 6* should be used to ensure that the $\overline{\text{RESET}}$ pin is held low until the power supply to the chip stabilizes.



TL/DD/12532-7

RC > 5 × Power Supply Rise Time

FIGURE 6. Recommended Reset Circuit

Oscillator Circuits

The chip can be driven by a clock input on the CKI input pin which can be between DC and 10 MHz. The CKO output clock is on pin G7 (crystal configuration). The CKI input frequency is divided down by 10 to produce the instruction cycle clock ($1/t_c$).

Figure 7 shows the Crystal and R/C oscillator diagrams.

CRYSTAL OSCILLATOR

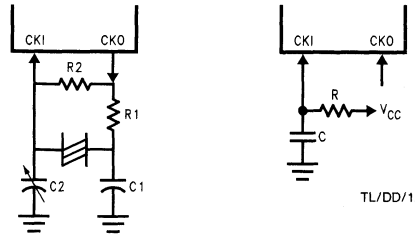
CKI and CKO can be connected to make a closed loop crystal (or resonator) controlled oscillator.

Table A shows the component values required for various standard crystal values.

R/C OSCILLATOR

By selecting CKI as a single pin oscillator input, a single pin R/C oscillator circuit can be connected to it. CKO is available as a general purpose input, and/or HALT restart input.

Table B shows the variation in the oscillator frequencies as functions of the component (R and C) values.



TL/DD/12532-8

TL/DD/12532-9

FIGURE 7. Crystal and R/C Oscillator Diagrams

TABLE A. Crystal Oscillator Configuration, T_A = 25°C

R1 (kΩ)	R2 (MΩ)	C1 (pF)	C2 (pF)	CKI Freq (MHz)	Conditions
0	1	30	30–36	10	V_{CC} = 5V
0	1	30	30–36	4	V_{CC} = 5V
0	1	200	100–150	0.455	V_{CC} = 5V

TABLE B. RC Oscillator Configuration, T_A = 25°C

R (kΩ)	C (pF)	CKI Freq (MHz)	Instr. Cycle (μs)	Conditions
3.3	82	2.2 to 2.7	3.7 to 4.6	V_{CC} = 5V
5.6	100	1.1 to 1.3	7.4 to 9.0	V_{CC} = 5V
6.8	100	0.9 to 1.1	8.8 to 10.8	V_{CC} = 5V

Note: 3k ≤ R ≤ 200k

50 pF ≤ C ≤ 200 pF

2

Control Registers

CNTRL Register (Address X'00EE)

The Timer1 (T1) and MICROWIRE/PLUS control register contains the following bits:

- SL1 & SL0 Select the MICROWIRE/PLUS clock divide by (00 = 2, 01 = 4, 1x = 8)
- IEDG External interrupt edge polarity select (0 = Rising edge, 1 = Falling edge)
- MSEL Selects G5 and G4 as MICROWIRE/PLUS signals SK and SO respectively
- T1C0 Timer T1 Start/Stop control in timer modes 1 and 2
  Timer T1 Underflow Interrupt Pending Flag in timer mode 3
- T1C1 Timer T1 mode control bit
- T1C2 Timer T1 mode control bit
- T1C3 Timer T1 mode control bit

T1C3	T1C2	T1C1	T1C0	MSEL	IEDG	SL1	SL0

Bit 7 Bit 0

PSW Register (Address X'00EF)

The PSW register contains the following select bits:

- GIE Global interrupt enable (enables interrupts)
- EXEN Enable external interrupt
- BUSY MICROWIRE/PLUS busy shifting flag
- EXPND External interrupt pending
- T1ENA Timer T1 Interrupt Enable for Timer Underflow or T1A Input capture edge
- T1PNDA Timer T1 Interrupt Pending Flag (Autoreload RA in mode 1, T1 Underflow in Mode 2, T1A capture edge in mode 3)
- C Carry Flag
- HC Half Carry Flag

HC	C	T1PNDA	T1ENA	EXPND	BUSY	EXEN	GIE

Bit 7 Bit 0

The Half-Carry bit is also affected by all the instructions that affect the Carry flag. The SC (Set Carry) and RC (Reset Carry) instructions will respectively set or clear both the carry flags. In addition to the SC and RC instructions, ADC, SUBC, RRC and RLC instructions affect the carry and Half Carry flags.

ICNTRL Register (Address X'00E8)

The ICNTRL register contains the following bits:

- T1ENB Timer T1 Interrupt Enable for T1B Input capture edge
- T1PNDB Timer T1 Interrupt Pending Flag for T1B capture edge
- μWEN Enable MICROWIRE/PLUS interrupt
- μWPND MICROWIRE/PLUS interrupt pending
- T0EN Timer T0 Interrupt Enable (Bit 12 toggle)
- T0PND Timer T0 Interrupt pending
- LPEN L Port Interrupt Enable (Multi-Input Wakeup/Interrupt)
  Bit 7 could be used as a flag

Unused	LPEN	T0PND	T0EN	μWPND	μWEN	T1PNDB	T1ENB

Bit 7 Bit 0

T2CNTRL Register (Address X'00C6)

The T2CNTRL register contains the following bits:

- T2ENB Timer T2 Interrupt Enable for T2B Input capture edge
- T2PNDB Timer T2 Interrupt Pending Flag for T2B capture edge
- T2ENA Timer T2 Interrupt Enable for Timer Underflow or T2A Input capture edge
- T2PNDA Timer T2 Interrupt Pending Flag (Autoreload RA in mode 1, T2 Underflow in mode 2, T2A capture edge in mode 3)
- T2C0 Timer T2 Start/Stop control in timer modes 1 and 2 Timer T2 Underflow Interrupt Pending Flag in timer mode 3
- T2C1 Timer T2 mode control bit
- T2C2 Timer T2 mode control bit
- T2C3 Timer T2 mode control bit

T2C3	T2C2	T2C1	T2C0	T2PNDA	T2ENA	T2PNDB	T2ENB

Bit 7 Bit 0

T3CNTRL Register (Address X'00B6)

The T3CNTRL register contains the following bits:

- T3ENB Timer T3 Interrupt Enable for T3B Input capture edge
- T3PNDB Timer T3 Interrupt Pending Flag for T3B capture edge
- T3ENA Timer T3 Interrupt Enable for Timer Underflow or T3A Input capture edge
- T3PNDA Timer T3 Interrupt Pending Flag (Autoload RA in mode 1, T3 Underflow in mode 2, T3A capture edge in mode 3)
- T3C0 Timer T3 Start/Stop control in timer modes 1 and 2
  Timer T3 Underflow Interrupt Pending Flag in timer mode 3
- T3C1 Timer T3 mode control bit
- T3C2 Timer T3 mode control bit
- T3C3 Timer T3 mode control bit

T3C3	T3C2	T3C1	T3C0	T3PNDA	T3ENA	T3PNDB	T3ENB

Bit 7 Bit 0

Timers

The device contains a very versatile set of timers (T0, T1, T2, T3). All timers and associated autoreload/capture registers power up containing random data.

TIMER T0 (IDLE TIMER)

The device supports applications that require maintaining real time and low power with the IDLE mode. This IDLE mode support is furnished by the IDLE timer T0, which is a 16-bit timer. The Timer T0 runs continuously at the fixed rate of the instruction cycle clock, t_C. The user cannot read or write to the IDLE Timer T0, which is a count down timer.

The Timer T0 supports the following functions:

- Exit out of the Idle Mode (See Idle Mode description)
- WATCHDOG logic (See WATCHDOG description)
- Start up delay out of the HALT mode

The IDLE Timer T0 can generate an interrupt when the thirteenth bit toggles. This toggle is latched into the T0PND pending flag, and will occur every 4 ms at the maximum clock frequency (t_C = 1 μs). A control flag T0EN allows the interrupt from the thirteenth bit of Timer T0 to be enabled or disabled. Setting T0EN will enable the interrupt, while resetting it will disable the interrupt.

TIMER T1, TIMER T2 AND TIMER T3

The device has a set of three powerful timer/counter blocks, T1, T2 and T3. The associated features and functioning of a timer block are described by referring to the timer block Tx. Since the three timer blocks, T1, T2 and T3 are identical, all comments are equally applicable to any of the three timer blocks.

Each timer block consists of a 16-bit timer, Tx, and two supporting 16-bit autoreload/capture registers, RxA and RxB. Each timer block has two pins associated with it, TxA and TxB. The pin TxA supports I/O required by the timer block, while the pin TxB is an input to the timer block. The powerful and flexible timer block allows the device to easily perform all timer functions with minimal software overhead. The timer block has three operating modes: Processor Independent PWM mode, External Event Counter mode, and Input Capture mode.

The control bits TxC3, TxC2, and TxC1 allow selection of the different modes of operation.

Mode 1. Processor Independent PWM Mode

As the name suggests, this mode allows the device to generate a PWM signal with very minimal user intervention. The user only has to define the parameters of the PWM signal (ON time and OFF time). Once begun, the timer block will continuously generate the PWM signal completely independent of the microcontroller. The user software services the timer block only when the PWM parameters require updating.

In this mode the timer Tx counts down at a fixed rate of t_C. Upon every underflow the timer is alternately reloaded with the contents of supporting registers, RxA and RxB. The very first underflow of the timer causes the timer to reload from the register RxA. Subsequent underflows cause the timer to be reloaded from the registers alternately beginning with the register RxB.

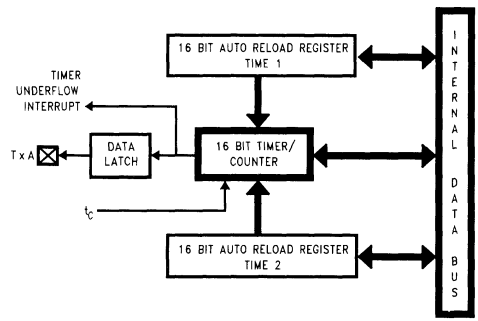
The Tx Timer control bits, TxC3, TxC2 and TxC1 set up the timer for PWM mode operation.

Figure 8 shows a block diagram of the timer in PWM mode.

The underflows can be programmed to toggle the TxA output pin. The underflows can also be programmed to generate interrupts.

Underflows from the timer are alternately latched into two pending flags, TxPNDA and TxPNDB. The user must reset these pending flags under software control. Two control enable flags, TxENA and TxENB, allow the interrupts from the timer underflow to be enabled or disabled. Setting the timer enable flag TxENA will cause an interrupt when a timer underflow causes the RxA register to be reloaded into the timer. Setting the timer enable flag TxENB will cause an interrupt when a timer underflow causes the RxB register to be reloaded into the timer. Resetting the timer enable flags will disable the associated interrupts.

Either or both of the timer underflow interrupts may be enabled. This gives the user the flexibility of interrupting once per PWM period on either the rising or falling edge of the PWM output. Alternatively, the user may choose to interrupt on both edges of the PWM output.



TL/DD/12532–10

FIGURE 8. Timer in PWM Mode

Mode 2. External Event Counter Mode

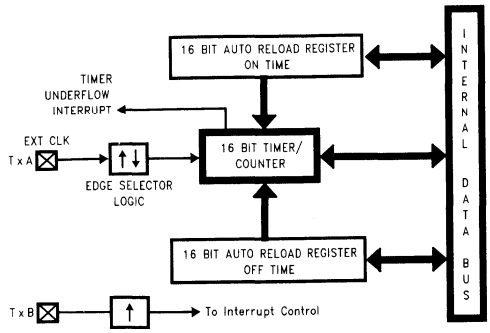
This mode is quite similar to the processor independent PWM mode described above. The main difference is that the timer, Tx, is clocked by the input signal from the TxA pin. The Tx timer control bits, TxC3, TxC2 and TxC1 allow the timer to be clocked either on a positive or negative edge from the TxA pin. Underflows from the timer are latched into the TxPNDA pending flag. Setting the TxENA control flag will cause an interrupt when the timer underflows.

In this mode the input pin TxB can be used as an independent positive edge sensitive interrupt input if the TxENB control flag is set. The occurrence of a positive edge on the TxB input pin is latched into the TxPNDB flag.

Timers (Continued)

Figure 9 shows a block diagram of the timer in External Event Counter mode.

Note: The PWM output is not available in this mode since the TxA pin is being used as the counter input clock.



TL/DD/12532–11

FIGURE 9. Timer in External Event Counter Mode

Mode 3. Input Capture Mode

The device can precisely measure external frequencies or time external events by placing the timer block, Tx, in the input capture mode.

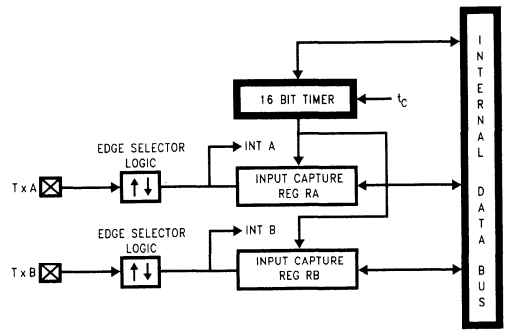
In this mode, the timer Tx is constantly running at the fixed t_C rate. The two registers, RxA and RxB, act as capture registers. Each register acts in conjunction with a pin. The register RxA acts in conjunction with the TxA pin and the register RxB acts in conjunction with the TxB pin.

The timer value gets copied over into the register when a trigger event occurs on its corresponding pin. Control bits, TxC3, TxC2 and TxC1, allow the trigger events to be specified either as a positive or a negative edge. The trigger condition for each input pin can be specified independently.

The trigger conditions can also be programmed to generate interrupts. The occurrence of the specified trigger condition on the TxA and TxB pins will be respectively latched into the pending flags, TxPNDA and TxPNDB. The control flag TxENA allows the interrupt on TxA to be either enabled or disabled. Setting the TxENA flag enables interrupts to be generated when the selected trigger condition occurs on the TxA pin. Similarly, the flag TxENB controls the interrupts from the TxB pin.

Underflows from the timer can also be programmed to generate interrupts. Underflows are latched into the timer TxC0 pending flag (the TxC0 control bit serves as the timer underflow interrupt pending flag in the Input Capture mode). Consequently, the TxC0 control bit should be reset when entering the Input Capture mode. The timer underflow interrupt is enabled with the TxENA control flag. When a TxA interrupt occurs in the Input Capture mode, the user must check both the TxPNDA and TxC0 pending flags in order to determine whether a TxA input capture or a timer underflow (or both) caused the interrupt.

Figure 10 shows a block diagram of the timer in Input Capture mode.



TL/DD/12532–12

FIGURE 10. Timer in Input Capture Mode

TIMER CONTROL FLAGS

The timers T1, T2 and T3 have indentical control structures. The control bits and their functions are summarized below.

TxC0	Timer Start/Stop control in Modes 1 and 2 (Processor Independent PWM and External Event Counter), where 1 = Start, 0 = Stop Timer Underflow Interrupt Pending Flag in Mode 3 (Input Capture)
TxPNDA	Timer Interrupt Pending Flag
TxPNDB	Timer Interrupt Pending Flag
TxENA	Timer Interrupt Enable Flag
TxENB	Timer Interrupt Enable Flag 1 = Timer Interrupt Enabled 0 = Timer Interrupt Disabled
TxC3	Timer mode control
TxC2	Timer mode control
TxC1	Timer mode control

Timers (Continued)

The timer mode control bits (TxC3, TxC2 and TxC1) are detailed below:

TxC3	TxC2	TxC1	Timer Mode	Interrupt A Source	Interrupt B Source	Timer Counts On
0	0	0	MODE 2 (External Event Counter)	Timer Underflow	Pos. TxB Edge	TxA Pos. Edge
0	0	1	MODE 2 (External Event Counter)	Timer Underflow	Pos. TxB Edge	TxA Neg. Edge
1	0	1	MODE 1 (PWM) TxA Toggle	Autoreload RA	Autoreload RB	t_C
1	0	0	MODE 1 (PWM) No TxA Toggle	Autoreload RA	Autoreload RB	t_C
0	1	0	MODE 3 (Capture) Captures: TxA Pos. Edge TxB Pos. Edge	Pos. TxA Edge or Timer Underflow	Pos. TxB Edge	t_C
1	1	0	MODE 3 (Capture) Captures: TxA Pos. Edge TxB Neg. Edge	Pos. TxA Edge or Timer Underflow	Neg. TxB Edge	t_C
0	1	1	MODE 3 (Capture) Captures: TxA Neg. Edge TxB Pos. Edge	Neg. TxA Edge or Timer Underflow	Pos. TxB Edge	t_C
1	1	1	MODE 3 (Capture) Captures: TxA Neg. Edge TxB Neg. Edge	Neg. TxA Edge or Timer Underflow	Neg. TxB Edge	t_C

Power Save Modes

The device offers the user two power save modes of operation: HALT and IDLE. In the HALT mode, all microcontroller activities are stopped. In the IDLE mode, the on-board oscillator circuitry the WATCHDOG logic, the Clock Monitor and timer T0 are active but all other microcontroller activities are stopped. In either mode, all on-board RAM, registers, I/O states, and timers (with the exception of T0) are unaltered.

HALT MODE

The device can be placed in the HALT mode by writing a "1" to the HALT flag (G7 data bit). All microcontroller activities, including the clock and timers, are stopped. The WATCHDOG logic is disabled during the HALT mode. However, the clock monitor circuitry if enabled remains active and will cause the WATCHDOG output pin (WDOUT) to go low. If the HALT mode is used and the user does not want to activate the WDOUT pin, the Clock Monitor should be disabled after the device comes out of reset (resetting the Clock Monitor control bit with the first write to the WDSVR register). In the HALT mode, the power requirements of the device are minimal and the applied voltage (V_{CC}) may be decreased to V_r (V_r = 2.0V) without altering the state of the machine.

The device supports three different ways of exiting the HALT mode. The first method of exiting the HALT mode is with the Multi-Input Wakeup feature on the L port. The second method is with a low to high transition on the CKO (G7) pin. This method precludes the use of the crystal clock configuration (since CKO becomes a dedicated output), and so may be used with an RC clock configuration. The third method of exiting the HALT mode is by pulling the $\overline{\text{RESET}}$ pin low.

Since a crystal or ceramic resonator may be selected as the oscillator, the Wakeup signal is not allowed to start the chip running immediately since crystal oscillators and ceramic resonators have a delayed start up time to reach full amplitude and frequency stability. The IDLE timer is used to generate a fixed delay to ensure that the oscillator has indeed stabilized before allowing instruction execution. In this case, upon detecting a valid Wakeup signal, only the oscillator circuitry is enabled. The IDLE timer is loaded with a value of 256 and is clocked with the t_C instruction cycle clock. The t_C clock is derived by dividing the oscillator clock down by a factor of 10. The Schmitt trigger following the CKI inverter on the chip ensures that the IDLE timer is clocked only when the oscillator has a sufficiently large amplitude to meet the Schmitt trigger specifications. This Schmitt trigger is not part of the oscillator closed loop. The startup timeout from the IDLE timer enables the clock signals to be routed to the rest of the chip.

If an RC clock option is being used, the fixed delay is introduced optionally. A control bit, CLKDLY, mapped as configuration bit G7, controls whether the delay is to be introduced or not. The delay is included if CLKDLY is set, and excluded if CLKDLY is reset. The CLKDLY bit is cleared on reset.

Power Save Modes (Continued)

The device has two mask options associated with the HALT mode. The first mask option enables the HALT mode feature, while the second mask option disables the HALT mode. With the HALT mode enable mask option, the device will enter and exit the HALT mode as described above. With the HALT disable mask option, the device cannot be placed in the HALT mode (writing a "1" to the HALT flag will have no effect, the HALT flag will remain "0").

IDLE MODE

The device is placed in the IDLE mode by writing a "1" to the IDLE flag (G6 data bit). In this mode, all activities, except the associated on-board oscillator circuitry and the IDLE Timer T0, are stopped. The power supply requirements of the micro-controller in this mode of operation are typically around 30% of normal power requirement of the microcontroller.

As with the HALT mode, the device can be returned to normal operation with a reset, or with a Multi-Input Wakeup from the L Port. Alternately, the microcontroller resumes normal operation from the IDLE mode when the thirteenth bit (representing 4.096 ms at internal clock frequency of 1 MHz, t_c = 1 μs) of the IDLE Timer toggles.

This toggle condition of the thirteenth bit of the IDLE Timer T0 is latched into the T0PND pending flag.

The user has the option of being interrupted with a transition on the thirteenth bit of the IDLE Timer T0. The interrupt can be enabled or disabled via the T0EN control bit. Setting the T0EN flag enables the interrupt and vice versa.

The user can enter the IDLE mode with the Timer T0 interrupt enabled. In this case, when the T0PND bit gets set, the device will first execute the Timer T0 interrupt service routine and then return to the instruction following the "Enter Idle Mode" instruction.

Alternatively, the user can enter the IDLE mode with the IDLE Timer T0 interrupt disabled. In this case, the device will resume normal operation with the instruction immediately following the "Enter IDLE Mode" instruction.

Note: It is necessary to program two NOP instructions following both the set HALT mode and set IDLE mode instructions. These NOP instructions are necessary to allow clock resynchronization following the HALT or IDLE modes.

Multi-Input Wakeup

The Multi-Input Wakeup feature is ued to return (wakeup) the device from either the HALT or IDLE modes. Alternately Multi-Input Wakeup/Interrupt feature may also be used to generate up to 8 edge selectable external interrupts.

Figure 11 shows the Multi-Input Wakeup logic.

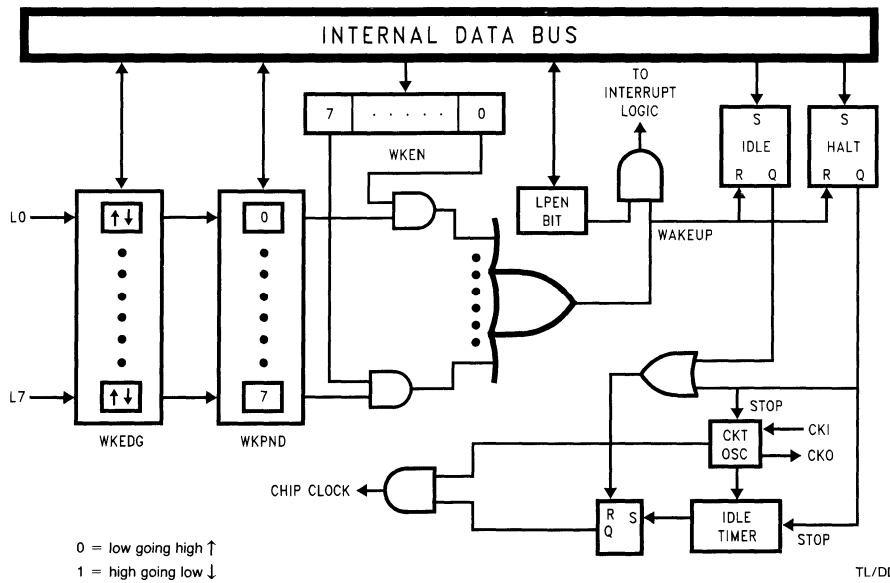


FIGURE 11. Multi-Input Wake Up Logic

Multi-Input Wakeup (Continued)

The Multi-Input Wakeup feature utilizes the L Port. The user selects which particular L port bit (or combination of L Port bits) will cause the device to exit the HALT or IDLE modes. The selection is done through the Reg: WKEN. The Reg: WKEN is an 8-bit read/write register, which contains a control bit for every L port bit. Setting a particular WKEN bit enables a Wakeup from the associated L port pin.

The user can select whether the trigger condition on the selected L Port pin is going to be either a positive edge (low to high transition) or a negative edge (high to low transition). This selection is made via the Reg: WKEDG, which is an 8-bit control register with a bit assigned to each L Port pin. Setting the control bit will select the trigger condition to be a negative edge on that particular L Port pin. Resetting the bit selects the trigger condition to be a positive edge. Changing an edge select entails several steps in order to avoid a pseudo Wakeup condition as a result of the edge change. First, the associated WKEN bit should be reset, followed by the edge select change in WKEDG. Next, the associated WKPND bit should be cleared, followed by the associated WKEN bit being re-enabled.

An example may serve to clarify this procedure. Suppose we wish to change the edge select from positive (low going high) to negative (high going low) for L Port bit 5, where bit 5 has previously been enabled for an input interrupt. The program would be as follows:

```
RBIT  5, WKEN
SBIT  5, WKEDG
RBIT  5, WKPND
SBIT  5, WKEN
```

If the L port bits have been used as outputs and then changed to inputs with Multi-Input Wakeup/Interrupt, a safety procedure should also be followed to avoid inherited pseudo wakeup conditions. After the selected L port bits have been changed from output to input but before the associated WKEN bits are enabled, the associated edge select bits in WKEDG should be set or reset for the desired edge selects, followed by the associated WKPND bits being cleared.

This same procedure should be used following reset, since the L port inputs are left floating as a result of reset.

The occurrence of the selected trigger condition for Multi-Input Wakeup is latched into a pending register called WKPND. The respective bits of the WKPND register will be set on the occurrence of the selected trigger edge on the corresponding Port L pin. The user has the responsibility of clearing these pending flags. Since WKPND is a pending register for the occurrence of selected wakeup conditions, the device will not enter the HALT mode if any Wakeup bit is both enabled and pending. Consequently, the user has the responsibility of clearing the pending flags before attempting to enter the HALT mode.

WKEN, WKPND and WKEDG are all read/write registers, and are cleared at reset.

PORT L INTERRUPTS

Port L provides the user with an additional eight fully selectable, edge sensitive interrupts which are all vectored into the same service subroutine.

The interrupt from Port L shares logic with the wake up circuitry. The register WKEN allows interrupts from Port L to be individually enabled or disabled. The register WKEDG specifies the trigger condition to be either a positive or a negative edge. Finally, the register WKPND latches in the pending trigger conditions.

The GIE (Global Interrupt Enable) bit enables the interrupt function.

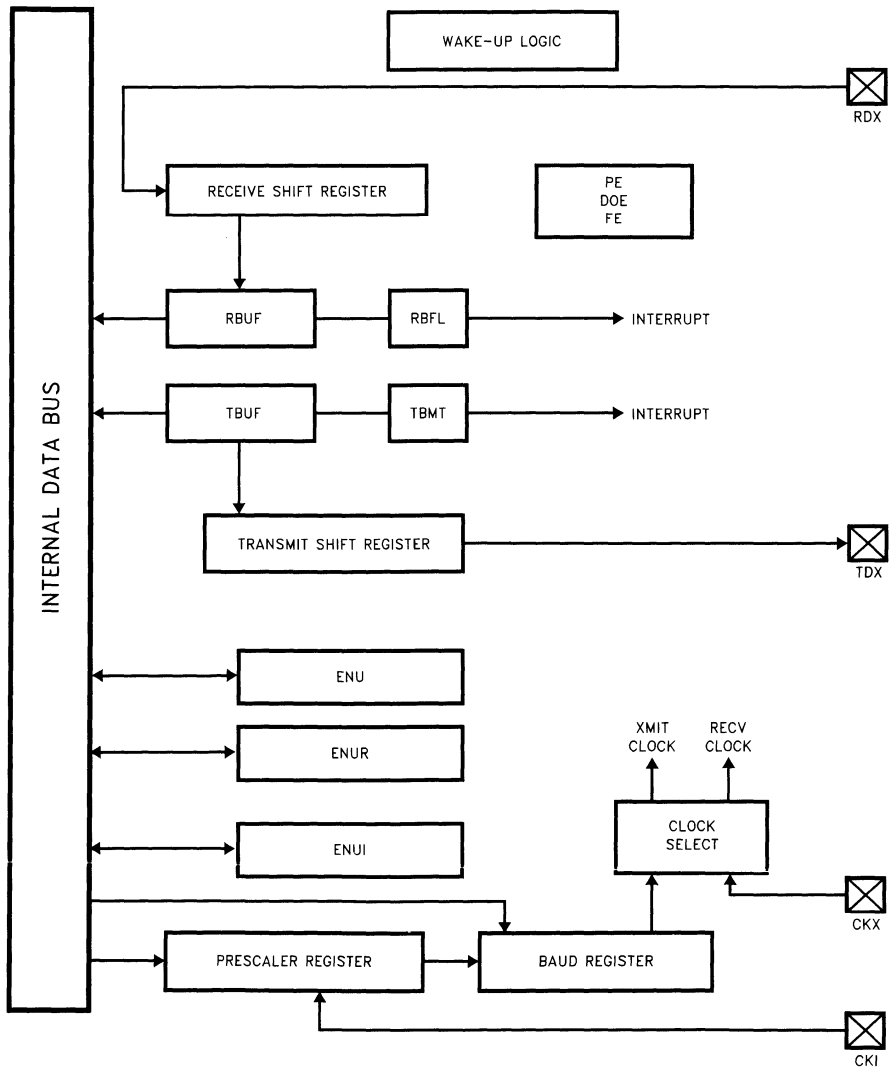
A control flag, LPEN, functions as a global interrupt enable for Port L interrupts. Setting the LPEN flag will enable interrupts and vice versa. A separate global pending flag is not needed since the register WKPND is adequate.

Since Port L is also used for waking the device out of the HALT or IDLE modes, the user can elect to exit the HALT or IDLE modes either with or without the interrupt enabled. If he elects to disable the interrupt, then the device will restart execution from the instruction immediately following the instruction that placed the microcontroller in the HALT or IDLE modes. In the other case, the device will first execute the interrupt service routine and then revert to normal operation. (See HALT MODE for clock option wakeup information.)

UART

The device contains a full-duplex software programmable UART. The UART *(Figure 12)* consists of a transmit shift register, a receive shift register and seven addressable registers, as follows: a transmit buffer register (TBUF), a receiver buffer register (RBUF), a UART control and status register (ENU), a UART receive control and status register (ENUR), a UART interrupt and clock source register (ENUI), a prescaler select register (PSR) and baud (BAUD) register. The ENU register contains flags for transmit and receive functions; this register also determines the length of the data frame (7, 8 or 9 bits), the value of the ninth bit in transmission, and parity selection bits. The ENUR register flags framing, data overrun and parity errors while the UART is receiving.

Other functions of the ENUR register include saving the ninth bit received in the data frame, enabling or disabling the UART's attention mode of operation and providing additional receiver/transmitter status information via RCVG and XMTG bits. The determination of an internal or external clock source is done by the ENUI register, as well as selecting the number of stop bits and enabling or disabling transmit and receive interrupts. A control flag in this register can also select the UART mode of operation: asynchronous or synchronous.



TL/DD/12532-17

FIGURE 12. UART Block Diagram

UART (Continued)

UART CONTROL AND STATUS REGISTERS

The operation of the UART is programmed through three registers: ENU, ENUR and ENUI. The function of the individual bits in these registers is as follows:

ENU-UART Control and Status Register (Address at 0BA)

PEN	PSEL1	XBIT9/ PSEL0	CHL1	CHL0	ERR	RBFL	TBMT
0RW	0RW	0RW	0RW	0RW	0R	0R	1R

Bit 7 Bit 0

ENUR-UART Receive Control and Status Register (Address at 0BB)

DOE	FE	PE	SPARE	RBIT9	ATTN	XMTG	RCVG
0RD	0RD	0RD	0RW*	0R	0RW	0R	0R

Bit7 Bit0

ENUI-UART Interrupt and Clock Source Register (Address at 0BC)

STP2	STP78	ETDX	SSEL	XRCLK	XTCLK	ERI	ETI
0RW	0RW	0RW	0RW	0RW	0RW	0RW	0RW

Bit7 Bit0

*Bit is not used.

0 Bit is cleared on reset.

1 Bit is set to one on reset.

R Bit is read-only; it cannot be written by software.

RW Bit is read/write.

D Bit is cleared on read; when read by software as a one, it is cleared automatically. Writing to the bit does not affect its state.

DESCRIPTION OF UART REGISTER BITS

ENU—UART CONTROL AND STATUS REGISTER

TBMT: This bit is set when the UART transfers a byte of data from the TBUF register into the TSFT register for transmission. It is automatically reset when software writes into the TBUF register.

RBFL: This bit is set when the UART has received a complete character and has copied it into the RBUF register. It is automatically reset when software reads the character from RBUF.

ERR: This bit is a global UART error flag which gets set if any or a combination of the errors (DOE, FE, PE) occur.

CHL1, CHL0: These bits select the character frame format. Parity is not included and is generated/verified by hardware.

CHL1 = 0, CHL0 = 0 The frame contains eight data bits.
CHL1 = 0, CHL0 = 1 The frame contains seven data bits.
CHL1 = 1, CHL0 = 0 The frame contains nine data bits.
CHL1 = 1, CHL0 = 1 Loopback Mode selected. Transmitter output internally looped back to receiver input. Nine bit framing format is used.

XBIT9/PSEL0: Programs the ninth bit for transmission when the UART is operating with nine data bits per frame. For seven or eight data bits per frame, this bit in conjunction with PSEL1 selects parity.

PSEL1, PSEL0: Parity select bits.

PSEL1 = 0, PSEL0 = 0 Odd Parity (if Parity enabled)
PSEL1 = 0, PSEL0 = 1 Odd Parity (if Parity enabled)
PSEL1 = 1, PSEL0 = 0 Mark(1) (if Parity enabled)
PSEL1 = 1, PSEL0 = 1 Space(0) (if Parity enabled)

PEN: This bit enables/disables Parity (7- and 8-bit modes only).

PEN = 0 Parity disabled.
PEN = 1 Parity enabled.

ENUR—UART RECEIVE CONTROL AND STATUS REGISTER

RCVG: This bit is set high whenever a framing error occurs and goes low when RDX goes high.

XMTG: This bit is set to indicate that the UART is transmitting. It gets reset at the end of the last frame (end of last Stop bit).

ATTN: ATTENTION Mode is enabled while this bit is set. This bit is cleared automatically on receiving a character with data bit nine set.

RBIT9: Contains the ninth data bit received when the UART is operating with nine data bits per frame.

SPARE: Reserved for future use.

PE: Flags a Parity Error.

PE = 0 Indicates no Parity Error has been detected since the last time the ENUR register was read.
PE = 1 Indicates the occurrence of a Parity Error.

FE: Flags a Framing Error.

FE = 0 Indicates no Framing Error has been detected since the last time the ENUR register was read.
FE = 1 Indicates the occurrence of a Framing Error.

DOE: Flags a Data Overrun Error.

DOE = 0 Indicates no Data Overrun Error has been detected since the last time the ENUR register was read.
DOE = 1 Indicates the occurrence of a Data Overrun Error.

ENUI—UART INTERRUPT AND CLOCK SOURCE REGISTER

ETI: This bit enables/disables interrupt from the transmitter section.

ETI = 0 Interrupt from the transmitter is disabled.
ETI = 1 Interrupt from the transmitter is enabled.

ERI: This bit enables/disables interrupt from the receiver section.

ERI = 0 Interrupt from the receiver is disabled.
ERI = 1 Interrupt from the receiver is enabled.

XTCLK: This bit selects the clock source for the transmitter section.

XTCLK = 0 The clock source is selected through the PSR and BAUD registers.
XTCLK = 1 Signal on CKX (L1) pin is used as the clock.

XRCLK: This bit selects the clock source for the receiver section.

XRCLK = 0 The clock source is selected through the PSR and BAUD registers.
XRCLK = 1 Signal on CKX (L1) pin is used as the clock.

SSEL: UART mode select.

SSEL = 0 Asynchronous Mode.
SSEL = 1 Synchronous Mode.

UART (Continued)

ETDX: TDX (UART Transmit Pin) is the alternate function assigned to Port L pin L2; it is selected by setting ETDX bit. To simulate line break generation, software should reset ETDX bit and output logic zero to TDX pin through Port L data and configuration registers.

STP78: This bit is set to program the last Stop bit to be 7/8th of a bit in length.

STP2: This bit programs the number of Stop bits to be transmitted.

STP2 = 0 One Stop bit transmitted.
STP2 = 1 Two Stop bits transmitted.

Associated I/O Pins

Data is transmitted on the TDX pin and received on the RDX pin. TDX is the alternate function assigned to Port L pin L2; it is selected by setting ETDX (in the ENUI register) to one. RDX is an inherent function of Port L pin L3, requiring no setup.

The baud rate clock for the UART can be generated on-chip, or can be taken from an external source. Port L pin L1 (CKX) is the external clock I/O pin. The CKX pin can be either an input or an output, as determined by Port L Configuration and Data registers (Bit 1). As an input, it accepts a clock signal which may be selected to drive the transmitter and/or receiver. As an output, it presents the internal Baud Rate Generator output.

UART Operation

The UART has two modes of operation: asynchronous mode and synchronous mode.

ASYNCHRONOUS MODE

This mode is selected by resetting the SSEL (in the ENUI register) bit to zero. The input frequency to the UART is 16 times the baud rate.

The TSFT and TBUF registers double-buffer data for transmission. While TSFT is shifting out the current character on the TDX pin, the TBUF register may be loaded by software with the next byte to be transmitted. When TSFT finishes transmitting the current character the contents of TBUF are transferred to the TSFT register and the Transmit Buffer Empty Flag (TBMT in the ENU register) is set. The TBMT flag is automatically reset by the UART when software loads a new character into the TBUF register. There is also the XMTG bit which is set to indicate that the UART is transmitting. This bit gets reset at the end of the last frame (end of last Stop bit). TBUF is a read/write register.

The RSFT and RBUF registers double-buffer data being received. The UART receiver continually monitors the signal on the RDX pin for a low level to detect the beginning of a Start bit. Upon sensing this low level, it waits for half a bit time and samples again. If the RDX pin is still low, the receiver considers this to be a valid Start bit, and the remaining bits in the character frame are each sampled a single time, at the mid-bit position. Serial data input on the RDX pin is shifted into the RSFT register. Upon receiving the complete character, the contents of the RSFT register are copied into the RBUF register and the Received Buffer Full Flag (RBFL) is set. RBFL is automatically reset when software reads the character from the RBUF register. RBUF is a read only register. There is also the RCVG bit which is set high when a framing error occurs and goes low once RDX goes high. TBMT, XMTG, RBFL and RCVG are read only bits.

SYNCHRONOUS MODE

In this mode data is transferred synchronously with the clock. Data is transmitted on the rising edge and received on the falling edge of the synchronous clock.

This mode is selected by setting SSEL bit in the ENUI register. The input frequency to the UART is the same as the baud rate.

When an external clock input is selected at the CKX pin, data transmit and receive are performed synchronously with this clock through TDX/RDX pins.

If data transmit and receive are selected with the CKX pin as clock output, the device generates the synchronous clock output at the CKX pin. The internal baud rate generator is used to produce the synchronous clock. Data transmit and receive are performed synchronously with this clock.

FRAMING FORMATS

The UART supports several serial framing formats *(Figure 13)*. The format is selected using control bits in the ENU, ENUR and ENUI registers.

The first format (1, 1a, 1b, 1c) for data transmission (CHL0 = 1, CHL1 = 0) consists of Start bit, seven Data bits (excluding parity) and 7/8, one or two Stop bits. In applications using parity, the parity bit is generated and verified by hardware.

The second format (CHL0 = 0, CHL1 = 0) consists of one Start bit, eight Data bits (excluding parity) and 7/8, one or two Stop bits. Parity bit is generated and verified by hardware.

The third format for transmission (CHL0 = 0, CHL1 = 1) consists of one Start bit, nine Data bits and 7/8, one or two Stop bits. This format also supports the UART "ATTENTION" feature. When operating in this format, all eight bits of TBUF and RBUF are used for data. The ninth data bit is transmitted and received using two bits in the ENU and ENUR registers, called XBIT9 and RBIT9. RBIT9 is a read only bit. Parity is not generated or verified in this mode.

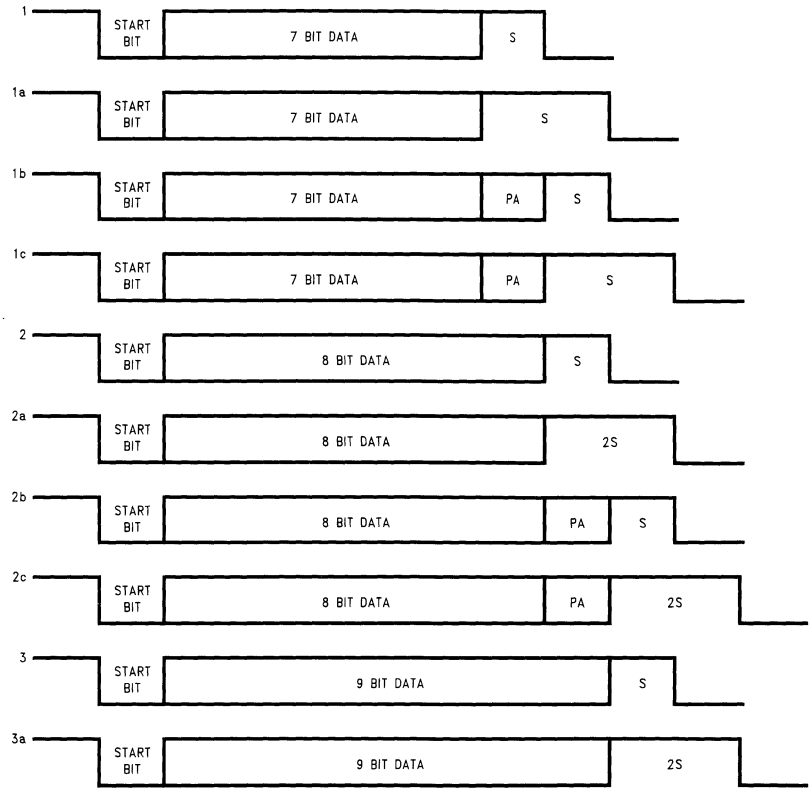
For any of the above framing formats, the last Stop bit can be programmed to be 7/8th of a bit in length. If two Stop bits are selected and the 7/8th bit is set (selected), the second Stop bit will be 7/8th of a bit in length.

The parity is enabled/disabled by PEN bit located in the ENU register. Parity is selected for 7- and 8-bit modes only. If parity is enabled (PEN = 1), the parity selection is then performed by PSEL0 and PSEL1 bits located in the ENU register.

Note that the XBIT9/PSEL0 bit located in the ENU register serves two mutually exclusive functions. This bit programs the ninth bit for transmission when the UART is operating with nine data bits per frame. There is no parity selection in this framing format. For other framing formats XBIT9 is not needed and the bit is PSEL0 used in conjunction with PSEL1 to select parity.

The frame formats for the receiver differ from the transmitter in the number of Stop bits required. The receiver only requires one Stop bit in a frame, regardless of the setting of the Stop bit selection bits in the control register. Note that an implicit assumption is made for full duplex UART operation that the framing formats are the same for the transmitter and receiver.

UART Operation (Continued)



TL/DD/12532–18

FIGURE 13. Framing Formats

UART INTERRUPTS

The UART is capable of generating interrupts. Interrupts are generated on Receive Buffer Full and Transmit Buffer Empty. Both interrupts have individual interrupt vectors. Two bytes of program memory space are reserved for each interrupt vector. The two vectors are located at addresses 0xEC to 0xEF Hex in the program memory space. The interrupts can be individually enabled or disabled using Enable Transmit Interrupt (ETI) and Enable Receive Interrupt (ERI) bits in the ENUI register.

The interrupt from the Transmitter is set pending, and remains pending, as long as both the TBMT and ETI bits are set. To remove this interrupt, software must either clear the ETI bit or write to the TBUF register (thus clearing the TBMT bit).

The interrupt from the receiver is set pending, and remains pending, as long as both the RBFL and ERI bits are set. To remove this interrupt, software must either clear the ERI bit or read from the RBUF register (thus clearing the RBFL bit).

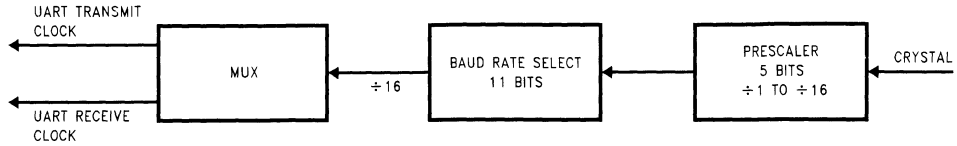
Baud Clock Generation

The clock inputs to the transmitter and receiver sections of the UART can be individually selected to come either from an external source at the CKX pin (port L, pin L1) or from a source selected in the PSR and BAUD registers. Internally, the basic baud clock is created from the oscillator frequency through a two-stage divider chain consisting of a 1–16 (increments of 0.5) prescaler and an 11-bit binary counter. *(Figure 14).* The divide factors are specified through two read/write registers shown in *Figure 15.* Note that the 11-bit Baud Rate Divisor spills over into the Prescaler Select Register (PSR). PSR is cleared upon reset.

As shown in Table V, a Prescaler Factor of 0 corresponds to NO CLOCK. This condition is the UART power down mode where the UART clock is turned off for power saving purpose. The user must also turn the UART clock off when a different baud rate is chosen.

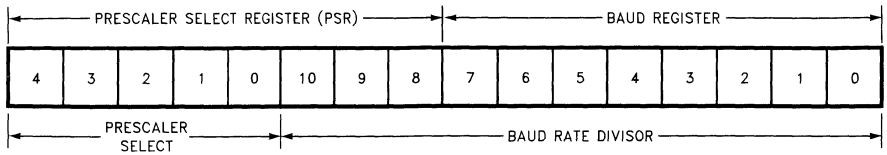
The correspondences between the 5-bit Prescaler Select and Prescaler factors are shown in Table V. There are many ways to calculate the two divisor factors, but one particularly effective method would be to achieve a 1.8432 MHz frequency coming out of the first stage. The 1.8432 MHz prescaler output is then used to drive the software programmable baud rate counter to create a 16x clock for the following baud rates: 110, 134.5, 150, 300, 600, 1200, 1800, 2400, 3600, 4800, 7200, 9600, 19200 and 38400 (Table IV). Other baud rates may be created by using appropriate divisors. The 16x clock is then divided by 16 to provide the rate for the serial shift registers of the transmitter and receiver.

Baud Clock Generation (Continued)



TL/DD/12532-22

FIGURE 14. UART BAUD Clock Generation



TL/DD/12532-20

FIGURE 15. UART BAUD Clock Divisor Registers

Baud Clock Generation (Continued)

TABLE IV. Baud Rate Divisors (1.8432 MHz Prescaler Output)

Baud Rate	Baud Rate Divisor − 1 (N-1)
110 (110.03)	1046
134.5 (134.58)	855
150	767
300	383
600	191
1200	95
1800	63
2400	47
3600	31
4800	23
7200	15
9600	11
19200	5
38400	2

Note: The entries in Table IV assume a prescaler output of 1.8432 MHz. In the asynchronous mode the baud rate could be as high as 625k.

TABLE V. Prescaler Factors

Prescaler Select	Prescaler Factor
00000	NO CLOCK
00001	1
00010	1.5
00011	2
00100	2.5
00101	3
00110	3.5
00111	4
01000	4.5
01001	5
01010	5.5
01011	6
01100	6.5
01101	7
01110	7.5
01111	8
10000	8.5
10001	9
10010	9.5
10011	10
10100	10.5
10101	11
10110	11.5
10111	12
11000	12.5
11001	13
11010	13.5
11011	14
11100	14.5
11101	15
11110	15.5
11111	16

As an example, considering Asynchronous Mode and a CKI clock of 4.608 MHz, the prescaler factor selected is:

$$4.608/1.8432 = 2.5$$

The 2.5 entry is available in Table V. The 1.8432 MHz prescaler output is then used with proper Baud Rate Divisor (Table V) to obtain different baud rates. For a baud rate of 19200 e.g., the entry in Table IV is 5.

$N - 1 = 5$ ($N - 1$ is the value from Table IV)

$N = 6$ (N is the Baud Rate Divisor)

$$\text{Baud Rate} = 1.8432\ \text{MHz}/(16 \times 6) = 19200$$

The divide by 16 is performed because in the asynchronous mode, the input frequency to the UART is 16 times the baud rate. The equation to calculate baud rates is given below.

The actual Baud Rate may be found from:

$$BR = Fc/(16 \times N \times P)$$

Baud Clock Generation (Continued)

Where:

BR is the Baud Rate

Fc is the CKI frequency

N is the Baud Rate Divisor (Table IV).

P is the Prescaler Divide Factor selected by the value in the Prescaler Select Register (Table V)

Note: In the Synchronous Mode, the divisor 16 is replaced by two.

Example:

Asynchronous Mode:

Crystal Frequency = 5 MHz

Desired baud rate = 9600

Using the above equation N × P can be calculated first.

$$N \times P = (5 \times 106)/(16 \times 9600) = 32.552$$

Now 32.552 is divided by each Prescaler Factor (Table III) to obtain a value closest to an integer. This factor happens to be 6.5 (P = 6.5).

$$N = 32.552/6.5 = 5.008 \ (N = 5)$$

The programmed value (from Table IV) should be 4 (N − 1).

Using the above values calculated for N and P:

$$BR = (5 \times 106)/(16 \times 5 \times 6.5) = 9615.384$$

$$\% \text{ error} = (9615.385 - 9600)/9600 = 0.16$$

Effect of HALT/IDLE

The UART logic is reinitialized when either the HALT or IDLE modes are entered. This reinitialization sets the TBMT flag and resets all read only bits in the UART control and status registers. Read/Write bits remain unchanged. The Transmit Buffer (TBUF) is not affected, but the Transmit Shift register (TSFT) bits are set to one. The receiver registers RBUF and RSFT are not affected.

The device will exit from the HALT/IDLE modes when the Start bit of a character is detected at the RDX (L3) pin. This feature is obtained by using the Multi-Input Wakeup scheme provided on the device.

Before entering the HALT or IDLE modes the user program must select the Wakeup source to be on the RDX pin. This selection is done by setting bit 3 of WKEN (Wakeup Enable) register. The Wakeup trigger condition is then selected to be high to low transition. This is done via the WKEDG register (Bit 3 is one.)

If the device is halted and crystal oscillator is used, the Wakeup signal will not start the chip running immediately because of the finite start up time requirement of the crystal oscillator. The idle timer (T0) generates a fixed (256 t_C) delay to ensure that the oscillator has indeed stabilized before allowing the device to execute code. The user has to consider this delay when data transfer is expected immediately after exiting the HALT mode.

Diagnostic

Bits CHARL0 and CHARL1 in the ENU register provide a loopback feature for diagnostic testing of the UART. When these bits are set to one, the following occur: The receiver input pin (RDX) is internally connected to the transmitter output pin (TDX); the output of the Transmitter Shift Register is "looped back" into the Receive Shift Register input. In this mode, data that is transmitted is immediately received. This feature allows the processor to verify the transmit and receive data paths of the UART.

Note that the framing format for this mode is the nine bit format; one Start bit, nine data bits, and 7/8, one or two Stop bits. Parity is not generated or verified in this mode.

Attention Mode

The UART Receiver section supports an alternate mode of operation, referred to as ATTENTION Mode. This mode of operation is selected by the ATTN bit in the ENUR register. The data format for transmission must also be selected as having nine Data bits and either 7/8, one or two Stop bits.

The ATTENTION mode of operation is intended for use in networking the device with other processors. Typically in such environments the messages consists of device addresses, indicating which of several destinations should receive them, and the actual data. This Mode supports a scheme in which addresses are flagged by having the ninth bit of the data field set to a 1. If the ninth bit is reset to a zero the byte is a Data byte.

While in ATTENTION mode, the UART monitors the communication flow, but ignores all characters until an address character is received. Upon receiving an address character, the UART signals that the character is ready by setting the RBFL flag, which in turn interrupts the processor if UART Receiver interrupts are enabled. The ATTN bit is also cleared automatically at this point, so that data characters as well as address characters are recognized. Software examines the contents of the RBUF and responds by deciding either to accept the subsequent data stream (by leaving the ATTN bit reset) or to wait until the next address character is seen (by setting the ATTN bit again).

Operation of the UART Transmitter is not affected by selection of this Mode. The value of the ninth bit to be transmitted is programmed by setting XBIT9 appropriately. The value of the ninth bit received is obtained by reading RBIT9. Since this bit is located in ENUR register where the error flags reside, a bit operation on it will reset the error flags.

Comparators

The device contains two differential comparators, each with a pair of inputs (positive and negative) and an output. Ports I1–I3 and I4–I6 are used for the comparators. The following is the Port I assignment:

- I1 Comparator1 negative input
- I2 Comparator1 positive input
- I3 Comparator1 output
- I4 Comparator2 negative input
- I5 Comparator2 positive input
- I6 Comparator2 output

A Comparator Select Register (CMPSL) is used to enable the comparators, read the outputs of the comparators internally, and enable the outputs of the comparators to the pins. Two control bits (enable and output enable) and one result bit are associated with each comparator. The comparator result bits (CMP1RD and CMP2RD) are read only bits which will read as zero if the associated comparator is not enabled. The Comparator Select Register is cleared with reset, resulting in the comparators being disabled. The comparators should also be disabled before entering either the HALT or IDLE modes in order to save power. The configuration of the CMPSL register is as follows:

Comparators (Continued)

CMPSL REGISTER (ADDRESS X'00B7)

The CMPSL register contains the following bits:

CMP1EN Enable comparator 1

CMP1RD Comparator 1 result (this is a read only bit, which will read as 0 if the comparator is not enabled)

CMP10E Selects pin I3 as comparator 1 output provided that CMPIEN is set to enable the comparator

CMP2EN Enable comparator 2

CMP2RD Comparator 2 result (this is a read only bit, which will read as 0 if the comparator is not enabled)

CMP20E Selects pin I6 as comparator 2 output provided that CMP2EN is set to enable the comparator

Unused	CMP20E	CMP2RD	CMP2EN	CMP10E	CMP1RD	CMP1EN	Unused
Bit 7							Bit 0

Note that the two unused bits of CMPSL may be used as software flags.

Comparator outputs have the same spec as Ports L and G except that the rise and fall times are symmetrical.

Interrupts

The device supports a vectored interrupt scheme. It supports a total of fourteen interrupt sources. The following table lists all the possible interrupt sources, their arbitration ranking and the memory locations reserved for the interrupt vector for each source.

Two bytes of program memory space are reserved for each interrupt source. All interrupt sources except the software interrupt are maskable. Each of the maskable interrupts have an Enable bit and a Pending bit. A maskable interrupt is active if its associated enable and pending bits are set. If GIE = 1 and an interrupt is active, then the processor will be interrupted as soon as it is ready to start executing an instruction except if the above conditions happen during the Software Trap service routine. This exception is described in the Software Trap sub-section.

The interruption process is accomplished with the INTR instruction (opcode 00), which is jammed inside the Instruction Register and replaces the opcode about to be executed. The following steps are performed for every interrupt:

1. The GIE (Global Interrupt Enable) bit is reset.
2. The address of the instruction about to be executed is pushed into the stack.
3. The PC (Program Counter) branches to address 00FF.

This procedure takes 7 t_C cycles to execute.

Arbitration Ranking	Source	Description	Vector* Address Hi-Low Byte
(1) Highest	Software	INTR Instruction	0yFE–0yFF
(2)	Reserved		0yFC–0yFD
(3)	External	G0	0yFA–0yFB
(4)	Timer T0	Underflow	0yF8–0yF9
(5)	Timer T1	T1A/Underflow	0yF6–0yF7
(6)	Timer T1	T1B	0yF4–0yF5
(7)	MICROWIRE/PLUS	BUSY Low	0yF2–0yF3
(8)	Reserved		0yF0–0yF1
(9)	UART	Receive	0yEE–0yEF
(10)	UART	Transmit	0yEC–0yED
(11)	Timer T2	T2A/Underflow	0yEA–0yEB
(12)	Timer T2	T2B	0yE8–0yE9
(13)	Timer T3	T3A/Underflow	0yE6–0yE7
(14)	Timer T3	T3B	0yE4–0yE5
(15)	Port L/Wakeup	Port L Edge	0yE2–0yE3
(16) Lowest	Default VIS	Reserved	0yE0–0yE1

*y is a variable which represents the VIS block. VIS and the vector table must be located in the same 256-byte block except if VIS is located at the last address of a block. In this case, the table must be in the next block.

Interrupts (Continued)

At this time, since GIE = 0, other maskable interrupts are disabled. The user is now free to do whatever context switching is required by saving the context of the machine in the stack with PUSH instructions. The user would then program a VIS (Vector Interrupt Select) instruction in order to branch to the interrupt service routine of the highest priority interrupt enabled and pending at the time of the VIS. Note that this is not necessarily the interrupt that caused the branch to address location 00FF Hex prior to the context switching.

Thus, if an interrupt with a higher rank than the one which caused the interruption becomes active before the decision of which interrupt to service is made by the VIS, then the interrupt with the higher rank will override any lower ones and will be acknowledged. The lower priority interrupt(s) are still pending, however, and will cause another interrupt immediately following the completion of the interrupt service routine associated with the higher priority interrupt just serviced. This lower priority interrupt will occur immediately following the RETI (Return from Interrupt) instruction at the end of the interrupt service routine just completed.

Inside the interrupt service routine, the associated pending bit has to be cleared by software. The RETI (Return from Interrupt) instruction at the end of the interrupt service routine will set the GIE (Global Interrupt Enable) bit, allowing the processor to be interrupted again if another interrupt is active and pending.

The VIS instruction looks at all the active interrupts at the time it is executed and performs an indirect jump to the beginning of the service routine of the one with the highest rank.

The addresses of the different interrupt service routines, called vectors, are chosen by the user and stored in ROM in a table starting at 01E0 (assuming that VIS is located between 00FF and 01DF). The vectors are 15-bit wide and therefore occupy 2 ROM locations.

VIS and the vector table must be located in the same 256-byte block (0y00 to 0yFF) except if VIS is located at the last address of a block. In this case, the table must be in the next block. The vector table cannot be inserted in the first 256-byte block (y ≠ 0).

The vector of the maskable interrupt with the lowest rank is located at 0yE0 (Hi-Order byte) and 0yE1 (Lo-Order byte) and so forth in increasing rank number. The vector of the maskable interrupt with the highest rank is located at 0yFA (Hi-Order byte) and 0yFB (Lo-Order byte).

The Software Trap has the highest rank and its vector is located at 0yFE and 0yFF.

If, by accident, a VIS gets executed and no interrupt is active, then the PC (Program Counter) will branch to a vector located at 0yE0–0yE1.

WARNING

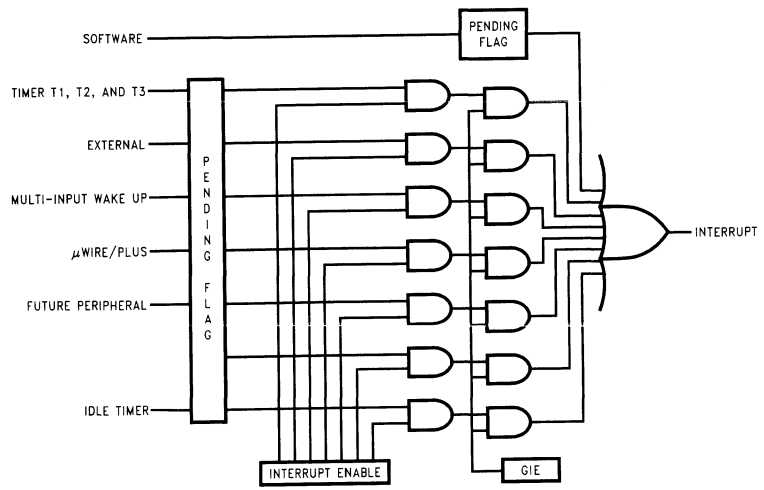
A Default VIS interrupt handler routine must be present. As a minimum, this handler should confirm that the GIE bit is cleared (this indicates that the interrupt sequence has been taken), take care of any required housekeeping, restore context and return. Some sort of Warm Restart procedure should be implemented. These events can occur without any error on the part of the system designer or programmer.

Note: There is always the possibility of an interrupt occurring during an instruction which is attempting to reset the GIE bit or any other interrupt enable bit. If this occurs when a single cycle instruction is being used to reset the interrupt enable bit, the interrupt enable bit will be reset but an interrupt may still occur. This is because interrupt processing is started at the same time as the interrupt bit is being reset. To avoid this scenario, the user should always use a two, three, or four cycle instruction to reset interrupt enable bits.

Figure 16 shows the Interrupt block diagram.

SOFTWARE TRAP

The Software Trap (ST) is a special kind of non-maskable interrupt which occurs when the INTR instruction (used to acknowledge interrupts) is fetched from ROM and placed inside the instruction register. This may happen when the PC is pointing beyond the available ROM address space or when the stack is over-popped.



TL/DD/12532–14

FIGURE 16. Interrupt Block Diagram

Interrupts (Continued)

When an ST occurs, the user can re-initialize the stack pointer and do a recovery procedure (similar to reset, but not necessarily containing all of the same initialization procedures) before restarting.

The occurrence of an ST is latched into the ST pending bit. The GIE bit is not affected and the ST pending bit **(not accessible by the user)** is used to inhibit other interrupts and to direct the program to the ST service routine with the VIS instruction. The RPND instruction is used to clear the software interrupt pending bit. This pending bit is also cleared on reset.

The ST has the highest rank among all interrupts.

Nothing (except another ST) can interrupt an ST being serviced.

WATCHDOG

The device contains a WATCHDOG and clock monitor. The WATCHDOG is designed to detect the user program getting stuck in infinite loops resulting in loss of program control or "runaway" programs. The Clock Monitor is used to detect the absence of a clock or a very slow clock below a specified rate on the CKI pin.

The WATCHDOG consists of two independent logic blocks: WD UPPER and WD LOWER. WD UPPER establishes the upper limit on the service window and WD LOWER defines the lower limit of the service window.

Servicing the WATCHDOG consists of writing a specific value to a WATCHDOG Service Register named WDSVR which is memory mapped in the RAM. This value is composed of three fields, consisting of a 2-bit Window Select, a 5-bit Key Data field, and the 1-bit Clock Monitor Select field. Table II shows the WDSVR register.

TABLE II. WATCHDOG Service Register (WDSVR)

Window Select		Key Data					Clock Monitor
X	X	0	1	1	0	0	Y
7	6	5	4	3	2	1	0

The lower limit of the service window is fixed at 2048 instruction cycles. Bits 7 and 6 of the WDSVR register allow the user to pick an upper limit of the service window.

Table III shows the four possible combinations of lower and upper limits for the WATCHDOG service window. This flexibility in choosing the WATCHDOG service window prevents any undue burden on the user software.

Bits 5, 4, 3, 2 and 1 of the WDSVR register represent the 5-bit Key Data field. The key data is fixed at 01100. Bit 0 of the WDSVR Register is the Clock Monitor Select bit.

TABLE III. WATCHDOG Service Window Select

WDSVR Bit 7	WDSVR Bit 6	Service Window (Lower-Upper Limits)
0	0	2k–8k t_C Cycles
0	1	2k–16k t_C Cycles
1	0	2k–32k t_C Cycles
1	1	2k–64k t_C Cycles

Clock Monitor

The Clock Monitor aboard the device can be selected or deselected under program control. The Clock Monitor is guaranteed not to reject the clock if the instruction cycle clock ($1/t_C$) is greater or equal to 10 kHz. This equates to a clock input rate on CKI of greater or equal to 100 kHz.

WATCHDOG Operation

The WATCHDOG and Clock Monitor are disabled during reset. The device comes out of reset with the WATCHDOG armed, the WATCHDOG Window Select bits (bits 6, 7 of the WDSVR Register) set, and the Clock Monitor bit (bit 0 of the WDSVR Register) enabled. Thus, a Clock Monitor error will occur after coming out of reset, if the instruction cycle clock frequency has not reached a minimum specified value, including the case where the oscillator fails to start.

The WDSVR register can be written to only once after reset and the key data (bits 5 through 1 of the WDSVR Register) must match to be a valid write. This write to the WDSVR register involves two irrevocable choices: (i) the selection of the WATCHDOG service window (ii) enabling or disabling of the Clock Monitor. Hence, the first write to WDSVR Register involves selecting or deselecting the Clock Monitor, select the WATCHDOG service window and match the WATCHDOG key data. Subsequent writes to the WDSVR register will compare the value being written by the user to the WATCHDOG service window value and the key data (bits 7 through 1) in the WDSVR Register. Table IV shows the sequence of events that can occur.

The user must service the WATCHDOG at least once before the upper limit of the service window expires. The WATCHDOG may not be serviced more than once in every lower limit of the service window. The user may service the WATCHDOG as many times as wished in the time period between the lower and upper limits of the service window. The first write to the WDSVR Register is also counted as a WATCHDOG service.

The WATCHDOG has an output pin associated with it. This is the WDOUT pin, on pin 1 of the port G. WDOUT is active low. The WDOUT pin is in the high impedance state in the inactive state. Upon triggering the WATCHDOG, the logic will pull the WDOUT (G1) pin low for an additional 16 t_C–32 t_C cycles after the signal level on WDOUT pin goes below the lower Schmitt trigger threshold. After this delay, the device will stop forcing the WDOUT output low.

The WATCHDOG service window will restart when the WDOUT pin goes high. It is recommended that the user tie the WDOUT pin back to V_{CC} through a resistor in order to pull WDOUT high.

A WATCHDOG service while the WDOUT signal is active will be ignored. The state of the WDOUT pin is not guaranteed on reset, but if it powers up low then the WATCHDOG will time out and WDOUT will enter high impedance state.

The Clock Monitor forces the G1 pin low upon detecting a clock frequency error. The Clock Monitor error will continue until the clock frequency has reached the minimum specified value, after which the G1 output will enter the high impedance TRI-STATE mode following 16 t_C–32 t_C clock cycles. The Clock Monitor generates a continual Clock Monitor error if the oscillator fails to start, or fails to reach the minimum specified frequency. The specification for the Clock Monitor is as follows:

$1/t_C > 10$ kHz—No clock rejection.

$1/t_C < 10$ Hz—Guaranteed clock rejection.

WATCHDOG Operation (Continued)

WATCHDOG AND CLOCK MONITOR SUMMARY

The following salient points regarding the WATCHDOG and CLOCK MONITOR should be noted:

- Both the WATCHDOG and CLOCK MONITOR detector circuits are inhibited during RESET.
- Following RESET, the WATCHDOG and CLOCK MONITOR are both enabled, with the WATCHDOG having he maximum service window selected.
- The WATCHDOG service window and CLOCK MONITOR enable/disable option can only be changed once, during the initial WATCHDOG service following RESET.
- The initial WATCHDOG service must match the key data value in the WATCHDOG Service register WDSVR in order to avoid a WATCHDOG error.
- Subsequent WATCHDOG services must match all three data fields in WDSVR in order to avoid WATCHDOG errors.
- The correct key data value cannot be read from the WATCHDOG Service register WDSVR. Any attempt to read this key data value of 01100 from WDSVR will read as key data value of all 0's.
- The WATCHDOG detector circuit is inhibited during both the HALT and IDLE modes.
- The CLOCK MONITOR detector circuit is active during both the HALT and IDLE modes. Consequently, the device inadvertently entering the HALT mode will be detected as a CLOCK MONITOR error (provided that the CLOCK MONITOR enable option has been selected by the program).
- With the single-pin R/C oscillator mask option selected and the CLKDLY bit reset, the WATCHDOG service window will resume following HALT mode from where it left off before entering the HALT mode.
- With the crystal oscillator mask option selected, or with the single-pin R/C oscillator mask option selected and the CLKDLY bit set, the WATCHDOG service window will be set to its selected value from WDSVR following HALT. Consequently, the WATCHDOG should not be serviced for at least 2048 instruction cycles following HALT, but must be serviced within the selected window to avoid a WATCHDOG error.
- The IDLE timer T0 is not initialized with RESET.
- The user can sync in to the IDLE counter cycle with an IDLE counter (T0) interrupt or by monitoring the T0PND flag. The T0PND flag is set whenever the thirteenth bit of the IDLE counter toggles (every 4096 instruction cycles). The user is responsible for resetting the T0PND flag.
- A hardware WATCHDOG service occurs just as the device exits the IDLE mode. Consequently, the WATCHDOG should not be serviced for at least 2048 instruction cycles following IDLE, but must be serviced within the selected window to avoid a WATCHDOG error.
- Following RESET, the initial WATCHDOG service (where the service window and the CLOCK MONITOR enable/disable must be selected) may be programmed anywhere within the maximum service window (65,536 instruction cycles) initialized by RESET. Note that this initial WATCHDOG service may be programmed within the initial 2048 instruction cycles without causing a WATCHDOG error.

Detection of Illegal Conditions

The device can detect various illegal conditions resulting from coding errors, transient noise, power supply voltage drops, runaway programs, etc.

Reading of undefined ROM gets zeros. The opcode for software interrupt is zero. If the program fetches instructions from undefined ROM, this will force a software interrupt, thus signaling that an illegal condition has occurred.

The subroutine stack grows down for each call (jump to subroutine), interrupt, or PUSH, and grows up for each return or POP. The stack pointer is initialized to RAM location 06F Hex during reset. Consequently, if there are more returns than calls, the stack pointer will point to addresses 070 and 071 Hex (which are undefined RAM). Undefined RAM from addresses 070 to 07F (Segment 0), and all other segments (i.e., Segments 4 . . . etc.) is read as all 1's, which in turn will cause the program to return to address 7FFF Hex. This is an undefined ROM location and the instruction fetched (all 0's) from this location will generate a software interrupt signaling an illegal condition.

Thus, the chip can detect the following illegal conditions:

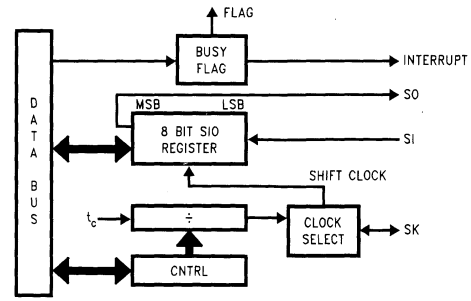
a. Executing from undefined ROM

b. Over "POP"ing the stack by having more returns than calls.

When the software interrupt occurs, the user can re-initialize the stack pointer and do a recovery procedure before restarting (this recovery program is probably similar to that following reset, but might not contain the same program initialization procedures). The recovery program should reset the software interrupt pending bit using the RPND instruction.

MICROWIRE/PLUS

MICROWIRE/PLUS is a serial synchronous communications interface. The MICROWIRE/PLUS capability enables the device to interface with any of National Semiconductor's MICROWIRE peripherals (i.e. A/D converters, display drivers, E^{2}PROMs etc.) and with other microcontrollers which support the MICROWIRE interface. It consists of an 8-bit serial shift register (SIO) with serial data input (SI), serial data output (SO) and serial shift clock (SK). *Figure 17* shows a block diagram of the MICROWIRE/PLUS logic.



TL/DD/12532–15

FIGURE 17. MICROWIRE/PLUS Block Diagram

The shift clock can be selected from either an internal source or an external source. Operating the MICROWIRE/PLUS arrangement with the internal clock source is called the Master mode of operation. Similarly, operating the MICROWIRE/PLUS arrangement with an external shift clock is called the Slave mode of operation.

The CNTRL register is used to configure and control the MICROWIRE/PLUS mode. To use the MICROWIRE/PLUS, the MSEL bit in the CNTRL register is set to one. In the master mode, the SK clock rate is selected by the two bits, SL0 and SL1, in the CNTRL register. Table V details the different clock rates that may be selected.

TABLE IV. WATCHDOG Service Actions

Key Data	Window Data	Clock Monitor	Action
Match	Match	Match	Valid Service: Restart Service Window
Don't Care	Mismatch	Don't Care	Error: Generate WATCHDOG Output
Mismatch	Don't Care	Don't Care	Error: Generate WATCHDOG Output
Don't Care	Don't Care	Mismatch	Error: Generate WATCHDOG Output

TABLE V. MICROWIRE/PLUS Master Mode Clock Select

SL1	SL0	SK
0	0	$2 \times t_c$
0	1	$4 \times t_c$
1	x	$8 \times t_c$

Where t_c is the instruction cycle clock

MICROWIRE/PLUS (Continued)

MICROWIRE/PLUS OPERATION

Setting the BUSY bit in the PSW register causes the MICROWIRE/PLUS to start shifting the data. It gets reset when eight data bits have been shifted. The user may reset the BUSY bit by software to allow less than 8 bits to shift. If enabled, an interrupt is generated when eight data bits have been shifted. The device may enter the MICROWIRE/PLUS mode either as a Master or as a Slave. *Figure 18* shows how two microcontroller devices and several peripherals may be interconnected using the MICROWIRE/PLUS arrangements.

Warning:

The SIO register should only be loaded when the SK clock is low. Loading the SIO register while the SK clock is high will result in undefined data in the SIO register. SK clock is normally low when not shifting.

Setting the BUSY flag when the input SK clock is high in the MICROWIRE/PLUS slave mode may cause the current SK clock for the SIO shift register to be narrow. For safety, the BUSY flag should only be set when the input SK clock is low.

MICROWIRE/PLUS Master Mode Operation

In the MICROWIRE/PLUS Master mode of operation the shift clock (SK) is generated internally by the device. The MICROWIRE Master always initiates all data exchanges. The MSEL bit in the CNTRL register must be set to enable the SO and SK functions onto the G Port. The SO and SK pins must also be selected as outputs by setting appropriate bits in the Port G configuration register. Table VI summarizes the bit settings required for Master mode of operation.

MICROWIRE/PLUS Slave Mode Operation

In the MICROWIRE/PLUS Slave mode of operation the SK clock is generated by an external source. Setting the MSEL bit in the CNTRL register enables the SO and SK functions onto the G Port. The SK pin must be selected as an input and the SO pin is selected as an output pin by setting and resetting the appropriate bits in the Port G configuration register. Table VI summarizes the settings required to enter the Slave mode of operation.

The user must set the BUSY flag immediately upon entering the Slave mode. This will ensure that all data bits sent by the Master will be shifted properly. After eight clock pulses the BUSY flag will be cleared and the sequence may be repeated.

Alternate SK Phase Operation

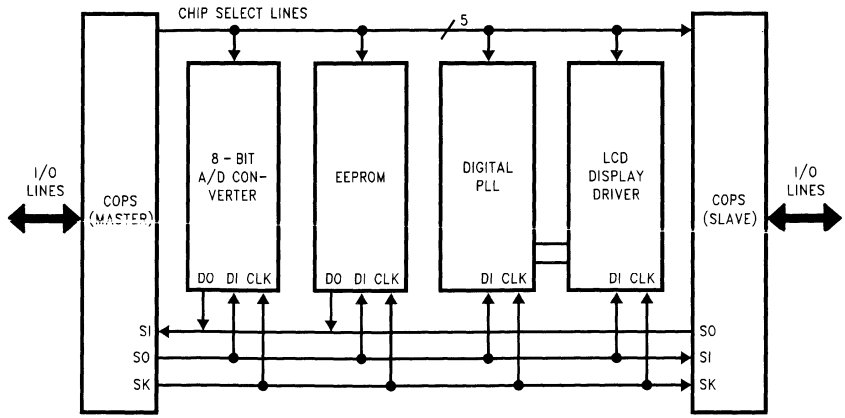
The device allows either the normal SK clock or an alternate phase SK clock to shift data in and out of the SIO register. In both the modes the SK is normally low. In the normal mode data is shifted in on the rising edge of the SK clock and the data is shifted out on the falling edge of the SK clock. The SIO register is shifted on each falling edge of the SK clock. In the alternate SK phase operation, data is shifted in on the falling edge of the SK clock and shifted out on the rising edge of the SK clock.

A control flag, SKSEL, allows either the normal SK clock or the alternate SK clock to be selected. Resetting SKSEL causes the MICROWIRE/PLUS logic to be clocked from the normal SK signal. Setting the SKSEL flag selects the alternate SK clock. The SKSEL is mapped into the G6 configuration bit. The SKSEL flag will power up in the reset condition, selecting the normal SK signal.

TABLE VI

This table assumes that the control flag MSEL is set.

G4 (SO) Config. Bit	G5 (SK) Config. Bit	G4 Fun.	G5 Fun.	Operation
1	1	SO	Int. SK	MICROWIRE/PLUS Master
0	1	TRI-STATE	Int. SK	MICROWIRE/PLUS Master
1	0	SO	Ext. SK	MICROWIRE/PLUS Slave
0	0	TRI-STATE	Ext. SK	MICROWIRE/PLUS Slave



TL/DD/12532–23

FIGURE 18. MICROWIRE/PLUS Application

Memory Map

All RAM, ports and registers (except A and PC) are mapped into data memory address space.

Address S/ADD REG	Contents
0000 to 006F	On-Chip RAM bytes (112 bytes)
0070 to 007F	Unused RAM Address Space (Reads As All Ones)
xx80 to xxAF	Unused RAM Address Space (Reads Undefined Data)
xxB0	Timer T3 Lower Byte
XXB1	Timer T3 Upper Byte
xxB2	Timer T3 Autoload Register T3RA Lower Byte
xxB3	Timer T3 Autoload Register T3RA Upper Byte
xxB4	Timer T3 Autoload Register T3RB Lower Byte
xxB5	Timer T3 Autoload Register T3RB Upper Byte
xxB6	Timer T3 Control Register
xxB7	Comparator Select Register (CMPSL)
xxB8	UART Transmit Buffer (TBUF)
xxB9	UART Receive Buffer (RBUF)
xxBA	UART Control and Status Register (ENU)
xxBB	UART Receive Control and Status Register (ENUR)
xxBC	UART Interrupt and Clock Source Register (ENUI)
xxBD	UART Baud Register (BAUD)
xxBE	UART Prescale Select Register (PSR)
xxBF	Reserved for UART
xxC0	Timer T2 Lower Byte
xxC1	Timer T2 Upper Byte
xxC2	Timer T2 Autoload Register T2RA Lower Byte
xxC3	Timer T2 Autoload Register T2RA Upper Byte
xxC4	Timer T2 Autoload Register T2RB Lower Byte
xxC5	Timer T2 Autoload Register T2RB Upper Byte
xxC6	Timer T2 Control Register
xxC7	WATCHDOG Service Register (Reg:WDSVR)
xxC8	MIWU Edge Select Register (Reg:WKEDG)
xxC9	MIWU Enable Register (Reg:WKEN)
xxCA	MIWU Pending Register (Reg:WKPND)
xxCB	Reserved
xxCC	Reserved
xxCD to xxCF	Reserved

Address S/ADD REG	Contents
xxD0	Port L Data Register
xxD1	Port L Configuration Register
xxD2	Port L Input Pins (Read Only)
xxD3	Reserved for Port L
xxD4	Port G Data Register
xxD5	Port G Configuration Register
xxD6	Port G Input Pins (Read Only)
xxD7	Port I Input Pins (Read Only)
xxD8	Port C Data Register
xxD9	Port C Configuration Register
xxDA	Port C Input Pins (Read Only)
xxDB	Reserved for Port C
xxDC	Port D
xxDD to xxDF	Reserved for Port D
xxE0 to xxE5	Reserved for EE Control Registers
xxE6	Timer T1 Autoload Register T1RB Lower Byte
xxE7	Timer T1 Autoload Register T1RB Upper Byte
xxE8	ICNTRL Register
xxE9	MICROWIRE/PLUS Shift Register
xxEA	Timer T1 Lower Byte
xxEB	Timer T1 Upper Byte
xxEC	Timer T1 Autoload Register T1RA Lower Byte
xxED	Timer T1 Autoload Register T1RA Upper Byte
xxEE	CNTRL Control Register
xxEF	PSW Register
xxF0 to FB	On-Chip RAM Mapped as Registers
xxFC	X Register
xxFD	SP Register
xxFE	B Register
xxFF	S Register
0100 to 017F	On-Chip 128 RAM Bytes
0200 to 027F	On-Chip 128 RAM Bytes
0300 to 037F	On-Chip 128 RAM Bytes

Reading memory locations 0070H–007FH (Segment 0) will return all ones. Reading unused memory locations 0080H–00AFH (Segment 0) will return undefined data. Reading memory locations from other Segments (i.e., Segment 4, Segment 5, ... etc.) will return all ones.

Addressing Modes

There are ten addressing modes, six for operand addressing and four for transfer of control.

OPERAND ADDRESSING MODES

Register Indirect

This is the "normal" addressing mode. The operand is the data memory addressed by the B pointer or X pointer.

Register Indirect (with auto post increment or decrement of pointer)

This addressing mode is used with the LD and X instructions. The operand is the data memory addressed by the B pointer or X pointer. This is a register indirect mode that automatically post increments or decrements the B or X register after executing the instruction.

Direct

The instruction contains an 8-bit address field that directly points to the data memory for the operand.

Immediate

The instruction contains an 8-bit immediate field as the operand.

Short Immediate

This addressing mode is used with the Load B Immediate instruction. The instruction contains a 4-bit immediate field as the operand.

Indirect

This addressing mode is used with the LAID instruction. The contents of the accumulator are used as a partial address (lower 8 bits of PC) for accessing a data operand from the program memory.

TRANSFER OF CONTROL ADDRESSING MODES

Relative

This mode is used for the JP instruction, with the instruction field being added to the program counter to get the new program location. JP has a range from −31 to +32 to allow a 1-byte relative jump (JP + 1 is implemented by a NOP instruction). There are no "pages" when using JP, since all 15 bits of PC are used.

Absolute

This mode is used with the JMP and JSR instructions, with the instruction field of 12 bits replacing the lower 12 bits of the program counter (PC). This allows jumping to any location in the current 4k program memory segment.

Absolute Long

This mode is used with the JMPL and JSRL instructions, with the instruction field of 15 bits replacing the entire 15 bits of the program counter (PC). This allows jumping to any location up to 32k in the program memory space.

Indirect

This mode is used with the JID instruction. The contents of the accumulator are used as a partial address (lower 8 bits of PC) for accessing a location in the program memory. The contents of this program memory location serve as a partial address (lower 8 bits of PC) for the jump to the next instruction.

Note: The VIS is a special case of the Indirect Transfer of Control addressing mode, where the double byte vector associated with the interrupt is transferred from adjacent addresses in the program memory into the program counter (PC) in order to jump to the associated interrupt service routine.

Instruction Set

Register and Symbol Definition

Registers	
A	8-Bit Accumulator Register
B	8-Bit Address Register
X	8-Bit Address Register
S	8-Bit Segment Register
SP	8-Bit Stack Pointer Register
PC	15-Bit Program Counter Register
PU	Upper 7 Bits of PC
PL	Lower 8 Bits of PC
C	1 Bit of PSW Register for Carry
HC	1 Bit of PSW Register for Half Carry
GIE	1 Bit of PSW Register for Global Interrupt Enable
VU	Interrupt Vector Upper Byte
VL	Interrupt Vector Lower Byte

Symbols	
[B]	Memory Indirectly Addressed by B Register
[X]	Memory Indirectly Addressed by X Register
MD	Direct Addressed Memory
Mem	Direct Addressed Memory or [B]
Meml	Direct Addressed Memory or [B] or Immediate Data
Imm	8-Bit Immediate Data
Reg	Register Memory: Addresses F0 to FF (Includes B, X and SP)
Bit	Bit Number (0 to 7)
←	Loaded with
↔	Exchanged with

Instruction Set (Continued)

INSTRUCTION SET

ADD	A,Meml	ADD	A ← A + Meml
ADC	A,Meml	ADD with Carry	A ← A + Meml + C, C ← Carry HC ← Half Carry
SUBC	A,Meml	Subtract with Carry	A ← A − $\overline{\text{Meml}}$ + C, C ← Carry HC ← Half Carry
AND	A,Meml	Logical AND	A ← A and Meml
ANDSZ	A,Imm	Logical AND Immed., Skip if Zero	Skip next if (A and Imm) = 0
OR	A,Meml	Logical OR	A ← A or Meml
XOR	A,Meml	Logical EXclusive OR	A ← A xor Meml
IFEQ	MD,Imm	IF EQual	Compare MD and Imm, Do next if MD = Imm
IFEQ	A,Meml	IF EQual	Compare A and Meml, Do next if A = Meml
IFNE	A,Meml	IF Not Equal	Compare A and Meml, Do next if A ≠ Meml
IFGT	A,Meml	IF Greater Than	Compare A and Meml, Do next if A > Meml
IFBNE	#	If B Not Equal	Do next if lower 4 bits of B ≠ Imm
DRSZ	Reg	Decrement Reg., Skip if Zero	Reg ← Reg − 1, Skip if Reg = 0
SBIT	#,Mem	Set BIT	1 to bit, Mem (bit = 0 to 7 immediate)
RBIT	#,Mem	Reset BIT	0 to bit, Mem
IFBIT	#,Mem	IF BIT	If bit in A or Mem is true do next instruction
RPND		Reset PeNDing Flag	Reset Software Interrupt Pending Flag
X	A,Mem	EXchange A with Memory	A ⟷ Mem
X	A,[X]	EXchange A with Memory [X]	A ⟷ [X]
LD	A,Meml	LoaD A with Memory	A ← Meml
LD	A,[X]	LoaD A with Memory [X]	A ← [X]
LD	B,Imm	LoaD B with Immed.	B ← Imm
LD	Mem,Imm	LoaD Memory Immed	Mem ← Imm
LD	Reg,Imm	LoaD Register Memory Immed.	Reg ← Imm
X	A, [B ±]	EXchange A with Memory [B]	A ⟷ [B], (B ← B ±1)
X	A, [X ±]	EXchange A with Memory [X]	A ⟷ [X], (X ← X ±1)
LD	A, [B±]	LoaD A with Memory [B]	A ← [B], (B ← B ±1)
LD	A, [X±]	LoaD A with Memory [X]	A ← [X], (X ← X±1)
LD	[B±],Imm	LoaD Memory [B] Immed.	[B] ← Imm, (B ← B±1)
CLR	A	CLeaR A	A ← 0
INC	A	INCrement A	A ← A + 1
DEC	A	DECrement A	A ← A − 1
LAID		Load A InDirect from ROM	A ← ROM (PU,A)
DCOR	A	Decimal CORrect A	A ← BCD correction of A (follows ADC, SUBC)
RRC	A	Rotate A Right thru C	C → A7 → . . . → A0 → C
RLC	A	Rotate A Left thru C	C ← A7 ← . . . ← A0 ← C
SWAP	A	SWAP nibbles of A	A7 . . . A4 ⟷ A3 . . . A0
SC		Set C	C ← 1, HC ← 1
RC		Reset C	C ← 0, HC ← 0
IFC		IF C	IF C is true, do next instruction
IFNC		IF Not C	If C is not true, do next instruction
POP	A	POP the stack into A	SP ← SP + 1, A ← [SP]
PUSH	A	PUSH A onto the stack	[SP] ← A, SP ← SP − 1
VIS		Vector to Interrupt Service Routine	PU ← [VU], PL ← [VL]
JMPL	Addr.	Jump absolute Long	PC ← ii (ii = 15 bits, 0 to 32k)
JMP	Addr.	Jump absolute	PC9 . . . 0 ← i (i = 12 bits)
JP	Disp.	Jump relative short	PC ← PC + r (r is −31 to +32, except 1)
JSRL	Addr.	Jump SubRoutine Long	[SP] ← PL, [SP−1] ← PU,SP−2, PC ← ii
JSR	Addr	Jump SubRoutine	[SP] ← PL, [SP−1] ← PU,SP−2, PC9 . . . 0 ← i
JID		Jump InDirect	PL ← ROM (PU,A)
RET		RETurn from subroutine	SP + 2, PL ← [SP], PU ← [SP−1]
RETSK		RETurn and SKip	SP + 2, PL ← [SP],PU ← [SP−1], skip next instruction
RETI		RETurn from Interrupt	SP + 2, PL ← [SP],PU ← [SP−1],GIE ← 1
INTR		Generate an Interrupt	[SP] ← PL, [SP−1] ← PU, SP−2, PC ← 0FF
NOP		No OPeration	PC ← PC + 1

Instruction Execution Time

Most instructions are single byte (with immediate addressing mode instructions taking two bytes).

Most single byte instructions take one cycle time to execute.

Skipped instructions require x number of cycles to be skipped, where x equals the number of bytes in the skipped instruction opcode.

See the BYTES and CYCLES per INSTRUCTION table for details.

Bytes and Cycles per Instruction

The following table shows the number of bytes and cycles for each instruction in the format of byte/cycle.

Arithmetic and Logic Instructions

	[B]	Direct	Immed.
ADD	1/1	3/4	2/2
ADC	1/1	3/4	2/2
SUBC	1/1	3/4	2/2
AND	1/1	3/4	2/2
OR	1/1	3/4	2/2
XOR	1/1	3/4	2/2
IFEQ	1/1	3/4	2/2
IFNE	1/1	3/4	2/2
IFGT	1/1	3/4	2/2
IFBNE	1/1		
DRSZ		1/3	
SBIT	1/1	3/4	
RBIT	1/1	3/4	
IFBIT	1/1	3/4	

RPND	1/1

Instructions Using A & C

CLRA	1/1
INCA	1/1
DECA	1/1
LAID	1/3
DCOR	1/1
RRCA	1/1
RLCA	1/1
SWAPA	1/1
SC	1/1
RC	1/1
IFC	1/1
IFNC	1/1
PUSHA	1/3
POPA	1/3
ANDSZ	2/2

Transfer of Control Instructions

JMPL	3/4
JMP	2/3
JP	1/3
JSRL	3/5
JSR	2/5
JID	1/3
VIS	1/5
RET	1/5
RETSK	1/5
RETI	1/5
INTR	1/7
NOP	1/1

Memory Transfer Instructions

	Register Indirect		Direct	Immed.	Register Indirect Auto Incr. & Decr.		
	[B]	[X]			[B+, B−]	[X+, X−]	
X A,*	1/1	1/3	2/3		1/2	1/3	
LD A,*	1/1	1/3	2/3	2/2	1/2	1/3	
LD B, Imm				1/1			(IF B < 16)
LD B, Imm				2/2			(IF B > 15)
LD Mem, Imm	2/2		3/3		2/2		
LD Reg, Imm			2/3				
IFEQ MD, Imm			3/3				

* = > Memory location addressed by B or X or directly.

Opcode Table

Upper Nibble																Lower Nibble
F	E	D	C	B	A	9	8	7	6	5	4	3	2	1	0	
JP−15	JP−31	LD 0F0, # i	DRSZ 0F0	RRCA	RC	ADC A,#i	ADC A,[B]	IFBIT 0,[B]	ANDSZ A, #i	LD B,#0F	IFBNE 0	JSR x000−x0FF	JMP x000−x0FF	JP+17	INTR	0
JP−14	JP−30	LD 0F1, # i	DRSZ 0F1	*	SC	SUBC A, #i	SUB A,[B]	IFBIT 1,[B]	*	LD B,#0E	IFBNE 1	JSR x100−x1FF	JMP x100−x1FF	JP+18	JP+2	1
JP−13	JP−29	LD 0F2, # i	DRSZ 0F2	X A,[X+]	X A,[B+]	IFEQ A,#i	IFEQ A,[B]	IFBIT 2,[B]	*	LD B,#0D	IFBNE 2	JSR x200−x2FF	JMP x200−x2FF	JP+19	JP+3	2
JP−12	JP−28	LD 0F3, # i	DRSZ 0F3	X A,[X−]	X A,[B−]	IFGT A,#i	IFGT A,[B]	IFBIT 3,[B]	*	LD B,#0C	IFBNE 3	JSR x300−x3FF	JMP x300−x3FF	JP+20	JP+4	3
JP−11	JP−27	LD 0F4, # i	DRSZ 0F4	VIS	LAID	ADD A,#i	ADD A,[B]	IFBIT 4,[B]	CLRA	LD B,#0B	IFBNE 4	JSR x400−x4FF	JMP x400−x4FF	JP+21	JP+5	4
JP−10	JP−26	LD 0F5, # i	DRSZ 0F5	RPND	JID	AND A,#i	AND A,[B]	IFBIT 5,[B]	SWAPA	LD B,#0A	IFBNE 5	JSR x500−x5FF	JMP x500−x5FF	JP+22	JP+6	5
JP−9	JP−25	LD 0F6, # i	DRSZ 0F6	X A,[X]	X A,[B]	XOR A,#i	XOR A,[B]	IFBIT 6,[B]	DCORA	LD B,#09	IFBNE 6	JSR x600−x6FF	JMP x600−x6FF	JP+23	JP+7	6
JP−8	JP−24	LD 0F7, # i	DRSZ 0F7	*	*	OR A,#i	OR A,[B]	IFBIT 7,[B]	PUSHA	LD B,#08	IFBNE 7	JSR x700−x7FF	JMP x700−x7FF	JP+24	JP+8	7
JP−7	JP−23	LD 0F8, # i	DRSZ 0F8	NOP	RLCA	LD A,#i	IFC	SBIT 0,[B]	RBIT 0,[B]	LD B,#07	IFBNE 8	JSR x800−x8FF	JMP x800−x8FF	JP+25	JP+9	8
JP−6	JP−22	LD 0F9, # i	DRSZ 0F9	IFNE A,[B]	IFEQ Md,#i	IFNE A,#i	IFNC	SBIT 1,[B]	RBIT 1,[B]	LD B,#06	IFBNE 9	JSR x900−x9FF	JMP x900−x9FF	JP+26	JP+10	9
JP−5	JP−21	LD 0FA, # i	DRSZ 0FA	LD A,[X+]	LD A,[B+]	LD [B+],#i	INCA	SBIT 2,[B]	RBIT 2,[B]	LD B,#05	IFBNE 0A	JSR xA00−xAFF	JMP xA00−xAFF	JP+27	JP+11	A
JP−4	JP−20	LD 0FB, # i	DRSZ 0FB	LD A,[X−]	LD A,[B−]	LD [B−],#i	DECA	SBIT 3,[B]	RBIT 3,[B]	LD B,#04	IFBNE 0B	JSR xB00−xBFF	JMP xB00−xBFF	JP+28	JP+12	B
JP−3	JP−19	LD 0FC, # i	DRSZ 0FC	LD Md,#i	JMPL	X A,Md	POPA	SBIT 4,[B]	RBIT 4,[B]	LD B,#03	IFBNE 0C	JSR xC00−xCFF	JMP xC00−xCFF	JP+29	JP+13	C
JP−2	JP−18	LD 0FD, # i	DRSZ 0FD	DIR	JSRL	LD A,Md	RETSK	SBIT 5,[B]	RBIT 5,[B]	LD B,#02	IFBNE 0D	JSR xD00−xDFF	JMP xD00−xDFF	JP+30	JP+14	D
JP−1	JP−17	LD 0FE, # i	DRSZ 0FE	LD A,[X]	LD A,[B]	LD [B],#i	RET	SBIT 6,[B]	RBIT 6,[B]	LD B,#01	IFBNE 0E	JSR xE00−xEFF	JMP xE00−xEFF	JP+31	JP+15	E
JP−0	JP−16	LD 0FF, # i	DRSZ 0FF	*	*	LD B,#i	RETI	SBIT 7,[B]	RBIT 7,[B]	LD B,#00	IFBNE 0F	JSR xF00−xFFF	JMP xF00−xFFF	JP+32	JP+16	F

Where,

i is the immediate data

Md is a directly addressed memory location

* is an unused opcode

Note: The opcode 60 Hex is also the opcode for IFBIT #i,A

Development Support

SUMMARY

- iceMASTER™: IM-COP8/400—Full feature in-circuit emulation for all COP8 products. A full set of COP8 Basic and Feature Family device and package specific probes are available.
- COP8 Debug Module: Moderate cost in-circuit emulation and development programming unit.
- COP8 Evaluation and Programming Unit: EPU-COP888GG—low cost in-circuit simulation and development programming unit.
- Assembler: COP8-DEV-IBMA. A DOS installable cross development Assembler, Linker, Librarian and Utility Software Development Tool Kit.
- C Compiler: COP8C. A DOS installable cross development Software Tool Kit.
- OTP/EPROM Programmer Support: Covering needs from engineering prototype, pilot production to full production environments.

iceMASTER (IM) IN-CIRCUIT EMULATION

The iceMASTER IM-COP8/400 is a full feature, PC based, in-circuit emulation tool developed and marketed by MetaLink Corporation to support the whole COP8 family of products. National is a resale vendor for these products.

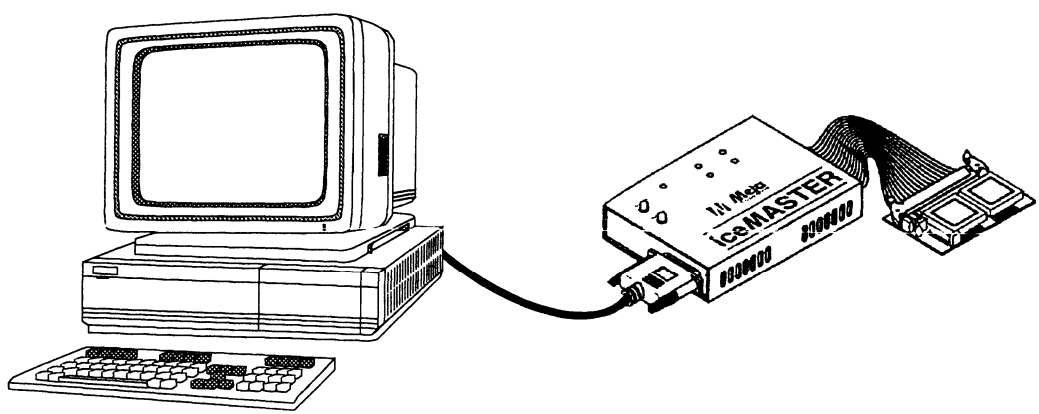
See *Figure 19* for configuration.

The iceMASTER IM-COP8/400 with its device specific COP8 Probe provides a rich feature set for developing, testing and maintaining product:

- Real-time in-circuit emulation; full 2.4V–5.5V operation range, full DC-10 MHz clock. Chip options are programmable or jumper selectable.
- Direct connection to application board by package compatible socket or surface mount assembly.
- Full 32 kbytes of loadable programming space that overlays (replaces) the on-chip ROM or EPROM. On-chip RAM and I/O blocks are used directly or recreated on the probe as necessary.
- Full 4k frame synchronous trace memory. Address, instruction, and 8 unspecified, circuit connectable trace lines. Display can be HLL source (e.g., C source), assembly or mixed.
- A full 64k hardware configurable break, trace on, trace off control, and pass count increment events.
- Tool set integrated interactive symbolic debugger—supports both assembler (COFF) and C Compiler (.COD) linked object formats.
- Real time performance profiling analysis; selectable bucket definition.
- Watch windows, content updated automatically at each execution break.
- Instruction by instruction memory/register changes displayed on source window when in single step operation.
- Single base unit and debugger software reconfigurable to support the entire COP8 family; only the probe personality needs to change. Debugger software is processor customized, and reconfigured from a master model file.
- Processor specific symbolic display of registers and bit level assignments, configured from master model file.
- Halt/Idle mode notification.
- On-line HELP customized to specific processor using master model file.
- Includes a copy of COP8-DEV-IBMA assembler and linker SDK.

IM Order Information

Base Unit	
IM-COP8/400-1	iceMASTER base unit, 110V power supply
IM-COP8/400-2	iceMASTER base unit, 220V power supply
iceMASTER Probe	
MHW-888GG40DWPC	40 DIP
MHW-888GG44PWPC	44 PLCC



TL/DD/12532–24

FIGURE 19. COP8 iceMASTER Environment

Development Support (Continued)

iceMASTER DEBUG MODULE (DM)

The iceMASTER Debug Module is a PC based, combination in-circuit emulation tool and COP8 based OTP/EPROM programming tool developed and marketed by MetaLink Corporation to support the whole COP8 family of products. National is a resale vendor for these products.

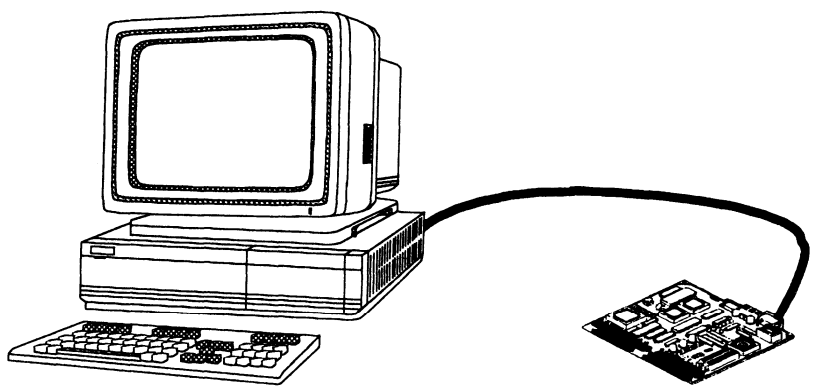
See *Figure 20* for configuration.

The iceMASTER Debug Module is a moderate cost development tool. It has the capability of in-circuit emulation for a specific COP8 microcontroller and in addition serves as a programming tool for COP8 OTP and EPROM product families. Summary of features is as follows:

- Real-time in-circuit emulation; full operating voltage range operation, full DC-10 MHz clock.
- All processor I/O pins can be cabled to an application development board with package compatible cable to socket and surface mount assembly.
- Full 32 kbytes of loadable programming space that overlays (replaces) the on-chip ROM or EPROM. On-chip RAM and I/O blocks are used directly or recreated as necessary.
- 100 frames of synchronous trace memory. The display can be HLL source (C source), assembly or mixed. The most recent history prior to a break is available in the trace memory.
- Configured break points; uses INTR instruction which is modestly intrusive.
- Software—only supported features are selectable.
- Tool set integrated interactive symbolic debugger—supports both assembler (COFF) and C Compiler (.COD) SDK linked object formats.
- Instruction by instruction memory/register changes displayed when in single step operation.
- Debugger software is processor customized, and reconfigured from a master model file.
- Processor specific symbolic display of registers and bit level assignments, configured from master model file.
- Halt/Idle mode notification.
- Programming menu supports full product line of programmable OTP and EPROM COP8 products. Program data is taken directly from the overlay RAM.
- Programming of 44 PLCC and 68 PLCC parts requires external programming adapters.
- Includes wallmount power supply.
- On-board V_{PP} generator from 5V input or connection to external supply supported. Requires V_{PP} level adjustment per the family programming specification (correct level is provided on an on-screen pop-down display).
- On-line HELP customized to specific processor using master model file.
- Includes a copy of COP8-DEV-IBMA assembler and linker SDK.

DM Order Information

Debug Model Unit	
COP8-DM/888GG	
Cable Adapters	
DM-COP8/40D	40 DIP
DM-COP8/44P	44 PLCC



TL/DD/12532–25

FIGURE 20. COP8-DM Environment

iceMASTER EVALUATION PROGRAMMING UNIT (EPU)

The iceMASTER EPU-COP888GG is a PC based, in-circuit simulation tool to support the feature family COP8 products.

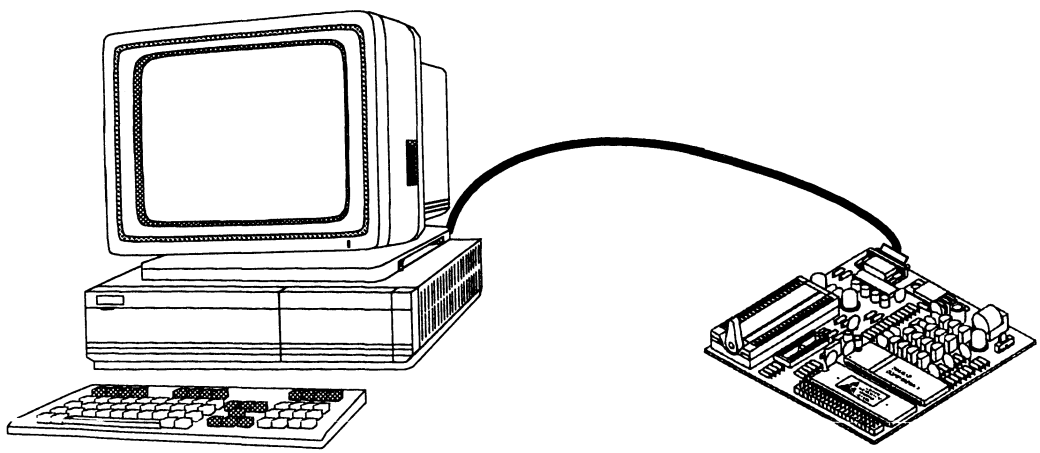
See *Figure 21* for configuration.

The simulation capability is a very low cost means of evaluating the general COP8 architecture. In additions the EPU has programming capability, with added adapters, for programming the whole COP8 product family of OTP and EPROM products. The product includes the following features:

- Non real-time in-circuit simulation. Program overlay memory is PC resident; instructions are downloaded over RS-232 as executed. Approximate performance is 20 kHz.
- Includes a 40-pin DIP cable adapter. Other target packages are not supported. All processor I/O pins are cabled to an application development environment.
- Full 32 kbytes of loadable programming space that overlays (replaces) the on-chip ROM or EPROM. On-chip RAM and I/O blocks are used directly or recreated as necessary.
- On-chip timer and WATCHDOG execution are not well synchronized to the instruction simulation.
- 100 frames of synchronous trace memory. The display can be HLL source (e.g., C source), assembly or mixed. The most recent history prior to a break is available in the trace memory.
- Up to eight software configured break points; uses INTR instruction which is modestly instrusive.
- Common look-feel debugger software across all MetaLink products—only supported features are selectable.
- Tool set integrated interactive symbolic debugger—supports both assembler (.OFF) and C Compiler (.COD) SDK linked object formats.
- Instruction by instruction memory/register changes displayed when in single step operation.
- Processor specific symbolic display of registers and bit level assignments, configured from master model file.
- Halt/Idle mode notification. Restart requires special handling.
- Programming menu supports full product line of programmable OTP and EPROM COP8 products. Only a 40 ZIF socket is available on the EPU unit. Adapters are available for other part package configurations.
- Integral wall mount power supply provides 5V and develops the required V_{PP} to program parts.
- Includes a copy of COP8-DEV-IBMA assembler, linker SDK.

EPU Order Information

Evaluation Programming Unit	
EPU-COP888GG	Evaluation Programming Unit with debugger and programmer control software and 40 ZIF programming socket.
General Programming Adapters	
COP8-PGMA-DS44P	28 and 20 DIP and SOIC plus 44 PLCC adapter.



TL/DD/12532-26

FIGURE 21. EPU-COP8 Tool Environment

Development Support (Continued)

COP8 ASSEMBLER/LINKER SOFTWARE DEVELOPMENT TOOL KIT

National Semiconductor offers a relocateable COP8 macro cross assembler, linker, librarian and utility software development tool kit. Features are summarized as follows:

- Basic and Feature Family instruction set by "device" type.
- Nested macro capability.
- Extensive set of assembler directives.
- Supported on PC/DOS platform.
- Generates National standard COFF output files.
- Integrated Linker and Librarian.
- Integrated utilities to generate ROM code file outputs.
- DUMPCOFF utility.

This product is integrated as a part of MetaLink tools as a development kit, fully supported by the MetaLink debugger. It may be ordered separately or it is bundled with the MetaLink products at no additional cost.

Order Information

Assembler SDK	
COP8-DEV-IBMA	Assembler SDK on installable 3.5" PC®/DOS Floppy Disk Drive format. Periodic upgrades and most recent version is available on National's BBS and Internet.

COP8 C COMPILER

A C Compiler is developed and marketed by Byte Craft Limited. The COP8C compiler is a fully integrated development tool specifically designed to support the compact embedded configuration of the COP8 family of products.

Features are summarized as follows:

- ANSI C with some restrictions and extensions that optimize development for the COP8 embedded application.
- BITS data type extension. Register declaration #pragma with direct bit level definitions.
- C language support for interrupt routines.
- Expert system, rule based code geration and optimization.
- Performs consistency checks against the architectural definitions of the target COP8 device.
- Generates program memory code.
- Supports linking of compiled object or COP8 assembled object formats.
- Global optimization of linked code.
- Symbolic debug load format fully sourced level supported by the MetaLink debugger.

OTP/EMULATOR SUPPORT

The COP87L88GG provides emulation and OTP support for the COP888GG/COP888HG mask programmable devices.

Approved List

Manufacturer	North America	Europe	Asia
BP Microsystems	(800) 225-2102 (713) 688-4600 Fax: (713) 688-0920	+49-8152-4183 +49-8856-932616	+852-234-16611 +852-2710-8121
Data I/O	(800) 426-1045 (206) 881-6444 Fax: (206) 882-1043	+44-0734-440011	Call North America
HI-LO	(510) 623-8860	Call Asia	+886-2-764-0215 Fax: +886-2-756-6403
ICE Technology	(800) 624-8949 (919) 430-7915	+44-1226-767404 Fax: 0-1226-370-434	
MetaLink	(800) 638-2423 (602) 926-0797 Fax: (602) 693-0681	+49-80 9156 96-0 Fax: +49-80 9123 86	+852-737-1800
Systems General	(408) 263-6667	+41-1-9450300	+886-2-917-3005 Fax: +886-2-911-1283
Needhams	(916) 924-8037 Fax: (916) 924-8065		

Development Support (Continued)

OTP Emulator Ordering Information

Device Number	Clock Option	Package	Emulates
COP87L88GGV-XE	Crystal/HALT En	44 PLCC	COP888GG/ COP888HG
COP87L88GGN-XE	Crystal/HALT En	40 DIP	COP888GG/ COP888HG

INDUSTRY WIDE OTP/EPROM PROGRAMMING SUPPORT

Programming support, in addition to the MetaLink development tools, is provided by a full range of independent approved vendors to meet the needs from the engineering laboratory to full production.

AVAILABLE LITERATURE

For more information, please see the COP8 Basic Family User's Manual, Literature Number 620895, COP8 Feature Family User's Manual, Literature Number 620897 and National's Family of 8-bit Microcontrollers COP8 Selection Guide, Literature Number 630009.

DIAL-A-HELPER

Dial-A-Helper is a service provided by the Microcontroller Applications group. The Dial-A-Helper is an Electronic Information System that may be accessed as a Bulletin Board System (BBS) via data modem, as an FTP site on the Internet via standard FTP client application or as an FTP site on the Internet using a standard Internet browser such as Netscape or Mosaic.

The Dial-A-Helper system provides access to an automated information storage and retrieval system. The system capabilities include a MESSAGE SECTION (electronic mail, when accessed as a BBS) for communication to and from the Microcontroller Applications Group and a FILE SECTION which consists of several file areas where valuable application software and utilities could be found.

DIAL-A-HELPER BBS via a Standard Modem

Modem: CANADA/U.S.: (800) NSC-MICRO
(800) 672-6427
EUROPE: (+49) 0-8141-351332
Baud: 14.4k
Set-up: Length: 8-Bit
Parity: None
Stop Bit: 1
Operation: 24 Hrs., 7 Days

DIAL-A-HELPER via FTP

ftp nscmicro.nsc.com
user: anonymous
password: username@yourhost.site.domain

DIAL-A-HELPER via a WorldWide Web Browser

ftp://nscmicro.nsc.com

National Semiconductor on the WorldWide Web

See us on the WorldWide Web at: http://www.national.com

CUSTOMER RESPONSE CENTER

Complete product information and technical support is available from National's customer response centers.

CANADA/U.S.:	Tel:	(800) 272-9959
	email:	support @tevm2.nsc.com
EUROPE:	email:	europe.support@nsc.com
	Deutsch Tel:	+49 (0) 180-530 85 85
	English Tel:	+49 (0) 180-532 78 32
	Français Tel:	+49 (0) 180-532 93 58
	Italiano Tel:	+49 (0) 180-534 16 80
JAPAN:	Tel:	+81-043-299-2309
S.E. ASIA:	Beijing Tel:	(+86) 10-6856-8601
	Shanghai Tel:	(+86) 21-6415-4092
	Hong Kong Tel:	(+852) 2737-1600
	Korea Tel:	(+82) 2-3771-6909
	Malaysia Tel:	(+60-4) 644-9061
	Singapore Tel:	(+65) 255-2226
	Taiwan Tel:	+886-2-521-3288
AUSTRALIA:	Tel:	(+61) 3-9558-9999
INDIA:	Tel:	(+91) 80-559-9467

National Semiconductor

ADVANCE INFORMATION

COP87L88GD
8-Bit One-Time Programmable (OTP) Microcontroller with A/D Converter

General Description

The COP87L88GD is a member of the COP8™ 8-bit OTP microcontroller family. It is pin and software compatible to the mask ROM COP888GD product family. (Continued)

Key Features

- 8-channel A/D converter with prescaler and both differential and single ended modes
- Idle Timer with 5 selectable Wake-Up periods
- Three 16-bit timers, each with two 16-bit registers supporting:
  - — Processor Independent PWM mode
  - — External Event counter mode
  - — Input Capture mode
- 16 kbytes on-board OTP EPROM with security feature
- 256 bytes on-board RAM

Additional Peripheral Features

- Multi-Input Wakeup (MIWU) with optional interrupts (8)
- WATCHDOG™ and clock monitor logic
- MICROWIRE/PLUS™ serial I/O

I/O Features

- Memory mapped I/O
- Software selectable I/O options (TRI-STATE® Output, Push-Pull Output, Weak Pull Up Input, High Impedance Input)
- Schmitt trigger inputs on ports G and L
- Package:
  - — 44 PLCC with 40 I/O pins
  - — 40 DIP with 36 I/O pins

CPU/Instruction Set Features

- 1 μs instruction cycle time
- Twelve multi-source vectored interrupts servicing
  - — External Interrupt
  - — Idle Timer T0
  - — Three Timers (each with 2 Interrupts)
  - — MICROWIRE/PLUS
  - — Multi-Input Wake Up
  - — Software Trap
  - — Default VIS (default interrupt)
- Versatile and easy to use instruction set
- 8-bit Stack Pointer (SP) – stack in RAM
- Two 8-bit Register Indirect Data Memory Pointers (B and X)

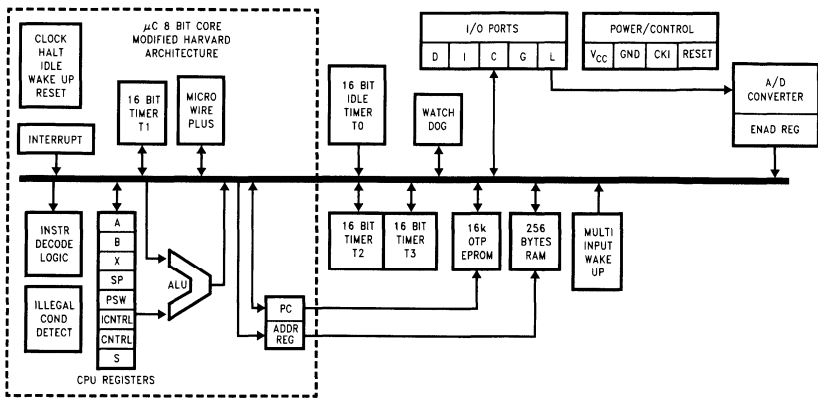
Fully Static CMOS

- Two power saving modes: HALT and IDLE
- Single supply operation: 2.7V to 5.5V
- Temperature range: −40°C to +85°C

Development Support

- Emulation device for COP888GD
- Real time emulation and full program debug offered by MetaLink Development System

Block Diagram



TL/DD/12526–1

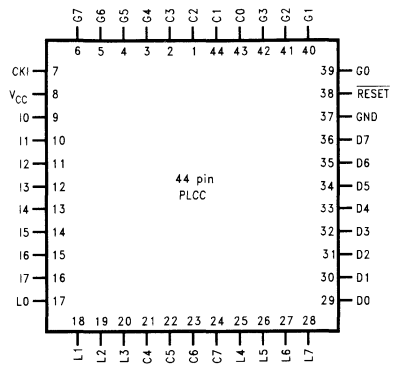
FIGURE 1. Block Diagram

General Description (Continued)

It is a fully static part, fabricated using double-metal silicon gate microCMOS technology. Features include an 8-bit memory mapped architecture, MICROWIRE/PLUS serial I/O, three 16-bit timer/counters supporting three modes (Processor Independent PWM generation, External Event counter, and Input Capture mode capabilities), 8 channel analog to digital converter, and two power saving modes (HALT and IDLE), both with a multi-sourced wakeup/interrupt capability. This multi-sourced interrupt capability may also be used independent of the HALT or IDLE modes. The device includes an IDLE Timer with 5 selectable interrupt periods which can be used to wake the device from IDLE mode. Each I/O pin has software selectable configurations. The devices operate over a voltage range of 2.7V to 5.5V. High throughput is achieved with an efficient, regular instruction set operating at a maximum rate of 1 μs per instruction.

Connection Diagrams

Plastic Chip Carrier



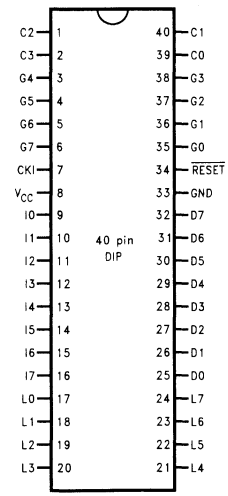
TL/DD/12526–2

Top View

Order Number COP87L88GDV-XE
See NS Plastic Chip Package Number V44A

Note: -X Crystal Oscillator
-E Halt Enable

Dual-In-Line Package



TL/DD/12526–3

Top View

Order Number COP87L88GDN-XE
See NS Molded Package Number N40A

FIGURE 2. Connection Diagrams

National Semiconductor

PRELIMINARY

COP87L88KG
8-Bit One-Time Programmable (OTP) Microcontroller with UART and Three Multi-Function Timers

General Description

The COP87L88KG OTP microcontroller is a member of the COP8™ feature family using an 8-bit core architecture. It is pin and software compatible to the mask ROM COP888KG product family. (Continued)

Key Features

- Full duplex UART
- Three 16-bit timers, each with two 16-bit registers supporting:
  - — Processor Independent PWM mode
  - — External Event counter mode
  - — Input Capture mode
- 24 kbytes on-board OTP EPROM with security feature
- 1088 bytes on-board RAM

Additional Peripheral Features

- Idle Timer
- Multi-Input Wake-Up (MIWU) with optional interrupts (8)
- Two analog comparators
- WATCHDOG™ and Clock Monitor logic
- MICROWIRE/PLUS™ serial I/O

I/O Features

- Memory mapped I/O
- Software selectable I/O options (TRI-STATE® Output, Push-Pull Output, Weak Pull-Up Input, High Impedance Input)
- Schmitt trigger inputs on ports G and L
- Packages:
  - — 40 DIP with 35 I/O pins
  - — 44 PLCC with 39 I/O pins

CPU/Instruction Set Features

- 1 μs instruction cycle time
- Fourteen multi-source vectored interrupts servicing
  - — External Interrupt with selectable edge
  - — Idle Timer T0
  - — Three Timers (each with 2 interrupts)
  - — MICROWIRE/PLUS
  - — Multi-Input Wake-Up
  - — Software Trap
  - — UART (2)
  - — Default VIS (default interrupt)
- Versatile and easy to use instruction set
- 8-bit Stack Pointer SP—(stack in RAM)
- Two 8-bit Register Indirect Data Memory Pointers (B and X)

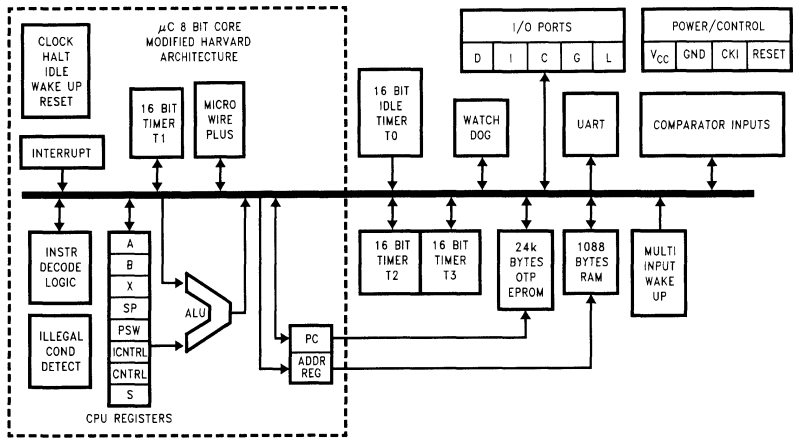
Fully Static CMOS

- Two power saving modes: HALT and IDLE
- Low current drain (typically <1 μA)
- Single supply operation: 2.7V–5.5V
- Temperature ranges:
  $-40°C$ to $+85°C$

Development Support

- Emulation device for COP888KG
- Real time emulation and full program debug offered by MetaLink's Development System

Block Diagram



TL/DD12864-1

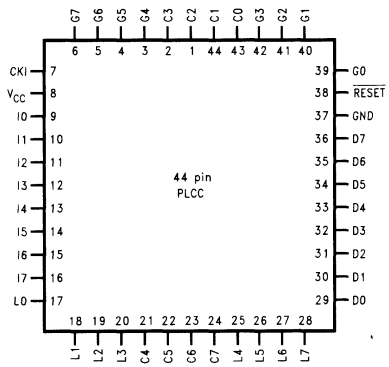
FIGURE 1. COP888KG Block Diagram

General Description (Continued)

It is a fully static part, fabricated using double-metal silicon gate microCMOS technology. Features include an 8-bit memory mapped architecture, MICROWIRE/PLUS serial I/O, three 16-bit timer/counters supporting three modes (Processor Independent PWM generation, External Event counter, and Input Capture mode capabilities), full duplex UART, two comparators, and two power saving modes (HALT and IDLE), both with a multi-sourced wakeup/interrupt capability. This multi-sourced interrupt capability may also be used independent of the HALT or IDLE modes. Each I/O pin has software selectable configurations. The devices operate over a voltage range of 2.7V–5.5V. High throughput is achieved with an efficient, regular instruction set operating at a maximum rate of 1 μs per instruction.

Connection Diagrams

Plastic Chip Carrier



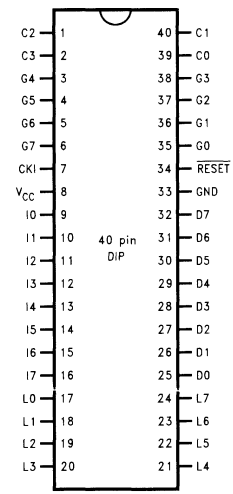
TL/DD12864-2

Top View

Order Number COP87L88KGV-XE
See NS Package Number V44A

Note: -X Crystal Oscillator
-E Halt Enable

Dual-In-Line Package



TL/DD12864-3

Top View

Order Number COP87L88KGN-XE
See NS Package Number N40A

FIGURE 2. Connection Diagrams

Connection Diagrams (Continued)

Pinouts for 40- and 44-Pin Packages

Port	Type	Alt. Fun	Alt. Fun	40-Pin DIP	44-Pin PLCC
L0	I/O	MIWU		17	17
L1	I/O	MIWU	CKX	18	18
L2	I/O	MIWU	TDX	19	19
L3	I/O	MIWU	RDX	20	20
L4	I/O	MIWU	T2A	21	25
L5	I/O	MIWU	T2B	22	26
L6	I/O	MIWU	T3A	23	27
L7	I/O	MIWU	T3B	24	28
G0	I/O	INT		35	39
G1	WDOUT			36	40
G2	I/O	T1B		37	41
G3	I/O	T1A		38	42
G4	I/O	SO		3	3
G5	I/O	SK		4	4
G6	I	SI		5	5
G7	I/CKO	HALT Restart		6	6
D0	O			25	29
D1	O			26	30
D2	O			27	31
D3	O			28	32
D4	O			29	33
D5	O			30	34
D6	O			31	35
D7	O			32	36
I0	I			9	9
I1	I	COMP1IN−		10	10
I2	I	COMP1IN+		11	11
I3	I	COMP1OUT		12	12
I4	I	COMP2IN−		13	13
I5	I	COMP2IN+		14	14
I6	I	COMP2OUT		15	15
I7	I			16	16
C0	I/O			39	43
C1	I/O			40	44
C2	I/O			1	1
C3	I/O			2	2
C4	I/O				21
C5	I/O				22
C6	I/O				23
C7	I/O				24
V_{CC}				8	8
GND				33	37
CKI				7	7
$\overline{\text{RESET}}$				34	38

Absolute Maximum Ratings

If Military/Aerospace specified devices are required, please contact the National Semiconductor Sales Office/Distributors for availability and specifications.

Supply Voltage (V_{CC})	7V
Voltage at Any Pin	−0.3V to V_{CC} + 0.3V
Total Current into V_{CC} Pin (Source)	100 mA
Total Current out of GND Pin (Sink)	110 mA
Storage Temperature Range	−65°C to +140°C

Note: *Absolute maximum ratings indicate limits beyond which damage to the device may occur. DC and AC electrical specifications are not ensured when operating the device at absolute maximum ratings.*

DC Electrical Characteristics

−40°C ≤ T_A ≤ +85°C unless otherwise specified

Parameter	Conditions	Min	Typ	Max	Units
Operating Voltage		2.7		5.5	V
Power Supply Ripple (Note 1)	Peak-to-Peak			0.1 V_{CC}	V
Supply Current (Note 2)					
CKI = 10 MHz	V_{CC} = 5.5V, t_c = 1 μs			14	mA
CKI = 4 MHz	V_{CC} = 5.5V, t_c = 2.5 μs			5.5	mA
HALT Current (Note 3)	V_{CC} = 5.5V, CKI = 0 MHz		<1	12	μA
	V_{CC} = 4.0V, CKI = 0 MHz		<0.5	8	μA
IDLE Current					
CKI = 10 MHz	V_{CC} = 5.5V, t_c = 1 μs			3.5	mA
CKI = 1 MHz	V_{CC} = 4.0V, t_c = 10 μs			0.7	mA
Input Levels (V_{IH}, V_{IL})					
$\overline{\text{RESET}}$					
Logic High		0.8 V_{CC}			V
Logic Low				0.2 V_{CC}	V
CKI, All Other Inputs					
Logic High		0.7 V_{CC}			V
Logic Low				0.2 V_{CC}	V
Hi-Z Input Leakage	V_{CC} = 5.5V, V_{IN} = 0V	−2		+2	μA
Input Pullup Current	V_{CC} = 5.5V, V_{IN} = 0V	−40		−250	μA
G and L Port Input Hysteresis	(Note 5)			0.35 V_{CC}	V
Output Current Levels					
D Outputs					
Source	V_{CC} = 4.5V, V_{OH} = 3.3V	−0.4			mA
	V_{CC} = 2.7V, V_{OH} = 1.8V	−0.2			mA
Sink	V_{CC} = 4.5V, V_{OL} = 1V	10			mA
	V_{CC} = 2.7V, V_{OL} = 0.4V	2.0			mA
All Others					
Source (Weak Pull-Up Mode)	V_{CC} = 4.5V, V_{OH} = 2.7V	−10		−110	μA
	V_{CC} = 2.7V, V_{OH} = 1.8V	−2.5		−33	μA
Source (Push-Pull Mode)	V_{CC} = 4.5V, V_{OH} = 3.3V	−0.4			mA
	V_{CC} = 2.7V, V_{OH} = 1.8V	−0.2			mA
Sink (Push-Pull Mode)	V_{CC} = 4.5V, V_{OL} = 0.4V	1.6			mA
	V_{CC} = 2.7V, V_{OL} = 0.4V	0.7			mA
TRI-STATE Leakage	V_{CC} = 5.5V	−2		+2	μA
Allowable Sink/Source Current per Pin					
D Outputs (Sink)				15	mA
All others				3	mA
Maximum Input Current without Latchup (Notes 4, 5)	Room Temperature			±200	mA
RAM Retention Voltage, V_r	500 ns Rise and Fall Time (min)	2			V
Input Capacitance	(Note 5)			7	pF
Load Capacitance on D2	(Note 5)			1000	pF

AC Electrical Characteristics COP888KG: $-40°C \le T_A \le +85°C$ unless otherwise specified

Parameter	Conditions	Min	Typ	Max	Units
Instruction Cycle Time (t_c)					
Crystal, Resonator	$4.5V \le V_{CC} \le 5.5V$	2.5		DC	μs
	$2.7V \le V_{CC} < 4.5V$	1.0		DC	μs
R/C Oscillator	$4.5V \le V_{CC} \le 5.5V$	7.5		DC	μs
	$2.7V \le V_{CC} < 4.5V$	3.0		DC	μs
Inputs					
t_{SETUP}	$4.5V \le V_{CC} \le 5.5V$	200			ns
	$2.7V \le V_{CC} < 4.5V$	500			ns
t_{HOLD}	$4.5V \le V_{CC} \le 5.5V$	60			ns
	$2.7V \le V_{CC} < 4.5V$	150			ns
Output Propagation Delay	$R_L = 2.2k$, $C_L = 100$ pF				
t_{PD1}, t_{PD0}					
SO, SK	$4.5V \le V_{CC} \le 5.5V$			0.7	μs
	$2.7V \le V_{CC} < 4.5V$			1.75	μs
All Others	$4.5V \le V_{CC} \le 5.5V$			1	μs
	$2.7V \le V_{CC} < 4.5V$			2.5	μs
MICROWIRE Setup Time (t_{UWS}) (Note 5)	$V_{CC} \ge 4.5V$	20			ns
MICROWIRE Hold Time (t_{UWH}) (Note 5)	$V_{CC} \ge 4.5V$	56			ns
MICROWIRE Output Propagation Delay (t_{UPD})	$V_{CC} \ge 4.5V$			220	ns
Input Pulse Width (Note 6)					
Interrupt Input High Time		1			t_c
Interrupt Input Low Time		1			t_c
Timer 1, 2, 3 Input High Time		1			t_c
Timer 1, 2, 3 Input Low Time		1			t_c
Reset Pulse Width		1			μs

Note 1: Maximum rate of voltage change must be less than 0.5 V/ms.

Note 2: Supply current is measured after running 2000 cycles with a square wave CKI input, CKO open, inputs at rails and outputs open.

Note 3: The HALT mode will stop CKI from oscillating in the RC and the Crystal configurations. Measurement of IDD HALT is done with device neither sourcing or sinking current; with L, C, and G0–G5 programmed as low outputs and not driving a load; all outputs programmed low and not driving a load; all inputs tied to V_{CC}; clock monitor and comparators disabled. Parameter refers to HALT mode entered via setting bit 7 of the G Port data register. Part will pull up CKI during HALT in crystal clock mode.

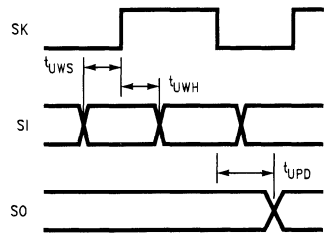
Note 4: Pins G6 and $\overline{\text{RESET}}$ are designed with a high voltage input network. These pins allow input voltages greater than V_{CC} and the pins will have sink current to V_{CC} when biased at voltages greater than V_{CC} (the pins do not have source current when biased at a voltage below V_{CC}). The effective resistance to V_{CC} is 750Ω (typical). These two pins will not latch up. The voltage at the pins must be limited to less than 14V. **WARNING: Voltages in excess of 14V will cause damage to the pins. This warning excludes ESD transients.**

Note 5: Parameter characterized but not tested.

Note 6: t_c Instruction Cycle Time

Comparators AC and DC Characteristics V_{CC} = 5V, −40°C ≤ T_A ≤ +85°C.

Parameter	Conditions	Min	Typ	Max	Units
Input Offset Voltage	0.4V ≤ V_{IN} ≤ V_{CC} − 1.5V		±10	±25	mV
Input Common Mode Voltage Range		0.4		V_{CC} − 1.5	V
Voltage Gain			300k		V/V
Low Level Output Current	V_{OL} = 0.4V	1.6			mA
High Level Output Current	V_{OH} = 4.6V	1.6			mA
DC Supply Current per Comparator (When Enabled)				250	μA
Response Time	100 mV Overdrive, 100 pF Load			1	μs



TL/DD12864–4

FIGURE 3. MICROWIRE/PLUS Timing

Pin Descriptions

V_{CC} and GND are the power supply pins. All V_{CC} and GND pins must be connected.

CKI is the clock input. This can come from an R/C generated oscillator, or a crystal oscillator (in conjunction with CKO). See Oscillator Description section.

$\overline{\text{RESET}}$ is the master reset input. See Reset Description section.

The device contains three bidirectional 8-bit I/O ports (C, G and L), where each individual bit may be independently configured as an input (Schmitt Trigger inputs on ports L and G), output or TRI-STATE under program control. Three data memory address locations are allocated for each of these I/O ports. Each I/O port has two associated 8-bit memory mapped registers, the CONFIGURATION register and the output DATA register. A memory mapped address is also reserved for the input pins of each I/O port. (See the memory map for the various addresses associated with the I/O ports.) *Figure 4* shows the I/O port configurations. The DATA and CONFIGURATION registers allow for each port bit to be individually configured under software control as shown below:

CONFIGURATION Register	DATA Register	Port Set-Up
0	0	Hi-Z Input (TRI-STATE Output)
0	1	Input with Weak Pull-Up
1	0	Push-Pull Zero Output
1	1	Push-Pull One Output

PORT L is an 8-bit I/O port. All L-pins have Schmitt triggers on the inputs.

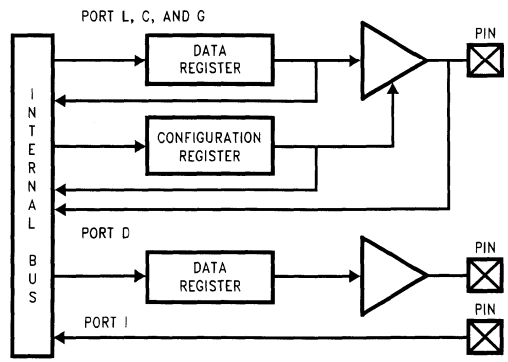
The Port L supports Multi-Input Wake Up on all eight pins. L1 is used for the UART external clock. L2 and L3 are used for the UART transmit and receive. L4 and L5 are used for the timer input functions T2A and T2B. L6 and L7 are used for the timer input functions T3A and T3B.

The Port L has the following alternate features:

- L0 MIWU
- L1 MIWU or CKX
- L2 MIWU or TDX
- L3 MIWU or RDX
- L4 MIWU or T2A
- L5 MIWU or T2B
- L6 MIWU or T3A
- L7 MIWU or T3B

Port G is an 8-bit port with 5 I/O pins (G0, G2–G5), an input pin (G6), and a dedicated output pin (G7). Pins G0 and G2–G6 all have Schmitt Triggers on their inputs. Pin G1 serves as the dedicated WDOUT WATCHDOG output, while pin G7 is either input or output depending on the oscillator mask option selected. With the crystal oscillator option selected, G7 serves as the dedicated output pin for the CKO clock output. With the single-pin R/C oscillator mask option selected, G7 serves as a general purpose input pin but is also used to bring the device out of HALT mode with a low to high transition on G7. There are two registers associated with the G Port, a data register and a configuration register. Therefore, each of the 5 I/O bits (G0, G2–G5) can be individually configured under software control.

Since G6 is an input only pin and G7 is the dedicated CKO clock output pin (crystal clock option) or general purpose input (R/C clock option), the associated bits in the data and configuration registers for G6 and G7 are used for special purpose functions as outlined on the next page. Reading the G6 and G7 data bits will return zeros.



TL/DD12864–5

FIGURE 4. I/O Port Configurations

Pin Descriptions (Continued)

Note that the chip will be placed in the HALT mode by writing a "1" to bit 7 of the Port G Data Register. Similarly the chip will be placed in the IDLE mode by writing a "1" to bit 6 of the Port G Data Register.

Writing a "1" to bit 6 of the Port G Configuration Register enables the MICROWIRE/PLUS to operate with the alternate phase of the SK clock. The G7 configuration bit, if set high, enables the clock start up delay after HALT when the R/C clock configuration is used.

	Config Reg.	Data Reg.
G7	CLKDLY	HALT
G6	Alternate SK	IDLE

Port G has the following alternate features:

- G0 INTR (External Interrupt Input)
- G2 T1B (Timer T1 Capture Input)
- G3 T1A (Timer T1 I/O)
- G4 SO (MICROWIRE Serial Data Output)
- G5 SK (MICROWIRE Serial Clock)
- G6 SI (MICROWIRE Serial Data Input)

Port G has the following dedicated functions:

- G1 WDOUT WATCHDOG and/or Clock Monitor dedicated output
- G7 CKO Oscillator dedicated output or general purpose input

Port C is an 8-bit I/O port. The 40-pin device does not have a full complement of Port C pins. The unavailable pins are not terminated. A read operation for these unterminated pins will return unpredicatable values.

Port I is an eight-bit Hi-Z input port.

Port I1–I3 are used for Comparator 1. Port I4–I6 are used for Comparator 2.

The Port I has the following alternate features:

- I1 COMP1 − IN (Comparator 1 Negative Input)
- I2 COMP1 + IN (Comparator 1 Positive Input)
- I3 COMP1OUT (Comparator 1 Output)
- I4 COMP2 − IN (Comparator 2 Negative Input)
- I5 COMP2 + IN (Comparator 2 Positive Input)
- I6 COMP2OUT (Comparator 2 Output)

Port D is an 8-bit output port that is preset high when $\overline{\text{RESET}}$ goes low. The user can tie two or more D port outputs (except D2) together in order to get a higher drive.

Functional Description

The architecture of the device is modified Harvard architecture. With the Harvard architecture, the control store program memory (ROM) is separated from the data store memory (RAM). Both ROM and RAM have their own separate addressing space with separate address buses. The architecture, though based on Harvard architecture, permits transfer of data from ROM to RAM.

CPU REGISTERS

The CPU can do an 8-bit addition, subtraction, logical or shift operation in one instruction (t_C) cycle time.

There are six CPU registers:

A is the 8-bit Accumulator Register

PC is the 15-bit Program Counter Register

- PU is the upper 7 bits of the program counter (PC)
- PL is the lower 8 bits of the program counter (PC)

B is an 8-bit RAM address pointer, which can be optionally post auto incremented or decremented.

X is an 8-bit alternate RAM address pointer, which can be optionally post auto incremented or decremented.

SP is the 8-bit stack pointer, which points to the subroutine/interrupt stack (in RAM). The SP is initialized to RAM address 06F with reset.

S is the 8-bit Data Segment Address Register used to extend the lower half of the address range (00 to 7F) into 256 data segments of 128 bytes each.

All the CPU registers are memory mapped with the exception of the Accumulator (A) and the Program Counter (PC).

PROGRAM MEMORY

The program memory consists of 24 kbytes of OTP EPROM. These bytes may hold program instructions or constant data (data tables for the LAID instruction, jump vectors for the JID instruction, and interrupt vectors for the VIS instruction). The program memory is addressed by the 15-bit program counter (PC). All interrupts in the devices vector to program memory location 0FF Hex.

The device can be configured to inhibit external reads of the program memory. This is done by programming the Security Byte.

SECURITY FEATURE

The program memory array has an associate Security Byte that is located outside of the program address range. This byte can be addressed only from programming mode by a programmer tool.

Security is an optional feature and can only be asserted after the memory array has been programmed and verified. A secured part will read all 00(hex) by a programmer. The part will fail Blank Check an will fail Verify operations. A Read operation will fill the programmer's memory with 00(hex). The Security Byte itself is always readable with value of 00(hex) if unsecure and FF(hex) if secure.

DATA MEMORY

The data memory address space includes the on-chip RAM and data registers, the I/O registers (Configuration, Data and Pin), the control registers, the MICROWIRE/PLUS SIO shift register, and the various registers, and counters associated with the timers (with the exception of the IDLE timer). Data memory is addressed directly by the instruction or indirectly by the B, X, SP pointers and S register.

The data memory consists of 1088 bytes of RAM. Sixteen bytes of RAM are mapped as "registers" at addresses 0F0 to 0FF Hex. These registers can be loaded immediately, and also decremented and tested with the DRSZ (decrement register and skip if zero) instruction. The memory pointer registers X, SP, B and S are memory mapped into this space at address locations 0FC to 0FF Hex respectively, with the other registers being available for general usage.

The instruction set permits any bit in memory to be set, reset or tested. All I/O and registers (except A and PC) are memory mapped; therefore, I/O bits and register bits can be directly and individually set, reset and tested. The accumulator (A) bits can also be directly and individually tested.

Note: RAM contents are undefined upon power-up.

Data Memory Segment RAM Extension

Data memory address 0FF is used as a memory mapped location for the Data Segment Address Register (S).

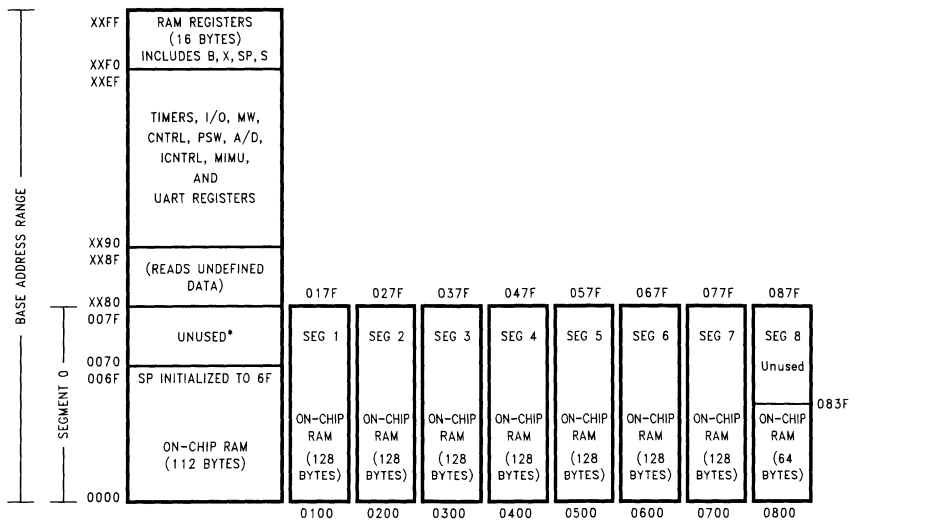
The data store memory is either addressed directly by a single byte address within the instruction, or indirectly relative to the reference of the B, X, or SP pointers (each contains a single-byte address). This single-byte address allows an addressing range of 256 locations from 00 to FF hex. The upper bit of this single-byte address divides the data store memory into two separate sections as outlined previously. With the exception of the RAM register memory from address locations 00F0 to 00FF, all RAM memory is memory mapped with the upper bit of the single-byte address being equal to zero. This allows the upper bit of the single-byte address to determine whether or not the base address range (from 0000 to 00FF) is extended. If this upper bit equals one (representing address range 0080 to 00FF), then address extension does not take place. Alternatively, if this upper bit equals zero, then the data segment extension register S is used to extend the base address range (from 0000 to 007F) from XX00 to XX7F, where XX represents the 8 bits from the S register. Thus the 128-byte data segment extensions are located from addresses 0100 to 017F for data segment 1, 0200 to 027F for data segment 2, etc., up to FF00 to FF7F for data segment 255. The base address range from 0000 to 007F represents data segment 0.

Figure 5 illustrates how the S register data memory extension is used in extending the lower half of the base address range (00 to 7F hex) into 256 data segments of 128 bytes each, with a total addressing range of 32 kbytes from XX00 to XX7F. This organization allows a total of 256 data segments of 128 bytes each with an additional upper base segment of 128 bytes. Furthermore, all addressing modes are available for all data segments. The S register must be changed under program control to move from one data segment (128 bytes) to another. However, the upper base segment (containing the 16 memory registers, I/O registers, control registers, etc.) is always available regardless of the contents of the S register, since the upper base segment (address range 0080 to 00FF) is independent of data segment extension.

The instructions that utilize the stack pointer (SP) always reference the stack as part of the base segment (Segment 0), regardless of the contents of the S register. The S register is not changed by these instructions. Consequently, the stack (used with subroutine linkage and interrupts) is always located in the base segment. The stack pointer will be initialized to point at data memory location 006F as a result of reset.

The 128 bytes of RAM contained in the base segment are split between the lower and upper base segments. The first 112 bytes of RAM are resident from address 0000 to 006F in the lower base segment, while the remaining 16 bytes of RAM represent the 16 data memory registers located at addresses 00F0 to 00FF of the upper base segment. No RAM is located at the upper sixteen addresses (0070 to 007F) of the lower base segment.

Additional RAM beyond these initial 128 bytes, however, will always be memory mapped in groups of 128 bytes (or less) at the data segment address extensions (XX00 to XX7F) of the lower base segment. The additional 960 bytes of RAM are memory mapped at segment 1 through segment 8 (see *Figure 5*).



TL/DD12864-6

*Reads as all ones.

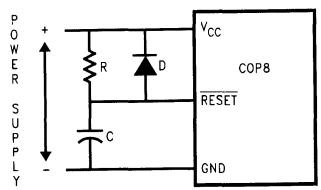
FIGURE 5. RAM Organization

Reset

The $\overline{\text{RESET}}$ input when pulled low initializes the microcontroller. Initialization will occur whenever the $\overline{\text{RESET}}$ input is pulled low. Upon initialization, the data and configuration registers for ports L, G and C are cleared, resulting in these Ports being initialized to the TRI-STATE mode. Pin G1 of the G Port is an exception (as noted below) since pin G1 is dedicated as the WATCHDOG and/or Clock Monitor error output pin. Port D is set high. The PC, PSW, ICNTRL, CNTRL, T2CNTRL and T3CNTRL control registers are cleared. The UART registers PSR, ENU (except that TBMT bit is set), ENUR and ENUI are cleared. The Comparator Select Register is cleared. The S register is initialized to zero. The Multi-Input Wakeup registers WKEN and WKEDG are cleared. Wakeup register WKPND is unknown. The stack pointer, SP, is initialized to 6F hex.

The device comes out of reset with both the WATCHDOG logic and the Clock Monitor detector armed, with the WATCHDOG service window bits set and the Clock Monitor bit set. The WATCHDOG and Clock Monitor circuits are inhibited during reset. The WATCHDOG service window bits being initialized high default to the maximum WATCHDOG service window of 64k t_C clock cycles. The Clock Monitor bit being initialized high will cause a Clock Monitor error following reset if the clock has not reached the minimum specified frequency at the termination of reset. A Clock Monitor error will cause an active low error output on pin G1. This error output will continue until 16 t_C–32 t_C clock cycles following the clock frequency reaching the minimum specified value, at which time the G1 output will enter the TRI-STATE mode.

The external RC network shown in *Figure 6* should be used to ensure that the $\overline{\text{RESET}}$ pin is held low until the power supply to the chip stabilizes.



TL/DD12864–7

RC > 5 × Power Supply Rise Time

FIGURE 6. Recommended Reset Circuit

Oscillator Circuits

The chip can be driven by a clock input on the CKI input pin which can be between DC and 10 MHz. The CKO output clock is on pin G7 (crystal configuration). The CKI input frequency is divided down by 10 to produce the instruction cycle clock ($1/t_C$).

Figure 7 shows the Crystal and R/C oscillator diagrams.

CRYSTAL OSCILLATOR

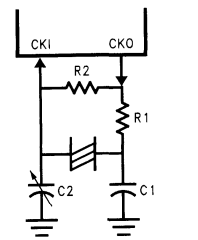
CKI and CKO can be connected to make a closed loop crystal (or resonator) controlled oscillator.

Table A shows the component values required for various standard crystal values.

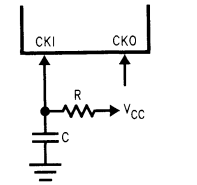
R/C OSCILLATOR

By selecting CKI as a single pin oscillator input, a single pin R/C oscillator circuit can be connected to it. CKO is available as a general purpose input, and/or HALT restart input.

Table II shows the variation in the oscillator frequencies as functions of the component (R and C) values.



TL/DD12864–8



TL/DD12864–9

FIGURE 7. Crystal and R/C Oscillator Diagrams

TABLE I. Crystal Oscillator Configuration, T_A = 25°C

R1 (kΩ)	R2 (MΩ)	C1 (pF)	C2 (pF)	CKI Freq (MHz)	Conditions
0	1	30	30–36	10	V_{CC} = 5V
0	1	30	30–36	4	V_{CC} = 5V
0	1	200	100–150	0.455	V_{CC} = 5V

TABLE II. RC Oscillator Configuration, T_A = 25°C

R (kΩ)	C (pF)	CKI Freq (MHz)	Instr. Cycle (μs)	Conditions
3.3	82	2.2 to 2.7	3.7 to 4.6	V_{CC} = 5V
5.6	100	1.1 to 1.3	7.4 to 9.0	V_{CC} = 5V
6.8	100	0.9 to 1.1	8.8 to 10.8	V_{CC} = 5V

Note: 3k ≤ R ≤ 200k

50 pF ≤ C ≤ 200 pF

Control Registers

CNTRL Register (Address X'00EE)

The Timer1 (T1) and MICROWIRE/PLUS control register contains the following bits:

- SL1 & SL0 Select the MICROWIRE/PLUS clock divide by (00 = 2, 01 = 4, 1x = 8)
- IEDG External interrupt edge polarity select (0 = Rising edge, 1 = Falling edge)
- MSEL Selects G5 and G4 as MICROWIRE/PLUS signals SK and SO respectively
- T1C0 Timer T1 Start/Stop control in timer modes 1 and 2
  Timer T1 Underflow Interrupt Pending Flag in timer mode 3
- T1C1 Timer T1 mode control bit
- T1C2 Timer T1 mode control bit
- T1C3 Timer T1 mode control bit

T1C3	T1C2	T1C1	T1C0	MSEL	IEDG	SL1	SL0

Bit 7 Bit 0

PSW Register (Address X'00EF)

The PSW register contains the following select bits:

- GIE Global interrupt enable (enables interrupts)
- EXEN Enable external interrupt
- BUSY MICROWIRE/PLUS busy shifting flag
- EXPND External interrupt pending
- T1ENA Timer T1 Interrupt Enable for Timer Underflow or T1A Input capture edge
- T1PNDA Timer T1 Interrupt Pending Flag (Autoreload RA in mode 1, T1 Underflow in Mode 2, T1A capture edge in mode 3)
- C Carry Flag
- HC Half Carry Flag

HC	C	T1PNDA	T1ENA	EXPND	BUSY	EXEN	GIE

Bit 7 Bit 0

The Half-Carry bit is also affected by all the instructions that affect the Carry flag. The SC (Set Carry) and RC (Reset Carry) instructions will respectively set or clear both the carry flags. In addition to the SC and RC instructions, ADC, SUBC, RRC and RLC instructions affect the carry and Half Carry flags.

ICNTRL Register (Address X'00E8)

The ICNTRL register contains the following bits:

- T1ENB Timer T1 Interrupt Enable for T1B Input capture edge
- T1PNDB Timer T1 Interrupt Pending Flag for T1B capture edge
- μWEN Enable MICROWIRE/PLUS interrupt
- μWPND MICROWIRE/PLUS interrupt pending
- T0EN Timer T0 Interrupt Enable (Bit 12 toggle)
- T0PND Timer T0 Interrupt pending
- LPEN L Port Interrupt Enable (Multi-Input Wakeup/Interrupt)
  Bit 7 could be used as a flag

Unused	LPEN	T0PND	T0EN	μWPND	μWEN	T1PNDB	T1ENB

Bit 7 Bit 0

T2CNTRL Register (Address X'00C6)

The T2CNTRL register contains the following bits:

- T2ENB Timer T2 Interrupt Enable for T2B Input capture edge
- T2PNDB Timer T2 Interrupt Pending Flag for T2B capture edge
- T2ENA Timer T2 Interrupt Enable for Timer Underflow or T2A Input capture edge
- T2PNDA Timer T2 Interrupt Pending Flag (Autoreload RA in mode 1, T2 Underflow in mode 2, T2A capture edge in mode 3)
- T2C0 Timer T2 Start/Stop control in timer modes 1 and 2 Timer T2 Underflow Interrupt Pending Flag in timer mode 3
- T2C1 Timer T2 mode control bit
- T2C2 Timer T2 mode control bit
- T2C3 Timer T2 mode control bit

T2C3	T2C2	T2C1	T2C0	T2PNDA	T2ENA	T2PNDB	T2ENB

Bit 7 Bit 0

T3CNTRL Register (Address X'00B6)

The T3CNTRL register contains the following bits:

- T3ENB Timer T3 Interrupt Enable for T3B Input capture edge
- T3PNDB Timer T3 Interrupt Pending Flag for T3B capture edge
- T3ENA Timer T3 Interrupt Enable for Timer Underflow or T3A Input capture edge
- T3PNDA Timer T3 Interrupt Pending Flag (Autoreload RA in mode 1, T3 Underflow in mode 2, T3A capture edge in mode 3)
- T3C0 Timer T3 Start/Stop control in timer modes 1 and 2
  Timer T3 Underflow Interrupt Pending Flag in timer mode 3
- T3C1 Timer T3 mode control bit
- T3C2 Timer T3 mode control bit
- T3C3 Timer T3 mode control bit

T3C3	T3C2	T3C1	T3C0	T3PNDA	T3ENA	T3PNDB	T3ENB

Bit 7 Bit 0

Timers

The device contains a very versatile set of timers (T0, T1, T2, T3). All timers and associated autoreload/capture registers power up containing random data.

TIMER T0 (IDLE TIMER)

The device supports applications that require maintaining real time and low power with the IDLE mode. This IDLE mode support is furnished by the IDLE timer T0, which is a 16-bit timer. The Timer T0 runs continuously at the fixed rate of the instruction cycle clock, t_C. The user cannot read or write to the IDLE Timer T0, which is a count down timer.

The Timer T0 supports the following functions:

- Exit out of the Idle Mode (See Idle Mode description)
- WATCHDOG logic (See WATCHDOG description)
- Start up delay out of the HALT mode

The IDLE Timer T0 can generate an interrupt when the thirteenth bit toggles. This toggle is latched into the T0PND pending flag, and will occur every 4 ms at the maximum clock frequency (t_C = 1 μs). A control flag T0EN allows the interrupt from the thirteenth bit of Timer T0 to be enabled or disabled. Setting T0EN will enable the interrupt, while resetting it will disable the interrupt.

TIMER T1, TIMER T2 AND TIMER T3

The device has a set of three powerful timer/counter blocks, T1, T2 and T3. The associated features and functioning of a timer block are described by referring to the timer block Tx. Since the three timer blocks, T1, T2 and T3 are identical, all comments are equally applicable to any of the three timer blocks.

Each timer block consists of a 16-bit timer, Tx, and two supporting 16-bit autoreload/capture registers, RxA and RxB. Each timer block has two pins associated with it, TxA and TxB. The pin TxA supports I/O required by the timer block, while the pin TxB is an input to the timer block. The powerful and flexible timer block allows the device to easily perform all timer functions with minimal software overhead. The timer block has three operating modes: Processor Independent PWM mode, External Event Counter mode, and Input Capture mode.

The control bits TxC3, TxC2, and TxC1 allow selection of the different modes of operation.

Mode 1. Processor Independent PWM Mode

As the name suggests, this mode allows the device to generate a PWM signal with very minimal user intervention. The user only has to define the parameters of the PWM signal (ON time and OFF time). Once begun, the timer block will continuously generate the PWM signal completely independent of the microcontroller. The user software services the timer block only when the PWM parameters require updating.

In this mode the timer Tx counts down at a fixed rate of t_C. Upon every underflow the timer is alternately reloaded with the contents of supporting registers, RxA and RxB. The very first underflow of the timer causes the timer to reload from the register RxA. Subsequent underflows cause the timer to be reloaded from the registers alternately beginning with the register RxB.

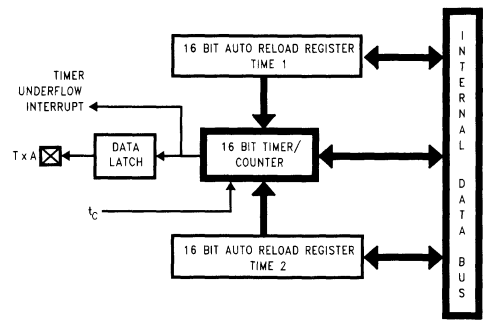
The Tx Timer control bits, TxC3, TxC2 and TxC1 set up the timer for PWM mode operation.

Figure 8 shows a block diagram of the timer in PWM mode.

The underflows can be programmed to toggle the TxA output pin. The underflows can also be programmed to generate interrupts.

Underflows from the timer are alternately latched into two pending flags, TxPNDA and TxPNDB. The user must reset these pending flags under software control. Two control enable flags, TxENA and TxENB, allow the interrupts from the timer underflow to be enabled or disabled. Setting the timer enable flag TxENA will cause an interrupt when a timer underflow causes the RxA register to be reloaded into the timer. Setting the timer enable flag TxENB will cause an interrupt when a timer underflow causes the RxB register to be reloaded into the timer. Resetting the timer enable flags will disable the associated interrupts.

Either or both of the timer underflow interrupts may be enabled. This gives the user the flexibility of interrupting once per PWM period on either the rising or falling edge of the PWM output. Alternatively, the user may choose to interrupt on both edges of the PWM output.



TL/DD12864–10

FIGURE 8. Timer in PWM Mode

Mode 2. External Event Counter Mode

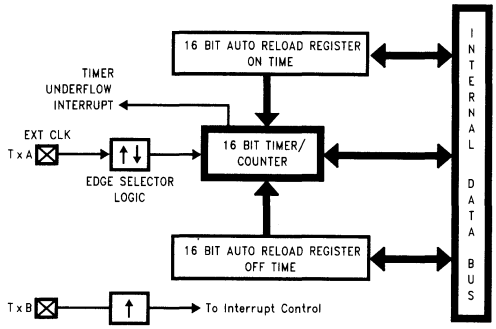
This mode is quite similar to the processor independent PWM mode described above. The main difference is that the timer, Tx, is clocked by the input signal from the TxA pin. The Tx timer control bits, TxC3, TxC2 and TxC1 allow the timer to be clocked either on a positive or negative edge from the TxA pin. Underflows from the timer are latched into the TxPNDA pending flag. Setting the TxENA control flag will cause an interrupt when the timer underflows.

In this mode the input pin TxB can be used as an independent positive edge sensitive interrupt input if the TxENB control flag is set. The occurrence of a positive edge on the TxB input pin is latched into the TxPNDB flag.

Timers (Continued)

Figure 9 shows a block diagram of the timer in External Event Counter mode.

Note: The PWM output is not available in this mode since the TxA pin is being used as the counter input clock.



TL/DD12864-11

FIGURE 9. Timer in External Event Counter Mode

Mode 3. Input Capture Mode

The device can precisely measure external frequencies or time external events by placing the timer block, Tx, in the input capture mode.

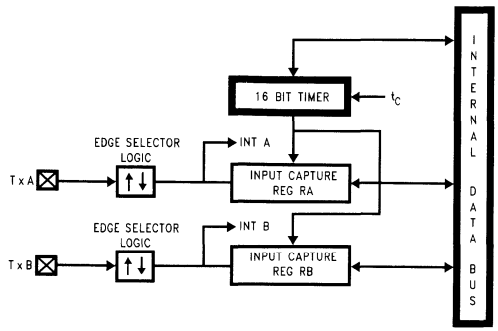
In this mode, the timer Tx is constantly running at the fixed t_C rate. The two registers, RxA and RxB, act as capture registers. Each register acts in conjunction with a pin. The register RxA acts in conjunction with the TxA pin and the register RxB acts in conjunction with the TxB pin.

The timer value gets copied over into the register when a trigger event occurs on its corresponding pin. Control bits, TxC3, TxC2 and TxC1, allow the trigger events to be specified either as a positive or a negative edge. The trigger condition for each input pin can be specified independently.

The trigger conditions can also be programmed to generate interrupts. The occurrence of the specified trigger condition on the TxA and TxB pins will be respectively latched into the pending flags, TxPNDA and TxPNDB. The control flag TxENA allows the interrupt on TxA to be either enabled or disabled. Setting the TxENA flag enables interrupts to be generated when the selected trigger condition occurs on the TxA pin. Similarly, the flag TxENB controls the interrupts from the TxB pin.

Underflows from the timer can also be programmed to generate interrupts. Underflows are latched into the timer TxC0 pending flag (the TxC0 control bit serves as the timer underflow interrupt pending flag in the Input Capture mode). Consequently, the TxC0 control bit should be reset when entering the Input Capture mode. The timer underflow interrupt is enabled with the TxENA control flag. When a TxA interrupt occurs in the Input Capture mode, the user must check both the TxPNDA and TxC0 pending flags in order to determine whether a TxA input capture or a timer underflow (or both) caused the interrupt.

Figure 10 shows a block diagram of the timer in Input Capture mode.



TL/DD12864-12

FIGURE 10. Timer in Input Capture Mode

TIMER CONTROL FLAGS

The timers T1, T2 and T3 have indentical control structures. The control bits and their functions are summarized below.

TxC0	Timer Start/Stop control in Modes 1 and 2 (Processor Independent PWM and External Event Counter), where 1 = Start, 0 = Stop Timer Underflow Interrupt Pending Flag in Mode 3 (Input Capture)
TxPNDA	Timer Interrupt Pending Flag
TxPNDB	Timer Interrupt Pending Flag
TxENA	Timer Interrupt Enable Flag
TxENB	Timer Interrupt Enable Flag 1 = Timer Interrupt Enabled 0 = Timer Interrupt Disabled
TxC3	Timer mode control
TxC2	Timer mode control
TxC1	Timer mode control

Timers (Continued)

The timer mode control bits (TxC3, TxC2 and TxC1) are detailed below:

TxC3	TxC2	TxC1	Timer Mode	Interrupt A Source	Interrupt B Source	Timer Counts On
0	0	0	MODE 2 (External Event Counter)	Timer Underflow	Pos. TxB Edge	TxA Pos. Edge
0	0	1	MODE 2 (External Event Counter)	Timer Underflow	Pos. TxB Edge	TxA Neg. Edge
1	0	1	MODE 1 (PWM) TxA Toggle	Autoreload RA	Autoreload RB	t_C
1	0	0	MODE 1 (PWM) No TxA Toggle	Autoreload RA	Autoreload RB	t_C
0	1	0	MODE 3 (Capture) Captures: TxA Pos. Edge TxB Pos. Edge	Pos. TxA Edge or Timer Underflow	Pos. TxB Edge	t_C
1	1	0	MODE 3 (Capture) Captures: TxA Pos. Edge TxB Neg. Edge	Pos. TxA Edge or Timer Underflow	Neg. TxB Edge	t_C
0	1	1	MODE 3 (Capture) Captures: TxA Neg. Edge TxB Pos. Edge	Neg. TxA Edge or Timer Underflow	Pos. TxB Edge	t_C
1	1	1	MODE 3 (Capture) Captures: TxA Neg. Edge TxB Neg. Edge	Neg. TxA Edge or Timer Underflow	Neg. TxB Edge	t_C

Power Save Modes

The device offers the user two power save modes of operation: HALT and IDLE. In the HALT mode, all microcontroller activities are stopped. In the IDLE mode, the on-board oscillator circuitry and timer T0 are active but all other microcontroller activities are stopped. In either mode, all on-board RAM, registers, I/O states, and timers (with the exception of T0) are unaltered.

HALT MODE

The device can be placed in the HALT mode by writing a "1" to the HALT flag (G7 data bit). All microcontroller activities, including the clock and timers, are stopped. The WATCHDOG logic on the device is disabled during the HALT mode. However, the clock monitor circuitry if enabled remains active and will cause the WATCHDOG output pin (WDOUT) to go low. If the HALT mode is used and the user does not want to activate the WDOUT pin, the Clock Monitor should be disabled after the device comes out of reset (resetting the Clock Monitor control bit with the first write to the WDSVR register). In the HALT mode, the power requirements of the device are minimal and the applied voltage (V_{CC}) may be decreased to V_r (V_r = 2.0V) without altering the state of the machine.

The device supports three different ways of exiting the HALT mode. The first method of exiting the HALT mode is with the Multi-Input Wakeup feature on the L port. The second method is with a low to high transition on the CKO (G7) pin. This method precludes the use of the crystal clock configuration (since CKO becomes a dedicated output), and so may only be used with an RC clock configuration. The third method of exiting the HALT mode is by pulling the $\overline{\text{RESET}}$ pin low.

Since a crystal or ceramic resonator may be selected as the oscillator, the Wakeup signal is not allowed to start the chip running immediately since crystal oscillators and ceramic resonators have a delayed start up time to reach full amplitude and frequency stability. The IDLE timer is used to generate a fixed delay to ensure that the oscillator has indeed stabilized before allowing instruction execution. In this case, upon detecting a valid Wakeup signal, only the oscillator circuitry is enabled. The IDLE timer is loaded with a value of 256 and is clocked with the t_C instruction cycle clock. The t_C clock is derived by dividing the oscillator clock down by a factor of 10. The Schmitt trigger following the CKI inverter on the chip ensures that the IDLE timer is clocked only when the oscillator has a sufficiently large amplitude to meet the Schmitt trigger specifications. This Schmitt trigger is not part of the oscillator closed loop. The startup timeout from the IDLE timer enables the clock signals to be routed to the rest of the chip.

If an RC clock option is being used, the fixed delay is introduced optionally. A control bit, CLKDLY, mapped as configuration bit G7, controls whether the delay is to be introduced or not. The delay is included if CLKDLY is set, and excluded if CLKDLY is reset. The CLKDLY bit is cleared on reset.

Power Save Modes (Continued)

The device has two mask options associated with the HALT mode. The first mask option enables the HALT mode feature, while the second mask option disables the HALT mode. With the HALT mode enable mask option, the device will enter and exit the HALT mode as described above. With the HALT disable mask option, the device cannot be placed in the HALT mode (writing a "1" to the HALT flag will have no effect, the HALT flag will remain "0").

IDLE MODE

The device is placed in the IDLE mode by writing a "1" to the IDLE flag (G6 data bit). In this mode, all activities, except the associated on-board oscillator circuitry and the IDLE Timer T0, are stopped. The power supply requirements of the micro-controller in this mode of operation are typically around 30% of normal power requirement of the microcontroller.

As with the HALT mode, the device can be returned to normal operation with a reset, or with a Multi-Input Wakeup from the L Port. Alternately, the microcontroller resumes normal operation from the IDLE mode when the thirteenth bit (representing 4.096 ms at internal clock frequency of 10 MHz, t_c = 1 μs) of the IDLE Timer toggles.

This toggle condition of the thirteenth bit of the IDLE Timer T0 is latched into the T0PND pending flag.

The user has the option of being interrupted with a transition on the thirteenth bit of the IDLE Timer T0. The interrupt can be enabled or disabled via the T0EN control bit. Setting the T0EN flag enables the interrupt and vice versa.

The user can enter the IDLE mode with the Timer T0 interrupt enabled. In this case, when the T0PND bit gets set, the device will first execute the Timer T0 interrupt service routine and then return to the instruction following the "Enter Idle Mode" instruction.

Alternatively, the user can enter the IDLE mode with the IDLE Timer T0 interrupt disabled. In this case, the device will resume normal operation with the instruction immediately following the "Enter IDLE Mode" instruction.

Note: It is necessary to program two NOP instructions following both the set HALT mode and set IDLE mode instructions. These NOP instructions are necessary to allow clock resynchronization following the HALT or IDLE modes.

Multi-Input Wakeup

The Multi-Input Wakeup feature is ued to return (wakeup) the device from either the HALT or IDLE modes. Alternately Multi-Input Wakeup/Interrupt feature may also be used to generate up to 8 edge selectable external interrupts.

Figure 11 shows the Multi-Input Wakeup logic.

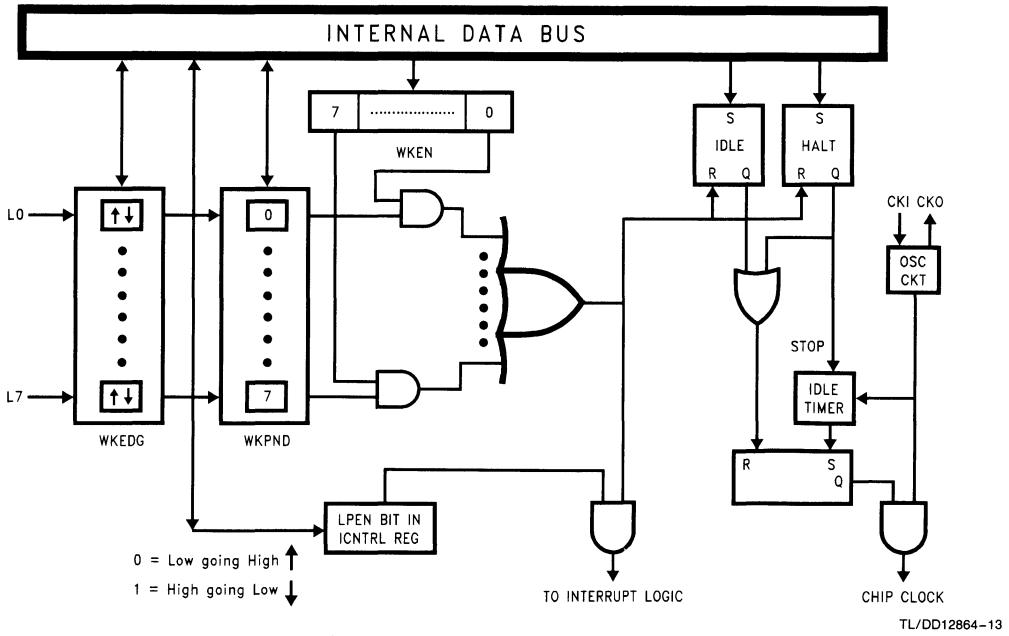


FIGURE 11. Multi-Input Wake Up Logic

Multi-Input Wakeup (Continued)

The Multi-Input Wakeup feature utilizes the L Port. The user selects which particular L port bit (or combination of L Port bits) will cause the device to exit the HALT or IDLE modes. The selection is done through the Register WKEN. The Register WKEN is an 8-bit read/write register, which contains a control bit for every L port bit. Setting a particular WKEN bit enables a Wakeup from the associated L port pin.

The user can select whether the trigger condition on the selected L Port pin is going to be either a positive edge (low to high transition) or a negative edge (high to low transition). This selection is made via the Register WKEDG, which is an 8-bit control register with a bit assigned to each L Port pin. Setting the control bit will select the trigger condition to be a negative edge on that particular L Port pin. Resetting the bit selects the trigger condition to be a positive edge. Changing an edge select entails several steps in order to avoid a Wakeup condition as a result of the edge change. First, the associated WKEN bit should be reset, followed by the edge select change in WKEDG. Next, the associated WKPND bit should be cleared, followed by the associated WKEN bit being re-enabled.

An example may serve to clarify this procedure. Suppose we wish to change the edge select from positive (low going high) to negative (high going low) for L Port bit 5, where bit 5 has previously been enabled for an input interrupt. The program would be as follows:

```
RBIT  5, WKEN      ; Disable Port bit 5 for
                   ; Wakeup
SBIT  5, WKEDG     ; Select neg going edge
                   ; sensitivity
RBIT  5, WKPND     ; Clear pending bit
SBIT  5, WKEN      ; Re-enable the bit for
                   ; Wakeup
```

If the L port bits have been used as outputs and then changed to inputs with Multi-Input Wakeup/Interrupt, a safety procedure should also be followed to avoid wakeup conditions. After the selected L port bits have been changed from output to input but before the associated WKEN bits are enabled, the associated edge select bits in WKEDG should be set or reset for the desired edge selects, followed by the associated WKPND bits being cleared.

This same procedure should be used following reset, since the L port inputs are left floating as a result of reset.

The occurrence of the selected trigger condition for Multi-Input Wakeup is latched into a pending register called WKPND. The respective bits of the WKPND register will be set on the occurrence of the selected trigger edge on the corresponding Port L pin. The user has the responsibility of clearing these pending flags. Since WKPND is a pending register for the occurrence of selected wakeup conditions, the device will not enter the HALT mode if any Wakeup bit is both enabled and pending. Consequently, the user must clear the pending flags before attempting to enter the HALT mode.

WKEN, WKPND and WKEDG are all read/write registers, and are cleared at reset.

PORT L INTERRUPTS

Port L provides the user with an additional eight fully selectable, edge sensitive interrupts which are all vectored into the same service subroutine.

The interrupt from Port L shares logic with the wake up circuitry. The register WKEN allows interrupts from Port L to be individually enabled or disabled. The register WKEDG specifies the trigger condition to be either a positive or a negative edge. Finally, the register WKPND latches in the pending trigger conditions.

The GIE (Global Interrupt Enable) bit enables the interrupt function.

A control flag, LPEN, functions as a global interrupt enable for Port L interrupts. Setting the LPEN flag will enable interrupts and vice versa. A separate global pending flag is not needed since the register WKPND is adequate.

Since Port L is also used for waking the device out of the HALT or IDLE modes, the user can elect to exit the HALT or IDLE modes either with or without the interrupt enabled. If he elects to disable the interrupt, then the device will restart execution from the instruction immediately following the instruction that placed the microcontroller in the HALT or IDLE modes. In the other case, the device will first execute the interrupt service routine and then revert to normal operation. (See HALT MODE for clock option wakeup information.)

UART

The device contains a full-duplex software programmable UART. The UART *(Figure 12)* consists of a transmit shift register, a receive shift register and seven addressable registers, as follows: a transmit buffer register (TBUF), a receiver buffer register (RBUF), a UART control and status register (ENU), a UART receive control and status register (ENUR), a UART interrupt and clock source register (ENUI), a prescaler select register (PSR) and baud (BAUD) register. The ENU register contains flags for transmit and receive functions; this register also determines the length of the data frame (7, 8 or 9 bits), the value of the ninth bit in transmission, and parity selection bits. The ENUR register flags framing, data overrun and parity errors while the UART is receiving.

Other functions of the ENUR register include saving the ninth bit received in the data frame, enabling or disabling the UART's attention mode of operation and providing additional receiver/transmitter status information via RCVG and XMTG bits. The determination of an internal or external clock source is done by the ENUI register, as well as selecting the number of stop bits and enabling or disabling transmit and receive interrupts. A control flag in this register can also select the UART mode of operation: asynchronous or synchronous.

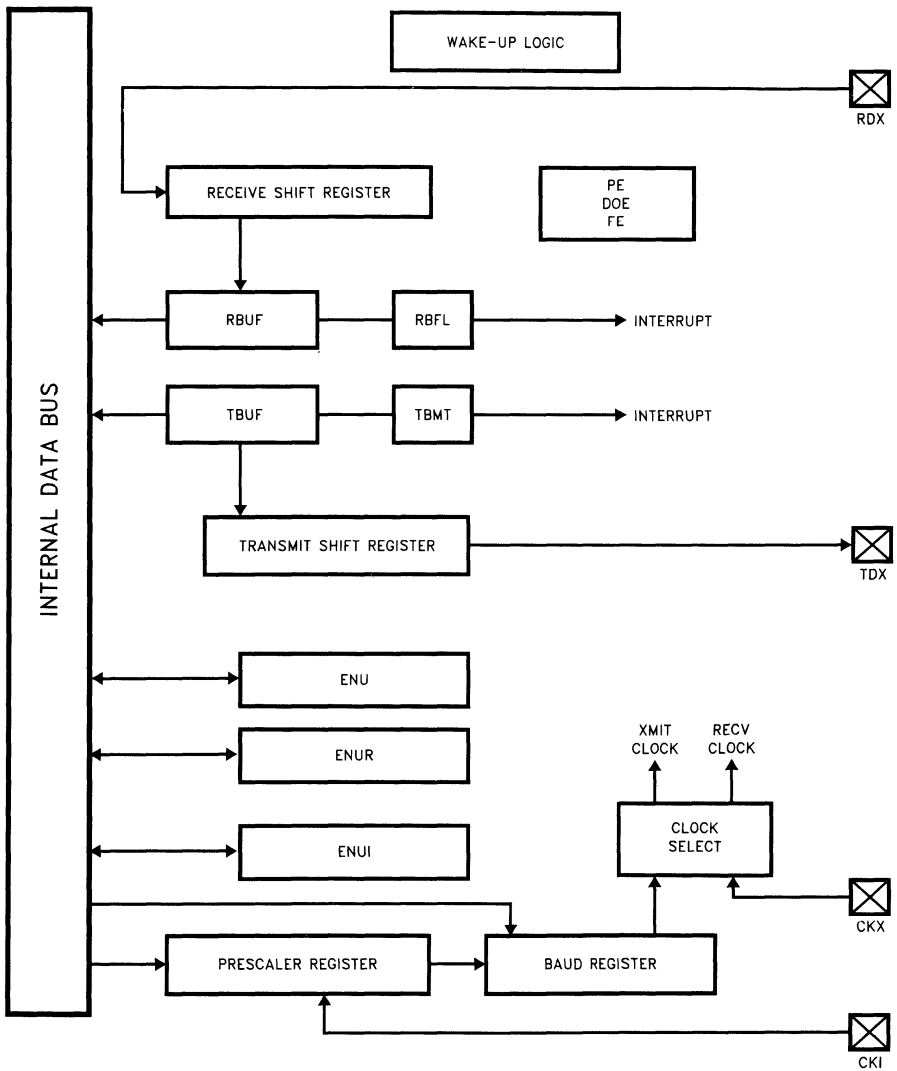


FIGURE 12. UART Block Diagram

UART (Continued)

UART CONTROL AND STATUS REGISTERS

The operation of the UART is programmed through three registers: ENU, ENUR and ENUI. The function of the individual bits in these registers is as follows:

ENU-UART Control and Status Register (Address at 0BA)

PEN	PSEL1	XBIT9/ PSEL0	CHL1	CHL0	ERR	RBFL	TBMT
0RW	0RW	0RW	0RW	0RW	0R	0R	1R

Bit 7 Bit 0

ENUR-UART Receive Control and Status Register (Address at 0BB)

DOE	FE	PE	SPARE	RBIT9	ATTN	XMTG	RCVG
0RD	0RD	0RD	0RW*	0R	0RW	0R	0R

Bit7 Bit0

ENUI-UART Interrupt and Clock Source Register (Address at 0BC)

STP2	STP78	ETDX	SSEL	XRCLK	XTCLK	ERI	ETI
0RW	0RW	0RW	0RW	0RW	0RW	0RW	0RW

Bit7 Bit0

*Bit is not used.

0 Bit is cleared on reset.

1 Bit is set to one on reset.

R Bit is read-only; it cannot be written by software.

RW Bit is read/write.

D Bit is cleared on read; when read by software as a one, it is cleared automatically. Writing to the bit does not affect its state.

DESCRIPTION OF UART REGISTER BITS

ENU—UART CONTROL AND STATUS REGISTER

TBMT: This bit is set when the UART transfers a byte of data from the TBUF register into the TSFT register for transmission. It is automatically reset when software writes into the TBUF register.

RBFL: This bit is set when the UART has received a complete character and has copied it into the RBUF register. It is automatically reset when software reads the character from RBUF.

ERR: This bit is a global UART error flag which gets set if any or a combination of the errors (DOE, FE, PE) occur.

CHL1, CHL0: These bits select the character frame format. Parity is not included and is generated/verified by hardware.

CHL1 = 0, CHL0 = 0 The frame contains eight data bits.
CHL1 = 0, CHL0 = 1 The frame contains seven data bits.
CHL1 = 1, CHL0 = 0 The frame contains nine data bits.
CHL1 = 1, CHL0 = 1 Loopback Mode selected. Transmitter output internally looped back to receiver input. Nine bit framing format is used.

XBIT9/PSEL0: Programs the ninth bit for transmission when the UART is operating with nine data bits per frame. For seven or eight data bits per frame, this bit in conjunction with PSEL1 selects parity.

PSEL1, PSEL0: Parity select bits.

PSEL1 = 0, PSEL0 = 0 Odd Parity (if Parity enabled)
PSEL1 = 0, PSEL0 = 1 Odd Parity (if Parity enabled)
PSEL1 = 1, PSEL0 = 0 Mark(1) (if Parity enabled)
PSEL1 = 1, PSEL0 = 1 Space(0) (if Parity enabled)

PEN: This bit enables/disables Parity (7- and 8-bit modes only).

PEN = 0 Parity disabled.
PEN = 1 Parity enabled.

ENUR—UART RECEIVE CONTROL AND STATUS REGISTER

RCVG: This bit is set high whenever a framing error occurs and goes low when RDX goes high.

XMTG: This bit is set to indicate that the UART is transmitting. It gets reset at the end of the last frame (end of last Stop bit).

ATTN: ATTENTION Mode is enabled while this bit is set. This bit is cleared automatically on receiving a character with data bit nine set.

RBIT9: Contains the ninth data bit received when the UART is operating with nine data bits per frame.

SPARE: Reserved for future use.

PE: Flags a Parity Error.

PE = 0 Indicates no Parity Error has been detected since the last time the ENUR register was read.
PE = 1 Indicates the occurrence of a Parity Error.

FE: Flags a Framing Error.

FE = 0 Indicates no Framing Error has been detected since the last time the ENUR register was read.
FE = 1 Indicates the occurrence of a Framing Error.

DOE: Flags a Data Overrun Error.

DOE = 0 Indicates no Data Overrun Error has been detected since the last time the ENUR register was read.
DOE = 1 Indicates the occurrence of a Data Overrun Error.

ENUI—UART INTERRUPT AND CLOCK SOURCE REGISTER

ETI: This bit enables/disables interrupt from the transmitter section.

ETI = 0 Interrupt from the transmitter is disabled.
ETI = 1 Interrupt from the transmitter is enabled.

ERI: This bit enables/disables interrupt from the receiver section.

ERI = 0 Interrupt from the receiver is disabled.
ERI = 1 Interrupt from the receiver is enabled.

XTCLK: This bit selects the clock source for the transmitter section.

XTCLK = 0 The clock source is selected through the PSR and BAUD registers.
XTCLK = 1 Signal on CKX (L1) pin is used as the clock.

XRCLK: This bit selects the clock source for the receiver section.

XRCLK = 0 The clock source is selected through the PSR and BAUD registers.
XRCLK = 1 Signal on CKX (L1) pin is used as the clock.

SSEL: UART mode select.

SSEL = 0 Asynchronous Mode.
SSEL = 1 Synchronous Mode.

UART (Continued)

ETDX: TDX (UART Transmit Pin) is the alternate function assigned to Port L pin L2; it is selected by setting ETDX bit. To simulate line break generation, software should reset ETDX bit and output logic zero to TDX pin through Port L data and configuration registers.

STP78: This bit is set to program the last Stop bit to be 7/8th of a bit in length.

STP2: This bit programs the number of Stop bits to be transmitted.

STP2 = 0 One Stop bit transmitted.
STP2 = 1 Two Stop bits transmitted.

Associated I/O Pins

Data is transmitted on the TDX pin and received on the RDX pin. TDX is the alternate function assigned to Port L pin L2; it is selected by setting ETDX (in the ENUI register) to one. RDX is an inherent function of Port L pin L3, requiring no setup.

The baud rate clock for the UART can be generated on-chip, or can be taken from an external source. Port L pin L1 (CKX) is the external clock I/O pin. The CKX pin can be either an input or an output, as determined by Port L Configuration and Data registers (Bit 1). As an input, it accepts a clock signal which may be selected to drive the transmitter and/or receiver. As an output, it presents the internal Baud Rate Generator output.

UART Operation

The UART has two modes of operation: asynchronous mode and synchronous mode.

ASYNCHRONOUS MODE

This mode is selected by resetting the SSEL (in the ENUI register) bit to zero. The input frequency to the UART is 16 times the baud rate.

The TSFT and TBUF registers double-buffer data for transmission. While TSFT is shifting out the current character on the TDX pin, the TBUF register may be loaded by software with the next byte to be transmitted. When TSFT finishes transmitting the current character the contents of TBUF are transferred to the TSFT register and the Transmit Buffer Empty Flag (TBMT in the ENU register) is set. The TBMT flag is automatically reset by the UART when software loads a new character into the TBUF register. There is also the XMTG bit which is set to indicate that the UART is transmitting. This bit gets reset at the end of the last frame (end of last Stop bit). TBUF is a read/write register.

The RSFT and RBUF registers double-buffer data being received. The UART receiver continually monitors the signal on the RDX pin for a low level to detect the beginning of a Start bit. Upon sensing this low level, it waits for half a bit time and samples again. If the RDX pin is still low, the receiver considers this to be a valid Start bit, and the remaining bits in the character frame are each sampled a single time, at the mid-bit position. Serial data input on the RDX pin is shifted into the RSFT register. Upon receiving the complete character, the contents of the RSFT register are copied into the RBUF register and the Received Buffer Full Flag (RBFL) is set. RBFL is automatically reset when software reads the character from the RBUF register. RBUF is a read only register. There is also the RCVG bit which is set high when a framing error occurs and goes low once RDX goes high. TBMT, XMTG, RBFL and RCVG are read only bits.

SYNCHRONOUS MODE

In this mode data is transferred synchronously with the clock. Data is transmitted on the rising edge and received on the falling edge of the synchronous clock.

This mode is selected by setting SSEL bit in the ENUI register. The input frequency to the UART is the same as the baud rate.

When an external clock input is selected at the CKX pin, data transmit and receive are performed synchronously with this clock through TDX/RDX pins.

If data transmit and receive are selected with the CKX pin as clock output, the device generates the synchronous clock output at the CKX pin. The internal baud rate generator is used to produce the synchronous clock. Data transmit and receive are performed synchronously with this clock.

FRAMING FORMATS

The UART supports several serial framing formats *(Figure 13)*. The format is selected using control bits in the ENU, ENUR and ENUI registers.

The first format (1, 1a, 1b, 1c) for data transmission (CHL0 = 1, CHL1 = 0) consists of Start bit, seven Data bits (excluding parity) and 7/8, one or two Stop bits. In applications using parity, the parity bit is generated and verified by hardware.

The second format (CHL0 = 0, CHL1 = 0) consists of one Start bit, eight Data bits (excluding parity) and 7/8, one or two Stop bits. Parity bit is generated and verified by hardware.

The third format for transmission (CHL0 = 0, CHL1 = 1) consists of one Start bit, nine Data bits and 7/8, one or two Stop bits. This format also supports the UART "ATTENTION" feature. When operating in this format, all eight bits of TBUF and RBUF are used for data. The ninth data bit is transmitted and received using two bits in the ENU and ENUR registers, called XBIT9 and RBIT9. RBIT9 is a read only bit. Parity is not generated or verified in this mode.

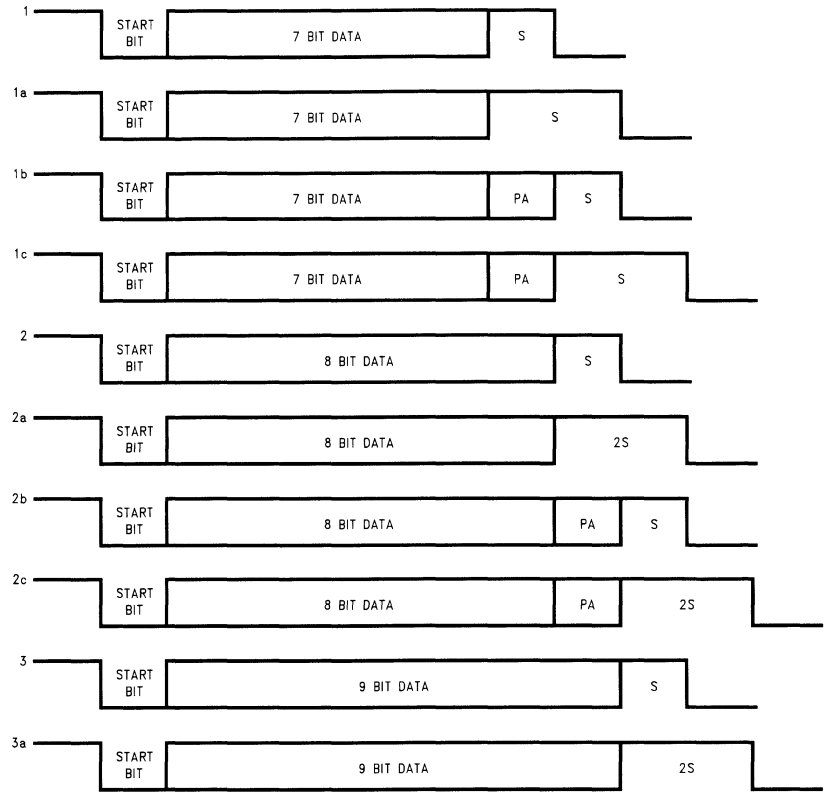
For any of the above framing formats, the last Stop bit can be programmed to be 7/8th of a bit in length. If two Stop bits are selected and the 7/8th bit is set (selected), the second Stop bit will be 7/8th of a bit in length.

The parity is enabled/disabled by PEN bit located in the ENU register. Parity is selected for 7- and 8-bit modes only. If parity is enabled (PEN = 1), the parity selection is then performed by PSEL0 and PSEL1 bits located in the ENU register.

Note that the XBIT9/PSEL0 bit located in the ENU register serves two mutually exclusive functions. This bit programs the ninth bit for transmission when the UART is operating with nine data bits per frame. There is no parity selection in this framing format. For other framing formats XBIT9 is not needed and the bit is PSEL0 used in conjunction with PSEL1 to select parity.

The frame formats for the receiver differ from the transmitter in the number of Stop bits required. The receiver only requires one Stop bit in a frame, regardless of the setting of the Stop bit selection bits in the control register. Note that an implicit assumption is made for full duplex UART operation that the framing formats are the same for the transmitter and receiver.

UART Operation (Continued)



TL/DD12864–15

FIGURE 13. Framing Formats

UART INTERRUPTS

The UART is capable of generating interrupts. Interrupts are generated on Receive Buffer Full and Transmit Buffer Empty. Both interrupts have individual interrupt vectors. Two bytes of program memory space are reserved for each interrupt vector. The two vectors are located at addresses 0xEC to 0xEF Hex in the program memory space. The interrupts can be individually enabled or disabled using Enable Transmit Interrupt (ETI) and Enable Receive Interrupt (ERI) bits in the ENUI register.

The interrupt from the Transmitter is set pending, and remains pending, as long as both the TBMT and ETI bits are set. To remove this interrupt, software must either clear the ETI bit or write to the TBUF register (thus clearing the TBMT bit).

The interrupt from the receiver is set pending, and remains pending, as long as both the RBFL and ERI bits are set. To remove this interrupt, software must either clear the ERI bit or read from the RBUF register (thus clearing the RBFL bit).

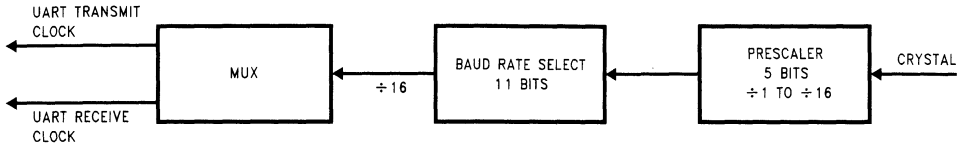
Baud Clock Generation

The clock inputs to the transmitter and receiver sections of the UART can be individually selected to come either from an external source at the CKX pin (port L, pin L1) or from a source selected in the PSR and BAUD registers. Internally, the basic baud clock is created from the oscillator frequency through a two-stage divider chain consisting of a 1–16 (increments of 0.5) prescaler and an 11-bit binary counter. *(Figure 14).* The divide factors are specified through two read/write registers shown in *Figure 15.* Note that the 11-bit Baud Rate Divisor spills over into the Prescaler Select Register (PSR). PSR is cleared upon reset.

As shown in Table V, a Prescaler Factor of 0 corresponds to NO CLOCK. This condition is the UART power down mode where the UART clock is turned off for power saving purpose. The user must also turn the UART clock off when a different baud rate is chosen.

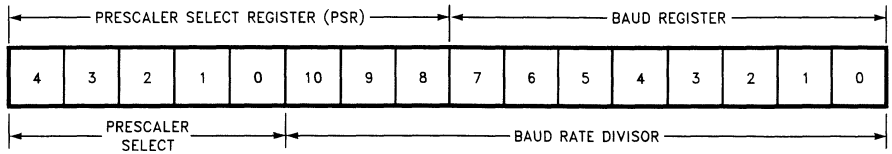
The correspondences between the 5-bit Prescaler Select and Prescaler factors are shown in Table V. There are many ways to calculate the two divisor factors, but one particularly effective method would be to achieve a 1.8432 MHz frequency coming out of the first stage. The 1.8432 MHz prescaler output is then used to drive the software programmable baud rate counter to create a 16x clock for the following baud rates: 110, 134.5, 150, 300, 600, 1200, 1800, 2400, 3600, 4800, 7200, 9600, 19200 and 38400 (Table I). Other baud rates may be created by using appropriate divisors. The 16x clock is then divided by 16 to provide the rate for the serial shift registers of the transmitter and receiver.

Baud Clock Generation (Continued)



TL/DD12864–16

FIGURE 14. UART BAUD Clock Generation



TL/DD12864–17

FIGURE 15. UART BAUD Clock Divisor Registers

Baud Clock Generation (Continued)

TABLE III. Baud Rate Divisors (1.8432 MHz Prescaler Output)

Baud Rate	Baud Rate Divisor − 1 (N-1)
110 (110.03)	1046
134.5 (134.58)	855
150	767
300	383
600	191
1200	95
1800	63
2400	47
3600	31
4800	23
7200	15
9600	11
19200	5
38400	2

Note: The entries in Table IV assume a prescaler output of 1.8432 MHz. In the asynchronous mode the baud rate could be as high as 625k.

TABLE IV. Prescaler Factors

Prescaler Select	Prescaler Factor
00000	NO CLOCK
00001	1
00010	1.5
00011	2
00100	2.5
00101	3
00110	3.5
00111	4
01000	4.5
01001	5
01010	5.5
01011	6
01100	6.5
01101	7
01110	7.5
01111	8
10000	8.5
10001	9
10010	9.5
10011	10
10100	10.5
10101	11
10110	11.5
10111	12
11000	12.5
11001	13
11010	13.5
11011	14
11100	14.5
11101	15
11110	15.5
11111	16

As an example, considering Asynchronous Mode and a CKI clock of 4.608 MHz, the prescaler factor selected is:

$$4.608/1.8432 = 2.5$$

The 2.5 entry is available in Table IV. The 1.8432 MHz prescaler output is then used with proper Baud Rate Divisor (Table V) to obtain different baud rates. For a baud rate of 19200 e.g., the entry in Table IV is 5.

$N - 1 = 5$ ($N - 1$ is the value from Table III)

$N = 6$ (N is the Baud Rate Divisor)

$$\text{Baud Rate} = 1.8432\ \text{MHz}/(16 \times 6) = 19200$$

The divide by 16 is performed because in the asynchronous mode, the input frequency to the UART is 16 times the baud rate. The equation to calculate baud rates is given below.

The actual Baud Rate may be found from:

$$BR = Fc/(16 \times N \times P)$$

Baud Clock Generation (Continued)

Where:

BR is the Baud Rate

Fc is the CKI frequency

N is the Baud Rate Divisor (Table IV).

P is the Prescaler Divide Factor selected by the value in the Prescaler Select Register (Table III)

Note: In the Synchronous Mode, the divisor 16 is replaced by two.

Example:

Asynchronous Mode:

Crystal Frequency = 5 MHz

Desired baud rate = 9600

Using the above equation N × P can be calculated first.

$$N \times P = (5 \times 106)/(16 \times 9600) = 32.552$$

Now 32.552 is divided by each Prescaler Factor (Table III) to obtain a value closest to an integer. This factor happens to be 6.5 (P = 6.5).

$$N = 32.552/6.5 = 5.008 \ (N = 5)$$

The programmed value (from Table III) should be 4 (N − 1).

Using the above values calculated for N and P:

$$BR = (5 \times 106)/(16 \times 5 \times 6.5) = 9615.384$$

$$\% \text{ error} = (9615.385 - 9600)/9600 = 0.16$$

Effect of HALT/IDLE

The UART logic is reinitialized when either the HALT or IDLE modes are entered. This reinitialization sets the TBMT flag and resets all read only bits in the UART control and status registers. Read/Write bits remain unchanged. The Transmit Buffer (TBUF) is not affected, but the Transmit Shift register (TSFT) bits are set to one. The receiver registers RBUF and RSFT are not affected.

The device will exit from the HALT/IDLE modes when the Start bit of a character is detected at the RDX (L3) pin. This feature is obtained by using the Multi-Input Wakeup scheme provided on the device.

Before entering the HALT or IDLE modes the user program must select the Wakeup source to be on the RDX pin. This selection is done by setting bit 3 of WKEN (Wakeup Enable) register. The Wakeup trigger condition is then selected to be high to low transition. This is done via the WKEDG register (Bit 3 is one.)

If the device is halted and crystal oscillator is used, the Wakeup signal will not start the chip running immediately because of the finite start up time requirement of the crystal oscillator. The idle timer (T0) generates a fixed (256 t_C) delay to ensure that the oscillator has indeed stabilized before allowing the device to execute code. The user has to consider this delay when data transfer is expected immediately after exiting the HALT mode.

Diagnostic

Bits CHARL0 and CHARL1 in the ENU register provide a loopback feature for diagnostic testing of the UART. When these bits are set to one, the following occur: The receiver input pin (RDX) is internally connected to the transmitter output pin (TDX); the output of the Transmitter Shift Register is "looped back" into the Receive Shift Register input. In this mode, data that is transmitted is immediately received. This feature allows the processor to verify the transmit and receive data paths of the UART.

Note that the framing format for this mode is the nine bit format; one Start bit, nine data bits, and 7/8, one or two Stop bits. Parity is not generated or verified in this mode.

Attention Mode

The UART Receiver section supports an alternate mode of operation, referred to as ATTENTION Mode. This mode of operation is selected by the ATTN bit in the ENUR register. The data format for transmission must also be selected as having nine Data bits and either 7/8, one or two Stop bits.

The ATTENTION mode of operation is intended for use in networking the device with other processors. Typically in such environments the messages consists of device addresses, indicating which of several destinations should receive them, and the actual data. This Mode supports a scheme in which addresses are flagged by having the ninth bit of the data field set to a 1. If the ninth bit is reset to a zero the byte is a Data byte.

While in ATTENTION mode, the UART monitors the communication flow, but ignores all characters until an address character is received. Upon receiving an address character, the UART signals that the character is ready by setting the RBFL flag, which in turn interrupts the processor if UART Receiver interrupts are enabled. The ATTN bit is also cleared automatically at this point, so that data characters as well as address characters are recognized. Software examines the contents of the RBUF and responds by deciding either to accept the subsequent data stream (by leaving the ATTN bit reset) or to wait until the next address character is seen (by setting the ATTN bit again).

Operation of the UART Transmitter is not affected by selection of this Mode. The value of the ninth bit to be transmitted is programmed by setting XBIT9 appropriately. The value of the ninth bit received is obtained by reading RBIT9. Since this bit is located in ENUR register where the error flags reside, a bit operation on it will reset the error flags.

Comparators

The device contains two differential comparators, each with a pair of inputs (positive and negative) and an output. Ports I1–I3 and I4–I6 are used for the comparators. The following is the Port I assignment:

- I1 Comparator1 negative input
- I2 Comparator1 positive input
- I3 Comparator1 output
- I4 Comparator2 negative input
- I5 Comparator2 positive input
- I6 Comparator2 output

A Comparator Select Register (CMPSL) is used to enable the comparators, read the outputs of the comparators internally, and enable the outputs of the comparators to the pins. Two control bits (enable and output enable) and one result bit are associated with each comparator. The comparator result bits (CMP1RD and CMP2RD) are read only bits which will read as zero if the associated comparator is not enabled. The Comparator Select Register is cleared with reset, resulting in the comparators being disabled. The comparators should also be disabled before entering either the HALT or IDLE modes in order to save power. The configuration of the CMPSL register is as follows:

Comparators (Continued)

CMPSL REGISTER (ADDRESS X'00B7)

The CMPSL register contains the following bits:

CMP1EN Enable comparator 1

CMP1RD Comparator 1 result (this is a read only bit, which will read as 0 if the comparator is not enabled)

CMP10E Selects pin I3 as comparator 1 output provided that CMPIEN is set to enable the comparator

CMP2EN Enable comparator 2

CMP2RD Comparator 2 result (this is a read only bit, which will read as 0 if the comparator is not enabled)

CMP20E Selects pin I6 as comparator 2 output provided that CMP2EN is set to enable the comparator

Unused	CMP20E	CMP2RD	CMP2EN	CMP10E	CMP1RD	CMP1EN	Unused
Bit 7							Bit 0

Note that the two unused bits of CMPSL may be used as software flags.

Comparator outputs have the same spec as Ports L and G except that the rise and fall times are symmetrical.

Interrupts

The device supports a vectored interrupt scheme. It supports a total of fourteen interrupt sources. The following table lists all the possible interrupt sources, their arbitration ranking and the memory locations reserved for the interrupt vector for each source.

Two bytes of program memory space are reserved for each interrupt source. All interrupt sources except the software interrupt are maskable. Each of the maskable interrupts have an Enable bit and one or more Pending bit. A maskable interrupt is active if its associated enable and pending bits are set. If GIE = 1 and an interrupt is active, then the processor will be interrupted as soon as it is ready to start executing an instruction except if the above conditions happen during the Software Trap service routine. This exception is described in the Software Trap sub-section.

The interruption process is accomplished with the INTR instruction (opcode 00), which is jammed inside the Instruction Register and replaces the opcode about to be executed. The following steps are performed for every interrupt:

1. The GIE (Global Interrupt Enable) bit is reset.
2. The address of the instruction about to be executed is pushed into the stack.
3. The PC (Program Counter) branches to address 00FF.

This procedure takes 7 t_C cycles to execute.

Arbitration Ranking	Source	Description	Vector* Address Hi-Low Byte
(1) Highest	Software	INTR Instruction	0yFE–0yFF
(2)	Reserved		0yFC–0yFD
(3)	External	G0	0yFA–0yFB
(4)	Timer T0	Underflow	0yF8–0yF9
(5)	Timer T1	T1A/Underflow	0yF6–0yF7
(6)	Timer T1	T1B	0yF4–0yF5
(7)	MICROWIRE/PLUS	BUSY Low	0yF2–0yF3
(8)	Reserved		0yF0–0yF1
(9)	UART	Receive	0yEE–0yEF
(10)	UART	Transmit	0yEC–0yED
(11)	Timer T2	T2A/Underflow	0yEA–0yEB
(12)	Timer T2	T2B	0yE8–0yE9
(13)	Timer T3	T3A/Underflow	0yE6–0yE7
(14)	Timer T3	T3B	0yE4–0yE5
(15)	Port L/Wakeup	Port L Edge	0yE2–0yE3
(16) Lowest	Default VIS	Reserved	0yE0–0yE1

*y is a variable which represents the VIS block. VIS and the vector table must be located in the same 256-byte block except if VIS is located at the last address of a block. In this case, the table must be in the next block.

Interrupts (Continued)

At this time, since GIE = 0, other maskable interrupts are disabled. The user is now free to do whatever context switching is required by saving the context of the machine in the stack with PUSH instructions. The user would then program a VIS (Vector Interrupt Select) instruction in order to branch to the interrupt service routine of the highest priority interrupt enabled and pending at the time of the VIS. Note that this is not necessarily the interrupt that caused the branch to address location 00FF Hex prior to the context switching.

Thus, if an interrupt with a higher rank than the one which caused the interruption becomes active before the decision of which interrupt to service is made by the VIS, then the interrupt with the higher rank will override any lower ones and will be acknowledged. The lower priority interrupt(s) are still pending, however, and will cause another interrupt immediately following the completion of the interrupt service routine associated with the higher priority interrupt just serviced. This lower priority interrupt will occur immediately following the RETI (Return from Interrupt) instruction at the end of the interrupt service routine just completed.

Inside the interrupt service routine, the associated pending bit has to be cleared by software. The RETI (Return from Interrupt) instruction at the end of the interrupt service routine will set the GIE (Global Interrupt Enable) bit, allowing the processor to be interrupted again if another interrupt is active and pending.

The VIS instruction looks at all the active interrupts at the time it is executed and performs an indirect jump to the beginning of the service routine of the one with the highest rank.

The addresses of the different interrupt service routines, called vectors, are chosen by the user and stored in ROM in a table starting at 01E0 (assuming that VIS is located between 00FF and 01DF). The vectors are 15-bit wide and therefore occupy 2 ROM locations.

VIS and the vector table must be located in the same 256-byte block (0y00 to 0yFF) except if VIS is located at the last address of a block. In this case, the table must be in the next block. The vector table cannot be inserted in the first 256-byte block (y ≠ 0).

The vector of the maskable interrupt with the lowest rank is located at 0yE0 (Hi-Order byte) and 0yE1 (Lo-Order byte) and so forth in increasing rank number. The vector of the maskable interrupt with the highest rank is located at 0yFA (Hi-Order byte) and 0yFB (Lo-Order byte).

The Software Trap has the highest rank and its vector is located at 0yFE and 0yFF.

If, by accident, a VIS gets executed and no interrupt is active, then the PC (Program Counter) will branch to a vector located at 0yE0–0yE1.

WARNING

A Default VIS interrupt handler routine must be present. As a minimum, this handler should confirm that the GIE bit is cleared (this indicates that the interrupt sequence has been taken), take care of any required housekeeping, restore context and return. Some sort of Warm Restart procedure should be implemented. These events can occur without any error on the part of the system designer or programmer.

Note: There is always the possibility of an interrupt occurring during an instruction which is attempting to reset the GIE bit or any other interrupt enable bit. If this occurs when a single cycle instruction is being used to reset the interrupt enable bit, the interrupt enable bit will be reset but an interrupt may still occur. This is because interrupt processing is started at the same time as the interrupt bit is being reset. To avoid this scenario, the user should always use a two, three, or four cycle instruction to reset interrupt enable bits.

Figure 16 shows the Interrupt block diagram.

SOFTWARE TRAP

The Software Trap (ST) is a special kind of non-maskable interrupt which occurs when the INTR instruction (used to acknowledge interrupts) is fetched from ROM and placed inside the instruction register. This may happen when the PC is pointing beyond the available ROM address space or when the stack is over-popped.

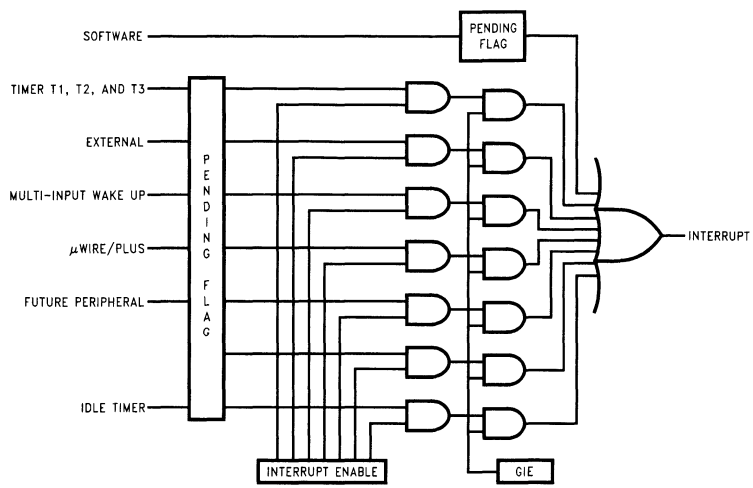


FIGURE 16. Interrupt Block Diagram

Interrupts (Continued)

When an ST occurs, the user can re-initialize the stack pointer and do a recovery procedure (similar to reset, but not necessarily containing all of the same initialization procedures) before restarting.

The occurrence of an ST is latched into the ST pending bit. The GIE bit is not affected and the ST pending bit **(not accessible by the user)** is used to inhibit other interrupts and to direct the program to the ST service routine with the VIS instruction. The RPND instruction is used to clear the software interrupt pending bit. This pending bit is also cleared on reset.

The ST has the highest rank among all interrupts.

Nothing (except another ST) can interrupt an ST being serviced.

WATCHDOG

The device contains a WATCHDOG and clock monitor. The WATCHDOG is designed to detect the user program getting stuck in infinite loops resulting in loss of program control or "runaway" programs. The Clock Monitor is used to detect the absence of a clock or a very slow clock below a specified rate on the CKI pin.

The WATCHDOG consists of two independent logic blocks: WD UPPER and WD LOWER. WD UPPER establishes the upper limit on the service window and WD LOWER defines the lower limit of the service window.

Servicing the WATCHDOG consists of writing a specific value to a WATCHDOG Service Register named WDSVR which is memory mapped in the RAM. This value is composed of three fields, consisting of a 2-bit Window Select, a 5-bit Key Data field, and the 1-bit Clock Monitor Select field. Table II shows the WDSVR register.

TABLE V. WATCHDOG Service Register (WDSVR)

Window Select		Key Data					Clock Monitor
X	X	0	1	1	0	0	Y
7	6	5	4	3	2	1	0

The lower limit of the service window is fixed at 2048 instruction cycles. Bits 7 and 6 of the WDSVR register allow the user to pick an upper limit of the service window.

Table III shows the four possible combinations of lower and upper limits for the WATCHDOG service window. This flexibility in choosing the WATCHDOG service window prevents any undue burden on the user software.

Bits 5, 4, 3, 2 and 1 of the WDSVR register represent the 5-bit Key Data field. The key data is fixed at 01100. Bit 0 of the WDSVR Register is the Clock Monitor Select bit.

TABLE VI. WATCHDOG Service Window Select

WDSVR Bit 7	WDSVR Bit 6	Service Window (Lower-Upper Limits)
0	0	2k–8k t_C Cycles
0	1	2k–16k t_C Cycles
1	0	2k–32k t_C Cycles
1	1	2k–64k t_C Cycles

Clock Monitor

The Clock Monitor aboard the device can be selected or deselected under program control. The Clock Monitor is guaranteed not to reject the clock if the instruction cycle clock ($1/t_C$) is greater or equal to 10 kHz. This equates to a clock input rate on CKI of greater or equal to 100 kHz.

WATCHDOG Operation

The WATCHDOG and Clock Monitor are disabled during reset. The device comes out of reset with the WATCHDOG armed, the WATCHDOG Window Select bits (bits 6, 7 of the WDSVR Register) set, and the Clock Monitor bit (bit 0 of the WDSVR Register) enabled. Thus, a Clock Monitor error will occur after coming out of reset, if the instruction cycle clock frequency has not reached a minimum specified value, including the case where the oscillator fails to start.

The WDSVR register can be written to only once after reset and the key data (bits 5 through 1 of the WDSVR Register) must match to be a valid write. This write to the WDSVR register involves two irrevocable choices: (i) the selection of the WATCHDOG service window (ii) enabling or disabling of the Clock Monitor. Hence, the first write to WDSVR Register involves selecting or deselecting the Clock Monitor, select the WATCHDOG service window and match the WATCHDOG key data. Subsequent writes to the WDSVR register will compare the value being written by the user to the WATCHDOG service window value and the key data (bits 7 through 1) in the WDSVR Register. Table IV shows the sequence of events that can occur.

The user must service the WATCHDOG at least once before the upper limit of the service window expires. The WATCHDOG may not be serviced more than once in every lower limit of the service window. The user may service the WATCHDOG as many times as wished in the time period between the lower and upper limits of the service window. The first write to the WDSVR Register is also counted as a WATCHDOG service.

The WATCHDOG has an output pin associated with it. This is the WDOUT pin, on pin 1 of the port G. WDOUT is active low. The WDOUT pin is in the high impedance state in the inactive state. Upon triggering the WATCHDOG, the logic will pull the WDOUT (G1) pin low for an additional 16 t_C–32 t_C cycles after the signal level on WDOUT pin goes below the lower Schmitt trigger threshold. After this delay, the device will stop forcing the WDOUT output low.

The WATCHDOG service window will restart when the WDOUT pin goes high. It is recommended that the user tie the WDOUT pin back to V_{CC} through a resistor in order to pull WDOUT high.

A WATCHDOG service while the WDOUT signal is active will be ignored. The state of the WDOUT pin is not guaranteed on reset, but if it powers up low then the WATCHDOG will time out and WDOUT will enter high impedance state.

The Clock Monitor forces the G1 pin low upon detecting a clock frequency error. The Clock Monitor error will continue until the clock frequency has reached the minimum specified value, after which the G1 output will enter the high impedance TRI-STATE mode following 16 t_C–32 t_C clock cycles. The Clock Monitor generates a continual Clock Monitor error if the oscillator fails to start, or fails to reach the minimum specified frequency. The specification for the Clock Monitor is as follows:

$1/t_C > 10$ kHz—No clock rejection.

$1/t_C < 10$ Hz—Guaranteed clock rejection.

WATCHDOG Operation (Continued)

WATCHDOG AND CLOCK MONITOR SUMMARY

The following salient points regarding the WATCHDOG and CLOCK MONITOR should be noted:

- Both the WATCHDOG and CLOCK MONITOR detector circuits are inhibited during RESET.
- Following RESET, the WATCHDOG and CLOCK MONITOR are both enabled, with the WATCHDOG having he maximum service window selected.
- The WATCHDOG service window and CLOCK MONITOR enable/disable option can only be changed once, during the initial WATCHDOG service following RESET.
- The initial WATCHDOG service must match the key data value in the WATCHDOG Service register WDSVR in order to avoid a WATCHDOG error.
- Subsequent WATCHDOG services must match all three data fields in WDSVR in order to avoid WATCHDOG errors.
- The correct key data value cannot be read from the WATCHDOG Service register WDSVR. Any attempt to read this key data value of 01100 from WDSVR will read as key data value of all 0's.
- The WATCHDOG detector circuit is inhibited during both the HALT and IDLE modes.
- The CLOCK MONITOR detector circuit is active during both the HALT and IDLE modes. Consequently, the device inadvertently entering the HALT mode will be detected as a CLOCK MONITOR error (provided that the CLOCK MONITOR enable option has been selected by the program).
- With the single-pin R/C oscillator mask option selected and the CLKDLY bit reset, the WATCHDOG service window will resume following HALT mode from where it left off before entering the HALT mode.
- With the crystal oscillator mask option selected, or with the single-pin R/C oscillator mask option selected and the CLKDLY bit set, the WATCHDOG service window will be set to its selected value from WDSVR following HALT. Consequently, the WATCHDOG should not be serviced for at least 2048 instruction cycles following HALT, but must be serviced within the selected window to avoid a WATCHDOG error.
- The IDLE timer T0 is not initialized with RESET.
- The user can sync in to the IDLE counter cycle with an IDLE counter (T0) interrupt or by monitoring the T0PND flag. The T0PND flag is set whenever the thirteenth bit of the IDLE counter toggles (every 4096 instruction cycles). The user is responsible for resetting the T0PND flag.
- A hardware WATCHDOG service occurs just as the device exits the IDLE mode. Consequently, the WATCHDOG should not be serviced for at least 2048 instruction cycles following IDLE, but must be serviced within the selected window to avoid a WATCHDOG error.
- Following RESET, the initial WATCHDOG service (where the service window and the CLOCK MONITOR enable/disable must be selected) may be programmed anywhere within the maximum service window (65,536 instruction cycles) initialized by RESET. Note that this initial WATCHDOG service may be programmed within the initial 2048 instruction cycles without causing a WATCHDOG error.

Detection of Illegal Conditions

The device can detect various illegal conditions resulting from coding errors, transient noise, power supply voltage drops, runaway programs, etc.

Reading of undefined ROM gets zeros. The opcode for software interrupt is zero. If the program fetches instructions from undefined ROM, this will force a software interrupt, thus signaling that an illegal condition has occurred.

The subroutine stack grows down for each call (jump to subroutine), interrupt, or PUSH, and grows up for each return or POP. The stack pointer is initialized to RAM location 06F Hex during reset. Consequently, if there are more returns than calls, the stack pointer will point to addresses 070 and 071 Hex (which are undefined RAM). Undefined RAM from addresses 070 to 07F (Segment 0), and all other segments (i.e., Segments 4 . . . etc.) is read as all 1's, which in turn will cause the program to return to address 7FFF Hex. This is an undefined ROM location and the instruction fetched (all 0's) from this location will generate a software interrupt signaling an illegal condition.

Thus, the chip can detect the following illegal conditions:

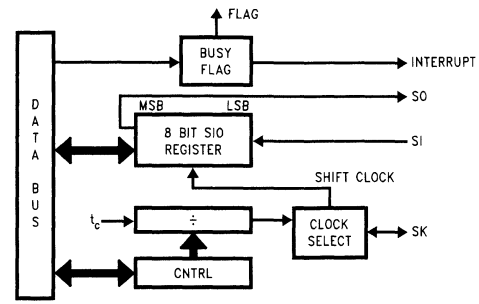
a. Executing from undefined ROM

b. Over "POP"ing the stack by having more returns than calls.

When the software interrupt occurs, the user can re-initialize the stack pointer and do a recovery procedure before restarting (this recovery program is probably similar to that following reset, but might not contain the same program initialization procedures). The recovery program should reset the software interrupt pending bit using the RPND instruction.

MICROWIRE/PLUS

MICROWIRE/PLUS is a serial synchronous communications interface. The MICROWIRE/PLUS capability enables the device to interface with any of National Semiconductor's MICROWIRE peripherals (i.e. A/D converters, display drivers, E^2PROMs etc.) and with other microcontrollers which support the MICROWIRE interface. It consists of an 8-bit serial shift register (SIO) with serial data input (SI), serial data output (SO) and serial shift clock (SK). *Figure 17* shows a block diagram of the MICROWIRE/PLUS logic.



TL/DD12864–19

FIGURE 17. MICROWIRE/PLUS Block Diagram

The shift clock can be selected from either an internal source or an external source. Operating the MICROWIRE/PLUS arrangement with the internal clock source is called the Master mode of operation. Similarly, operating the MICROWIRE/PLUS arrangement with an external shift clock is called the Slave mode of operation.

The CNTRL register is used to configure and control the MICROWIRE/PLUS mode. To use the MICROWIRE/PLUS, the MSEL bit in the CNTRL register is set to one. In the master mode, the SK clock rate is selected by the two bits, SL0 and SL1, in the CNTRL register. Table VIII details the different clock rates that may be selected.

TABLE VII. WATCHDOG Service Actions

Key Data	Window Data	Clock Monitor	Action
Match	Match	Match	Valid Service: Restart Service Window
Don't Care	Mismatch	Don't Care	Error: Generate WATCHDOG Output
Mismatch	Don't Care	Don't Care	Error: Generate WATCHDOG Output
Don't Care	Don't Care	Mismatch	Error: Generate WATCHDOG Output

TABLE VIII. MICROWIRE/PLUS Master Mode Clock Select

SL1	SL0	SK
0	0	$2 \times t_c$
0	1	$4 \times t_c$
1	x	$8 \times t_c$

Where t_c is the instruction cycle clock

MICROWIRE/PLUS (Continued)

MICROWIRE/PLUS OPERATION

Setting the BUSY bit in the PSW register causes the MICROWIRE/PLUS to start shifting the data. It gets reset when eight data bits have been shifted. The user may reset the BUSY bit by software to allow less than 8 bits to shift. If enabled, an interrupt is generated when eight data bits have been shifted. The device may enter the MICROWIRE/PLUS mode either as a Master or as a Slave. *Figure 18* shows how two microcontroller devices and several peripherals may be interconnected using the MICROWIRE/PLUS arrangements.

Warning:

The SIO register should only be loaded when the SK clock is low. Loading the SIO register while the SK clock is high will result in undefined data in the SIO register. SK clock is normally low when not shifting.

Setting the BUSY flag when the input SK clock is high in the MICROWIRE/PLUS slave mode may cause the current SK clock for the SIO shift register to be narrow. For safety, the BUSY flag should only be set when the input SK clock is low.

MICROWIRE/PLUS Master Mode Operation

In the MICROWIRE/PLUS Master mode of operation the shift clock (SK) is generated internally by the device. The MICROWIRE Master always initiates all data exchanges. The MSEL bit in the CNTRL register must be set to enable the SO and SK functions onto the G Port. The SO and SK pins must also be selected as outputs by setting appropriate bits in the Port G configuration register. Table IX summarizes the bit settings required for Master mode of operation.

MICROWIRE/PLUS Slave Mode Operation

In the MICROWIRE/PLUS Slave mode of operation the SK clock is generated by an external source. Setting the MSEL bit in the CNTRL register enables the SO and SK functions onto the G Port. The SK pin must be selected as an input and the SO pin is selected as an output pin by setting and resetting the appropriate bits in the Port G configuration register. Table VI summarizes the settings required to enter the Slave mode of operation.

The user must set the BUSY flag immediately upon entering the Slave mode. This will ensure that all data bits sent by the Master will be shifted properly. After eight clock pulses the BUSY flag will be cleared and the sequence may be repeated.

Alternate SK Phase Operation

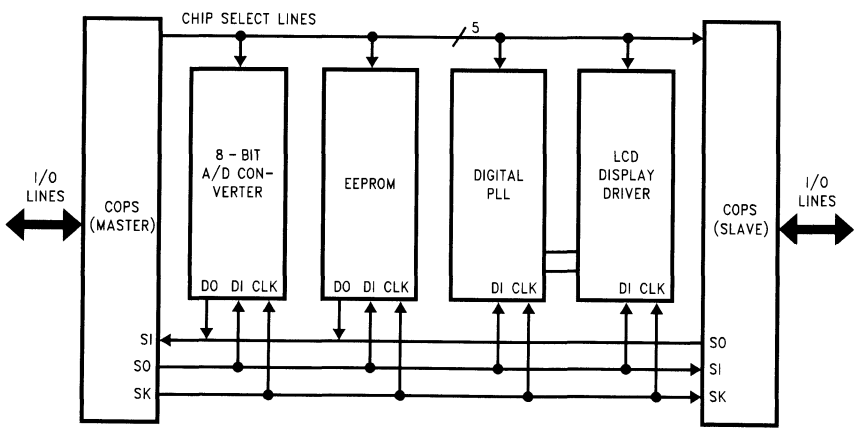
The device allows either the normal SK clock or an alternate phase SK clock to shift data in and out of the SIO register. In both the modes the SK is normally low. In the normal mode data is shifted in on the rising edge of the SK clock and the data is shifted out on the falling edge of the SK clock. The SIO register is shifted on each falling edge of the SK clock. In the alternate SK phase operation, data is shifted in on the falling edge of the SK clock and shifted out on the rising edge of the SK clock.

A control flag, SKSEL, allows either the normal SK clock or the alternate SK clock to be selected. Resetting SKSEL causes the MICROWIRE/PLUS logic to be clocked from the normal SK signal. Setting the SKSEL flag selects the alternate SK clock. The SKSEL is mapped into the G6 configuration bit. The SKSEL flag will power up in the reset condition, selecting the normal SK signal.

TABLE IX

This table assumes that the control flag MSEL is set.

G4 (SO) Config. Bit	G5 (SK) Config. Bit	G4 Fun.	G5 Fun.	Operation
1	1	SO	Int. SK	MICROWIRE/PLUS Master
0	1	TRI-STATE	Int. SK	MICROWIRE/PLUS Master
1	0	SO	Ext. SK	MICROWIRE/PLUS Slave
0	0	TRI-STATE	Ext. SK	MICROWIRE/PLUS Slave



TL/DD12864-20

FIGURE 18. MICROWIRE/PLUS Application

Memory Map

All RAM, ports and registers (except A and PC) are mapped into data memory address space.

Address S/ADD REG	Contents
0000 to 006F	On-Chip RAM bytes (112 bytes)
0070 to 007F	Unused RAM Address Space (Reads As All Ones)
xx80 to xxAF	Unused RAM Address Space (Reads Undefined Data)
xxB0	Timer T3 Lower Byte
XXB1	Timer T3 Upper Byte
xxB2	Timer T3 Autoload Register T3RA Lower Byte
xxB3	Timer T3 Autoload Register T3RA Upper Byte
xxB4	Timer T3 Autoload Register T3RB Lower Byte
xxB5	Timer T3 Autoload Register T3RB Upper Byte
xxB6	Timer T3 Control Register
xxB7	Comparator Select Register (CMPSL)
xxB8	UART Transmit Buffer (TBUF)
xxB9	UART Receive Buffer (RBUF)
xxBA	UART Control and Status Register (ENU)
xxBB	UART Receive Control and Status Register (ENUR)
xxBC	UART Interrupt and Clock Source Register (ENUI)
xxBD	UART Baud Register (BAUD)
xxBE	UART Prescale Select Register (PSR)
xxBF	Reserved for UART
xxC0	Timer T2 Lower Byte
xxC1	Timer T2 Upper Byte
xxC2	Timer T2 Autoload Register T2RA Lower Byte
xxC3	Timer T2 Autoload Register T2RA Upper Byte
xxC4	Timer T2 Autoload Register T2RB Lower Byte
xxC5	Timer T2 Autoload Register T2RB Upper Byte
xxC6	Timer T2 Control Register
xxC7	WATCHDOG Service Register (Reg:WDSVR)
xxC8	MIWU Edge Select Register (Reg:WKEDG)
xxC9	MIWU Enable Register (Reg:WKEN)
xxCA	MIWU Pending Register (Reg:WKPND)
xxCB	Reserved
xxCC	Reserved
xxCD to xxCF	Reserved

Address S/ADD REG	Contents
xxD0	Port L Data Register
xxD1	Port L Configuration Register
xxD2	Port L Input Pins (Read Only)
xxD3	Reserved for Port L
xxD4	Port G Data Register
xxD5	Port G Configuration Register
xxD6	Port G Input Pins (Read Only)
xxD7	Port I Input Pins (Read Only)
xxD8	Port C Data Register
xxD9	Port C Configuration Register
xxDA	Port C Input Pins (Read Only)
xxDB	Reserved for Port C
xxDC	Port D
xxDD to xxDF	Reserved for Port D
xxE0 to xxE5	Reserved for EE Control Registers
xxE6	Timer T1 Autoload Register T1RB Lower Byte
xxE7	Timer T1 Autoload Register T1RB Upper Byte
xxE8	ICNTRL Register
xxE9	MICROWIRE/PLUS Shift Register
xxEA	Timer T1 Lower Byte
xxEB	Timer T1 Upper Byte
xxEC	Timer T1 Autoload Register T1RA Lower Byte
xxED	Timer T1 Autoload Register T1RA Upper Byte
xxEE	CNTRL Control Register
xxEF	PSW Register
xxF0 to FB	On-Chip RAM Mapped as Registers
xxFC	X Register
xxFD	SP Register
xxFE	B Register
xxFF	S Register
0100–017F	On-Chip 128 RAM Bytes
0200–027F	On-Chip 128 RAM Bytes
0300–037F	On-Chip 128 RAM Bytes
0400–047F	On-Chip 128 RAM Bytes
0500–057F	On-Chip 128 RAM Bytes
0600–067F	On-Chip 128 RAM Bytes
0700–077F	On-Chip 128 RAM Bytes
0800–083F	On-Chip 64 RAM Bytes

Reading memory locations 0070H–007FH (Segment 0) will return all ones. Reading unused memory locations 0080H–00AFH (Segment 0) will return undefined data. Reading memory locations from other Segments (i.e., Segment 8 (084H–087FH), Segment 9, ... etc.) will return all ones.

Addressing Modes

There are ten addressing modes, six for operand addressing and four for transfer of control.

OPERAND ADDRESSING MODES

Register Indirect

This is the "normal" addressing mode. The operand is the data memory addressed by the B pointer or X pointer.

Register Indirect (with auto post increment or decrement of pointer)

This addressing mode is used with the LD and X instructions. The operand is the data memory addressed by the B pointer or X pointer. This is a register indirect mode that automatically post increments or decrements the B or X register after executing the instruction.

Direct

The instruction contains an 8-bit address field that directly points to the data memory for the operand.

Immediate

The instruction contains an 8-bit immediate field as the operand.

Short Immediate

This addressing mode is used with the Load B Immediate instruction. The instruction contains a 4-bit immediate field as the operand.

Indirect

This addressing mode is used with the LAID instruction. The contents of the accumulator are used as a partial address (lower 8 bits of PC) for accessing a data operand from the program memory.

TRANSFER OF CONTROL ADDRESSING MODES

Relative

This mode is used for the JP instruction, with the instruction field being added to the program counter to get the new program location. JP has a range from −31 to +32 to allow a 1-byte relative jump (JP + 1 is implemented by a NOP instruction). There are no "pages" when using JP, since all 15 bits of PC are used.

Absolute

This mode is used with the JMP and JSR instructions, with the instruction field of 12 bits replacing the lower 12 bits of the program counter (PC). This allows jumping to any location in the current 4k program memory segment.

Absolute Long

This mode is used with the JMPL and JSRL instructions, with the instruction field of 15 bits replacing the entire 15 bits of the program counter (PC). This allows jumping to any location up to 32k in the program memory space.

Indirect

This mode is used with the JID instruction. The contents of the accumulator are used as a partial address (lower 8 bits of PC) for accessing a location in the program memory. The contents of this program memory location serve as a partial address (lower 8 bits of PC) for the jump to the next instruction.

Note: The VIS is a special case of the Indirect Transfer of Control addressing mode, where the double byte vector associated with the interrupt is transferred from adjacent addresses in the program memory into the program counter (PC) in order to jump to the associated interrupt service routine.

Instruction Set

Register and Symbol Definition

Registers	
A	8-Bit Accumulator Register
B	8-Bit Address Register
X	8-Bit Address Register
S	8-Bit Segment Register
SP	8-Bit Stack Pointer Register
PC	15-Bit Program Counter Register
PU	Upper 7 Bits of PC
PL	Lower 8 Bits of PC
C	1 Bit of PSW Register for Carry
HC	1 Bit of PSW Register for Half Carry
GIE	1 Bit of PSW Register for Global Interrupt Enable
VU	Interrupt Vector Upper Byte
VL	Interrupt Vector Lower Byte

Symbols	
[B]	Memory Indirectly Addressed by B Register
[X]	Memory Indirectly Addressed by X Register
MD	Direct Addressed Memory
Mem	Direct Addressed Memory or [B]
Meml	Direct Addressed Memory or [B] or Immediate Data
Imm	8-Bit Immediate Data
Reg	Register Memory: Addresses F0 to FF (Includes B, X and SP)
Bit	Bit Number (0 to 7)
←	Loaded with
↔	Exchanged with

Instruction Set (Continued)

INSTRUCTION SET

ADD	A,Meml	ADD	A ← A + Meml
ADC	A,Meml	ADD with Carry	A ← A + Meml + C, C ← Carry HC ← Half Carry
SUBC	A,Meml	Subtract with Carry	A ← A − $\overline{\text{Meml}}$ + C, C ← Carry HC ← Half Carry
AND	A,Meml	Logical AND	A ← A and Meml
ANDSZ	A,Imm	Logical AND Immed., Skip if Zero	Skip next if (A and Imm) = 0
OR	A,Meml	Logical OR	A ← A or Meml
XOR	A,Meml	Logical EXclusive OR	A ← A xor Meml
IFEQ	MD,Imm	IF EQual	Compare MD and Imm, Do next if MD = Imm
IFEQ	A,Meml	IF EQual	Compare A and Meml, Do next if A = Meml
IFNE	A,Meml	IF Not Equal	Compare A and Meml, Do next if A ≠ Meml
IFGT	A,Meml	IF Greater Than	Compare A and Meml, Do next if A > Meml
IFBNE	#	If B Not Equal	Do next if lower 4 bits of B ≠ Imm
DRSZ	Reg	Decrement Reg., Skip if Zero	Reg ← Reg − 1, Skip if Reg = 0
SBIT	#,Mem	Set BIT	1 to bit, Mem (bit = 0 to 7 immediate)
RBIT	#,Mem	Reset BIT	0 to bit, Mem
IFBIT	#,Mem	IF BIT	If bit in A or Mem is true do next instruction
RPND		Reset PeNDing Flag	Reset Software Interrupt Pending Flag
X	A,Mem	EXchange A with Memory	A ⟷ Mem
X	A,[X]	EXchange A with Memory [X]	A ⟷ [X]
LD	A,Meml	LoaD A with Memory	A ← Meml
LD	A,[X]	LoaD A with Memory [X]	A ← [X]
LD	B,Imm	LoaD B with Immed.	B ← Imm
LD	Mem,Imm	LoaD Memory Immed	Mem ← Imm
LD	Reg,Imm	LoaD Register Memory Immed.	Reg ← Imm
X	A, [B ±]	EXchange A with Memory [B]	A ⟷ [B], (B ← B ±1)
X	A, [X ±]	EXchange A with Memory [X]	A ⟷ [X], (X ← X ±1)
LD	A, [B±]	LoaD A with Memory [B]	A ← [B], (B ← B ±1)
LD	A, [X±]	LoaD A with Memory [X]	A ← [X], (X ← X±1)
LD	[B±],Imm	LoaD Memory [B] Immed.	[B] ← Imm, (B ← B±1)
CLR	A	CLeaR A	A ← 0
INC	A	INCrement A	A ← A + 1
DEC	A	DECrement A	A ← A − 1
LAID		Load A InDirect from ROM	A ← ROM (PU,A)
DCOR	A	Decimal CORrect A	A ← BCD correction of A (follows ADC, SUBC)
RRC	A	Rotate A Right thru C	C → A7 → . . . → A0 → C
RLC	A	Rotate A Left thru C	C ← A7 ← . . . ← A0 ← C
SWAP	A	SWAP nibbles of A	A7 . . . A4 ⟷ A3 . . . A0
SC		Set C	C ← 1, HC ← 1
RC		Reset C	C ← 0, HC ← 0
IFC		IF C	IF C is true, do next instruction
IFNC		IF Not C	If C is not true, do next instruction
POP	A	POP the stack into A	SP ← SP + 1, A ← [SP]
PUSH	A	PUSH A onto the stack	[SP] ← A, SP ← SP − 1
VIS		Vector to Interrupt Service Routine	PU ← [VU], PL ← [VL]
JMPL	Addr.	Jump absolute Long	PC ← ii (ii = 15 bits, 0 to 32k)
JMP	Addr.	Jump absolute	PC9 . . . 0 ← i (i = 12 bits)
JP	Disp.	Jump relative short	PC ← PC + r (r is −31 to +32, except 1)
JSRL	Addr.	Jump SubRoutine Long	[SP] ← PL, [SP−1] ← PU,SP−2, PC ← ii
JSR	Addr	Jump SubRoutine	[SP] ← PL, [SP−1] ← PU,SP−2, PC9 . . . 0 ← i
JID		Jump InDirect	PL ← ROM (PU,A)
RET		RETurn from subroutine	SP + 2, PL ← [SP], PU ← [SP−1]
RETSK		RETurn and SKip	SP + 2, PL ← [SP],PU ← [SP−1], skip next instruction
RETI		RETurn from Interrupt	SP + 2, PL ← [SP],PU ← [SP−1],GIE ← 1
INTR		Generate an Interrupt	[SP] ← PL, [SP−1] ← PU, SP−2, PC ← 0FF
NOP		No OPeration	PC ← PC + 1

Instruction Execution Time

Most instructions are single byte (with immediate addressing mode instructions taking two bytes).

Most single byte instructions take one cycle time to execute.

Skipped instructions require x number of cycles to be skipped, where x equals the number of bytes in the skipped instruction opcode.

See the BYTES and CYCLES per INSTRUCTION table for details.

Bytes and Cycles per Instruction

The following table shows the number of bytes and cycles for each instruction in the format of byte/cycle.

Arithmetic and Logic Instructions

	[B]	Direct	Immed.
ADD	1/1	3/4	2/2
ADC	1/1	3/4	2/2
SUBC	1/1	3/4	2/2
AND	1/1	3/4	2/2
OR	1/1	3/4	2/2
XOR	1/1	3/4	2/2
IFEQ	1/1	3/4	2/2
IFNE	1/1	3/4	2/2
IFGT	1/1	3/4	2/2
IFBNE	1/1		
DRSZ		1/3	
SBIT	1/1	3/4	
RBIT	1/1	3/4	
IFBIT	1/1	3/4	

RPND	1/1

Instructions Using A & C

CLRA	1/1
INCA	1/1
DECA	1/1
LAID	1/3
DCOR	1/1
RRCA	1/1
RLCA	1/1
SWAPA	1/1
SC	1/1
RC	1/1
IFC	1/1
IFNC	1/1
PUSHA	1/3
POPA	1/3
ANDSZ	2/2

Transfer of Control Instructions

JMPL	3/4
JMP	2/3
JP	1/3
JSRL	3/5
JSR	2/5
JID	1/3
VIS	1/5
RET	1/5
RETSK	1/5
RETI	1/5
INTR	1/7
NOP	1/1

Memory Transfer Instructions

	Register Indirect		Direct	Immed.	Register Indirect Auto Incr. & Decr.		
	[B]	[X]			[B+, B−]	[X+, X−]	
X A,*	1/1	1/3	2/3		1/2	1/3	
LD A,*	1/1	1/3	2/3	2/2	1/2	1/3	
LD B, Imm				1/1			(IF B < 16)
LD B, Imm				2/2			(IF B > 15)
LD Mem, Imm	2/2		3/3		2/2		
LD Reg, Imm			2/3				
IFEQ MD, Imm			3/3				

* = > Memory location addressed by B or X or directly.

Opcode Table

Upper Nibble																Lower Nibble
F	E	D	C	B	A	9	8	7	6	5	4	3	2	1	0	
JP−15	JP−31	LD 0F0, #i	DRSZ 0F0	RRCA	RC	ADC A,#i	ADC A,[B]	IFBIT 0,[B]	ANDSZ A, #i	LD B,#0F	IFBNE 0	JSR x000−x0FF	JMP x000−x0FF	JP+17	INTR	0
JP−14	JP−30	LD 0F1, #i	DRSZ 0F1	*	SC	SUBC A, #i	SUB A,[B]	IFBIT 1,[B]	*	LD B,#0E	IFBNE 1	JSR x100−x1FF	JMP x100−x1FF	JP+18	JP+2	1
JP−13	JP−29	LD 0F2, #i	DRSZ 0F2	X A,[X+]	X A,[B+]	IFEQ A,#i	IFEQ A,[B]	IFBIT 2,[B]	*	LD B,#0D	IFBNE 2	JSR x200−x2FF	JMP x200−x2FF	JP+19	JP+3	2
JP−12	JP−28	LD 0F3, #i	DRSZ 0F3	X A,[X−]	X A,[B−]	IFGT A,#i	IFGT A,[B]	IFBIT 3,[B]	*	LD B,#0C	IFBNE 3	JSR x300−x3FF	JMP x300−x3FF	JP+20	JP+4	3
JP−11	JP−27	LD 0F4, #i	DRSZ 0F4	VIS	LAID	ADD A,#i	ADD A,[B]	IFBIT 4,[B]	CLRA	LD B,#0B	IFBNE 4	JSR x400−x4FF	JMP x400−x4FF	JP+21	JP+5	4
JP−10	JP−26	LD 0F5, #i	DRSZ 0F5	RPND	JID	AND A,#i	AND A,[B]	IFBIT 5,[B]	SWAPA	LD B,#0A	IFBNE 5	JSR x500−x5FF	JMP x500−x5FF	JP+22	JP+6	5
JP−9	JP−25	LD 0F6, #i	DRSZ 0F6	X A,[X]	X A,[B]	XOR A,#i	XOR A,[B]	IFBIT 6,[B]	DCORA	LD B,#09	IFBNE 6	JSR x600−x6FF	JMP x600−x6FF	JP+23	JP+7	6
JP−8	JP−24	LD 0F7, #i	DRSZ 0F7	*	*	OR A,#i	OR A,[B]	IFBIT 7,[B]	PUSHA	LD B,#08	IFBNE 7	JSR x700−x7FF	JMP x700−x7FF	JP+24	JP+8	7
JP−7	JP−23	LD 0F8, #i	DRSZ 0F8	NOP	RLCA	LD A,#i	IFC	SBIT 0,[B]	RBIT 0,[B]	LD B,#07	IFBNE 8	JSR x800−x8FF	JMP x800−x8FF	JP+25	JP+9	8
JP−6	JP−22	LD 0F9, #i	DRSZ 0F9	IFNE A,[B]	IFEQ Md,#i	IFNE A,#i	IFNC	SBIT 1,[B]	RBIT 1,[B]	LD B,#06	IFBNE 9	JSR x900−x9FF	JMP x900−x9FF	JP+26	JP+10	9
JP−5	JP−21	LD 0FA, #i	DRSZ 0FA	LD A,[X+]	LD A,[B+]	LD [B+],#i	INCA	SBIT 2,[B]	RBIT 2,[B]	LD B,#05	IFBNE 0A	JSR xA00−xAFF	JMP xA00−xAFF	JP+27	JP+11	A
JP−4	JP−20	LD 0FB, #i	DRSZ 0FB	LD A,[X−]	LD A,[B−]	LD [B−],#i	DECA	SBIT 3,[B]	RBIT 3,[B]	LD B,#04	IFBNE 0B	JSR xB00−xBFF	JMP xB00−xBFF	JP+28	JP+12	B
JP−3	JP−19	LD 0FC, #i	DRSZ 0FC	LD Md,#i	JMPL	X A,Md	POPA	SBIT 4,[B]	RBIT 4,[B]	LD B,#03	IFBNE 0C	JSR xC00−xCFF	JMP xC00−xCFF	JP+29	JP+13	C
JP−2	JP−18	LD 0FD, #i	DRSZ 0FD	DIR	JSRL	LD A,Md	RETSK	SBIT 5,[B]	RBIT 5,[B]	LD B,#02	IFBNE 0D	JSR xD00−xDFF	JMP xD00−xDFF	JP+30	JP+14	D
JP−1	JP−17	LD 0FE, #i	DRSZ 0FE	LD A,[X]	LD A,[B]	LD [B],#i	RET	SBIT 6,[B]	RBIT 6,[B]	LD B,#01	IFBNE 0E	JSR xE00−xEFF	JMP xE00−xEFF	JP+31	JP+15	E
JP−0	JP−16	LD 0FF, #i	DRSZ 0FF	*	*	LD B,#i	RETI	SBIT 7,[B]	RBIT 7,[B]	LD B,#00	IFBNE 0F	JSR xF00−xFFF	JMP xF00−xFFF	JP+32	JP+16	F

Where,

i is the immediate data

Md is a directly addressed memory location

* *is an unused opcode*

Note: The opcode 60 Hex is also the opcode for IFBIT #i,A

Mask Options

The mask programmable options are shown below. The options are programmed at the same time as the ROM pattern submission.

OPTION 1: CLOCK CONFIGURATION

= 1 Crystal Oscillator (CKI/10)
G7 (CKO) is clock generator output to crystal/ resonator with CKI being the clock input

= 2 Single-pin RC controlled oscillator (CKI/10)
G7 is available as a HALT restart and/or general purpose input

OPTION 2: HALT

= 1 Enable HALT mode

= 2 Disable HALT mode

The chip can be driven by a clock input on the CKI input pin which can be between DC and 10 MHz. The CKO output clock is on pin G7 (if clock option = 1 has been selected). The CKI input frequency is divided down by 10 to produce the instruction cycle clock (1/tc).

Development Support

Summary

- iceMASTER: IM-COP8/400—Full feature in-circuit emulation for all COP8 products. A full set of COP8 Basic and Feature Family device and package specific probes are available.
- COP8 Debug Module: Moderate cost in-circuit emulation and development programming unit.
- COP8 Evaluation and Programming Unit: EPU-COP888GG—low cost In-circuit simulation and development programming unit.
- Assembler: COP8-DEV-IBMA. A DOS installable cross development Assembler, Linker, Librarian and Utility Software Development Tool Kit.
- C Compiler: COP8C. A DOS installable cross development Software Tool Kit.
- OTP/EPROM Programmer Support: Covering needs from engineering prototype, pilot production to full production environments.

Development Support (Continued)

IceMASTER (IM) IN-CIRCUIT EMULATION

The iceMASTER IM-COP8/400 is a full feature, PC based, in-circuit emulation tool developed and marketed by MetaLink Corporation to support the whole COP8 family of products. National is a resale vendor for these products.

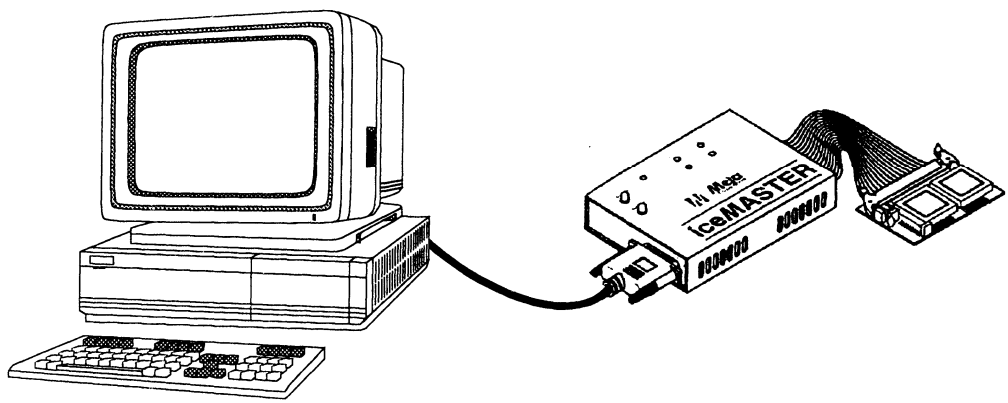
See *Figure 19* for configuration.

The iceMASTER IM-COP8/400 with its device specific COP8 Probe provides a rich feature set for developing, testing and maintaining product:

- Real-time in-circuit emulation; full 2.4V–5.5V operation range, full DC-10 MHz clock. Chip options are programmable or jumper selectable.
- Direct connection to application board by package compatible socket or surface mount assembly.
- Full 32 kbytes of loadable programming space that overlays (replaces) the on-chip ROM or EPROM. On-chip RAM and I/O blocks are used directly or recreated on the probe as necessary.
- Full 4K frame synchronous trace memory. Address, instruction, and 8 unspecified, circuit connectable trace lines. Display can be HLL source (e.g., C source), assembly or mixed.
- A full 64K hardware configurable break, trace on, trace off control, and pass count increment events.
- Tool set integrated interactive symbolic debugger—supports both assembler (COFF) and C Compiler (.COD) linked object formats.
- Real time peformance profiling analysis; selectable bucket definition.
- Watch windows, content updated automatically at each execution break.
- Instruction by instruction memory/register changes displayed on source window when in single step operation.
- Single base unit and debugger software reconfigurable to support the entire COP8 family; only the probe personality needs to change. Debugger software is processor customized, and reconfigured from a master model file.
- Processor specific symbolic display of registers and bit level assignments, configured from master model file.
- Halt/Idle mode notification.
- On-line HELP customized to specific processor using master model file.
- Includes a copy of COP8-DEV-IBMA assembler and linker SDK.

IM Order Information

Base Unit	
IM-COP8/400-1	iceMASTER base unit, 110V power supply
IM-COP8/400-2	iceMASTER base unit, 220V power supply
iceMASTER Probe	
MHW-888KG40DWPC MHW-888KG44PWPC	40 DIP 44 PLCC



TL/DD/12864–21

FIGURE 19. COP8 iceMASTER Environment

Development Support (Continued)

IceMASTER DEBUG MODULE (DM)

The iceMASTER Debug Module is a PC based, combination in-circuit emulation tool and COP8 based OTP/EPROM programming tool developed and marketed by MetaLink Corporation to support the whole COP8 family of products. National is a resale vendor for these products.

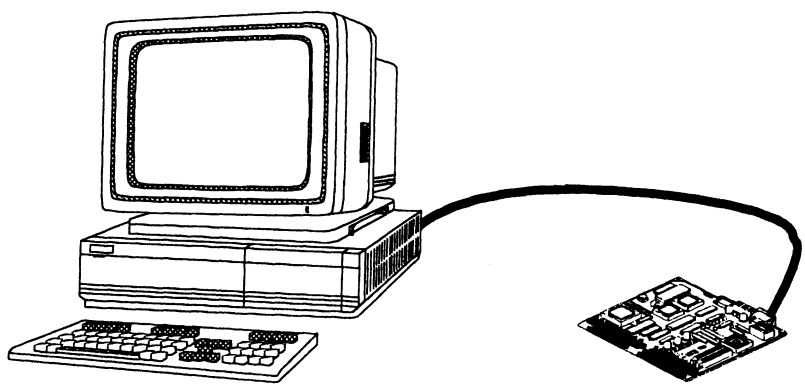
See *Figure 20* for configuration.

The iceMASTER Debug Module is a moderate cost development tool. It has the capability of in-circuit emulation for a specific COP8 microcontroller and in addition serves as a programming tool for COP8 OTP and EPROM product families. Summary of features is as follows:

- Real-time in-circuit emulation; full operating voltage range operation, full DC-10 MHz clock.
- All processor I/O pins can be cabled to an application development board with package compatible cable to socket and surface mount assembly.
- Full 32 kbytes of loadable programming space that overlays (replaces) the on-chip ROM or EPROM. On-chip RAM and I/O blocks are used directly or recreated as necessary.
- 100 frames of synchronous trace memory. The display can be HLL source (C source), assembly or mixed. The most recent history prior to a break is available in the trace memory.
- Configured break points; uses INTR instruction which is modestly intrusive.
- Software—only supported features are selectable.
- Tool set integrated interactive symbolic debugger—supports both Assembler (COFF) and C Compiler (.COD) SDK linked object formats.
- Instruction by instruction memory/register changes displayed when in single step operation.
- Debugger software is processor customized, and reconfigured from a master model file.
- Processor specific symbolic display of registers and bit level assignments, configured from master model file.
- Halt/Idle mode notification.
- Programming menu supports full product line of programmable OTP and EPROM COP8 products. Program data is taken directly from the overlay RAM.
- Programming of 44 PLCC and 68 PLCC parts requires external programming adapters.
- Includes wall mount power supply
- On-board V_{PP} generator from 5V input or connection to external supply supported. Requires V_{PP} level adjustment per the family programming specification (correct level is provided on an on-screen pop-down display).
- On-line HELP customized to specific processor using master model file.
- Includes a copy of COP8-DEV-IBMA assembler and linker SDK.

DM Order Information

Debug Module Unit	
COP8-DM/888KG	
Cable Adapters	
DM-COP8/40D	40 DIP
DM-COP8/44P	44 PLCC



TL/DD/12864-22

FIGURE 20. COP8-DM Environment

Development Support (Continued)

iceMASTER EVALUATION PROGRAMMING UNIT (EPU)

The iceMASTER EPU-COP888GG is a PC based, in-circuit simulation tool to support the feature family COP8 products.

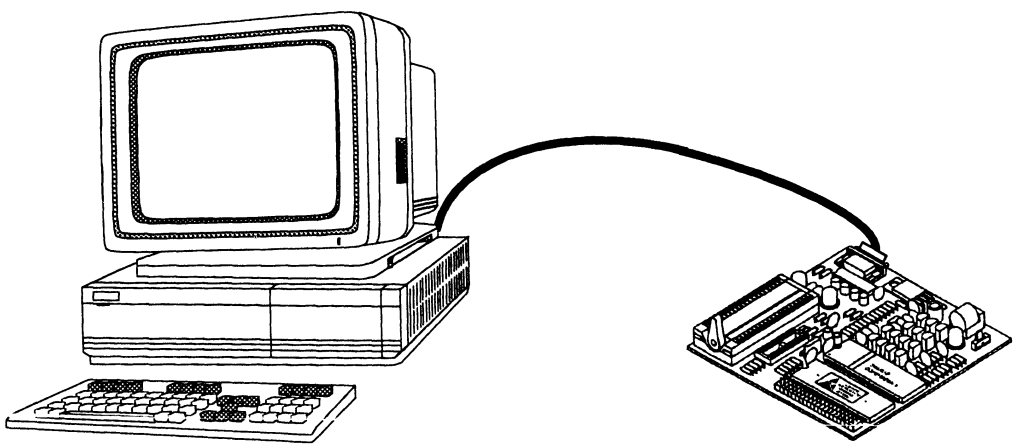
See *Figure 21* for configuration.

The simulation capability is a very low cost means of evaluation the general COP8 architecture. In addition, the EPU has programming capability, with added adapters, for programming the whole COP8 product family of OTP and EPROM products. The product includes the following features:

- Non-real-time in-circuit simulation. Program overlay memory is PC resident; instructions are downloaded over RS-232 as executed. Approximate performance is 20 kHz.
- Includes a 40 pin DIP cable adapter. Other target packages are not supported. All processor I/O pins are cabled to the application development environment.
- Full 32 kbyte of loadable programmable space that overlays (replaces) the on-chip ROM or EPROM. On-chip RAM and I/O blocks are used directly or recreated as necessary.
- On-chip timer and watch-dog execution are not well synchronized to the instruction simulation.
- 100 frames of synchronous trace memory. The display can be HLL source (e.g., C source), assembly or mixed. The most recent history prior to a break is available in the trace memory.
- Up to eight software configured break points; uses INTR instruction which is modestly intrusive.
- Common look-feel debugger software across all MetaLink products—only supported features are selectable.
- Tool set integrated interactive symbolic debugger—supports both assembler (COFF and C Compiler (.COD) SDK linked object formats.
- Instruction by instruction memory/register changes displayed when in single step operation.
- Processor specific symbolic display of registers and bit level assignments, configured from master model file.
- Halt/Idle mode notification. Restart requires special handling.
- Programming menu supports full product line of programmable OTP and EPROM COP8 products. Only a 40 ZIF socket is available on the EPU unit. Adapters are available for other part package configurations.
- Integral wall mount power supply provides 5V and develops the required V_{PP} to program parts.
- Includes a copy of COP8-DEV-IBMA assembler, linker SDK.

EPU Order Information

Evaluation Programming Unit	
EPU-COP888GG	Evaluation Programming Unit with debugger and programmer control software with 40 ZIF programming socket.
General Programming Adapters	
COP8-PGMA-DS	28 and 20 DIP and SOIC adapter
COP8-PGMA-DS44P	28 and 20 DIP and SOIC plus 44 PLCC adapter



TL/DD/12864–23

FIGURE 21. EPU-COP8 Tool Environment

Development Support (Continued)

COP8 ASSEMBLER/LINKER SOFTWARE DEVELOPMENT TOOL KIT

National Semiconductor offers a relocateable COP8 macro cross assembler, linker, librarian and utility software development tool kit. Features are summarized as follows:

- Basic and Feature Family instruction set by "device" type.
- Nested macro capability.
- Extensive set of assembler directives.
- Supported on PC/DOS platform.
- Generates National standard COFF output files.
- Integrated Linker & Librarian.
- Integrated utilities to generate ROM code file outputs.
- DUMPCOFF utility.

This product is integrated as a part of MetaLink tools as a development kit, fully supported by the MetaLink debugger. It may be ordered separately or it is bundled with the MetaLink products at no additional cost.

Order Information

Assembler SDK:	
COP8-DEV-IBMA	Assembler SDK on installable 3.5" PC/DOS Floppy Disk Drive format. Periodic upgrades and most recent version is available on National's BBS and Internet.

COP8 C COMPILER

A C Compiler is developed and marketed by Byte Craft Limited. The COP8C compiler is a fully integrated development tool specifically designed to support the compact embedded configuration of the COP8 family of products.

Features are summarized as follows:

- ANSI C wrth some restrictions and extensions that optimize development for the COP8 embedded application.
- BITS data type extension. Register declaration #pragma with direct bit level definitions.
- C language support for interrupt routines.
- Expert system, rule based code generation and optimization.
- Performs consistency checks against the architectural definitions of the target COP8 device.
- Generates program memory code.
- Supports linking of compiled object or COP8 assembled object formats.
- Global optimization of linked code.
- Symbolic debug load format fully source level supported by the MetaLink debugger.

INDUSTRY WIDE OTP/EPROM PROGRAMMING SUPPORT

Programming support, in addition to the MetaLink development tools, is provided by a full range of independent approved vendors to meet the needs from the engineering laboratory to full production.

Approved List

Manufacturer	North America	Europe	Asia
BP Microsystems	(800) 225-2102 (713) 688-4600 Fax: (713) 688-0920	+49-8152-4183 +49-8856-932616	+852-234-16611 +852-2710-8121
Data I/O	(800) 426-1045 (206) 881-6444 Fax: (206) 882-1043	+44-0734-440011	Call North America
HI-LO	(510) 623-8860	Call Asia	+886-2-764-0215 Fax: +886-2-756-6403
ICE Technology	(800) 624-8949 (919) 430-7915	+44-1226-767404 Fax: 0-1226-370-434	
MetaLink	(800) 638-2423 (602) 926-0797 Fax: (602) 693-0681	+49-80 9156 96-0 Fax: 49-80 9123 86	+852-737-1800
Systems General	(408) 263-6667	+41-1-9450300	+886-2-917-3005 Fax: +886-2-911-1283
Needhams	(916) 924-8037 Fax: (916) 924-8065		

Development Support (Continued)

AVAILABLE LITERATURE

For more information, please see the COP8 Basic Family User's Manual, Literature Number 620895, COP8 Feature Family User's Manual Literature Number 620897 and National's Family of 8-bit Microcontrollers COP8 Selection Guide, Literature Number 630009.

DIAL-A-HELPER

Dial-A-Helper is a service provided by the Microcontroller Applications group. The Dial-A-Helper is an Electronic Information System that may be accessed as a Bulletin Board System (BBS) via data modem, as an FTP site on the Internet via standard FTP client application or as an FTP site on the Internet using a standard Internet browser such as Netscape or Mosaic.

The Dial-A-Helper system provides access to an automated information storage and retrieval system. The system capabilities include a MESSAGE SECTION (electronic mail, when accessed as a BBS) for communication to and from the Microcontroller Applications Group and a FILE SECTION which consists of several file areas where valuable application software and utilities could be found.

DIAL-A-HELPER BBS via a Standard Modem

Modem: CANADA/U.S.: (800) NSC-MICRO
(800) 672-6427
EUROPE: (+49) 0-8141-351332
Baud: 14.4k
Set-up: Length: 8-Bit
Parity: None
Stop Bit: 1
Operation: 24 Hrs., 7 Days

DIAL-A-HELPER via FTP

ftp nscmicro.nsc.com
user: anonymous
password: username@yourhost.site.domain

DIAL-A-HELPER via a WorldWide Web Browser

ftp://nscmicro.nsc.com

National Semiconductor on the WorldWide Web

See us on the WorldWide Web at: http://www.national.com

CUSTOMER RESPONSE CENTER

Complete product information and technical support is available from National's customer response centers.

CANADA/U.S.:	Tel:	(800) 272-9959
	email:	support @tevm2.nsc.com
EUROPE:	email:	europe.support@nsc.com
	Deutsch Tel:	+49 (0) 180-530 85 85
	English Tel:	+49 (0) 180-532 78 32
	Français Tel:	+49 (0) 180-532 93 58
	Italiano Tel:	+49 (0) 180-534 16 80
JAPAN:	Tel:	+81-043-299-2309
S.E. ASIA:	Beijing Tel:	(+86) 10-6856-8601
	Shanghai Tel:	(+86) 21-6415-4092
	Hong Kong Tel:	(+852) 2737-1600
	Korea Tel:	(+82) 2-3771-6909
	Malaysia Tel:	(+60-4) 644-9061
	Singapore Tel:	(+65) 255-2226
	Taiwan Tel:	+886-2-521-3288
AUSTRALIA:	Tel:	(+61) 3-9558-9999
INDIA:	Tel:	(+91) 80-559-9467

Section 3

COP8™ 32K OTP Products

Section 3 Contents

COP87L40RJ/COP87L42RJ 8-Bit One-Time Programmable (OTP) Microcontroller with 32 Kbytes of Program Memory 3-3

COP87L88RK/COP87L84RK 8-Bit One-Time Programmable (OTP) Microcontroller with Analog Function Block and 32 Kbytes of Program Memory 3-29

COP87L88RG 8-Bit One-Time Programmable (OTP) Microcontroller with 32 Kbytes of Program Memory 3-61

COP87L84RG 8-Bit One-Time Programmable (OTP) Microcontroller with 32 Kbytes of Program Memory 3-98

COP87L88RD 8-Bit One-Time Programmable (OTP) Microcontroller with A/D Converter and 32 Kbytes of Program Memory 3-135

COP87L88RW 8-Bit One-Time Programmable (OTP) Microcontroller with Pulse Train Generators and Capture Modules 3-137

National Semiconductor

PRELIMINARY

COP87L40RJ/COP87L42RJ 8-Bit One-Time Programmable (OTP) Microcontroller with 32 kbyte of Program Memory

General Description

The COP87L40RJ/COP87L42RJ are members of the COP8™ 8-bit OTP microcontroller family. They are pin and software compatible to the mask ROM COP840CJ/COP842CJ product family.

They are fully static parts, fabricated using double-metal silicon gate microCMOS technology. Features include an 8-bit memory mapped architecture, MICROWIRE/PLUS™ serial I/O, a 16-bit timer/counter with capture register, multi-sourced interrupts, Comparator, WATCHDOG™ Timer, Modulator/Timer and Multi-Input Wakeup.

Each I/O pin has software selectable options to adapt the device to the specific application. The device operates over a voltage range of 2.7V to 5.5V. High throughput is achieved with an efficient, regular instruction set operating at a maximum rate of 1 μs per instruction.

Key Features

- Multi-Input Wakeup (on the 8-bit Port L)
- Analog comparator
- Modulator/Timer (high speed PWM timer for IR transmission)
- 16-bit multi-function timer supporting
  - — PWM mode
  - — External event counter mode
  - — Input capture mode
- Integrated capacitor for the R/C oscillator
- 32 kbytes on-chip OTP EPROM with security feature

Note: Mask ROMed devices with equivalent on-chip features and program memory size of 1k and 2k are available.

- 128 bytes on-chip RAM
- R/C oscillator with on-chip capacitor

I/O Features

- Memory mapped I/O
- Software selectable I/O options (TRI-STATE®, Push-Pull, Weak Pull-Up Input, High Impedance Input)
- High current outputs (8 pins)
- Schmitt trigger inputs on Port G
- MICROWIRE/PLUS serial I/O
- Packages:
  - — 20 DIP/SO with 16 I/O pins
  - — 28 DIP/SO with 24 I/O pins

CPU/Instruction Set Features

- 1 μs instruction cycle time
- Three multi-source interrupts servicing
  - — External interrupt with selectable edge
  - — Timer interrupt
  - — Software interrupt
- Versatile and easy to use instruction set
- 8-bit Stack Pointer (SP)—stack in RAM
- Two 8-bit Register Indirect Data Memory Pointers (B and X)

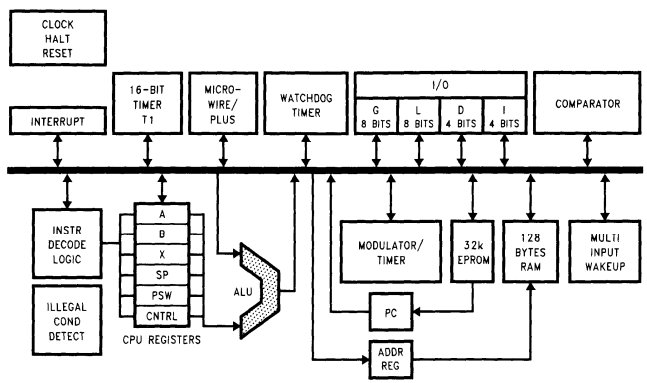
Fully Static CMOS

- Single supply operation: 2.7V to 5.5V
- Temperature range: −40°C to +85°C

Development Support

- Emulation device for the COP820CJ/COP822CJ and COP840CJ/COP842CJ
- Real time emulation and full program debug offered by MetaLink Development System

Block Diagram



TL/DD/12862–1

FIGURE 1. Block Diagram

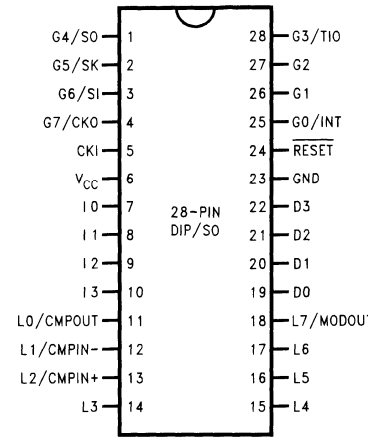
TABLE I. COP820CJ/840CJ Family Members

Device	ROM/EPROM (Bytes)	RAM (Bytes)	Key Common Features
COP820CJ	1k ROM	64	WATCHDOG
COP822CJ	1k ROM	64	Multi-Input
COP840CJ	2k ROM	128	Wakeup
COP842CJ	2k ROM	128	
COP87L40CJ	2k OTP EPROM	128	
COP87L42CJ	2k OTP EPROM	128	

Pin Assignment

Port Pin	Typ	ALT Funct.	20 Pin	28 Pin
L0	I/O	MIWU/CMPOUT	7	11
L1	I/O	MIWU/CMPIN−	8	12
L2	I/O	MIWU/CMPIN+	9	13
L3	I/O	MIWU	10	14
L4	I/O	MIWU	11	15
L5	I/O	MIWU	12	16
L6	I/O	MIWU	13	17
L7	I/O	MIWU/MODOUT	14	18
G0	I/O	INTR	17	25
G1	I/O		18	26
G2	I/O		19	27
G3	I/O	TIO	20	28
G4	I/O	SO	1	1
G5	I/O	SK	2	2
G6	I	SI	3	3
G7	I	CKO	4	4
I0	I			7
I1	I			8
I2	I			9
I3	I			10
D0	O			19
D1	O			20
D2	O			21
D3	O			22
V_{CC}			6	6
GND			15	23
CKI			5	5
$\overline{RESET}$			16	24

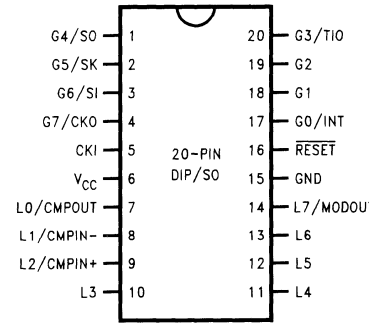
Connection Diagrams



TL/DD/12862-2

Top View

Order Number
COP87L40RJN (-1N, -2N, -3N)
COP87L40RJM (-1N, -2N, -3N)
See NS Package Number N28B or M28B



TL/DD/12862-3

Top View

Order Number
COP87L42RJN (-1N, -2N, -3N)
COP87L42RJM (-1N, -2N, -3N)
See NS Package Number N20A or M20B

Note: -1 Crystal Oscillator N - Brown out disabled
-2 External Oscillator
-3 R/C Oscillator

FIGURE 2. Connection Diagrams

Absolute Maximum Ratings

If Military/Aerospace specified devices are required, please contact the National Semiconductor Sales Office/Distributors for availability and specifications.

Supply Voltage (V_{CC})	7.0V
Voltage at any Pin	−0.3V to V_{CC} + 0.3V
Total Current into V_{CC} pin (Source)	80 mA
Total Current out of GND pin (sink)	80 mA
Storage Temperature Range	−65°C to +150°C

Note: *Absolute maximum ratings indicate limits beyond which damage to the device may occur.*

DC and AC electrical specifications are not ensured when operating the device at absolute maximum ratings.

DC Electrical Characteristics $-40°C \le T_A \le +85°C$ unless otherwise specified

Parameter	Conditions	Min	Typ	Max	Units
Operating Voltage		2.7		5.5	V
Power Supply Ripple 1 (Note 1)	Peak to Peak			0.1 V_{CC}	V
Supply Current (Note 2)					
CKI = 10 MHz	V_{CC} = 5.5V, tc = 1 μs			12	mA
CKI = 4 MHz	V_{CC} = 4.5V, tc = 2.5 μs			6.5	mA
HALT Current (Note 3)	V_{CC} = 5.5V, CKI = 0 MHz			12	μA
INPUT LEVELS (V_{IH}, V_{IL})					
Reset, CKI:					
Logic High		0.8 V_{CC}			V
Logic Low				0.2 V_{CC}	V
All Other Inputs					
Logic High		0.7 V_{CC}			V
Logic Low				0.2 V_{CC}	V
Hi-Z Input Leakage	V_{CC} = 5.5V	−2		+2	μA
Input Pullup Current	V_{CC} = 5.5V	−40		−250	μA
L- and G-Port Hysteresis (Note 6)				0.35 V_{CC}	V
Output Current Levels					
D Outputs:					
Source	V_{CC} = 4.5V, V_{OH} = 3.8V	−0.4			mA
Sink (Note 4)	V_{CC} = 4.5V, V_{OL} = 1.0V	10			mA
L4–L7 Output Sink	V_{CC} = 4.5V, V_{OL} = 2.5V	15			mA
All Others					
Source (Weak Pull-up Mode)	V_{CC} = 4.5V, V_{OH} = 3.2V	−10		−110	μA
Source (Push-pull Mode)	V_{CC} = 4.5V, V_{OH} = 3.8V	−0.4			mA
Sink (Push-pull Mode)	V_{CC} = 4.5V, V_{OL} = 0.4V	1.6			mA
TRI-STATE Leakage		−2.0		+2.0	μA
Allowable Sink/Source Current Per Pin					
D Outputs				15	mA
L4–L7 (Sink)				20	mA
All Others				3	mA
Maximum Input Current without Latchup (Note 5)	Room Temperature			±100	mA
RAM Retention Voltage, V_r	500 ns Rise and Fall Time (Min)	2.0			V
Input Capacitance				7	pF
Load Capacitance on D2				1000	pF

Note 1: Rate of voltage change must be less than 10 V/mS.

Note 2: Supply current is measured after running 2000 cycles with a square wave CKI input, CKO open, inputs at rails and outputs open.

Note 3: The HALT mode will stop CKI from oscillating in the RC and crystal configurations by bringing CKI high. HALT test conditions: L, and G0..G5 ports configured as outputs and set high. The D port set to zero. All inputs tied to V_{CC}. The comparator is disabled.

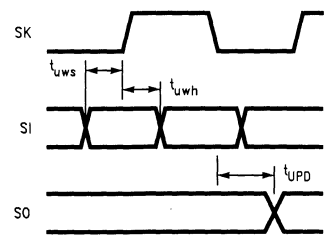
Note 4: The user must guarantee that D2 pin does not source more than 10 mA during RESET. If D2 sources more than 10 mA during reset, the device will go into programming mode.

Note 5: Pins G6 and $\overline{\text{RESET}}$ are designed with a high voltage input network. These pins allow input voltages greater than V_{CC} and the pins will have sink current to V_{CC} when biased at voltages greater than V_{CC} (the pins do not have source current when biased at a voltage below V_{CC}). The effective resistance to V_{CC} is 750Ω (typical). These two pins will not latch up. The voltage at the pins must be limited to less than 14V.

AC Electrical Characteristics $-40°C \le T_A \le +85°C$ unless otherwise specified

Parameter	Conditions	Min	Typ	Max	Units
Instruction Cycle Time (tc)					
Crystal/Resonator	$4.5V \le V_{CC} \le 5.5V$	1		DC	μs
R/C Oscillator	$4.5V \le V_{CC} \le 5.5V$	2		DC	μs
CKI Clock Duty Cycle (Note 6)	fr = Max	40		60	%
Rise Time (Note 6)	fr = 10 MHz ext. Clock			12	ns
Fall Time (Note 6)	fr = 10 MHz ext. Clock			8	ns
Inputs					
t_{Setup}	$4.5V \le V_{CC} \le 5.5V$	200			ns
t_{Hold}	$4.5V \le V_{CC} \le 5.5V$	60			ns
Output Propagation Delay	R_L = 2.2k, CL = 100 pF				
t_{PD1}, t_{PD0}					
SO, SK	$4.5V \le V_{CC} \le 5.5V$			0.7	μs
All Others	$4.5V \le V_{CC} \le 5.5V$			1	μs
Input Pulse Width					
Interrupt Input High Time		1			tc
Interrupt Input Low Time		1			tc
Timer Input High Time		1			tc
Timer Input Low Time		1			tc
MICROWIRE™ Setup Time ($t_{\mu WS}$)		20			ns
MICROWIRE Hold Time ($t_{\mu WH}$)		56			ns
MICROWIRE Output Propagation Delay ($t_{\mu PD}$)				220	ns
Reset Pulse Width		1			μs

Note 6: Parameter characterized but not production tested.



TL/DD/12862–4

FIGURE 3. MICROWIRE/PLUS Timing

Pin Description

V_{CC} and **GND** are the power supply pins.

CKI is the clock input. This can come from an external source, a R/C generated oscillator or a crystal (in conjunction with CKO). See Oscillator description.

$\overline{\text{RESET}}$ is the master reset input. See Reset description.

PORT I is a 4-bit Hi-Z input port.

PORT L is an 8-bit I/O port.

There are two registers associated with the L port: a data register and a configuration register. Therefore, each L I/O bit can be individually configured under software control as shown below:

Port L Config.	Port L Data	Port L Setup
0	0	Hi-Z Input (TRI-STATE)
0	1	Input with Weak Pull-up
1	0	Push-pull Zero Output
1	1	Push-pull One Output

Three data memory address locations are allocated for this port, one each for data register [00D0], configuration register [00D1] and the input pins [00D2].

Port L has the following alternate features:

L0 MIWU or CMPOUT

L1 MIWU or CMPIN−

L2 MIWU or CMPIN+

L3 MIWU

L4 MIWU (high sink current capability)

L5 MIWU (high sink current capability)

L6 MIWU (high sink current capability)

L7 MIWU or MODOUT (high sink current capability)

The selection of alternate Port L functions is done through registers WKEN [00C9] to enable MIWU and CNTRL2 [00CC] to enable comparator and modulator.

All eight L-pins have Schmitt Triggers on their inputs.

PORT G is an 8-bit port with 6 I/O pins (G0–G5) and 2 input pins (G6, G7).

All eight G-pins have Schmitt Triggers on the inputs.

There are two registers associated with the G port: a data register and a configuration register. Therefore each G port bit can be individually configured under software control as shown below:

Port G Config.	Port G Data	Port G Setup
0	0	Hi-Z Input (TRI-STATE)
0	1	Input with Weak Pull-up
1	0	Push-pull Zero Output
1	1	Push-pull One Output

Three data memory address locations are allocated for this port, one for data register [00D4], one for configuration register [00D5] and one for the input pins [00D6]. Since G6 and G7 are Hi-Z input only pins, any attempt by the user to configure them as outputs by writing a one to the configuration register will be disregarded. Reading the G6 and G7 configuration bits will return zeros. Note that the device will be placed in the Halt mode by writing a "1" to the G7 data bit.

Six pins of Port G have alternate features:

G0 INTR (an external interrupt)

G3 TIO (timer/counter input/output)

G4 SO (MICROWIRE serial data output)

G5 SK (MICROWIRE clock I/O)

G6 SI (MICROWIRE serial data input)

G7 CKO crystal oscillator output (selected by mask option) or HALT restart input/general purpose input (if clock option is R/C or external clock)

Pins G1 and G2 currently do not have any alternate functions.

The selection of alternate Port G functions are done through registers PSW [00EF] to enable external interrupt and CNTRL1 [00EE] to select TIO and MICROWIRE operations.

PORT D is a four bit output port that is preset when RESET goes low. One data memory address location is allocated for the data register [00DC]. The user can tie two or more D port outputs (except D2 pin) together in order to get a higher drive.

Note: Care must be exercised with the D2 pin operation. At RESET, the external loads on this pin must ensure that the output voltages stay above 0.8 V_{CC} to prevent the chip from entering special modes. Also keep the external loading on D2 to less than 1000 pF.

Functional Description

The internal architecture is shown in the block diagram. Data paths are illustrated in simplified form to depict how the various logic elements communicate with each other in implementing the instruction set of the device.

ALU and CPU Registers

The ALU can do an 8-bit addition, subtraction, logical or shift operations in one cycle time. There are five CPU registers:

A is the 8-bit Accumulator register

PC is the 15-bit Program Counter register

PU is the upper 7 bits of the program counter (PC)

PL is the lower 8 bits of the program counter (PC)

B is the 8-bit address register and can be auto incremented or decremented.

X is the 8-bit alternate address register and can be auto incremented or decremented.

SP is the 8-bit stack pointer which points to the subroutine stack (in RAM).

B, X and SP registers are mapped into the on chip RAM. The B and X registers are used to address the on chip RAM. The SP register is used to address the stack in RAM during subroutine calls and returns. The SP must be initialized by software before any subroutine call or interrupts occurs.

Memory

The memory is separated into two memory spaces: program and data.

PROGRAM MEMORY

Program memory consists of 32 kBytes of OTP EPROM. These bytes of ROM may be instructions or constant data. The memory is addressed by the 15-bit program counter (PC). ROM can be indirectly read by the LAID instruction for table lookup.

The device can be configured to inhibit external reads of the program memory. This is done by programming the Security Byte.

Note: Mask ROMed devices with equivalent on-chip features and program memory size of 1k and 2k are available.

SECURITY FEATURE

The memory array has an associate Security Byte that is located outside of the program address range. This byte can be addressed only from programming mode by a programmer tool.

Security is an optional feature and can only be asserted after the memory array has been programmed and verified. A secured part will read all 00(hex) by a programmer. The part will fail Blank Check and will fail Verify operations. A Read operation will fill the programmer's memory with 00(hex). The Security Byte itself is always readable with value of 00(hex) if unsecure and FF(hex) if secure.

DATA MEMORY

The data memory address space includes on chip RAM, I/O and registers. Data memory is addressed directly by the instruction or indirectly through B, X and SP registers. The device has 128 bytes of RAM. Sixteen bytes of RAM are mapped as "registers", these can be loaded immediately, decremented and tested. Three specific registers: X, B, and SP are mapped into this space, the other registers are available for general usage.

Any bit of data memory can be directly set, reset or tested. All I/O and registers (except A and PC) are memory mapped; therefore, I/O bits and register bits can be directly and individually set, reset and tested, except the write once only bit (WDREN, WATCHDOG Reset Enable), and the unused and read only bits in CNTRL2 and WDREG registers.

Note: RAM contents are undefined upon power-up.

Reset

EXTERNAL RESET

The RESET input pin when pulled low initializes the microcontroller. The user must insure that the RESET pin is held low until V_{CC} is within the specified voltage range and the clock is stabilized. An R/C circuit with a delay 5x greater than the power supply rise time is recommended *(Figure 4)*. The device immediately goes into reset state when the RESET input goes low. When the RESET pin goes high the device comes out of reset state synchronously. The device will be running within two instruction cycles of the RESET pin going high. The following actions occur upon reset:

Port L	TRI-STATE
Port G	TRI-STATE
Port D	HIGH
PC	CLEARED
RAM Contents	RANDOM with Power-On-Reset UNAFFECTED with external Reset (power already applied)
B, X, SP	Same as RAM
PSW, CNTRL1, CNTRL2 and WDREG Reg.	CLEARED
Multi-Input Wakeup Reg. WKEDG, WKEN WKPND	 CLEARED UNKNOWN
Data and Configuration Registers for L & G	CLEARED
WATCHDOG Timer	Prescaler/Counter each loaded with FF

The device comes out of the HALT mode when the RESET pin is pulled low. In this case, the user has to ensure that the RESET signal is low long enough to allow the oscillator to restart. An internal 256 t_C delay is normally used in conjunction with the two pin crystal oscillator. When the device comes out of the HALT mode through Multi-Input Wakeup, this delay allows the oscillator to stabilize.

The following additional actions occur after the device comes out of the HALT mode through the RESET pin.

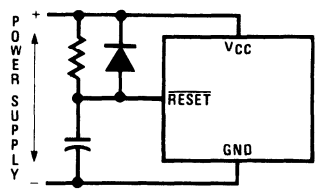
If a two pin crystal/resonator oscillator is being used:

RAM Contents	UNCHANGED
Timer T1 and A Contents	UNKNOWN
WATCHDOG Timer Prescaler/Counter	ALTERED

Functional Description (Continued)

If the external or RC Clock option is being used:

RAM Contents	UNCHANGED
Timer T1 and A Contents	UNCHANGED
WATCHDOG Timer Prescaler/Counter	ALTERED



RC > 5 × Power Supply Rise Time TL/DD/12862–5

FIGURE 4. Recommended Reset Circuit

WATCHDOG RESET

With WATCHDOG enabled, the WATCHDOG logic resets the device if the user program does not service the WATCHDOG timer within the selected service window. The WATCHDOG reset does not disable the WATCHDOG. Upon WATCHDOG reset, the WATCHDOG Prescaler/Counter are each initialized with FF Hex.

The following actions occur upon WATCHDOG reset that are different from external reset.

WDREN	WATCHDOG Reset Enable bit	UNCHANGED
WDUDF	WATCHDOG Underflow bit	UNCHANGED

Additional initialization actions that occur as a result of WATCHDOG reset are as follows:

Port L	TRI-STATE
Port G	TRI-STATE
Port D	HIGH
PC	CLEARED
Ram Contents	UNCHANGED
B, X, SP	UNCHANGED
PSW, CNTRL1 and CNTRL2 (except WDUDF Bit) Registers	CLEARED
Multi-Input Wakeup Registers WKEDG, WKEN WKPND	 CLEARED UNKNOWN
Data and Configuration Registers for L & G	CLEARED
WATCHDOG Timer	Prescalar/Counter each loaded with FF

Oscillator Circuits

EXTERNAL OSCILLATOR

By selecting the external oscillator option, the CKI pin can be driven by an external clock signal provided it meets the specified duty cycle, rise and fall times, and input levels. The G7/CKO is available as a general purpose input G7 and/or HALT control.

CRYSTAL OSCILLATOR

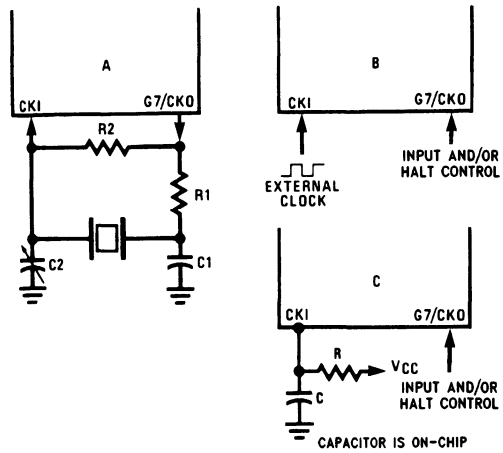
By selecting the crystal oscillator option, the G7/CKO pin is connected as a clock output, CKI and G7/CKO can be connected to make a crystal controlled oscillator. Table I shows the clock frequency for different component values. See *Figure 5* for the connections.

R/C OSCILLATOR

By selecting R/C oscillator option, connecting a resistor from the CKI pin to V_{CC} makes a R/C oscillator. The capacitor is on-chip. The G7/CKO pin is available as a general purpose input G7 and/or HALT control. Adding an external capacitor will jeopardize the clock frequency tolerance and increase EMI emissions.

Table II shows the clock frequency for the different resistor values. The capacitor is on-chip. See *Figure 5* for the connections.

Oscillator Circuits (Continued)



TL/DD/12862–6

FIGURE 5. Clock Oscillator Configurations

TABLE I. Crystal Oscillator Configuration (T_A = 25°C)

R1 (kΩ)	R2 (MΩ)	C1 (pF)	C2 (pF)	CKI Freq. (MHz)	Conditions
0	1	30	30–36	10	V_{CC} = 5V
0	1	30	30–36	4	V_{CC} = 5V
5.6	1	100	100–156	0.455	V_{CC} = 5V

TABLE II. RC Oscillator Configuration (Part-To-Part Variation)

R (kΩ) (Note 1)	CK1 Freq. (MHz)	Instr. Cycle (μs)	Conditions
2.2	7.0 ± 15%	−40°C– +85°C	4.5V to 5.5V
4.7	4.2 ± 10%	−40°C– +85°C	4.5V to 5.5V
20.0	1.1 ± 10%	−40°C– +85°C	4.5V to 5.5V

Note 1: The resistance level is calculated with a total of 5.3 pF capacitance added from the printed circuit board. It is important to take this into account when figuring the clock frequency.

Halt Mode

The device is a fully static device. The device enters the HALT mode by writing a one to the G7 bit of the G data register. Once in the HALT mode, the internal circuitry does not receive any clock signal and is therefore frozen in the exact state it was in when halted. In this mode the chip will only draw leakage current.

The device supports three different methods of exiting the HALT mode. The first method is with a low to high transition on the CKO (G7) pin. This method precludes the use of the crystal clock configuration (since CKO is a dedicated output). It may be used either with an RC clock configuration or an external clock configuration. The second method of exiting the HALT mode is with the multi-Input Wakeup feature on the L port. The third method of exiting the HALT mode is by pulling the RESET input low.

If the two pin crystal/resonator oscillator is being used and Multi-Input Wakeup causes the device to exit the HALT mode, the WAKEUP signal does not allow the chip to start running immediately since crystal oscillators have a delayed start up time to reach full amplitude and freuqency stability. The WATCHDOG timer (consisting of an 8-bit prescaler followed by an 8-bit counter) is used to generate a fixed delay of 256tc to ensure that the oscillator has indeed stabilized before allowing instruction execution. In this case, upon detecting a valid WAKEUP signal only the oscillator circuitry is enabled. The WATCHDOG Counter and Prescaler are each loaded with a value of FF Hex. The WATCHDOG prescaler is clocked with the tc instruction cycle. (The tc clock is derived by dividing the oscillator clock down by a factor of 10).

The Schmitt trigger following the CKI inverter on the chip ensures that the WATCHDOG timer is clocked only when the oscillator has a sufficiently large amplitude to meet the Schmitt trigger specs. This Schmitt trigger is not part of the oscillator closed loop. The start-up timeout from the WATCHDOG timer enables the clock signals to be routed to the rest of the chip. The delay is not activated when the device comes out of HALT mode through RESET pin. Also, if the clock option is either RC or External clock, the delay is not used, but the WATCHDOG Prescaler/-Counter contents are changed. The Development System will not emulate the 256tc delay.

The $\overline{\text{RESET}}$ pin will cause the device to reset and start executing from address X'0000. A low to high transition on the G7 pin (if single pin oscillator is used) or Multi-Input Wakeup will cause the device to start executing from the address following the HALT instruction.

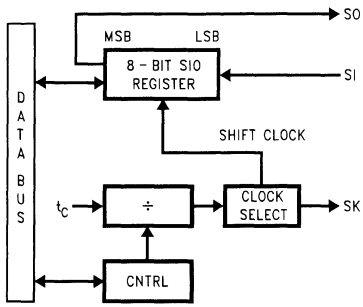
When $\overline{\text{RESET}}$ pin is used to exit the device from the HALT mode and the two pin crystal/resonator (CKI/CKO) clock option is selected, the contents of the Accumulator and the Timer T1 are undetermined following the reset. All other information except the WATCHDOG Prescaler/Counter contents is retained until continuing. All information except the WATCHDOG Prescaler/Counter contents is retained if the device exits the HALT mode through G7 pin or Multi-Input Wakeup.

G7 is the HALT-restart pin, but it can still be used as an input. If the device is not halted, G7 can be used as a general purpose input.

Note: To allow clock resynchronization, it is necessary to program two NOP's immediately after the device comes out of the HALT mode. The user must program two NOP's following the "enter HALT mode" (set G7 data bit) instruction.

MICROWIRE/PLUS

MICROWIRE/PLUS is a serial synchronous bidirectional communications interface. The MICROWIRE/PLUS capability enables the device to interface with any of National Semiconductor's MICROWIRE peripherals (i.e. A/D converters, display drivers, EEPROMS, etc.) and with other microcontrollers which support the MICROWIRE/PLUS interface. It consists of an 8-bit serial shift register (SIO) with serial data input (SI), serial data output (SO) and serial shift clock (SK). *Figure 6* shows the block diagram of the MICROWIRE/PLUS interface.



TL/DD/12862-7

FIGURE 6. MICROWIRE/PLUS Block Diagram

The shift clock can be selected from either an internal source or an external source. Operating the MICROWIRE/PLUS interface with the internal clock source is called the Master mode of operation. Operating the MICROWIRE/PLUS interface with an external shift clock is called the Slave mode of operation.

The CNTRL register is used to configure and control the MICROWIRE/PLUS mode. To use the MICROWIRE/PLUS, the MSEL bit in the CNTRL register is set to one. The SK clock rate is selected by the two bits, SL0 and SL1, in the CNTRL register. Table III details the different clock rates that may be selected.

TABLE III

SL1	SL0	SK Cycle Time
0	0	$2t_C$
0	1	$4t_C$
1	x	$8t_C$

where,

t_C is the instruction cycle time.

MICROWIRE/PLUS OPERATION

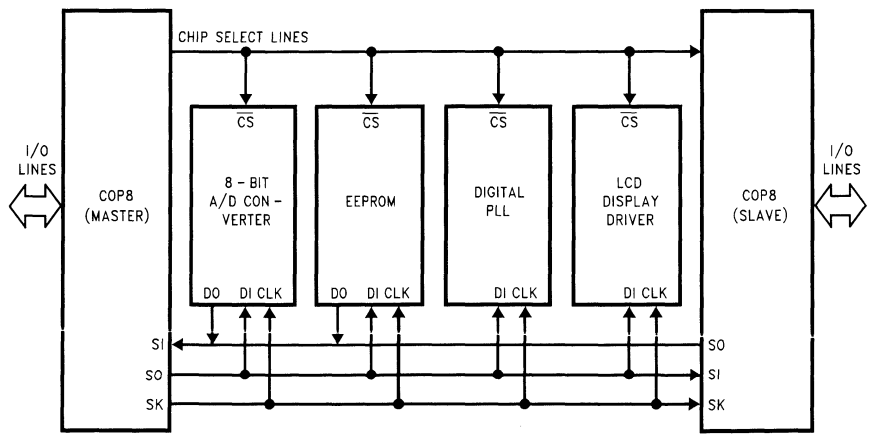
Setting the BUSY bit in the PSW register causes the MICROWIRE/PLUS arrangement to start shifting the data. It gets reset when eight data bits have been shifted. The user may reset the BUSY bit by software to allow less than 8 bits to shift. The device may enter the MICROWIRE/PLUS mode either as a Master or as a Slave. *Figure 7* shows how two device microcontrollers and several peripherals may be interconnected using the MICROWIRE/PLUS arrangement.

Master MICROWIRE/PLUS Operation

In the MICROWIRE/PLUS Master mode of operation the shift clock (SK) is generated internally by the device. The MICROWIRE/PLUS Master always initiates all data exchanges *(Figure 7)*. The MSEL bit in the CNTRL register must be set to enable the SO and SK functions on the G Port. The SO and SK pins must also be selected as outputs by setting appropriate bits in the Port G configuration register. Table IV summarizes the bit settings required for Master mode of operation.

Slave MICROWIRE/PLUS Operation

In the MICROWIRE/PLUS Slave mode of operation the SK clock is generated by an external source. Setting the MSEL bit in the CNTRL register enables the SO and SK functions on the G Port. The SK pin must be selected as an input and the SO pin selected as an output pin by appropriately setting up the Port G configuration register. Table IV summarizes the settings required to enter the Slave mode of operation.



TL/DD/12862-8

FIGURE 7. MICROWIRE/PLUS Application

MICROWIRE/PLUS (Continued)

The user must set the BUSY flag immediately upon entering the Slave mode. This will ensure that all data bits sent by the Master will be shifted properly. After eight clock pulses the BUSY flag will be cleared and the sequence may be repeated (see *Figure 7*).

TABLE IV

G4 Config. Bit	G5 Config. Bit	G4 Fun.	G5 Fun.	G6 Fun.	Operation
1	1	SO	Int. SK	SI	MICROWIRE Master
0	1	TRI-STATE	Int. SK	SI	MICROWIRE Master
1	0	SO	Ext. SK	SI	MICROWIRE Slave
0	0	TRI-STATE	Ext. SK	SI	MICROWIRE Slave

Timer/Counter

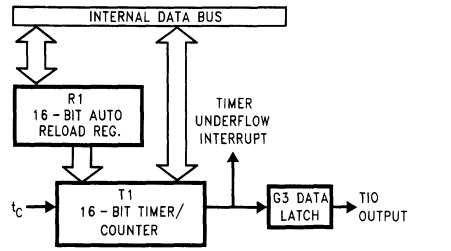
The device has a powerful 16-bit timer with an associated 16-bit register enabling it to perform extensive timer functions. The timer T1 and its register R1 are each organized as two 8-bit read/write registers. Control bits in the register CNTRL allow the timer to be started and stopped under software control. The timer-register pair can be operated in one of three possible modes. Table V details various timer operating modes and their requisite control settings.

MODE 1. TIMER WITH AUTO-LOAD REGISTER

In this mode of operation, the timer T1 counts down at the instruction cycle rate. Upon underflow the value in the register R1 gets automatically reloaded into the timer which continues to count down. The timer underflow can be programmed to interrupt the microcontroller. A bit in the control register CNTRL enables the TIO (G3) pin to toggle upon timer underflows. This allows the generation of square-wave outputs or pulse width modulated outputs under software control *(Figure 8)*.

MODE 2. EXTERNAL COUNTER

In this mode, the timer T1 becomes a 16-bit external event counter. The counter counts down upon an edge on the TIO pin. Control bits in the register CNTRL program the counter to decrement either on a positive edge or on a negative edge. Upon underflow the contents of the register R1 are automatically copied into the counter. The underflow can also be programmed to generate an interrupt *(Figure 8)*.



TL/DD/12862–9

FIGURE 8. Timer/Counter Auto Reload Mode Block Diagram

MODE 3. TIMER WITH CAPTURE REGISTER

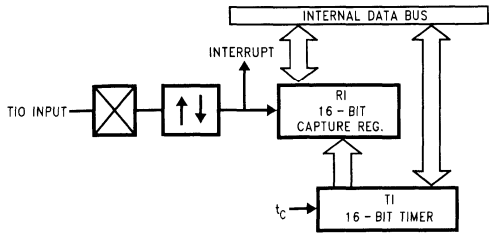
Timer T1 can be used to precisely measure external frequencies or events in this mode of operation. The timer T1 counts down at the instruction cycle rate. Upon the occurrence of a specified edge on the TIO pin the contents of the

TABLE V. Timer Operating Modes

CNTRL Bits 7 6 5	Operation Mode	T Interrupt	Timer Counter On
0 0 0	External Counter w/Auto-Load Reg.	Timer Underflow	TIO Pos. Edge
0 0 1	External Counter w/Auto-Load Reg.	Timer Underflow	TIO Neg. Edge
0 1 0	Not Allowed	Not Allowed	Not Allowed
0 1 1	Not Allowed	Not Allowed	Not Allowed
1 0 0	Timer w/Auto-Load Reg.	Timer Underflow	t_C
1 0 1	Timer w/Auto-Load Reg./Toggle TIO Out	Timer Underflow	t_C
1 1 0	Timer w/Capture Register	TIO Pos. Edge	t_C
1 1 1	Timer w/Capture Register	TIO Neg. Edge	t_C

Timer/Counter (Continued)

timer T1 are copied into the register R1. Bits in the control register CNTRL allow the trigger edge to be specified either as a positive edge or as a negative edge. In this mode the user can elect to be interrupted on the specified trigger edge *(Figure 9)*.

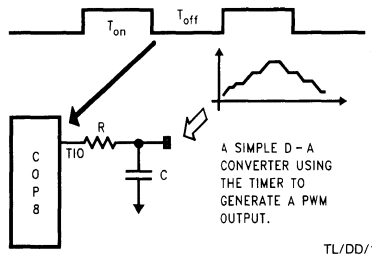


TL/DD/12862–10

FIGURE 9. Timer Capture Mode Block Diagram

TIMER PWM APPLICATION

Figure 10 shows how a minimal component D/A converter can be built out of the Timer-Register pair in the Auto-Reload mode. The timer is placed in the "Timer with auto reload" mode and the TIO pin is selected as the timer output. At the outset the TIO pin is set high, the timer T1 holds the on time and the register R1 holds the signal off time. Setting TRUN bit starts the timer which counts down at the instruction cycle rate. The underflow toggles the TIO output and copies the off time into the timer, which continues to run. By alternately loading in the on time and the off time at each successive interrupt a PWM frequency can be easily generated.



TL/DD/12862–11

FIGURE 10. Timer Application

WATCHDOG

The device has an on-board 8-bit WATCHDOG timer. The timer contains an 8-bit READ/WRITE down counter clocked by an 8-bit prescaler. Under software control the timer can be dedicated for the WATCHDOG or used as a general purpose counter. *Figure 11* shows the WATCHDOG timer block diagram.

MODE 1: WATCHDOG TIMER

The WATCHDOG is designed to detect user programs getting stuck in infinite loops resulting in loss of program control or "runaway" programs. The WATCHDOG can be enabled or disabled (only once) after the device is reset as a result of external reset. On power-up the WATCHDOG is disabled. The WATCHDOG is enabled by writing a "1" to WDREN bit (resides in WDREG register). Once enabled, the user program should write periodically into the 8-bit counter before the counter underflows. The 8-bit counter (WDCNT) is memory mapped at address 0CE Hex. The counter is loaded with n-1 to get n counts. The counter underflow resets the device, but does not disable the WATCHDOG. Loading the 8-bit counter initializes the prescaler with FF Hex and starts the prescaler/counter. Prescaler and counter are stopped upon counter underflow. Prescaler and counter are each loaded with FF Hex when the device goes into the HALT mode. The prescaler is used for crystal/resonator start-up when the device exits the HALT mode through Multi-Input Wakeup. In this case, the prescaler/counter contents are changed.

MODE 2: TIMER

In this mode, the prescaler/counter is used as a timer by keeping the WDREN (WATCHDOG reset enable) bit at 0. The counter underflow sets the WDUDF (underflow) bit and the underflow does not reset the device. Loading the 8-bit counter (load n-1 for n counts) sets the WDTEN bit (WATCHDOG Timer Enable) to "1", loads the prescaler with FF, and starts the timer. The counter underflow stops the timer. The WDTEN bit serves as a start bit for the WATCHDOG timer. This bit is set when the 8-bit counter is loaded by the user program. The load could be as a result of WATCHDOG service (WATCHDOG timer dedicated for WATCHDOG function) or write to the counter (WATCHDOG timer used as a general purpose counter). The bit is cleared upon Brown Out reset, WATCHDOG reset or external reset. The bit is not memory mapped and is transparent to the user program.

TABLE VI. WATCHDOG Control/Status

Parameter	HALT Mode	WD Reset	EXT Reset	Counter Load
8-Bit Prescaler	FF	FF	FF	FF
8-Bit WD Counter	FF	FF	FF	User Value
WDREN Bit	Unchanged	Unchanged	0	No Effect
WDUDF Bit	0	Unchanged	0	0
WDTEN Signal	Unchanged	0	0	1

WATCHDOG (Continued)

CONTROL/STATUS BITS

WDUDF: WATCHDOG Timer Underflow Bit

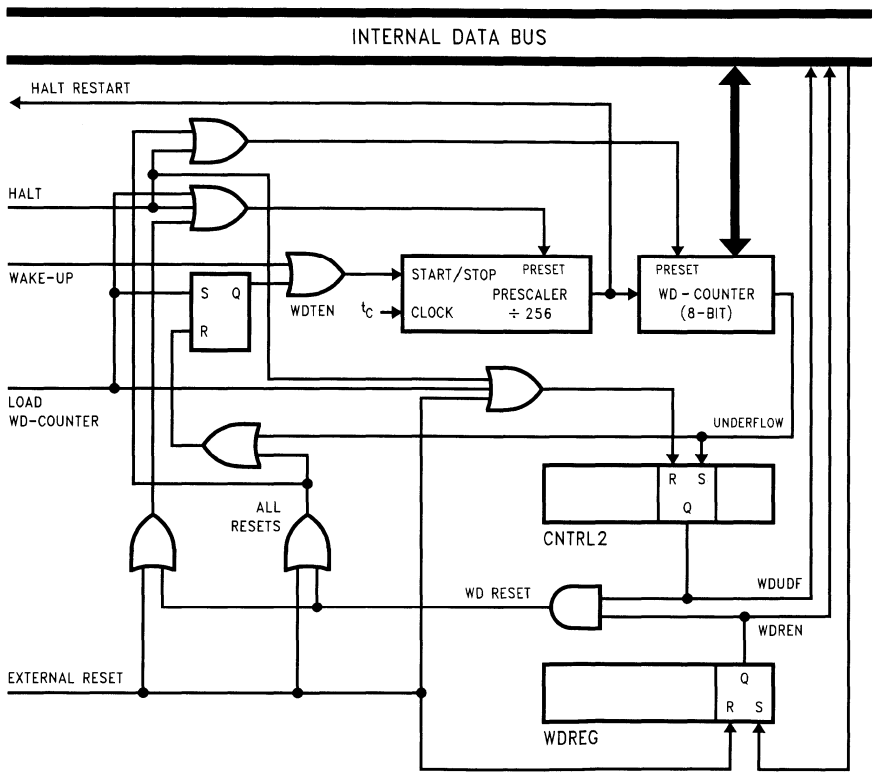
This bit resides in the CNTRL2 Register. The bit is set when the WATCHDOG timer underflows. The underflow resets the device if the WATCHDOG reset enable bit is set (WDREN = 1). Otherwise, WDUDF can be used as the timer underflow flag. The bit is cleared upon external reset, load to the 8-bit counter, or going into the HALT mode. It is a read only bit.

WDREN: WD Reset Enable

WDREN bit resides in a separate register (bit 0 of WDREG). This bit enables the WATCHDOG timer to generate a reset. The bit is cleared upon external reset. The bit under software control can be written to only once (once written to, the hardware does not allow the bit to be changed during program execution).

WDREN = 1 WATCHDOG reset is enabled.
WDREN = 0 WATCHDOG reset is disabled.

Table VI shows the impact of WATCHDOG Reset and External Reset on the Control/Status bits.



TL/DD/12862-12

FIGURE 11. WATCHDOG Timer Block Diagram

Modulator/Timer

The Modulator/Timer contains an 8-bit counter and an 8-bit autoreload register (MODRL address 0CF Hex). The Modulator/Timer has two modes of operation, selected by the control bit MC3. The Modulator/Timer Control bits MC1, MC2 and MC3 reside in CNTRL2 Register.

MODE 1: MODULATOR

The Modulator is used to generate high frequency pulses on the modulator output pin (L7). The L7 pin should be configured as an output. The number of pulses is determined by the 8-bit down counter. Under software control the modulator input clock can be either CKI or tC. The tC clock is derived by dividing down the oscillator clock by a factor of 10. Three control bits (MC1, MC2, and MC3) are used for the Modulator/Timer output control. When MC2 = 1 and MC3 = 1, CKI is used as the modulator input clock. When MC2 = 0, and MC3 = 1, tC is used as the modulator input clock. The user loads the counter with the desired number of counts (256 max) and sets MC1 to start the counter. The modulator autoreload register is loaded with n-1 to get n pulses. CKI or tc pulses are routed to the modulator output (L7) until the counter underflows *(Figure 12).* Upon underflow the hardware resets MC1 and stops the counter. The L7 pin goes low and stays low until the counter is restarted by the user program. The user program has the responsibility to timeout the low time. Unless the number of counts is changed, the user program does not have to load the counter each time the counter is started. The counter can simply be started by setting the MC1 bit. Setting MC1 by software will load the counter with the value of the autoreload register. The software can reset MC1 to stop the counter.

MODE 2: PWM TIMER

The counter can also be used as a PWM Timer. In this mode, an 8-bit register is used to serve as an autoreload register (MODRL).

a. 50% Duty Cycle:

When MC1 is 1 and MC2, MC3 are 0, a 50% duty cycle free running signal is generated on the L7 output pin *(Figure 13).* The L7 pin must be configured as an output pin. In this mode the 8-bit counter is clocked by tC. Setting the MC1 control bit by software loads the counter with the value of the autoreload register and starts the counter. The counter underflow toggles the (L7) output pin. The 50% duty cycle signal will be continuously generated until MC1 is reset by the user program.

b. Variable Duty Cycle:

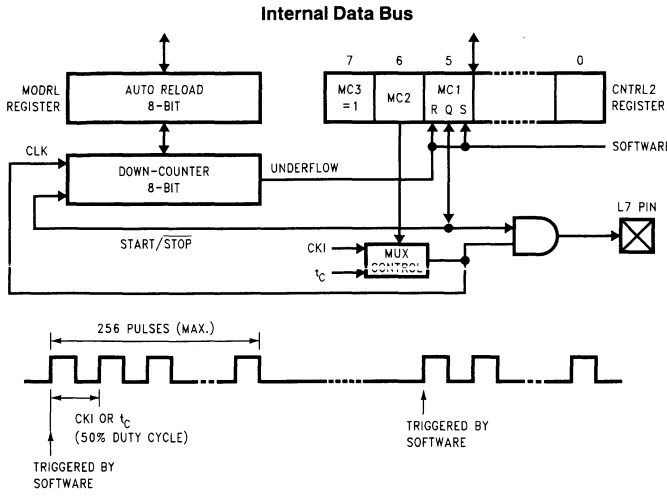
When MC3 = 0 and MC2 = 1, a variable duty cycle PWM signal is generated on the L7 output pin. The counter is clocked by tC. In this mode the 16-bit timer T1 along with the 8-bit down counter are used to generate a variable duty cycle PWM signal. The timer T1 underflow sets MC1 which starts the down counter and it also sets L7 high (L7 should be configured as an output).When the counter underflows the MC1 control bit is reset and the L7 output will go low until the next timer T1 underflow. Therefore, the width of the output pulse is controlled by the 8-bit counter and the pulse duration is controlled by the 16-bit timer T1 *(Figure 14).* Timer T1 must be configured in "PWM Mode/Toggle TIO Out" (CNTRL1 Bits 7,6,5 = 101).

Table VII shows the different operation modes for the Modulator/Timer.

TABLE VII. Modulator/Timer Modes

Control Bits in CNTRL2(00CC)			Operation Mode L7 Function
MC3	MC2	MC1	
0	0	0	Normal I/O
0	0	1	50% Duty Cycle Mode (Clocked by tc)
0	1	X	Variable Duty Cycle Mode (Clocked by tc) Using Timer 1 Underflow
1	0	X	Modulator Mode (Clocked by tc)
1	1	X	Modulator Mode (Clocked by CKI)

Note: MC1, MC2 and MC3 control bits are cleared upon reset.



TL/DD/12862-13

FIGURE 12. Mode 1: Modulator Block Diagram/Output Waveform

Modulator/Timer (Continued)

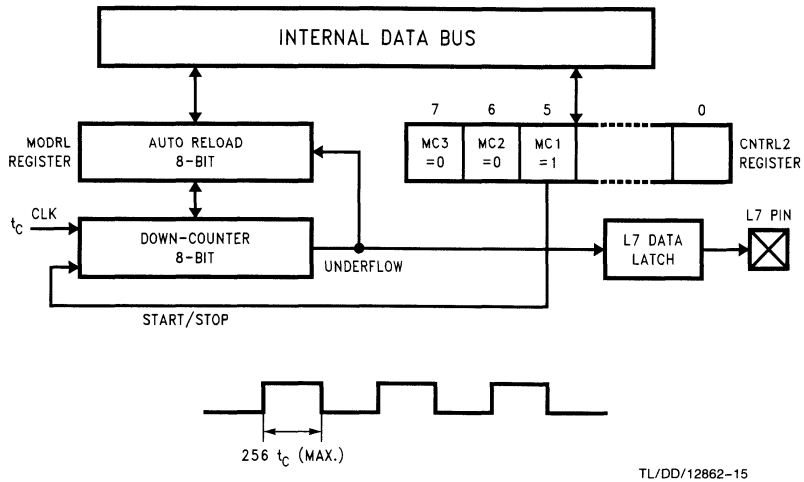


FIGURE 13. Mode 2a: 50% Duty Cycle Output

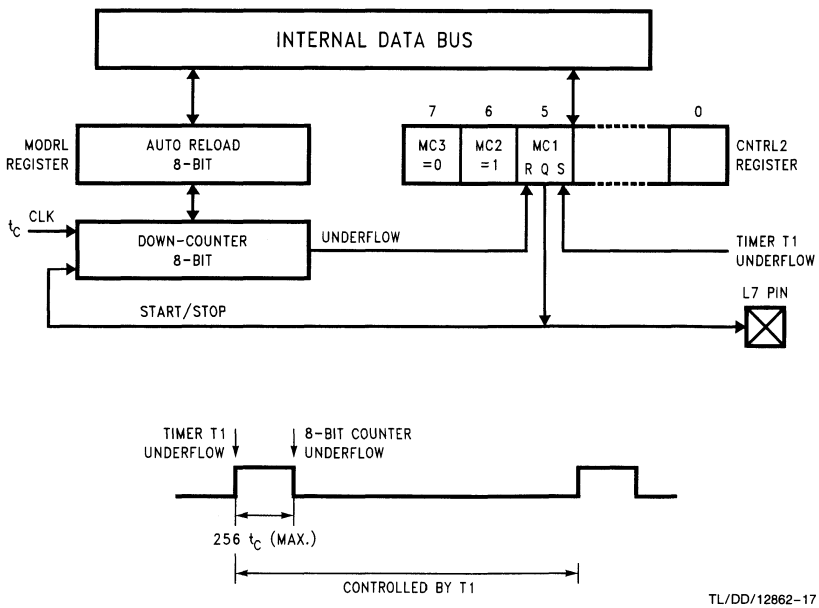


FIGURE 14. Mode 2b: Variable Duty Cycle Output

Comparator

The device has one differential comparator. Ports L0–L2 are used for the comparator. The output of the comparator is brought out to a pin. Port L has the following assignments:

L0 Comparator output
L1 Comparator negative input
L2 Comparator positive input

THE COMPARATOR STATUS/CONTROL BITS

These bits reside in the CNTRL2 Register (Address 0CC)

CMPEN Enables comparator ("1" = enable)

CMPRD Reads comparator output internally (CMPEN = 1, CMPOE=X)

CMPOE Enables comparator output to pin L0 ("1"=enable), CMPEN bit must be set to enable this function. If CMPEN=0, L0 will be 0.

The Comparator Select/Control bits are cleared on RESET (the comparator is disabled). To save power the program should also disable the comparator before the device enters the HALT mode.

The user program must set up L0, L1 and L2 ports correctly for comparator Inputs/Output: L1 and L2 need to be configured as inputs and L0 as output. Table VIII shows the DC and AC characteristics for the comparator.

Multi-Input Wake Up

The Multi-Input Wakeup feature is used to return (wakeup) the device from the HALT mode. *Figure 15* shows the Multi-Input Wakeup logic.

This feature utilizes the L Port. The user selects which particular L port bit or combination of L Port bits will cause the device to exit the HALT mode. Three 8-bit memory mapped registers, Reg:WKEN, Reg:WKEDG, and Reg:WKPND are used in conjunction with the L port to implement the Multi-Input Wakeup feature.

All three registers Reg:WKEN, Reg:WKPND, and Reg:WKEDG are read/write registers, and are cleared at reset, except WKPND. WKPND is unknown on reset.

The user can select whether the trigger condition on the selected L Port pin is going to be either a positive edge (low to high transition) or a negative edge (high to low transition). This selection is made via the Reg:WKEDG, which is an 8-bit control register with a bit assigned to each L Port pin. Setting the control bit will select the trigger condition to be a negative edge on that particular L Port pin. Resetting the bit selects the trigger condition to be a positive edge. Changing an edge select entails several steps in order to avoid a pseudo Wakeup condition as a result of the edge change. First, the associated WKEN bit should be reset, followed by the edge select change in WKEDG. Next, the associated WKPND bit should be cleared, followed by the associated WKEN bit being re-enabled.

An example may serve to clarify this procedure. Suppose we wish to change the edge select from positive (low going high) to negative (high going low) for L port bit 5, where bit 5 has previously been enabled for an input. The program would be as follows:

```
RBIT  5,WKEN
SBIT  5,WKEDG
RBIT  5,WKPND
SBIT  5,WKEN
```

If the L port bits have been used as outputs and then changed to inputs with Multi-Input Wakeup, a safety procedure should also be followed to avoid inherited pseudo wakeup conditions. After the selected L port bits have been changed from output to input but before the associated WKEN bits are enabled, the associated edge select bits in WKEDG should be set or reset for the desired edge selects, followed by the associated WKPND bits being cleared. This same procedure should be used following RESET, since the L port inputs are left floating as a result of RESET.

The occurrence of the selected trigger condition for Multi-Input Wakeup is latched into a pending register called Reg:WKPND. The respective bits of the WKPND register will be set on the occurrence of the selected trigger edge on the corresponding Port L pin. The user has the responsibility of clearing these pending flags. Since the Reg:WKPND is a pending register for the occurrence of selected wakeup conditions, the device will not enter the HALT mode if any Wakeup bit is both enabled and pending. Setting the G7 data bit under this condition will not allow the device to enter the HALT mode. Consequently, the user has the responsibility of clearing the pending flags before attempting to enter the HALT mode.

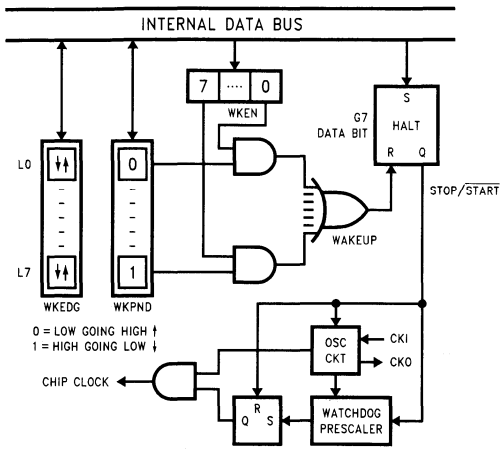
If a crystal oscillator is being used, the Wakeup signal will not start the chip running immediately since crystal oscillators have a finite start up time. The WATCHDOG timer prescaler generates a fixed delay to ensure that the oscillator has indeed stabilized before allowing the device to execute instructions. In this case, upon detecting a valid Wakeup signal only the oscillator circuitry and the WATCHDOG timer are enabled. The WATCHDOG timer prescaler is loaded with a value of FF Hex (256 counts) and is clocked from the tc instruction cycle clock. The tc clock is derived by dividing down the oscillator clock by a factor of 10. A Schmitt trigger following the CKI on chip inverter ensures that the WATCHDOG timer is clocked only when the oscillator has a sufficiently large amplitude to meet the Schmitt trigger specs. This Schmitt trigger is not part of the oscillator closed loop. The startup timeout from the WATCHDOG timer enables the clock signals to be routed to the rest of the chip.

TABLE VIII. DC and AC Characteristics (Note 1) $4.5V \le V_{CC} \le 5.5V$, $-40°C \le T_A \le +85°C$

Parameters	Conditioins	Min	Typ	Max	Units
Input Offset Voltage	$0.4V < V_{IN} < V_{CC} - 1.5V$		±10	±25	mV
Input Common Mode Voltage Range		0.4		$V_{CC} - 1.5$	V
Voltage Gain			300k		V/V
DC Supply Current (when enabled)	$V_{CC} = 5.5V$			250	μA
Response Time	TBD mV Step, TBD mV Overdrive, 100 pF Load			1	μs

Note 1: For comparator output current characteristics see L-Port specs.

Multi-Input Wakeup (Continued)



TL/DD/12862–18

FIGURE 15. Multi-Input Wakeup Logic

INTERRUPTS

The device has a sophisticated interrupt structure to allow easy interface to the real world. There are three possible interrupt sources, as shown below.

A maskable interrupt on external G0 input (positive or negative edge sensitive under software control)

A maskabie interrupt on timer carry or timer capture

A non-maskable software/error interrupt on opcode zero

INTERRUPT CONTROL

The GIE (global interrupt enable) bit enables the interrupt function. This is used in conjunction with ENI and ENTI to select one or both of the interrupt sources. This bit is reset when interrupt is acknowledged.

ENI and ENTI bits select external and timer interrupts respectively. Thus the user can select either or both sources to interrupt the microcontroller when GIE is enabled.

IEDG selects the external interrupt edge (0 = rising edge, 1 = falling edge). The user can get an interrupt on both rising and falling edges by toggling the state of IEDG bit after each interrupt.

IPND and TPND bits signal which interrupt is pending. After an interrupt is acknowledged, the user can check these two bits to determine which interrupt is pending. This permits the interrupts to be prioritized under software. The pending flags have to be cleared by the user. Setting the GIE bit high inside the interrupt subroutine allows nested interrupts.

The software interrupt does not reset the GIE bit. This means that the controller can be interrupted by other interrupt sources while servicing the software interrupt.

INTERRUPT PROCESSING

The interrupt, once acknowledged, pushes the program counter (PC) onto the stack and the stack pointer (SP) is decremented twice. The Global Interrupt Enable (GIE) bit is reset to disable further interrupts. The microcontroller then vectors to the address 00FFH and resumes execution from that address. This process takes 7 cycles to complete. At the end of the interrupt subroutine, any of the following three instructions return the processor back to the main program: RET, RETSK or RETI. Either one of the three instructions will pop the stack into the program counter (PC). The stack pointer is then incremented twice. The RETI instruction additionally sets the GIE bit to re-enable further interrupts.

Any of the three instructions can be used to return from a hardware interrupt subroutine. The RETSK instruction should be used when returning from a software interrupt subroutine to avoid entering an infinite loop.

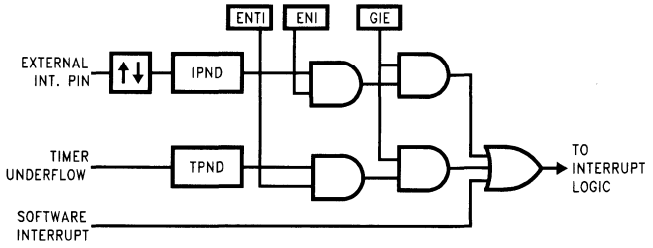
Note: There is always the possibility of an interrupt occurring during an instruction which is attempting to reset the GIE bit or any other interrupt enable bit. If this occurs when a single cycle instruction is being used to reset the interrupt enable bit, the interrupt enable bit will be reset but an interrupt may still occur. This is because interrupt processing is started at the same time as the interrupt bit is being reset. To avoid this scenario, the user should always use a two, three, or four cycle instruction to reset interrupt enable bits.

DETECTION OF ILLEGAL CONDITIONS

The device incorporates a hardware mechanism that allows it to detect illegal conditions which may occur from coding errors, noise, and "brown out" voltage drop situations. Specifically, it detects cases of executing out of undefined ROM area and unbalanced tack situations.

Reading an undefined ROM location returns 00 (hexadecimal) as its contents. The opcode for a software interrupt is also "00". Thus a program accessing undefined ROM will cause a software interrupt.

Reading an undefined RAM location returns an FF (hexadecimal). The subroutine stack on the device grows down for each subroutine call. By initializing the stack pointer to the top of RAM, the first unbalanced return instruction will cause the stack pointer to address undefined RAM. As a result the program will attempt to execute from FFFF (hexadecimal), which is an undefined ROM location and will trigger a software interrupt.



TL/DD/12862–19

FIGURE 16. Interrupt Block Diagram

Control Registers

CNTRL1 REGISTER (ADDRESS 00EE)

The Timer and MICROWIRE control register contains the following bits:

SL1 and SL0 Select the MICROWIRE clock divide-by (00 = 2, 01 = 4, 1x = 8)

IEDG External interrupt edge polarity select

MSEL Selects G5 and G4 as MICROWIRE signals SK and SO respectively

TRUN Used to start and stop the timer/counter (1 = run, 0 = stop)

TC1 Timer T1 Mode Control Bit

TC2 Timer T1 Mode Control Bit

TC3 Timer T1 Mode Control Bit

Bit 7							Bit 0
TC1	TC2	TC3	TRUN	MSEL	IEDG	SL1	SL0

PSW REGISTER (ADDRESS 00EF)

The PSW register contains the following select bits:

GIE Global interrupt enable (enables interrupts)

ENI External interrupt enable

BUSY MICROWIRE busy shifting flag

IPND External interrupt pending

ENTI Timer T1 interrupt enable

TPND Timer T1 interrupt pending (timer Underflow or capture edge)

C Carry Flip/Flop

HC Half-Carry Flip/Flop

Bit 7							Bit 0
HC	C	TPND	ENTI	IPND	BUSY	ENI	GIE

The Half-Carry bit is also effected by all the instructions that effect the Carry flag. The flag values depend upon the instruction. For example, after executing the ADC instruction the values of the Carry and the Half-Carry flag depend upon the operands involved. However, instructions like SET C and RESET C will set and clear both the carry flags. Table XIII lists the instructions that effect the HC and the C flags.

TABLE XIII. Instructions Effecting HC and C Flags

Instr.	HC Flag	C Flag
ADC	Depends on Operands	Depends on Operands
SUBC	Depends on Operands	Depends on Operands
SET C	Set	Set
RESET C	Set	Set
RRC	Depends on Operands	Depends on Operands

CNTRL2 REGISTER (ADDRESS 00CC)

Bit 7							Bit 0
MC3	MC2	MC1	CMPEN	CMPRD	CMPOE	WDUDF	unused
R/W	R/W	R/W	R/W	R/O	R/W	R/O	

MC3 Modulator/Timer Control Bit

MC2 Modulator/Timer Control Bit

MC1 Modulator/Timer Control Bit

CMPEN Comparator Enable Bit

CMPRD Comparator Read Bit

CMPOE Comparator Output Enable Bit

WDUDF WATCHDOG Timer Underflow Bit (Read Only)

WDREN REGISTER (ADDRESS 00CD)

WDREN WATCHDOG Reset Enable Bit (Write Once Only)

Bit 7	Bit 0
UNUSED	WDREN

Memory Map

All RAM, ports and registers (except A and PC) are mapped into data memory address space.

TABLE IX. Memory Map

Address	Contents
00 to 6F	On-chip RAM bytes (112 bytes)
70 to 7F	Unused RAM Address Space (Reads as All Ones)
80 to BF	Expansion Space for On-Chip EERAM (Reads Undefined Data)
C0 to C7	Reserved
C8	MIWU Edge Select Register (Reg:WKEDG)
C9	MIWU Enable Register (Reg:WKEN)
CA	MIWU Pending Register (Reg:WKPND)
CB	Reserved
CC	Control2 Register (CNTRL2)
CD	WATCHDOG Register (WDREG)
CE	WATCHDOG Counter (WDCNT)
CF	Modulator Reload (MODRL)
D0	Port L Data Register
D1	Port L Configuration Register
D2	Port L Input Pins (Read Only)
D3	Reserved for Port L
D4	Port G Data Register
D5	Port G Configuration Register
D6	Port G Input Pins (Read Only)
D7	Port I Input Pins (Read Only)
D8 to DB	Reserved for Port C
DC	Port D Data Register
DD to DF	Reserved for Port D
E0 to EF	On-Chip Functions and Registers
E0 to E7	Reserved for Future Parts
E8	Reserved
E9	MICROWIRE Shift Register
EA	Timer Lower Byte
EB	Timer Upper Byte
EC	Timer1 Autoreload Register Lower Byte
ED	Timer1 Autoreload Register Upper Byte
EE	CNTRL1 Control Register
EF	PSW Register
F0 to FF	On-Chip RAM Mapped as Registers
FC	X Register
FD	SP Register
FE	B Register

Reading other unused memory locations will return undefined data.

Addressing Modes

There are ten addressing modes, six for operand addressing and four for transfer of control.

OPERAND ADDRESSING MODES

REGISTER INDIRECT

This is the "normal" addressing mode for the chip. The operand is the data memory addressed by the **B** or **X** pointer.

REGISTER INDIRECT WITH AUTO POST INCREMENT OR DECREMENT

This addressing mode is used with the LD and X instructions. The operand is the data memory addressed by the **B** or **X** pointer. This is a register indirect mode that automatically post increments or post decrements the **B** or **X** pointer after executing the instruction.

DIRECT

The instruction contains an 8-bit address field that directly points to the data memory for the operand.

IMMEDIATE

The instruction contains an 8-bit immediate field as the operand.

SHORT IMMEDIATE

This addressing mode issued with the LD B,# instruction, where the immediate # is less than 16. The instruction contains a 4-bit immediate field as the operand.

INDIRECT

This addressing mode is used with the LAID instruction. The contents of the accumulator are used as a partial address (lower 8 bits of PC) for accessing a data operand from the program memory.

TRANSFER OF CONTROL ADDRESSING MODES

RELATIVE

This mode is used for the JP instruction with the instruction field being added to the program counter to produce the next instruction address. JP has a range from −31 to +32 to allow a one byte relative jump (JP + 1 is implemented by a NOP instruction). There are no "blocks" or "pages" when using JP since all 15 bits of the PC are used.

ABSOLUTE

This mode is used with the JMP and JSR instructions with the instruction field of 12 bits replacing the lower 12 bits of the program counter (PC). This allows jumping to any location in the current 4k program memory segment.

ABSOLUTE LONG

This mode is used with the JMPL and JSRL instructions with the instruction field of 15 bits replacing the entire 15 bits of the program counter (PC). This allows jumping to any location up to 32k in the program memory space.

INDIRECT

This mode is used with the JID instruction. The contents of the accumulator are used as a partial address (lower 8 bits of PC) for accessing a location in the program memory. The contents of this program memory location serves as a partial address (lower 8 bits of PC) for the jump to the next instruction.

Instruction Set

REGISTER AND SYMBOL DEFINITIONS

Registers

A	8-bit Accumulator register
B	8-bit Address register
X	8-bit Address register
SP	8-bit Stack pointer register
PC	15-bit Program counter register
PU	upper 7 bits of PC
PL	lower 8 bits of PC
C	1-bit of PSW register for carry
HC	Half Carry
GIE	1-bit of PSW register for global interrupt enable

Symbols

[B]	Memory indirectly addressed by B register
[X]	Memory indirectly addressed by X register
Mem	Direct address memory or [B]
Meml	Direct address memory or [B] or Immediate data
Imm	8-bit Immediate data
Reg	Register memory: addresses F0 to FF (Includes B, X and SP)
Bit	Bit number (0 to 7)
←	Loaded with
↔	Exchanged with

Instruction Set

ADD	add	A ← A + Meml
ADC	add with carry	A ← A + Meml + C, C ← Carry HC ← Half Carry
SUBC	subtract with carry	A ← A + $\overline{\text{Meml}}$ + C, C ← Carry HC ← Half Carry
AND	Logical AND	A ← A and Meml
OR	Logical OR	A ← A or Meml
XOR	Logical Exclusive-OR	A ← A xor Meml
IFEQ	IF equal	Compare A and Meml, Do next if A = Meml
IFGT	IF greater than	Compare A and Meml, Do next if A > Meml
IFBNE	IF B not equal	Do next if lower 4 bits of B ≠ Imm
DRSZ	Decrement Reg. ,skip if zero	Reg ← Reg − 1, skip if Reg goes to 0
SBIT	Set bit	1 to bit, Mem (bit = 0 to 7 immediate)
RBIT	Reset bit	0 to bit, Mem
IFBIT	If bit	If bit, Mem is true, do next instr.
X	Exchange A with memory	A ↔ Mem
LD A	Load A with memory	A ← Meml
LD mem	Load Direct memory Immed.	Mem ← Imm
LD Reg	Load Register memory Immed.	Reg ← Imm
X	Exchange A with memory [B]	A ↔ [B] (B ← B ± 1)
X	Exchange A with memory [X]	A ↔ [X] (X ← X ± 1)
LD A	Load A with memory [B]	A ← [B] (B ← B ± 1)
LD A	Load A with memory [X]	A ← [X] (X ← X ± 1)
LD M	Load Memory Immediate	[B] ← Imm (B ← B ± 1)
CLRA	Clear A	A ← 0
INCA	Increment A	A ← A + 1
DECA	Decrement A	A ← A − 1
LAID	Load A indirect from ROM	A ← ROM(PU,A)
DCORA	DECIMAL CORRECT A	A ← BCD correction (follows ADC, SUBC)
RRCA	ROTATE A RIGHT THRU C	C → A7 → . . . → A0 → C
SWAPA	Swap nibbles of A	A7 . . . A4 ↔ A3 . . . A0
SC	Set C	C ← 1, HC ← 1
RC	Reset C	C ← 0, HC ← 0
IFC	If C	If C is true, do next instruction
IFNC	If not C	If C is not true, do next instruction
JMPL	Jump absolute long	PC ← ii (ii = 15 bits, 0 to 32K)
JMP	Jump absolute	PC11..0 ← i (i = 12 bits)
JP	Jump relative short	PC ← PC + r (r is −31 to +32, not 1)
JSRL	Jump subroutine long	[SP] ← PL,[SP-1] ← PU,SP-2,PC ← ii
JSR	Jump subroutine	[SP] ← PL,[SP-1] ← PU,SP-2,PC11..0 ← i
JID	Jump indirect	PL ← ROM(PU,A)
RET	Return from subroutine	SP+2,PL ← [SP],PU ← [SP-1]
RETSK	Return and Skip	SP+2,PL ← [SP],PU ← [SP-1],Skip next instruction
RETI	Return from Interrupt	SP+2,PL ← [SP],PU ← [SP-1],GIE ← 1
INTR	Generate an interrupt	[SP] ← PL,[SP−1] ← PU,SP-2,PC ← 0FF
NOP	No operation	PC ← PC + 1

Bits 7–4

F	E	D	C	B	A	9	8	7	6	5	4	3	2	1	0	
JP -15	JP -31	LD 0F0, #i	DRSZ 0F0	RRCA	RC	ADC A, #i	ADC A, [B]	IFBIT 0,[B]	*	LD B, 0F	IFBNE 0	JSR 0000–00FF	JMP 0000–00FF	JP + 17	INTR	0
JP -14	JP -30	LD 0F1, #i	DRSZ 0F1	*	SC	SUBC A, #i	SUBC A,[B]	IFBIT 1,[B]	*	LD B, 0E	IFBNE 1	JSR 0100–01FF	JMP 0100–01FF	JP + 18	JP + 2	1
JP -13	JP -29	LD 0F2, #i	DRSZ 0F2	X A, [X+]	X A, [B+]	IFEQ A, #i	IFEQ A,[B]	IFBIT 2,[B]	*	LD B, 0D	IFBNE 2	JSR 0200–02FF	JMP 0200–02FF	JP + 19	JP + 3	2
JP -12	JP -28	LD 0F3, #i	DRSZ 0F3	X A, [X−]	X A, [B−]	IFGT A, #i	IFGT A,[B]	IFBIT 3,[B]	*	LD B, 0C	IFBNE 3	JSR 0300–03FF	JMP 0300–03FF	JP + 20	JP + 4	3
JP -11	JP -27	LD 0F4, #i	DRSZ 0F4	*	LAID	ADD A, #i	ADD A,[B]	IFBIT 4,[B]	CLRA	LD B, 0B	IFBNE 4	JSR 0400–04FF	JMP 0400–04FF	JP + 21	JP + 5	4
JP -10	JP -26	LD 0F5, #i	DRSZ 0F5	*	JID	AND A, #i	AND A,[B]	IFBIT 5,[B]	SWAPA	LD B, 0A	IFBNE 5	JSR 0500–05FF	JMP 0500–05FF	JP + 22	JP + 6	5
JP -9	JP -25	LD 0F6, #i	DRSZ 0F6	X A, [X]	X A, [B]	XOR A, #i	XOR A,[B]	IFBIT 6,[B]	DCORA	LD B, 9	IFBNE 6	JSR 0600–06FF	JMP 0600–06FF	JP + 23	JP + 7	6
JP -8	JP -24	LD 0F7, #i	DRSZ 0F7	*	*	OR A, #i	OR A,[B]	IFBIT 7,[B]	*	LD B, 8	IFBNE 7	JSR 0700–07FF	JMP 0700–07FF	JP + 24	JP + 8	7
JP -7	JP -23	LD 0F8, #i	DRSZ 0F8	NOP	*	LD A, #i	IFC	SBIT 0,[B]	RBIT 0,[B]	LD B, 7	IFBNE 8	JSR 0800–08FF	JMP 0800–08FF	JP + 25	JP + 9	8
JP -6	JP -22	LD 0F9, #i	DRSZ 0F9	*	*	*	IFNC	SBIT 1,[B]	RBIT 1,[B]	LD B, 6	IFBNE 9	JSR 0900–09FF	JMP 0900–09FF	JP + 26	JP + 10	9
JP -5	JP -21	LD 0FA, #i	DRSZ 0FA	LD A, [X+]	LD A, [B+]	LD [B+], #i	INCA	SBIT 2,[B]	RBIT 2,[B]	LD B, 5	IFBNE 0A	JSR 0A00–0AFF	JMP 0A00–0AFF	JP + 27	JP + 11	A
JP -4	JP -20	LD 0FB, #i	DRSZ 0FB	LD A, [X−]	LD A, [B−]	LD [B−], #i	DECA	SBIT 3,[B]	RBIT 3,[B]	LD B, 4	IFBNE 0B	JSR 0B00–0BFF	JMP 0B00–0BFF	JP + 28	JP + 12	B
JP -3	JP -19	LD 0FC, #i	DRSZ 0FC	LD Md, #i	JMPL	X A,Md	*	SBIT 4,[B]	RBIT 4,[B]	LD B, 3	IFBNE 0C	JSR 0C00–0CFF	JMP 0C00–0CFF	JP + 29	JP +13	C
JP -2	JP -18	LD 0FD, #i	DRSZ 0FD	DIR	JSRL	LD A, Md	RETSK	SBIT 5,[B]	RBIT 5,[B]	LD B, 2	IFBNE 0D	JSR 0D00–0DFF	JMP 0D00–0DFF	JP + 30	JP +14	D
JP -1	JP -17	LD 0FE, #i	DRSZ 0FE	LD A, [X]	LD A, [B]	LD [B], #i	RET	SBIT 6, [B]	RBIT 6, [B]	LD B, 1	IFBNE 0E	JSR 0E00–0EFF	JMP 0E00–0EFF	JP + 31	JP +15	E
JP -0	JP -16	LD 0FF, #1	DRSZ 0FF	*	*	*	RETI	SBIT 7,[B]	RBIT 7,[B]	LD B, 0	IFBNE 0F	JSR 0F00–0FFF	JMP 0F00–0FFF	JP + 32	JP + 16	F

Bits 3–0

where, i is the immediate data Md is a directly addressed memory location * is an unused opcode (see following table)

Instruction Execution Time

Most instructions are single byte (with immediate addressing mode instruction taking two bytes).

Most single instructions take one cycle time to execute.

Skipped instructions require x number of cycles to be skipped, where x equals the number of bytes in the skipped instruction opcode.

See the BYTES and CYCLES per INSTRUCTION table for details.

BYTES and CYCLES per INSTRUCTION

The following table shows the number of bytes and cycles for each instruction in the format of byte/cycle.

Arithmetic Instructions (Bytes/Cycles)

	[B]	Direct	Immed.
ADD	1/1	3/4	2/2
ADC	1/1	3/4	2/2
SUBC	1/1	3/4	2/2
AND	1/1	3/4	2/2
OR	1/1	3/4	2/2
XOR	1/1	3/4	2/2
IFEQ	1/1	3/4	2/2
IFGT	1/1	3/4	2/2
IFBNE	1/1		
DRSZ		1/3	
SBIT	1/1	3/4	
RBIT	1/1	3/4	
IFBIT	1/1	3/4	

Memory Transfer Instructions (Bytes/Cycles)

	Register Indirect		Direct	Immed.	Register Indirect Auto Incr & Decr		
	[B]	[X]			[B+, B−]	[X+, X−]	
X A,*	1/1	1/3	2/3		1/2	1/3	
LD A,*	1/1	1/3	2/3	2/2	1/2	1/3	
LD B,Imm				1/1			(If B < 16)
LD B,Imm				2/3			(If B > 15)
LD Mem,Imm			3/3		2/2		
LD Reg,Imm				2/3			

* => Memory location addressed by B or X or directly.

Instructions Using A & C

Instructions	Bytes/Cycles
CLRA	1/1
INCA	1/1
DECA	1/1
LAID	1/3
DCORA	1/1
RRCA	1/1
SWAPA	1/1
SC	1/1
RC	1/1
IFC	1/1
IFNC	1/1

Transfer of Control Instructions

Instructions	Bytes/Cycles
JMPL	3/4
JMP	2/3
JP	1/3
JSRL	3/5
JSR	2/5
JID	1/3
RET	1/5
RETSK	1/5
RETI	1/5
INTR	1/7
NOP	1/1

Development Support

SUMMARY

- iceMASTER™: IM-COP8/400—Full feature in-circuit emulation for all COP8 products. A full set of COP8 Basic and Feature Family device and package specific probes are available.
- COP8 Debug Module: Moderate cost in-circuit emulation and development programming unit.
- COP8 Evaluation and Programming Unit: EPU-COP880C—low cost In-circuit simulation and development programming unit.
- Assembler: COP8-DEV-IBMA. A DOS installable cross development Assembler, Linker, Librarian and Utility Software Development Tool Kit.
- C Compiler: COP8C. A DOS installable cross development Software Tool Kit.
- OTP/EPROM Programmer Support: Covering needs from engineering prototype, pilot production to full production environments.

iceMASTER (IM) IN-CIRCUIT EMULATION

The iceMASTER IM-COP8/400 is a full feature, PC based, in-circuit emulation tool developed and marketed by MetaLink Corporation to support the whole COP8 family of products. National is a resale vendor for these products.

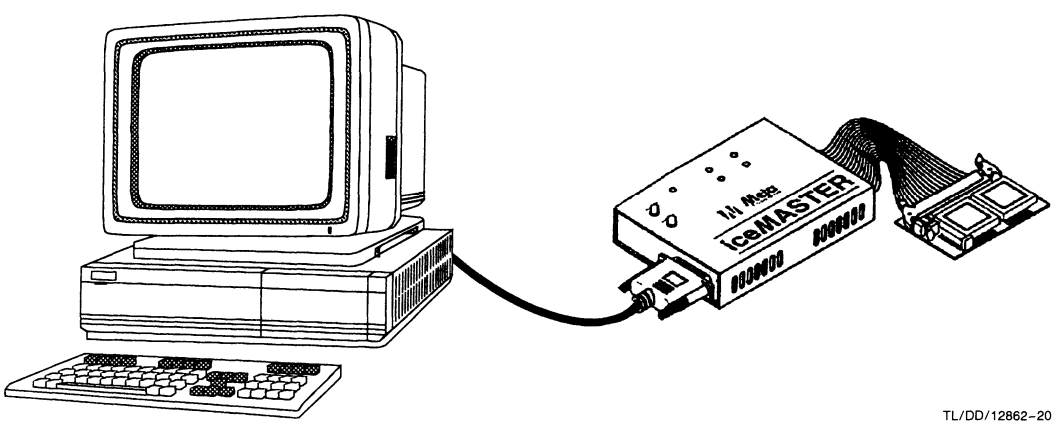
See *Figure 17* for configuration.

The iceMASTER IM-COP8/400 with its device specific COP8 Probe provides a rich feature set for developing, testing and maintaining product:

- Real-time in-circuit emulation; full 2.4VDC–5.5VDC operation range, full DC-10 MHz clock. Chip options are programmable or jumper selectable.
- Direct connection to application board by package compatible socket or surface mount assembly.
- Full 32 kbytes of loadable programming space that overlays (replaces) the on-chip ROM or EPROM. On-chip RAM and I/O blocks are used directly or recreated on the probe as necessary.
- Full 4k frame synchronous trace memory. Address, instruction, and 8 unspecified, circuit connectable trace lines. Display can be HLL source (e.g., C source), assembly or mixed.
- A full 64k hardware configurable break, trace on, trace off control, and pass count increment events.
- Tool set integrated interactive symbolic debugger—supports both assembler (COFF) and C Compiler (.COD) linked object formats.
- Real time performance profiling analysis; selectable bucket definition.
- Watch windows, content updated automatically at each execution break.
- Instruction by instruction memory/register changes displayed on source window when in single step operation.
- Single base unit and debugger software reconfigurable to support the entire COP8 family; only the probe personality needs to change. Debugger software is processor customized, and reconfigured from a master model file.
- Processor specific symbolic display of registers and bit level assignments, configured from master model file.
- Halt/Idle mode notification.
- On-line HELP customized to specific processor using master model file.
- Includes a copy of COP8-DEV-IBMA assembler and linker SDK.

IM Order Information

Base Unit	
IM-COP8/400-1	iceMASTER Base Unit, 110V Power Supply
IM-COP8/400-2	iceMASTER Base Unit, 220V Power Supply
iceMASTER Probe	
MHW-840RJ28DWPC MHW-840RJ20DWPC	28 DIP 20 DIP
MHW-SOIC28 MHW-SOIC20	28 SOIC Adapter Kit 20 SOIC Adapter Kit



TL/DD/12862–20

FIGURE 17. COP8 iceMASTER Environment

Development Support (Continued)

iceMASTER DEBUG MODULE (DM)

The iceMASTER Debug Module is a PC based, combination in-circuit emulation tool and COP8 based OTP/EPROM programming tool developed and marketed by MetaLink Corporation to support the whole COP8 family of products. National is a resale vendor for these products.

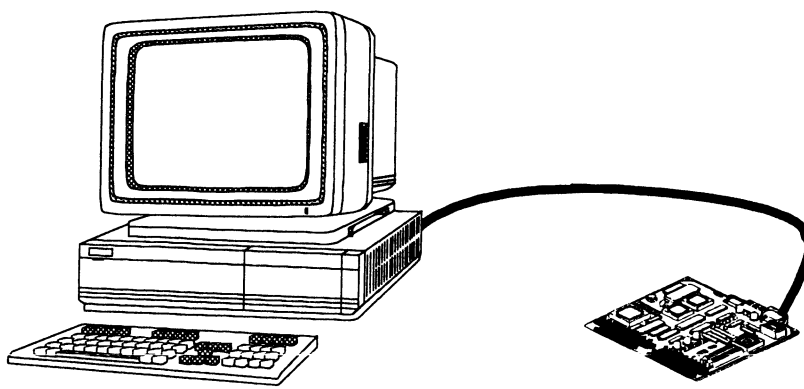
See *Figure 18* for configuration.

The iceMASTER Debug Module is a moderate cost development tool. It has the capability of in-circuit emulation for a specific COP8 microcontroller and in addition serves as a programming tool for COP8 OTP and EPROM product families. Summary of features is as follows:

- Real-time in-circuit emulation; full operating voltage range operation, full DC-10 MHz clock.
- All processor I/O pins can be cabled to an application development board with package compatible cable to socket and surface mount assembly.
- Full 32 kbytes of loadable programming space that overlays (replaces) the on-chip ROM or EPROM. On-chip RAM and I/O blocks are used directly or recreated as necessary.
- 100 frames of synchronous trace memory. The display can be HLL source (C source), assembly or mixed. The most recent history prior to a break is available in the trace memory.
- Configured break points; uses INTR instruction which is modestly intrusive.
- Software—only supported features are selectable.
- Tool set integrated interactive symbolic debugger—supports both assembler (COFF) and C Compiler (.COD) SDK linked object formats.
- Instruction by instruction memory/register changes displayed when in single step operation.
- Debugger software is processor customized, and reconfigured from a master model file.
- Processor specific symbolic display of registers and bit level assignments, configured from master model file.
- Halt/Idle mode notification.
- Programming menu supports full product line of programmable OTP and EPROM COP8 products. Program data is taken directly from the overlay RAM.
- Programming of 44 PLCC and 68 PLCC parts requires external programming adapters.
- Includes wall mount power supply.
- On-board V_{PP} generator from 5V input or connection to external supply supported. Requires V_{PP} level adjustment per the family programming specification (correct level is provided on an on-screen pop-down display).
- On-line HELP customized to specific processor using master model file.
- Includes a copy of COP8-DEV-IBMA assembler and linker SDK.

DM Order Information

Debug Model Unit	
COP8-DM/840CJ	
Cable Adapters	
DM-COP8/28D	28 DIP Cable
DM-COP8/28D-SO	28 DIP to 28 SOIC Adapter
DM-COP8/20D	20 DIP Cable
DM-COP8/20D-SO	20 DIP to 20 SOIC Adapter



TL/DD/12862-21

FIGURE 18. COP8-DM Environment

Development Support (Continued)

COP8 ASSEMBLER/LINKER SOFTWARE DEVELOPMENT TOOL KIT

National Semiconductor offers a relocatable COP8 macro cross assembler, linker, librarian and utility software development tool kit. Features are summarized as follows:

- Basic and Feature Family instruction set by "device" type.
- Nested macro capability.
- Extensive set of assembler directives.
- Supported on PC/DOS platform.
- Generates National standard COFF output files.
- Integrated Linker and Librarian.
- Integrated utilities to generate ROM code file outputs.
- DUMPCOFF utility.

This product is integrated as a part of MetaLink tools as a development kit, fully supported by the MetaLink debugger. It may be ordered separately or it is bundled with the MetaLink products at no additional cost.

Order Information

Assembler SDK:	
COP8-DEV-IBMA	Assembler SDK on installable 3.5" PC/DOS Floppy Disk Drive format. Periodic upgrades and most recent version is available on National's BBS and Internet.

COP8 C COMPILER

A C Compiler is developed and marketed by Byte Craft Limited. The COP8C compiler is a fully integrated development tool specifically designed to support the compact embedded configuration of the COP8 family of products.

Features are summarized as follows:

- ANSI C with some restrictions and extensions that optimize development for the COP8 embedded application.
- BITS data type extension. Register declaration #pragma with direct bit level definitions.
- C language support for interrupt routines.
- Expert system, rule based code generation and optimization.
- Performs consistency checks against the architectural definitions of the target COP8 device.
- Generates program memory code.
- Supports linking of compiled object or COP8 assembled object formats.
- Global optimization of linked code.
- Symbolic debug load format fully sourced level supported by the MetaLink debugger.

INDUSTRY WIDE OTP/EPROM PROGRAMMING SUPPORT

Programming support, in addition to the MetaLink development tools, is provided by a full range of independent approved vendors to meet the needs from the engineering laboratory to full production.

Approved List

Manufacturer	North America	Europe	Asia
BP Microsystems	(800) 225-2102 (713) 688-4600 Fax: (713) 688-0920	+49-8152-4183 +49-8856-932616	+852-234-16611 +852-2710-8121
Data I/O	(800) 426-1045 (206) 881-6444 Fax: (206) 882-1043	+44-0734-440011	Call North America
HI-LO	(510) 623-8860	Call Asia	+886-2-764-0215 Fax: +886-2-756-6403
ICE Technology	(800) 624-8949 (919) 430-7915	+44-1226-767404 Fax: 0-1226-370-434	
MetaLink	(800) 638-2423 (602) 926-0797 Fax: (602) 693-0681	+49-80 9156 96-0 Fax: +49-80 9123 86	+852-737-1800
Systems General	(408) 263-6667	+41-1-9450300	+886-2-917-3005 Fax: +886-2-911-1283
Needhams	(916) 924-8037 Fax: (916) 924-8065		

Development Support (Continued)

AVAILABLE LITERATURE

For more information, please see the *COP8 Basic Family User's Manual,* Literature Number 620895, *COP8 Feature Family User's Manual,* Literature Number 620897 and *National's Family of 8-Bit Microcontrollers COP8 Selection Selection Guide,* Literature Number 630009.

DIAL-A-HELPER SERVICE

Dial-A-Helper is a service provided by the Microcontroller Applications Group. The Dial-A-Helper is an Electronic Information System that may be accessed as a Bulletin Board System (BBS) via data modem, as an FTP site on the Internet via standard FTP client application or as an FTP site on the Internet using a standard Internet browser such as Netscape or Mosaic.

The Dial-A-Helper system provides access to an automated information storage and retrieval system. The system capabilities include a MESSAGE SECTION (electronic mail, when accessed as a BBS) for communications to and from the Microcontroller Applications Group and a FILE SECTION which consists of several file areas where valuable application software and utilities could be found.

DIAL-A-HELPER BBS VIA A STANDARD MODEM

Modem: CANADA/U.S.: (800) NSC-MICRO
(800) 672-6427
EUROPE: (+49) 0-8141-351332
Baud: 14.4k
Set-Up: Length: 8-Bit
Parity: None
Stop Bit: 1
Operation: 24 Hours, 7 Days

DIAL-A-HELPER VIA FTP

ftp nscmicro.nsc.com
user: anonymous
password: username@yourhost.site.domain

DIAL-A-HELPER VIA A WORLDWIDE WEB BROWSER

ftp://nscmicro.nsc.com

National Semiconductor on the WorldWide Web

See us on the WorldWide Web at: http://www.national.com

CUSTOMER RESPONSE CENTER

Complete product information and technical support is available from National's customer response centers.

CANADA/U.S.:	Tel:		(800) 272-9959
	email:		support@tevm2.nsc.com
EUROPE:	email:		europe.support@nsc.com
	Deutsch	Tel:	+49 (0) 180-530 85 85
	English	Tel:	+49 (0) 180-532 78 32
	Français	Tel:	+49 (0) 180-532 93 58
	Italiano	Tel:	+49 (0) 180-534 16 80
JAPAN:	Tel:		+81-043-299-2309
S.E. ASIA:	Beijing	Tel:	(+86) 10-6856-8601
	Shanghai	Tel:	(+86) 21-6415-4092
	Hong Kong	Tel:	(+852) 2737-1600
	Korea	Tel:	(+82) 2-3771-6909
	Malaysia	Tel:	(+60-4) 644-9061
	Singapore	Tel:	(+65) 255-2226
	Taiwan	Tel:	+886-2-521-3288
AUSTRALIA:	Tel:		(+61) 3-9558-9999
INDIA:	Tel:		(+91) 80-559-9467

National Semiconductor

PRELIMINARY

COP87L88RK/COP87L84RK
8-Bit One Time Programmable (OTP) Microcontroller with Analog Function Block and 32 kbytes of Program Memory

General Description

The COP87L88RK/COP87L84RK are members of the COP8™ OTP microcontroller family. They are pin and software compatible to the mask ROM COP888EK/COP884EK product family. (Continued)

Key Features

- Analog function block with
  - — Analog comparator with seven input multiplexor
  - — Constant current source and $V_{CC}/2$ reference
- Three 16-bit timers, each with two 16-bit registers supporting:
  - — Processor independent PWM mode
  - — External event counter mode
  - — Input capture mode
- 32 kbytes on-board EPROM with security feature

Note: Mask ROMed devices with equivalent on-chip features and program memory sizes of 8k is available.

- 256 bytes on-board RAM

Additional Peripheral Features

- Idle timer
- Multi-Input Wake-Up (MIWU) with optional interrupts (8)
- WATCHDOG™ and clock monitor logic
- MICROWIRE/PLUS™ serial I/O

Features

- Memory mapped I/O
- Software selectable I/O options (TRI-STATE® output, push-pull output, weak pull-up input, high impedance input)
- Schmitt trigger inputs on ports G and L
- Packages:
  - — 44 PLCC with 40 I/O pins
  - — 40 DIP with 36 I/O pins
  - — 28 DIP/SO, each with 24 I/O pins

CPU/Instruction Set Features

- 1 μs instruction cycle time
- Twelve multi-source vectored interrupts servicing
  - — External interrupt
  - — Idle timer T0
  - — Three timers (Each with 2 interrupts)
  - — MICROWIRE/PLUS
  - — Multi-Input Wake Up
  - — Software trap
  - — Default VIS (default interrupt)
- Versatile and easy to use instruction set
- 8-bit Stack Pointer (SP)—stack in RAM
- Two 8-bit register indirect data memory pointers (B and X)

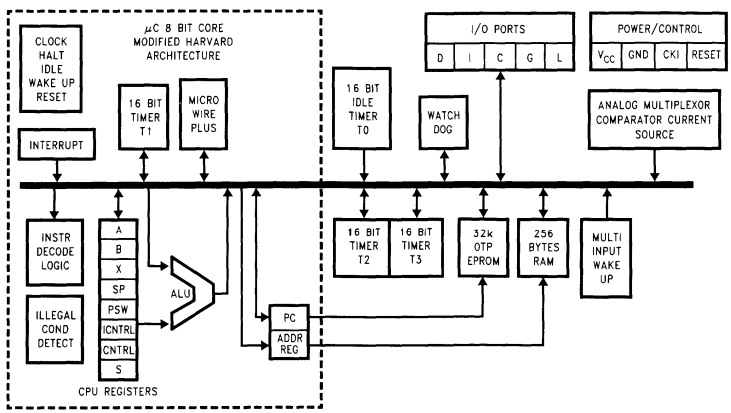
Fully Static CMOS

- Two power saving modes: HALT and IDLE
- Single supply operation: 2.7V to 5.5V
- Temperature ranges: −40° to +85°C

Development Support

- Emulation device for the COP888EK/COP884EK
- Real time emulation and full program debug offered by MetaLink Development System

Block Diagram



TL/DD/12859–1

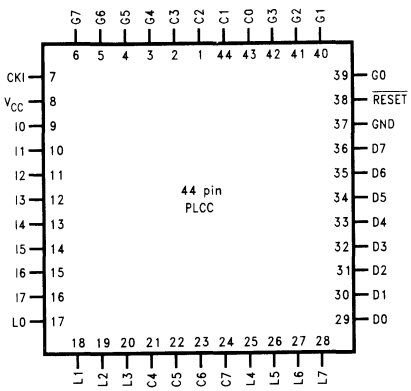
FIGURE 1. Block Diagram

General Description (Continued)

It is a fully static part, fabricated using double-metal silicon gate microCMOS technology. The device is available as One-Time Programmable (OTP). Features include an 8-bit memory mapped architecture, MICROWIRE/PLUS serial I/O, three 16-bit timer/counters supporting three modes (Processor Independent PWM generation, External Event counter, and Input Capture mode capabilities), one analog comparator with seven input multiplexor, and two power saving modes (HALT and IDLE), both with a multi-sourced wakeup/interrupt capability. This multi-sourced interrupt capability may also be used independent of the HALT or IDLE modes. Each I/O pin has software selectable configurations. The devices operate over a voltage range of 2.7V to 5.5V. High throughput is achieved with an efficient, regular instruction set operating at a maximum rate of 1 μs per instruction.

Connection Diagrams

Plastic Chip Carrier

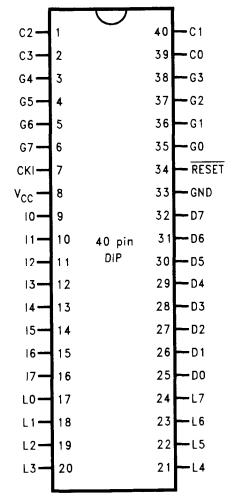


TL/DD/12859-2

Top View

Order Number COP87L88RKV-XE
See NS Plastic Chip Package Number V44A

Dual-In-Line Package

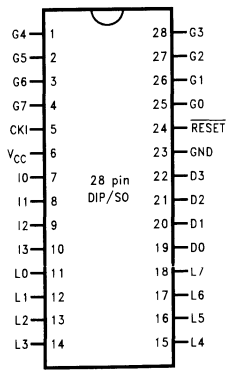


TL/DD/12859-3

Top View

Order Number COP87L84RKN-XE
See NS Molded Package Number N40A

Dual-In-Line Package



TL/DD/12859-4

Note: -X Crystal Oscillator
-E Halt Enable

Top View

Order Number COP87L84RKN-XE
See NS Molded Package Number N28B
Order Number COP87L84RKM-XE
See NS Molded Package Number M28B

FIGURE 2. Connection Diagrams

Connection Diagrams (Continued)

Pinouts for 28-, 40- and 44-Pin Packages

Port	Type	Alt. Fun	Alt. Fun	28-Pin DIP/SO	40-Pin DIP	44-Pin PLCC
L0	I/O	MIWU		11	17	17
L1	I/O	MIWU		12	18	18
L2	I/O	MIWU		13	19	19
L3	I/O	MIWU		14	20	20
L4	I/O	MIWU	T2A	15	21	25
L5	I/O	MIWU	T2B	16	22	26
L6	I/O	MIWU	T3A	17	23	27
L7	I/O	MIWU	T3B	18	24	28
G0	I/O	INT		25	35	39
G1	WDOUT			26	36	40
G2	I/O	T1B		27	37	41
G3	I/O	T1A		28	38	42
G4	I/O	SO		1	3	3
G5	I/O	SK		2	4	4
G6	I	SI		3	5	5
G7	I/CKO	HALT Restart		4	6	6
D0	O			19	25	29
D1	O			20	26	30
D2	O			21	27	31
D3	O			22	28	32
D4	O				29	33
D5	O				30	34
D6	O				31	35
D7	O				32	36
I0	I	COMPIN1+		7	9	9
I1	I	COMPIN−/Current Source Out		8	10	10
I2	I	COMPIN0+		9	11	11
I3	I	COMPOUT/COMPIN2+		10	12	12
I4	I	COMPIN3+			13	13
I5	I	COMPIN4+			14	14
I6	I	COMPIN5+			15	15
I7	I	COMPOUT			16	16
C0	I/O				39	43
C1	I/O				40	44
C2	I/O				1	1
C3	I/O				2	2
C4	I/O					21
C5	I/O					22
C6	I/O					23
C7	I/O					24
V_{CC}				6	8	8
GND				23	33	37
CKI				5	7	7
$\overline{\text{RESET}}$				24	34	38

Absolute Maximum Ratings

If Military/Aerospace specified devices are required, please contact the National Semiconductor Sales Office/Distributors for availability and specifications.

Supply Voltage (V_{CC})	7V
Voltage at Any Pin	−0.3V to V_{CC} + 0.3V
Total Current into V_{CC} Pin (Source)	100 mA
Total Current out of GND Pin (Sink)	110 mA
Storage Temperature Range	−65°C to +140°C

Note: *Absolute maximum ratings indicate limits beyond which damage to the device may occur. DC and AC electrical specifications are not ensured when operating the device at absolute maximum ratings.*

DC Electrical Characteristics

$-40°C \le T_A \le +85°C$ unless otherwise specified

Parameter	Conditions	Min	Typ	Max	Units
Operating Voltage		2.7		5.5	V
Power Supply Ripple (Note 1)	Peak-to-Peak			0.1 V_{CC}	V
Supply Current (Note 2)					
CKI = 10 MHz	V_{CC} = 5.5V, t_c = 1 μs			16.5	mA
CKI = 4 MHz	V_{CC} = 4.0V, t_c = 2.5 μs			6.5	mA
HALT Current (Note 3)	V_{CC} = 5.5V, CKI = 0 MHz			12	μA
	V_{CC} = 4.0V, CKI = 0 MHz			8	μA
IDLE Current (Note 2)					
CKI = 10 MHz	V_{CC} = 5.5V, t_c = 1 μs			3.5	mA
CKI = 1 MHz	V_{CC} = 4.0V, t_c = 10 μs			0.7	mA
Input Levels (V_{IH}, V_{IL})					
$\overline{RESET}$					
Logic High		0.8 V_{CC}			V
Logic Low				0.2 V_{CC}	V
CKI, All Other Inputs					
Logic High		0.7 V_{CC}			V
Logic Low				0.2 V_{CC}	V
Hi-Z Input Leakage	V_{CC} = 5.5V	−2		+2	μA
Input Pullup Current	V_{CC} = 5.5V, V_{IN} = 0V	−40		−250	μA
G and L Port Input Hysteresis (Note 7)				0.35 V_{CC}	V
Output Current Levels					
D Outputs					
Source	V_{CC} = 4.5V, V_{OH} = 3.3V	−0.4			mA
Sink (Note 4)	V_{CC} = 4.5V, V_{OL} = 1V	10			mA
All Others					
Source (Weak Pull-Up Mode)	V_{CC} = 4.5V, V_{OH} = 2.7V	−10		−110	μA
Source (Push-Pull Mode)	V_{CC} = 4.5V, V_{OH} = 3.3V	−0.4			mA
Sink (Push-Pull Mode)	V_{CC} = 4.5V, V_{OL} = 0.4V	1.6			mA
TRI-STATE Leakage	V_{CC} = 5.5V	−2		+2	μA
Allowable Sink/Source Current per Pin (Note 7)					
D Outputs (Sink)				15	mA
All others				3	mA
Maximum Input Current without Latchup (Note 5)	Room Temp			±200	mA
RAM Retention Voltage, V_r	500 ns Rise and Fall Time (Min)	2			V
Input Capacitance (Note 6)				7	pF
Load Capacitance on D2 (Note 6)				1000	pF

AC Electrical Characteristics $-40°C \le T_A \le +85°C$ unless otherwise specified

Parameter	Conditions	Min	Typ	Max	Units
Instruction Cycle Time (t_c)					
Crystal, Resonator,	$4.5V \le V_{CC} \le 5.5V$	1.0		DC	μs
R/C Oscillator	$4.5V \le V_{CC} \le 5.5V$	3.0		DC	μs
Inputs					
t_{SETUP}	$4.5V \le V_{CC} \le 5.5V$	200			ns
t_{HOLD}	$4.5V \le V_{CC} \le 5.5V$	60			ns
Output Propagation Delay (Note 6)	$R_L = 2.2k, C_L = 100$ pF				
t_{PD1}, t_{PD0}					
SO, SK	$4.5V \le V_{CC} \le 5.5V$			0.7	μs
All Others	$4.5V \le V_{CC} \le 5.5V$			1	μs
MICROWIRE™ Setup Time (t_{UWS}) (Note 7)	$V_{CC} \ge 4.5V$	20			ns
MICROWIRE Hold Time (t_{UWH}) (Note 7)	$V_{CC} \ge 4.5V$	56			ns
MICROWIRE Output Propagation Delay (t_{UPD})	$V_{CC} \ge 4.5V$			220	ns
Input Pulse Width (Note 7)					
Interrupt Input High Time		1.0			t_c
Interrupt Input Low Time		1.0			t_c
Timer 1, 2, 3 Input High Time		1.0			t_c
Timer 1, 2, 3 Input Low Time		1.0			t_c
Reset Pulse Width		1.0			μs

t_c = Instruction Cycle Time

Note 1: Maximum rate of voltage change must be < 0.5 V/ms.

Note 2: Supply and IDLE currents are measured with CKI driven with a square wave Oscillator, CKO driven 180° out of phase with CKI, inputs connected to V_{CC} and outputs driven low but not connected to a load.

Note 3: The HALT mode will stop CKI from oscillating in the RC and the Crystal configurations by bringing CKI high. Measurement of I_{DD} HALT is done with device neither sourcing nor sinking current; with L, C, and G0–G5 programmed as low outputs and not driving a load; all outputs programmed low and not driving a load; all inputs tied to V_{CC}; clock monitor and comparator disabled. Parameter refers to HALT mode entered via setting bit 7 of the G Port data register. Part will pull up CKI during HALT in crystal clock mode.

Note 4: The user must guarantee that D2 pin does not source more than 10 mA during RESET. If D2 sources more than 10 mA during reset, the device will go into programming mode.

Note 5: Pins G6 and $\overline{\text{RESET}}$ are designed with a high voltage input network. These pins allow input voltages > V_{CC} and the pins will have sink current to V_{CC} when biased at voltages > V_{CC} (the pins do not have source current when biased at a voltage below V_{CC}). The effective resistance to V_{CC} is 750Ω (typical). These two pins will not latch up. The voltage at the pins must be limited to < 14V. **WARNING: Voltages in excess of 14V will cause damage to the pins. This warning excludes ESD transients.**

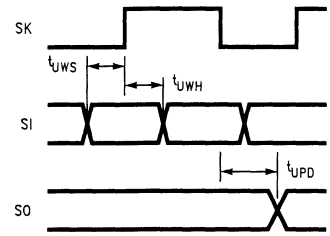
Note 6: The output propagation delay is referenced to the end of the instruction cycle where the output change occurs.

Note 7: Parameter characterized but not tested.

Analog Function Block V_{CC} = 5.0V, −40°C ≤ T_A ≤ +85°C

Parameter	Conditions	Min	Typ	Max	Units
Input Offset Voltage	0.4V < V_{IN} < V_{CC} − 1.5V		±10	±25	mV
Input Common Mode Voltage Range (Note 8)		0.4		V_{CC} − 1.5	V
V_{CC}/2 Reference	4.5V < V_{CC} < 5.5V	0.5 V_{CC} − 0.04	0.5 V_{CC}	0.5 V_{CC} + 0.04	V
DC Supply Current for Comparator (when enabled)	V_{CC} = 5.5V			250	μA
DC Supply Current for V_{CC}/2 Reference (when enabled)	V_{CC} = 5.5V		50	80	μA
DC Supply Current for Constant Current Source (when enabled)	V_{CC} = 5.5V			200	μA
Constant Current Source	4.5V < V_{CC} < 5.5V	10	20	40	μA
Current Source Variation over Common Mode Range	4.5V < V_{CC} < 5.5V Temp = Constant			±2	μA
Current Source Enable Time			1.5	2	μs
Comparator Response Time	100 mV Overdrive, 100 pF Load			1	μs

Note 8: The device is capable of operating over a common mode voltage range of 0 to V_{CC} − 1.5V, however increased offset voltage will be observed between 0V and 0.4V.



TL/DD/12859–5

FIGURE 3. MICROWIRE/PLUS Timing

Pin Descriptions

V_{CC} and GND are the power supply pins. All V_{CC} and GND pins must be connected.

CKI is the clock input. This can come from an R/C generated oscillator, or a crystal oscillator (in conjunction with CKO). See Oscillator Description section.

$\overline{\text{RESET}}$ is the master reset input. See Reset Description section.

The device contains three bidirectional 8-bit I/O ports (C, G and L), where each individual bit may be independently configured as an input (Schmitt Trigger inputs on ports L and G), output or TRI-STATE under program control. Three data memory address locations are allocated for each of these I/O ports. Each I/O port has two associated 8-bit memory mapped registers, the CONFIGURATION register and the output DATA register. A memory mapped address is also reserved for the input pins of each I/O port. (See the memory map for the various addresses associated with the I/O ports.) *Figure 4* shows the I/O port configurations. The DATA and CONFIGURATION registers allow for each port bit to be individually configured under software control as shown below:

CONFIGURATION Register	DATA Register	Port Set-Up
0	0	Hi-Z Input (TRI-STATE Output)
0	1	Input with Weak Pull-Up
1	0	Push-Pull Zero Output
1	1	Push-Pull One Output

PORT L is an 8-bit I/O port. All L-pins have Schmitt triggers on the inputs.

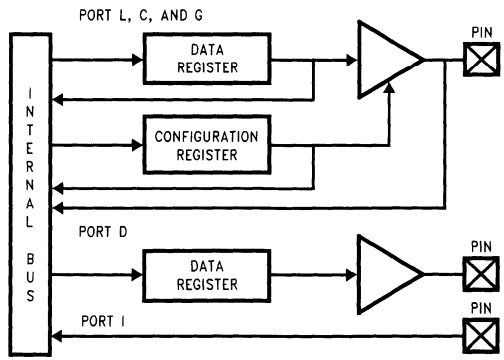
The Port L supports Multi-Input Wake Up on all eight pins. L4 and L5 are used for the timer input functions T2A and T2B. L6 and L7 are used for the timer input functions T3A and T3B.

The Port L has the following alternate features:

- L0 MIWU
- L1 MIWU
- L2 MIWU
- L3 MIWU
- L4 MIWU or T2A
- L5 MIWU or T2B
- L6 MIWU or T3A
- L7 MIWU or T3B

Port G is an 8-bit port with 5 I/O pins (G0, G2–G5), an input pin (G6), and a dedicated output pin (G7). Pins G0 and G2–G6 all have Schmitt Triggers on their inputs. Pin G1 serves as the dedicated WDOUT WATCHDOG output, while pin G7 is either input or output depending on the oscillator mask option selected. With the crystal oscillator option selected, G7 serves as the dedicated output pin for the CKO clock output. With the single-pin R/C oscillator mask option selected, G7 serves as a general purpose input pin but is also used to bring the device out of HALT mode with a low to high transition on G7. There are two registers associated with the G Port, a data register and a configuration register. Therefore, each of the 5 I/O bits (G0, G2–G5) can be individually configured under software control.

Since G6 is an input only pin and G7 is the dedicated CKO clock output pin (crystal clock option) or general purpose input (R/C clock option), the associated bits in the data and configuration registers for G6 and G7 are used for special purpose functions as outlined on the next page. Reading the G6 and G7 data bits will return zeros.



TL/DD/12859–6

FIGURE 4. I/O Port Configurations

Pin Descriptions (Continued)

Note that the chip will be placed in the HALT mode by writing a "1" to bit 7 of the Port G Data Register. Similarly the chip will be placed in the IDLE mode by writing a "1" to bit 6 of the Port G Data Register.

Writing a "1" to bit 6 of the Port G Configuration Register enables the MICROWIRE/PLUS to operate with the alternate phase of the SK clock. The G7 configuration bit, if set high, enables the clock start up delay after HALT when the R/C clock configuration is used.

	Config Reg.	Data Reg.
G7	CLKDLY	HALT
G6	Alternate SK	IDLE

Port G has the following alternate features:

- G0 INTR (External Interrupt Input)
- G2 T1B (Timer T1 Capture Input)
- G3 T1A (Timer T1 I/O)
- G4 SO (MICROWIRE Serial Data Output)
- G5 SK (MICROWIRE Serial Clock)
- G6 SI (MICROWIRE Serial Data Input)

Port G has the following dedicated functions:

- G1 WDOUT WATCHDOG and/or Clock Monitor dedicated output
- G7 CKO Oscillator dedicated output or general purpose input

Port C is an 8-bit I/O port. The 40-pin device does not have a full complement of Port C pins. The unavailable pins are not terminated. A read operation for these unterminated pins will return unpredicatable values.

Port I is an eight-bit Hi-Z input port.

Port I0–I7 are used for the analog function block.

The Port I has the following alternate features:

- I0 COMPIN1+ (Comparator Positive Input 1)
- I1 COMPIN− (Comparator Negative Input/Current Source Out)
- I2 COMPIN0+ (Comparator Positive Input 0)
- I3 COMPOUT/COMPIN2+ (Comparator Output/ Comparator Positive Input 2))
- I4 COMPIN3+ (Comparator Positive Input 3)
- I5 COMPIN4+ (Comparator Positive Input 4)
- I6 COMPIN5+ (Comparator Positive Input 5)
- I7 COMPOUT (Comparator Output)

Port D is an 8-bit output port that is preset high when $\overline{\text{RESET}}$ goes low. The user can tie two or more D port outputs (except D2) together in order to get a higher drive.

Note: Care must be exercised with the D2 pin operation. At RESET, the external loads on this pin must ensure that the output voltages stay above 0.8 V_{CC} to prevent the chip from entering special modes. Also keep the external loading on D2 to <1000 pF.

Functional Description

The architecture of the device is modified Harvard architecture. With the Harvard architecture, the control store program memory (ROM) is separated from the data store memory (RAM). Both ROM and RAM have their own separate addressing space with separate address buses. The architecture, though based on Harvard architecture, permits transfer of data from ROM to RAM.

CPU REGISTERS

The CPU can do an 8-bit addition, subtraction, logical or shift operation in one instruction (t_C) cycle time.

There are six CPU registers:

A is the 8-bit Accumulator Register

PC is the 15-bit Program Counter Register

PU is the upper 7 bits of the program counter (PC)
PL is the lower 8 bits of the program counter (PC)

B is an 8-bit RAM address pointer, which can be optionally post auto incremented or decremented.

X is an 8-bit alternate RAM address pointer, which can be optionally post auto incremented or decremented.

SP is the 8-bit stack pointer, which points to the subroutine/interrupt stack (in RAM). The SP is initialized to RAM address 06F with reset.

S is the 8-bit Data Segment Address Register used to extend the lower half of the address range (00 to 7F) into 256 data segments of 128 bytes each.

All the CPU registers are memory mapped with the exception of the Accumulator (A) and the Program Counter (PC).

PROGRAM MEMORY

The program memory consists of 32 kbytes of OTP EPROM. These bytes may hold program instructions or constant data (data tables for the LAID instruction, jump vectors for the JID instruction, and interrupt vectors for the VIS instruction). The program memory is addressed by the 15-bit program counter (PC). All interrupts in the devices vector to program memory location 0FF Hex.

The device can be configured to inhibit external reads of the program memory. This is done by programming the Security Byte.

Note: Mask ROMed devices with equivalent on-chip features and program memory sizes of 8k is available.

SECURITY FEATURE

The program memory array has an associate Security Byte that is located outside of the program address range. This byte can be addressed only from programming mode by a programmer tool.

Security is an optional feature and can only be asserted after the memory array has been programmed and verified. A secured part will read all 00(hex) by a programmer. The part will fail Blank Check and will fail Verify operations. A Read operation will fill the programmer's memory with 00(hex). The Security Byte itself is always readable with value of 00(hex) if unsecure and FF(hex) if secure.

DATA MEMORY

The data memory address space includes the on-chip RAM and data registers, the I/O registers (Configuration, Data and Pin), the control registers, the MICROWIRE/PLUS SIO shift register, and the various registers, and counters associated with the timers (with the exception of the IDLE timer). Data memory is addressed directly by the instruction or indirectly by the B, X, SP pointers and S register.

The data memory consists of 256 bytes of RAM. Sixteen bytes of RAM are mapped as "registers" at addresses 0F0 to 0FF Hex. These registers can be loaded immediately, and also decremented and tested with the DRSZ (decrement register and skip if zero) instruction. The memory pointer registers X, SP, B and S are memory mapped into this space at address locations 0FC to 0FF Hex respectively, with the other registers being available for general usage.

Functional Description (Continued)

The instruction set permits any bit in memory to be set, reset or tested. All I/O and registers (except A and PC) are memory mapped; therefore, I/O bits and register bits can be directly and individually set, reset and tested. The accumulator (A) bits can also be directly and individually tested.

Note: RAM contents are undefined upon power-up.

Data Memory Segment RAM Extension

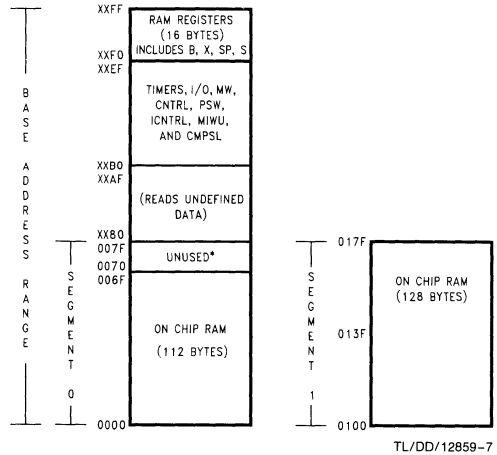
Data memory address 0FF is used as a memory mapped location for the Data Segment Address Register (S).

The data store memory is either addressed directly by a single byte address within the instruction, or indirectly relative to the reference of the B, X, or SP pointers (each contains a single-byte address). This single-byte address allows an addressing range of 256 locations from 00 to FF hex. The upper bit of this single-byte address divides the data store memory into two separate sections as outlined previously. With the exception of the RAM register memory from address locations 00F0 to 00FF, all RAM memory is memory mapped with the upper bit of the single-byte address being equal to zero. This allows the upper bit of the single-byte address to determine whether or not the base address range (from 0000 to 00FF) is extended. If this upper bit equals one (representing address range 0080 to 00FF), then address extension does not take place. Alternatively, if this upper bit equals zero, then the data segment extension register S is used to extend the base address range (from 0000 to 007F) from XX00 to XX7F, where XX represents the 8 bits from the S register. Thus the 128-byte data segment extensions are located from addresses 0100 to 017F for data segment 1, 0200 to 027F for data segment 2, etc., up to FF00 to FF7F for data segment 255. The base address range from 0000 to 007F represents data segment 0.

Figure 5 illustrates how the S register data memory extension is used in extending the lower half of the base address range (00 to 7F hex) into 256 data segments of 128 bytes each, with a total addressing range of 32 kbytes from XX00 to XX7F. This organization allows a total of 256 data segments of 128 bytes each with an additional upper base segment of 128 bytes. Furthermore, all addressing modes are available for all data segments. The S register must be changed under program control to move from one data segment (128 bytes) to another. However, the upper base segment (containing the 16 memory registers, I/O registers, control registers, etc.) is always available regardless of the contents of the S register, since the upper base segment (address range 0080 to 00FF) is independent of data segment extension.

The instructions that utilize the stack pointer (SP) always reference the stack as part of the base segment (Segment 0), regardless of the contents of the S register. The S register is not changed by these instructions. Consequently, the stack (used with subroutine linkage and interrupts) is always located in the base segment. The stack pointer will be initialized to point at data memory location 006F as a result of reset.

The 128 bytes of RAM contained in the base segment are split between the lower and upper base segments. The first 112 bytes of RAM are resident from address 0000 to 006F in the lower base segment, while the remaining 16 bytes of RAM represent the 16 data memory registers located at addresses 00F0 to 00FF of the upper base segment. No RAM is located at the upper sixteen addresses (0070 to 007F) of the lower base segment.



TL/DD/12859-7

*Reads as all ones.

FIGURE 5. RAM Organization

Additional RAM beyond these initial 128 bytes, however, will always be memory mapped in groups of 128 bytes (or less) at the data segment address extensions (XX00 to XX7F) of the lower base segment. The additional 128 bytes of RAM are memory mapped at address locations 0100 to 017F hex.

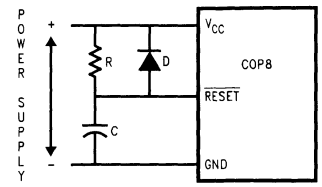
Reset

The $\overline{\text{RESET}}$ input when pulled low initializes the microcontroller. Initialization will occur whenever the $\overline{\text{RESET}}$ input is pulled low. Upon initialization, the data and configuration registers for ports L, G and C are cleared, resulting in these Ports being initialized to the TRI-STATE mode. Pin G1 of the G Port is an exception (as noted below) since pin G1 is dedicated as the WATCHDOG and/or Clock Monitor error output pin. Port D is set high. The PC, PSW, ICNTRL, CNTRL, T2CNTRL and T3CNTRL control registers are cleared. The Comparator Select Register is cleared. The S register is initialized to zero. The Multi-Input Wakeup registers WKEN and WKEDG are cleared. Wakeup register WKPND is unknown. The stack pointer, SP, is initialized to 6F hex.

The device comes out of reset with both the WATCHDOG logic and the Clock Monitor detector armed, with the WATCHDOG service window bits set and the Clock Monitor bit set. The WATCHDOG and Clock Monitor circuits are inhibited during reset. The WATCHDOG service window bits being initialized high default to the maximum WATCHDOG service window of 64k t_C clock cycles. The Clock Monitor bit being initialized high will cause a Clock Monitor error following reset if the clock has not reached the minimum specified frequency at the termination of reset. A Clock Monitor error will cause an active low error output on pin G1. This error output will continue until 16 t_C–32 t_C clock cycles following the clock frequency reaching the minimum specified value, at which time the G1 output will enter the TRI-STATE mode.

The external RC network shown in *Figure 6* should be used to ensure that the $\overline{\text{RESET}}$ pin is held low until the power supply to the chip stabilizes.

Reset (Continued)



TL/DD/12859-8

RC > 5 × Power Supply Rise Time

FIGURE 6. Recommended Reset Circuit

Oscillator Circuits

The chip can be driven by a clock input on the CKI input pin which can be between DC and 10 MHz. The CKO output clock is on pin G7 (crystal configuration). The CKI input frequency is divided down by 10 to produce the instruction cycle clock ($1/t_C$).

Figure 7 shows the Crystal and R/C oscillator diagrams.

CRYSTAL OSCILLATOR

CKI and CKO can be connected to make a closed loop crystal (or resonator) controlled oscillator.

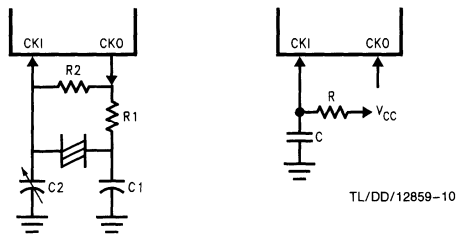
Table A shows the component values required for various standard crystal values.

R/C OSCILLATOR

By selecting CKI as a single pin oscillator input, a single pin R/C oscillator circuit can be connected to it. CKO is available as a general purpose input, and/or HALT restart input.

Note: Use of the R/C oscillator option will result in higher electromagnetic emissions.

Table B shows the variation in the oscillator frequencies as functions of the component (R and C) values.



TL/DD/12859-9

TL/DD/12859-10

FIGURE 7. Crystal and R/C Oscillator Diagrams

TABLE A. Crystal Oscillator Configuration, T_A = 25°C

R1 (kΩ)	R2 (MΩ)	C1 (pF)	C2 (pF)	CKI Freq (MHz)	Conditions
0	1	30	30-36	10	V_{CC} = 5V
0	1	30	30-36	4	V_{CC} = 5V
0	1	200	100-150	0.455	V_{CC} = 5V

TABLE B. RC Oscillator Configuration, T_A = 25°C

R (kΩ)	C (pF)	CKI Freq (MHz)	Instr. Cycle (μs)	Conditions
3.3	82	2.2 to 2.7	3.7 to 4.6	V_{CC} = 5V
5.6	100	1.1 to 1.3	7.4 to 9.0	V_{CC} = 5V
6.8	100	0.9 to 1.1	8.8 to 10.8	V_{CC} = 5V

Note: 3k ≤ R ≤ 200k

50 pF ≤ C ≤ 200 pF

Control Registers

CNTRL Register (Address X'00EE)

The Timer1 (T1) and MICROWIRE/PLUS control register contains the following bits:

SL1 & SL0 Select the MICROWIRE/PLUS clock divide by (00 = 2, 01 = 4, 1x = 8)

IEDG External interrupt edge polarity select (0 = Rising edge, 1 = Falling edge)

MSEL Selects G5 and G4 as MICROWIRE/PLUS signals SK and SO respectively

T1C0 Timer T1 Start/Stop control in timer modes 1 and 2

Timer T1 Underflow Interrupt Pending Flag in timer mode 3

T1C1 Timer T1 mode control bit

T1C2 Timer T1 mode control bit

T1C3 Timer T1 mode control bit

T1C3	T1C2	T1C1	T1C0	MSEL	IEDG	SL1	SL0

Bit 7 Bit 0

Control Registers (Continued)

PSW Register (Address X'00EF)

The PSW register contains the following select bits:

- GIE Global interrupt enable (enables interrupts)
- EXEN Enable external interrupt
- BUSY MICROWIRE/PLUS busy shifting flag
- EXPND External interrupt pending
- T1ENA Timer T1 Interrupt Enable for Timer Underflow or T1A Input capture edge
- T1PNDA Timer T1 Interrupt Pending Flag (Autoreload RA in mode 1, T1 Underflow in Mode 2, T1A capture edge in mode 3)
- C Carry Flag
- HC Half Carry Flag

HC	C	T1PNDA	T1ENA	EXPND	BUSY	EXEN	GIE

Bit 7 Bit 0

The Half-Carry bit is also affected by all the instructions that affect the Carry flag. The SC (Set Carry) and RC (Reset Carry) instructions will respectively set or clear both the carry flags. In addition to the SC and RC instructions, ADC, SUBC, RRC and RLC instructions affect the carry and Half Carry flags.

ICNTRL Register (Address X'00E8)

The ICNTRL register contains the following bits:

- T1ENB Timer T1 Interrupt Enable for T1B Input capture edge
- T1PNDB Timer T1 Interrupt Pending Flag for T1B capture edge
- μWEN Enable MICROWIRE/PLUS interrupt
- μWPND MICROWIRE/PLUS interrupt pending
- T0EN Timer T0 Interrupt Enable (Bit 12 toggle)
- T0PND Timer T0 Interrupt pending
- LPEN L Port Interrupt Enable (Multi-Input Wakeup/Interrupt)

  Bit 7 could be used as a flag

Unused	LPEN	T0PND	T0EN	μWPND	μWEN	T1PNDB	T1ENB

Bit 7 Bit 0

T2CNTRL Register (Address X'00C6)

The T2CNTRL register contains the following bits:

- T2ENB Timer T2 Interrupt Enable for T2B Input capture edge
- T2PNDB Timer T2 Interrupt Pending Flag for T2B capture edge
- T2ENA Timer T2 Interrupt Enable for Timer Underflow or T2A Input capture edge
- T2PNDA Timer T2 Interrupt Pending Flag (Autoreload RA in mode 1, T2 Underflow in mode 2, T2A capture edge in mode 3)
- T2C0 Timer T2 Start/Stop control in timer modes 1 and 2 Timer T2 Underflow Interrupt Pending Flag in timer mode 3
- T2C1 Timer T2 mode control bit
- T2C2 Timer T2 mode control bit
- T2C3 Timer T2 mode control bit

T2C3	T2C2	T2C1	T2C0	T2PNDA	T2ENA	T2PNDB	T2ENB

Bit 7 Bit 0

T3CNTRL Register (Address X'00B6)

The T3CNTRL register contains the following bits:

- T3ENB Timer T3 Interrupt Enable for T3B
- T3PNDB Timer T3 Interrupt Pending Flag for T3B pin (T3B capture edge)
- T3ENA Timer T3 Interrupt Enable for Timer Underflow or T3A pin
- T3PNDA Timer T3 Interrupt Pending Flag (Autoload RA in mode 1, T3 Underflow in mode 2, T3a capture edge in mode 3)
- T3C0 Timer T3 Start/Stop control in timer modes 1 and 2

  Timer T3 Underflow Interrupt Pending Flag in timer mode 3
- T3C1 Timer T3 mode control bit
- T3C2 Timer T3 mode control bit
- T3C3 Timer T3 mode control bit

T3C3	T3C2	T3C1	T3C0	T3PNDA	T3ENA	T3PNDB	T3ENB

Bit 7 Bit 0

Timers

The device contains a very versatile set of timers (T0, T1, T2, T3). All timers and associated autoreload/capture registers power up containing random data.

TIMER T0 (IDLE TIMER)

The device supports applications that require maintaining real time and low power with the IDLE mode. This IDLE mode support is furnished by the IDLE timer T0, which is a 16-bit timer. The Timer T0 runs continuously at the fixed rate of the instruction cycle clock, t_C. The user cannot read or write to the IDLE Timer T0, which is a count down timer. The Timer T0 supports the following functions:

- Exit out of the Idle Mode (See Idle Mode description)
- WATCHDOG logic (See WATCHDOG description)
- Start up delay out of the HALT mode

The IDLE Timer T0 can generate an interrupt when the thirteenth bit toggles. This toggle is latched into the T0PND pending flag, and will occur every 4 ms at the maximum clock frequency (t_C = 1 μs). A control flag T0EN allows the interrupt from the thirteenth bit of Timer T0 to be enabled or disabled. Setting T0EN will enable the interrupt, while resetting it will disable the interrupt.

Timers (Continued)

TIMER T1, TIMER T2 AND TIMER T3

The device has a set of three powerful timer/counter blocks, T1, T2 and T3. The associated features and functioning of a timer block are described by referring to the timer block Tx. Since the three timer blocks, T1, T2 and T3 are identical, all comments are equally applicable to any of the three timer blocks.

Each timer block consists of a 16-bit timer, Tx, and two supporting 16-bit autoreload/capture registers, RxA and RxB. Each timer block has two pins associated with it, TxA and TxB. The pin TxA supports I/O required by the timer block, while the pin TxB is an input to the timer block. The powerful and flexible timer block allows the device to easily perform all timer functions with minimal software overhead. The timer block has three operating modes: Processor Independent PWM mode, External Event Counter mode, and Input Capture mode.

The control bits TxC3, TxC2, and TxC1 allow selection of the different modes of operation.

Mode 1. Processor Independent PWM Mode

As the name suggests, this mode allows the device to generate a PWM signal with very minimal user intervention. The user only has to define the parameters of the PWM signal (ON time and OFF time). Once begun, the timer block will continuously generate the PWM signal completely independent of the microcontroller. The user software services the timer block only when the PWM parameters require updating.

In this mode the timer Tx counts down at a fixed rate of t_C. Upon every underflow the timer is alternately reloaded with the contents of supporting registers, RxA and RxB. The very first underflow of the timer causes the timer to reload from the register RxA. Subsequent underflows cause the timer to be reloaded from the registers alternately beginning with the register RxB.

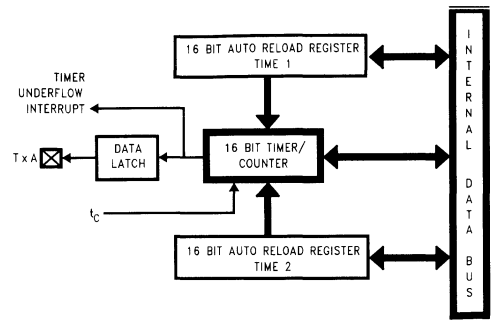
The Tx Timer control bits, TxC3, TxC2 and TxC1 set up the timer for PWM mode operation.

Figure 8 shows a block diagram of the timer in PWM mode.

The underflows can be programmed to toggle the TxA output pin. The underflows can also be programmed to generate interrupts.

Underflows from the timer are alternately latched into two pending flags, TxPNDA and TxPNDB. The user must reset these pending flags under software control. Two control enable flags, TxENA and TxENB, allow the interrupts from the timer underflow to be enabled or disabled. Setting the timer enable flag TxENA will cause an interrupt when a timer underflow causes the RxA register to be reloaded into the timer. Setting the timer enable flag TxENB will cause an interrupt when a timer underflow causes the RxB register to be reloaded into the timer. Resetting the timer enable flags will disable the associated interrupts.

Either or both of the timer underflow interrupts may be enabled. This gives the user the flexibility of interrupting once per PWM period on either the rising or falling edge of the PWM output. Alternatively, the user may choose to interrupt on both edges of the PWM output.



TL/DD/12859–11

FIGURE 8. Timer in PWM Mode

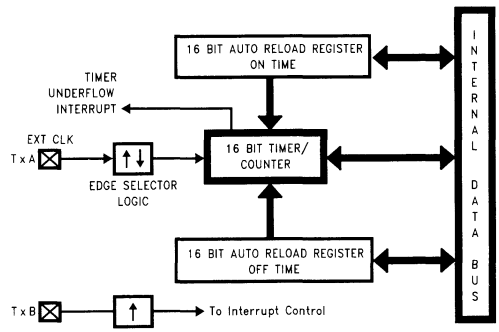
Mode 2. External Event Counter Mode

This mode is quite similar to the processor independent PWM mode described above. The main difference is that the timer, Tx, is clocked by the input signal from the TxA pin. The Tx timer control bits, TxC3, TxC2 and TxC1 allow the timer to be clocked either on a positive or negative edge from the TxA pin. Underflows from the timer are latched into the TxPNDA pending flag. Setting the TxENA control flag will cause an interrupt when the timer underflows.

In this mode the input pin TxB can be used as an independent positive edge sensitive interrupt input if the TxENB control flag is set. The occurrence of a positive edge on the TxB input pin is latched into the TxPNDB flag.

Figure 9 shows a block diagram of the timer in External Event Counter mode.

Note: The PWM output is not available in this mode since the TxA pin is being used as the counter input clock.



TL/DD/12859–12

FIGURE 9. Timer in External Event Counter Mode

Mode 3. Input Capture Mode

The device can precisely measure external frequencies or time external events by placing the timer block, Tx, in the input capture mode.

In this mode, the timer Tx is constantly running at the fixed t_C rate. The two registers, RxA and RxB, act as capture registers. Each register acts in conjunction with a pin. The register RxA acts in conjunction with the TxA pin and the register RxB acts in conjunction with the TxB pin.

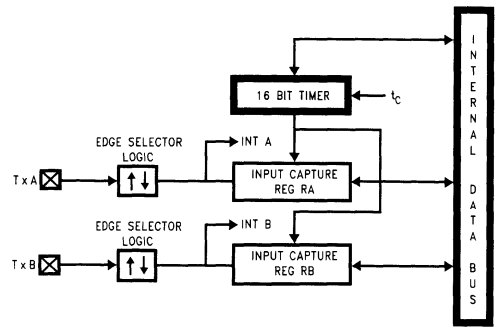
Timers (Continued)

The timer value gets copied over into the register when a trigger event occurs on its corresponding pin. Control bits, TxC3, TxC2 and TxC1, allow the trigger events to be specified either as a positive or a negative edge. The trigger condition for each input pin can be specified independently.

The trigger conditions can also be programmed to generate interrupts. The occurrence of the specified trigger condition on the TxA and TxB pins will be respectively latched into the pending flags, TxPNDA and TxPNDB. The control flag TxENA allows the interrupt on TxA to be either enabled or disabled. Setting the TxENA flag enables interrupts to be generated when the selected trigger condition occurs on the TxA pin. Similarly, the flag TxENB controls the interrupts from the TxB pin.

Underflows from the timer can also be programmed to generate interrupts. Underflows are latched into the timer TxC0 pending flag (the TxC0 control bit serves as the timer underflow interrupt pending flag in the Input Capture mode). Consequently, the TxC0 control bit should be reset when entering the Input Capture mode. The timer underflow interrupt is enabled with the TxENA control flag. When a TxA interrupt occurs in the Input Capture mode, the user must check both the TxPNDA and TxC0 pending flags in order to determine whether a TxA input capture or a timer underflow (or both) caused the interrupt.

Figure 10 shows a block diagram of the timer in Input Capture mode.



TL/DD/12859–13

FIGURE 10. Timer in Input Capture Mode

TIMER CONTROL FLAGS

The timers T1, T2 and T3 have indentical control structures. The control bits and their functions are summarized below.

TxC0	Timer Start/Stop control in Modes 1 and 2 (Processor Independent PWM and External Event Counter), where 1 = Start, 0 = Stop Timer Underflow Interrupt Pending Flag in Mode 3 (Input Capture)
TxPNDA	Timer Interrupt Pending Flag
TxPNDB	Timer Interrupt Pending Flag
TxENA	Timer Interrupt Enable Flag
TxENB	Timer Interrupt Enable Flag 1 = Timer Interrupt Enabled 0 = Timer Interrupt Disabled
TxC3	Timer mode control
TxC2	Timer mode control
TxC1	Timer mode control

Timers (Continued)

The timer mode control bits (TxC3, TxC2 and TxC1) are detailed below:

TxC3	TxC2	TxC1	Timer Mode	Interrupt A Source	Interrupt B Source	Timer Counts On
0	0	0	MODE 2 (External Event Counter)	Timer Underflow	Pos. TxB Edge	TxA Pos. Edge
0	0	1	MODE 2 (External Event Counter)	Timer Underflow	Pos. TxB Edge	TxA Neg. Edge
1	0	1	MODE 1 (PWM) TxA Toggle	Autoreload RA	Autoreload RB	t_C
1	0	0	MODE 1 (PWM) No TxA Toggle	Autoreload RA	Autoreload RB	t_C
0	1	0	MODE 3 (Capture) Captures: TxA Pos. Edge TxB Pos. Edge	Pos. TxA Edge or Timer Underflow	Pos. TxB Edge	t_C
1	1	0	MODE 3 (Capture) Captures: TxA Pos. Edge TxB Neg. Edge	Pos. TxA Edge or Timer Underflow	Neg. TxB Edge	t_C
0	1	1	MODE 3 (Capture) Captures: TxA Neg. Edge TxB Pos. Edge	Neg. TxA Edge or Timer Underflow	Pos. TxB Edge	t_C
1	1	1	MODE 3 (Capture) Captures: TxA Neg. Edge TxB Neg. Edge	Neg. TxA Edge or Timer Underflow	Neg. TxB Edge	t_C

Power Save Modes

The device offers the user two power save modes of operation: HALT and IDLE. In the HALT mode, all microcontroller activities are stopped. In the IDLE mode, the on-board oscillator circuitry the WATCHDOG logic, the Clock Monitor and timer T0 are active but all other microcontroller activities are stopped. In either mode, all on-board RAM, registers, I/O states, and timers (with the exception of T0) are unaltered.

HALT MODE

The device can be placed in the HALT mode by writing a "1" to the HALT flag (G7 data bit). All microcontroller activities, including the clock and timers, are stopped. The WATCHDOG logic is disabled during the HALT mode. However, the clock monitor circuitry if enabled remains active and will cause the WATCHDOG output pin (WDOUT) to go low. If the HALT mode is used and the user does not want to activate the WDOUT pin, the Clock Monitor should be disabled after the device comes out of reset (resetting the Clock Monitor control bit with the first write to the WDSVR register). In the HALT mode, the power requirements of the device are minimal and the applied voltage (V_{CC}) may be decreased to V_r (V_r = 2.0V) without altering the state of the machine.

The device supports three different ways of exiting the HALT mode. The first method of exiting the HALT mode is with the Multi-Input Wakeup feature on the L port. The second method is with a low to high transition on the CKO (G7) pin. This method precludes the use of the crystal clock configuration (since CKO becomes a dedicated output), and so may be used with an RC clock configuration. The third method of exiting the HALT mode is by pulling the $\overline{\text{RESET}}$ pin low.

Since a crystal or ceramic resonator may be selected as the oscillator, the Wakeup signal is not allowed to start the chip running immediately since crystal oscillators and ceramic resonators have a delayed start up time to reach full amplitude and frequency stability. The IDLE timer is used to generate a fixed delay to ensure that the oscillator has indeed stabilized before allowing instruction execution. In this case, upon detecting a valid Wakeup signal, only the oscillator circuitry is enabled. The IDLE timer is loaded with a value of 256 and is clocked with the t_C instruction cycle clock. The t_C clock is derived by dividing the oscillator clock down by a factor of 10. The Schmitt trigger following the CKI inverter on the chip ensures that the IDLE timer is clocked only when the oscillator has a sufficiently large amplitude to meet the Schmitt trigger specifications. This Schmitt trigger is not part of the oscillator closed loop. The startup timeout from the IDLE timer enables the clock signals to be routed to the rest of the chip.

If an RC clock option is being used, the fixed delay is introduced optionally. A control bit, CLKDLY, mapped as configuration bit G7, controls whether the delay is to be introduced or not. The delay is included if CLKDLY is set, and excluded if CLKDLY is reset. The CLKDLY bit is cleared on reset.

Power Save Modes (Continued)

The device has two mask options associated with the HALT mode. The first mask option enables the HALT mode feature, while the second mask option disables the HALT mode. With the HALT mode enable mask option, the device will enter and exit the HALT mode as described above. With the HALT disable mask option, the device cannot be placed in the HALT mode (writing a "1" to the HALT flag will have no effect, the HALT flag will remain "0").

IDLE MODE

The device is placed in the IDLE mode by writing a "1" to the IDLE flag (G6 data bit). In this mode, all activities, except the associated on-board oscillator circuitry, the WATCHDOG logic, the clock monitor and the IDLE Timer T0, are stopped.

As with the HALT mode, the device can be returned to normal operation with a reset, or with a Multi-Input Wakeup from the L Port. Alternately, the microcontroller resumes normal operation from the IDLE mode when the thirteenth bit (representing 4.096 ms at internal clock frequency of 1 MHz, $t_c = 1\ \mu s$) of the IDLE Timer toggles.

This toggle condition of the thirteenth bit of the IDLE Timer T0 is latched into the T0PND pending flag.

The user has the option of being interrupted with a transition on the thirteenth bit of the IDLE Timer T0. The interrupt can be enabled or disabled via the T0EN control bit. Setting the T0EN flag enables the interrupt and vice versa.

The user can enter the IDLE mode with the Timer T0 interrupt enabled. In this case, when the T0PND bit gets set, the device will first execute the Timer T0 interrupt service routine and then return to the instruction following the "Enter Idle Mode" instruction.

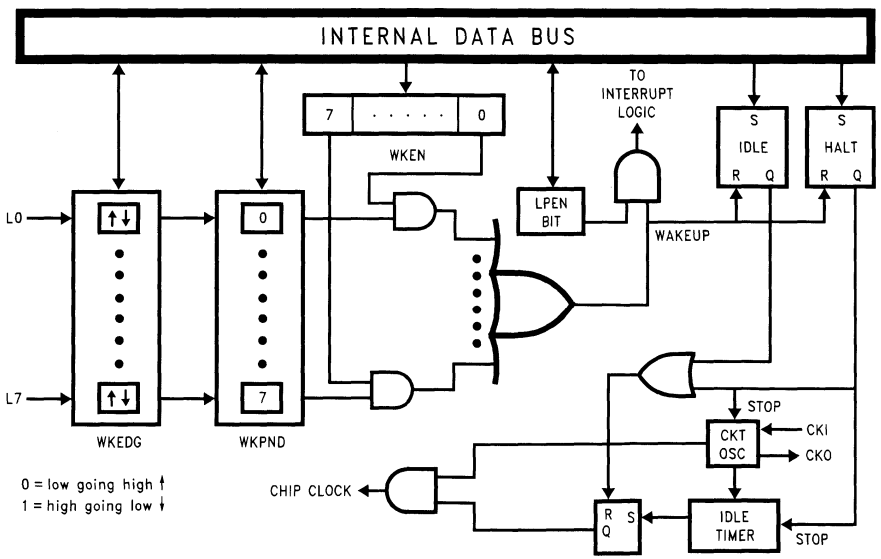
Alternatively, the user can enter the IDLE mode with the IDLE Timer T0 interrupt disabled. In this case, the device will resume normal operation with the instruction immediately following the "Enter IDLE Mode" instruction.

Note: It is necessary to program two NOP instructions following both the set HALT mode and set IDLE mode instructions. These NOP instructions are necessary to allow clock resynchronization following the HALT or IDLE modes.

Multi-Input Wakeup

The Multi-Input Wakeup feature is ued to return (wakeup) the device from either the HALT or IDLE modes. Alternately Multi-Input Wakeup/Interrupt feature may also be used to generate up to 8 edge selectable external interrupts.

Figure 11 shows the Multi-Input Wakeup logic.



TL/DD/12859–14

FIGURE 11. Multi-Input Wake Up Logic

Multi-Input Wakeup (Continued)

The Multi-Input Wakeup feature utilizes the L Port. The user selects which particular L port bit (or combination of L Port bits) will cause the device to exit the HALT or IDLE modes. The selection is done through the Reg: WKEN. The Reg: WKEN is an 8-bit read/write register, which contains a control bit for every L port bit. Setting a particular WKEN bit enables a Wakeup from the associated L port pin.

The user can select whether the trigger condition on the selected L Port pin is going to be either a positive edge (low to high transition) or a negative edge (high to low transition). This selection is made via the Reg: WKEDG, which is an 8-bit control register with a bit assigned to each L Port pin. Setting the control bit will select the trigger condition to be a negative edge on that particular L Port pin. Resetting the bit selects the trigger condition to be a positive edge. Changing an edge select entails several steps in order to avoid a pseudo Wakeup condition as a result of the edge change. First, the associated WKEN bit should be reset, followed by the edge select change in WKEDG. Next, the associated WKPND bit should be cleared, followed by the associated WKEN bit being re-enabled.

An example may serve to clarify this procedure. Suppose we wish to change the edge select from positive (low going high) to negative (high going low) for L Port bit 5, where bit 5 has previously been enabled for an input interrupt. The program would be as follows:

```
RBIT  5, WKEN
SBIT  5, WKEDG
RBIT  5, WKPND
SBIT  5, WKEN
```

If the L port bits have been used as outputs and then changed to inputs with Multi-Input Wakeup/Interrupt, a safety procedure should also be followed to avoid inherited pseudo wakeup conditions. After the selected L port bits have been changed from output to input but before the associated WKEN bits are enabled, the associated edge select bits in WKEDG should be set or reset for the desired edge selects, followed by the associated WKPND bits being cleared.

This same procedure should be used following reset, since the L port inputs are left floating as a result of reset.

The occurrence of the selected trigger condition for Multi Input Wakeup is latched into a pending register called WKPND. The respective bits of the WKPND register will be set on the occurrence of the selected trigger edge on the corresponding Port L pin. The user has the responsibility of clearing these pending flags. Since WKPND is a pending register for the occurrence of selected wakeup conditions, the device will not enter the HALT mode if any Wakeup bit is both enabled and pending. Consequently, the user has the responsibility of clearing the pending flags before attempting to enter the HALT mode.

WKEN, WKPND and WKEDG are all read/write registers, and are cleared at reset.

PORT L INTERRUPTS

Port L provides the user with an additional eight fully selectable, edge sensitive interrupts which are all vectored into the same service subroutine.

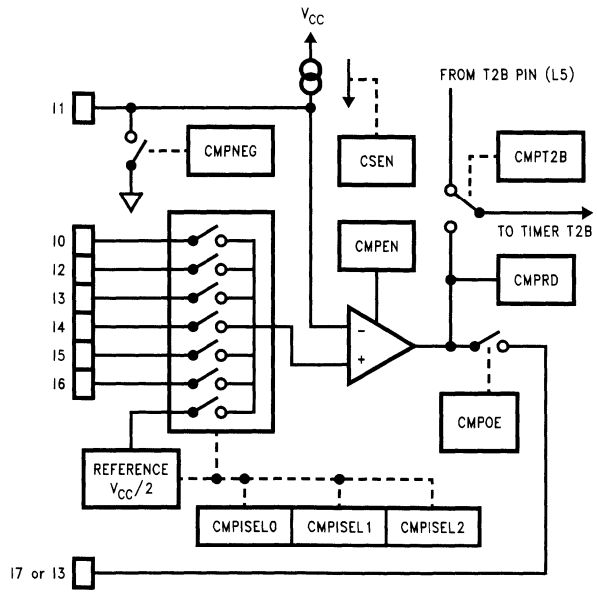
The interrupt from Port L shares logic with the wake up circuitry. The register WKEN allows interrupts from Port L to be individually enabled or disabled. The register WKEDG specifies the trigger condition to be either a positive or a negative edge. Finally, the register WKPND latches in the pending trigger conditions.

The GIE (Global Interrupt Enable) bit enables the interrupt function.

A control flag, LPEN, functions as a global interrupt enable for Port L interrupts. Setting the LPEN flag will enable interrupts and vice versa. A separate global pending flag is not needed since the register WKPND is adequate.

Since Port L is also used for waking the device out of the HALT or IDLE modes, the user can elect to exit the HALT or IDLE modes either with or without the interrupt enabled. If he elects to disable the interrupt, then the device will restart execution from the instruction immediately following the instruction that placed the microcontroller in the HALT or IDLE modes. In the other case, the device will first execute the interrupt service routine and then revert to normal operation. (See HALT MODE for clock option wakeup information.)

Analog Function Block



TL/DD/12859–15

FIGURE 12. COP87L88RK Analog Function Block

This device contains an analog function block with the intent to provide a function which allows for single slope, low cost, A/D conversion of up to 6 channels.

CMPSL REGISTER (ADDRESS X'00B7)

The CMPSL register contains the following bits:

CMPNEG — Will drive I1 to a low level. This bit can be used to discharge an external capacitor. This bit is disabled if the comparator is not enabled (CMPEN = 0).

CMPEN — Enable the comparator ("1" = enable).

CSEN — Enables the internal constant current source. This current source provides a nominal 20 μA constant current at the I1 pin. This current can be used to ensure a linear charging rate on an external capacitor. This bit has no affect and the current source is disabled if the comparator is not enabled (CMPEN = 0).

CMPOE — Enables the comparator output to either pin I3 or pin I7 ("1" = enable) depending on the value of CMPISEL0/1/2.

CMPISEL0/1/2 — Will select one of seven possible sources (I0/I2/I3/I4/I5/I6/internal reference) as a positive input to the comparator (see Table I for more information.)

CMPT2B — Selects the timer T2B input to be driven directly by the comparator output. If the comparator is disabled (CMPEN = 0), this function is disabled, i.e., the T2B input is connected to Port L5.

CMPT2B	CMPISEL2	CMPISEL1	CMPISEL0	CMPOE	CSEN	CMPEN	CMPNEG

Bit 7 Bit 0

The Comparator Select Register is cleared on RESET (the comparator is disabled). To save power the program should also disable the comparator before the μC enters the HALT/IDLE modes. Disabling the comparator will turn off the constant current source and the $V_{CC}/2$ reference, disconnect the comparator output from the T2B input and pin I3 or I7 and remove the low on I1 caused by CMPNEG.

It is often useful for the user's program to read the result of a comparator operation. Since I1 is always selected to be COMPIN− when the comparator is enabled (CMPEN = 1), the comparator output can be read internally by reading bit 1 (CMPRD) of register PORTI (RAM address 0 x D7).

The following table lists the comparator inputs and outputs vs. the value of the CMPISEL0/1/2 bits. The output will only be driven if the CMPOE bit is set to 1.

Analog Function Block (Continued)

TABLE I. Comparator Input Selection

Control Bit			Comparator Input Source		Comparator Output
CMPISEL2	CMPISEL1	CMPISEL0	Neg. Input	Pos. Input	
0	0	0	I1	I2	I3
0	0	1	I1	I2	I7
0	1	0	I1	I3	I7
0	1	1	I1	I0	I7
1	0	0	I1	I4	I7
1	0	1	I1	I5	I7
1	1	0	I1	I6	I7
1	1	1	I1	$V_{CC}/2$ Ref.	I7

Reset

The state of the Comparator Block immediately after RESET is as follows:

1. The CMPSL Register is set to all zeros
2. The Comparator is disabled
3. The Constant Current Source is disabled
4. CMPNEG is turned off
5. The Port I inputs are electrically isolated from the comparator
6. The T2B input is as normally selected by the T2CNTRL Register
7. CMPISEL0–CMPISEL2 are set to zero
8. All Port I inputs are selected to the default digital input mode

The comparator outputs have the same specification as Ports L and G except that the rise and fall times are symmetrical.

Interrupts

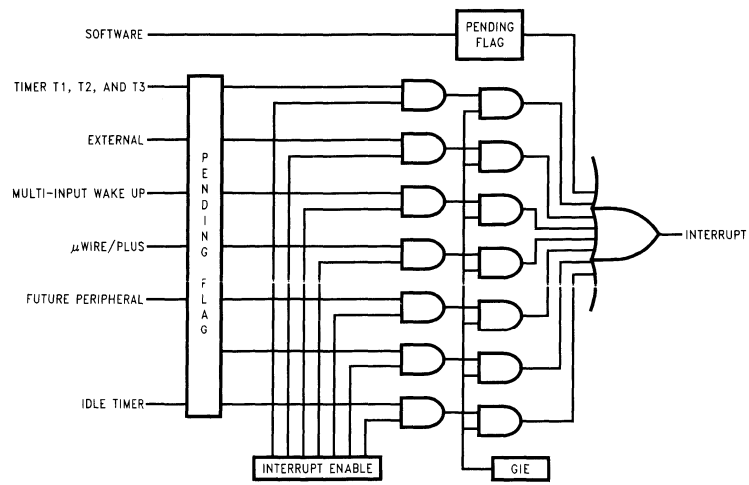
The device supports a vectored interrupt scheme. It supports a total of fourteen interrupt sources. The following table lists all the possible interrupt sources, their arbitration ranking and the memory locations reserved for the interrupt vector for each source.

Two bytes of program memory space are reserved for each interrupt source. All interrupt sources except the software interrupt are maskable. Each of the maskable interrupts have an Enable bit and a Pending bit. A maskable interrupt is active if its associated enable and pending bits are set. If GIE = 1 and an interrupt is active, then the processor will be interrupted as soon as it is ready to start executing an instruction except if the above conditions happen during the Software Trap service routine. This exception is described in the Software Trap sub-section.

The interruption process is accomplished with the INTR instruction (opcode 00), which is jammed inside the Instruction Register and replaces the opcode about to be executed. The following steps are performed for every interrupt:

1. The GIE (Global Interrupt Enable) bit is reset.
2. The address of the instruction about to be executed is pushed into the stack.
3. The PC (Program Counter) branches to address 00FF.

This procedure takes 7 t_C cycles to execute.



TL/DD/12859–16

FIGURE 13. Interrupt Block Diagram

Interrupts (Continued)

Arbitration Ranking	Source	Description	Vector* Address Hi-Low Byte
(1) Highest	Software	INTR Instruction	0yFE-0yFF
(2)	Reserved		0yFC-0yFD
(3)	External	G0	0yFA-0yFB
(4)	Timer T0	Underflow	0yF8-0yF9
(5)	Timer T1	T1A/Underflow	0yF6-0yF7
(6)	Timer T1	T1B	0yF4-0yF5
(7)	MICROWIRE/PLUS	BUSY Low	0yF2-0yF3
(8)	Reserved		0yF0-0yF1
(9)	Reserved		0yEE-0yEF
(10)	Reserved		0yEC-0yED
(11)	Timer T2	T2A/Underflow	0yEA-0yEB
(12)	Timer T2	T2B	0yE8-0yE9
(13)	Timer T3	T3A/Underflow	0yE6-0yE7
(14)	Timer T3	T3B	0yE4-0yE5
(15)	Port L/Wakeup	Port L Edge	0yE2-0yE3
(16) Lowest	Default	VIS Instr. Execution without Any Interrupts	0yE0-0yE1

*y is a variable which represents the VIS block. VIS and the vector table must be located in the same 256-byte block except if VIS is located at the last address of a block. In this case, the table must be in the next block.

At this time, since GIE = 0, other maskable interrupts are disabled. The user is now free to do whatever context switching is required by saving the context of the machine in the stack with PUSH instructions. The user would then program a VIS (Vector Interrupt Select) instruction in order to branch to the interrupt service routine of the highest priority interrupt enabled and pending at the time of the VIS. Note that this is not necessarily the interrupt that caused the branch to address location 00FF Hex prior to the context switching.

Thus, if an interrupt with a higher rank than the one which caused the interruption becomes active before the decision of which interrupt to service is made by the VIS, then the interrupt with the higher rank will override any lower ones and will be acknowledged. The lower priority interrupt(s) are still pending, however, and will cause another interrupt immediately following the completion of the interrupt service routine associated with the higher priority interrupt just serviced. This lower priority interrupt will occur immediately following the RETI (Return from Interrupt) instruction at the end of the interrupt service routine just completed.

Inside the interrupt service routine, the associated pending bit has to be cleared by software. The RETI (Return from Interrupt) instruction at the end of the interrupt service routine will set the GIE (Global Interrupt Enable) bit, allowing the processor to be interrupted again if another interrupt is active and pending.

The VIS instruction looks at all the active interrupts at the time it is executed and performs an indirect jump to the beginning of the service routine of the one with the highest rank.

The addresses of the different interrupt service routines, called vectors, are chosen by the user and stored in ROM in a table starting at 01E0 (assuming that VIS is located between 00FF and 01DF). The vectors are 15-bit wide and therefore occupy 2 ROM locations.

VIS and the vector table must be located in the same 256-byte block (0y00 to 0yFF) except if VIS is located at the last address of a block. In this case, the table must be in the next block. The vector table cannot be inserted in the first 256-byte block (y ≠ 0).

The vector of the maskable interrupt with the lowest rank is located at 0yE0 (Hi-Order byte) and 0yE1 (Lo-Order byte) and so forth in increasing rank number. The vector of the maskable interrupt with the highest rank is located at 0yFA (Hi-Order byte) and 0yFB (Lo-Order byte).

The Software Trap has the highest rank and its vector is located at 0yFE and 0yFF.

If, by accident, a VIS gets executed and no interrupt is active, then the PC (Program Counter) will branch to a vector located at 0yE0-0yE1.

WARNING

A Default VIS interrupt handler routine must be present. As a minimum, this handler should confirm that the GIE bit is cleared (this indicates that the interrupt sequence has been taken), take care of any required housekeeping, restore context and return. Some sort of Warm Restart procedure should be implemented. These events can occur without any error on the part of the system designer or programmer.

Note: There is always the possibility of an interrupt occurring during an instruction which is attempting to reset the GIE bit or any other interrupt enable bit. If this occurs when a single cycle instruction is being used to reset the interrupt enable bit, the interrupt enable bit will be reset but an interrupt may still occur. This is because interrupt processing is started at the same time as the interrupt bit is being reset. To avoid this scenario, the user should always use a two, three, or four cycle instruction to reset interrupt enable bits.

Figure 13 shows the Interrupt block diagram.

Interrupts (Continued)

SOFTWARE TRAP

The Software Trap (ST) is a special kind of non-maskable interrupt which occurs when the INTR instruction (used to acknowledge interrupts) is fetched from ROM and placed inside the instruction register. This may happen when the PC is pointing beyond the available ROM address space or when the stack is over-popped.

When an ST occurs, the user can re-initialize the stack pointer and do a recovery procedure (similar to reset, but not necessarily containing all of the same initialization procedures) before restarting.

The occurrence of an ST is latched into the ST pending bit. The GIE bit is not affected and the ST pending bit **(not accessible by the user)** is used to inhibit other interrupts and to direct the program to the ST service routine with the VIS instruction. The RPND instruction is used to clear the software interrupt pending bit. This pending bit is also cleared on reset.

The ST has the highest rank among all interrupts.

Nothing (except another ST) can interrupt an ST being serviced.

WATCHDOG

The device contains a WATCHDOG and clock monitor. The WATCHDOG is designed to detect the user program getting stuck in infinite loops resulting in loss of program control or "runaway" programs. The Clock Monitor is used to detect the absence of a clock or a very slow clock below a specified rate on the CKI pin.

The WATCHDOG consists of two independent logic blocks: WD UPPER and WD LOWER. WD UPPER establishes the upper limit on the service window and WD LOWER defines the lower limit of the service window.

Servicing the WATCHDOG consists of writing a specific value to a WATCHDOG Service Register named WDSVR which is memory mapped in the RAM. This value is composed of three fields, consisting of a 2-bit Window Select, a 5-bit Key Data field, and the 1-bit Clock Monitor Select field. Table II shows the WDSVR register.

TABLE II. WATCHDOG Service Register (WDSVR)

Window Select		Key Data					Clock Monitor
X	X	0	1	1	0	0	Y
7	6	5	4	3	2	1	0

The lower limit of the service window is fixed at 2048 instruction cycles. Bits 7 and 6 of the WDSVR register allow the user to pick an upper limit of the service window.

Table III shows the four possible combinations of lower and upper limits for the WATCHDOG service window. This flexibility in choosing the WATCHDOG service window prevents any undue burden on the user software.

Bits 5, 4, 3, 2 and 1 of the WDSVR register represent the 5-bit Key Data field. The key data is fixed at 01100. Bit 0 of the WDSVR Register is the Clock Monitor Select bit.

TABLE III. WATCHDOG Service Window Select

WDSVR Bit 7	WDSVR Bit 6	Service Window (Lower-Upper Limits)
0	0	2k–8k t_C Cycles
0	1	2k–16k t_C Cycles
1	0	2k–32k t_C Cycles
1	1	2k–64k t_C Cycles

Clock Monitor

The Clock Monitor aboard the device can be selected or deselected under program control. The Clock Monitor is guaranteed not to reject the clock if the instruction cycle clock ($1/t_C$) is greater or equal to 10 kHz. This equates to a clock input rate on CKI of greater or equal to 100 kHz.

WATCHDOG Operation

The WATCHDOG and Clock Monitor are disabled during reset. The device comes out of reset with the WATCHDOG armed, the WATCHDOG Window Select bits (bits 6, 7 of the WDSVR Register) set, and the Clock Monitor bit (bit 0 of the WDSVR Register) enabled. Thus, a Clock Monitor error will occur after coming out of reset, if the instruction cycle clock frequency has not reached a minimum specified value, including the case where the oscillator fails to start.

The WDSVR register can be written to only once after reset and the key data (bits 5 through 1 of the WDSVR Register) must match to be a valid write. This write to the WDSVR register involves two irrevocable choices: (i) the selection of the WATCHDOG service window (ii) enabling or disabling of the Clock Monitor. Hence, the first write to WDSVR Register involves selecting or deselecting the Clock Monitor, select the WATCHDOG service window and match the WATCHDOG key data. Subsequent writes to the WDSVR register will compare the value being written by the user to the WATCHDOG service window value and the key data (bits 7 through 1) in the WDSVR Register. Table IV shows the sequence of events that can occur.

The user must service the WATCHDOG at least once before the upper limit of the service window expires. The WATCHDOG may not be serviced more than once in every lower limit of the service window. The user may service the WATCHDOG as many times as wished in the time period between the lower and upper limits of the service window. The first write to the WDSVR Register is also counted as a WATCHDOG service.

The WATCHDOG has an output pin associated with it. This is the WDOUT pin, on pin 1 of the port G. WDOUT is active low. The WDOUT pin is in the high impedance state in the inactive state. Upon triggering the WATCHDOG, the logic will pull the WDOUT (G1) pin low for an additional 16 t_C–32 t_C cycles after the signal level on WDOUT pin goes below the lower Schmitt trigger threshold. After this delay, the device will stop forcing the WDOUT output low.

The WATCHDOG service window will restart when the WDOUT pin goes high. It is recommended that the user tie the WDOUT pin back to V_{CC} through a resistor in order to pull WDOUT high.

WATCHDOG Operation (Continued)

A WATCHDOG service while the WDOUT signal is active will be ignored. The state of the WDOUT pin is not guaranteed on reset, but if it powers up low then the WATCHDOG will time out and WDOUT will enter high impedance state.

The Clock Monitor forces the G1 pin low upon detecting a clock frequency error. The Clock Monitor error will continue until the clock frequency has reached the minimum specified value, after which the G1 output will enter the high impedance TRI-STATE mode following 16 t_c–32 t_c clock cycles. The Clock Monitor generates a continual Clock Monitor error if the oscillator fails to start, or fails to reach the minimum specified frequency. The specification for the Clock Monitor is as follows:

$1/t_c > 10$ kHz—No clock rejection.

$1/t_c < 10$ Hz—Guaranteed clock rejection.

WATCHDOG AND CLOCK MONITOR SUMMARY

The following salient points regarding the WATCHDOG and CLOCK MONITOR should be noted:

- Both the WATCHDOG and CLOCK MONITOR detector circuits are inhibited during RESET.
- Following RESET, the WATCHDOG and CLOCK MONITOR are both enabled, with the WATCHDOG having he maximum service window selected.
- The WATCHDOG service window and CLOCK MONITOR enable/disable option can only be changed once, during the initial WATCHDOG service following RESET.
- The initial WATCHDOG service must match the key data value in the WATCHDOG Service register WDSVR in order to avoid a WATCHDOG error.
- Subsequent WATCHDOG services must match all three data fields in WDSVR in order to avoid WATCHDOG errors.
- The correct key data value cannot be read from the WATCHDOG Service register WDSVR. Any attempt to read this key data value of 01100 from WDSVR will read as key data value of all 0's.
- The WATCHDOG detector circuit is inhibited during both the HALT and IDLE modes.
- The CLOCK MONITOR detector circuit is active during both the HALT and IDLE modes. Consequently, the device inadvertently entering the HALT mode will be detected as a CLOCK MONITOR error (provided that the CLOCK MONITOR enable option has been selected by the program).
- With the single-pin R/C oscillator mask option selected and the CLKDLY bit reset, the WATCHDOG service window will resume following HALT mode from where it left off before entering the HALT mode.
- With the crystal oscillator mask option selected, or with the single-pin R/C oscillator mask option selected and the CLKDLY bit set, the WATCHDOG service window will be set to its selected value from WDSVR following HALT. Consequently, the WATCHDOG should not be serviced for at least 2048 instruction cycles following HALT, but must be serviced within the selected window to avoid a WATCHDOG error.
- The IDLE timer T0 is not initialized with RESET.
- The user can sync in to the IDLE counter cycle with an IDLE counter (T0) interrupt or by monitoring the T0PND flag. The T0PND flag is set whenever the thirteenth bit of the IDLE counter toggles (every 4096 instruction cycles). The user is responsible for resetting the T0PND flag.
- A hardware WATCHDOG service occurs just as the device exits the IDLE mode. Consequently, the WATCHDOG should not be serviced for at least 2048 instruction cycles following IDLE, but must be serviced within the selected window to avoid a WATCHDOG error.
- Following RESET, the initial WATCHDOG service (where the service window and the CLOCK MONITOR enable/disable must be selected) may be programmed anywhere within the maximum service window (65,536 instruction cycles) initialized by RESET. Note that this initial WATCHDOG service may be programmed within the initial 2048 instruction cycles without causing a WATCHDOG error.

Detection of Illegal Conditions

The device can detect various illegal conditions resulting from coding errors, transient noise, power supply voltage drops, runaway programs, etc.

Reading of undefined ROM gets zeros. The opcode for software interrupt is zero. If the program fetches instructions from undefined ROM, this will force a software interrupt, thus signaling that an illegal condition has occurred.

The subroutine stack grows down for each call (jump to subroutine), interrupt, or PUSH, and grows up for each return or POP. The stack pointer is initialized to RAM location 06F Hex during reset. Consequently, if there are more returns than calls, the stack pointer will point to addresses 070 and 071 Hex (which are undefined RAM). Undefined RAM from addresses 070 to 07F (Segment 0), 140 to 17F (Segment 1), and all other segments (i.e., Segments 2 . . . etc.) is read as all 1's, which in turn will cause the program to return to address 7FFF Hex. This is an undefined ROM location and the instruction fetched (all 0's) from this location will generate a software interrupt signaling an illegal condition.

Thus, the chip can detect the following illegal conditions:

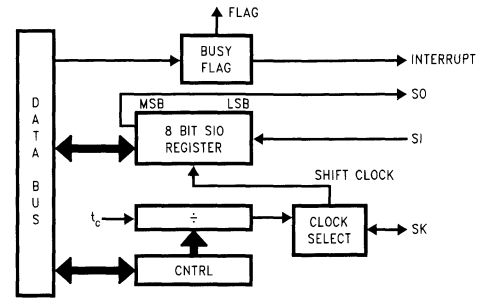
a. Executing from undefined ROM

b. Over "POP"ing the stack by having more returns than calls.

When the software interrupt occurs, the user can re-initialize the stack pointer and do a recovery procedure before restarting (this recovery program is probably similar to that following reset, but might not contain the same program initialization procedures). The recovery program should reset the software interrupt pending bit using the RPND instruction.

MICROWIRE/PLUS

MICROWIRE/PLUS is a serial synchronous communications interface. The MICROWIRE/PLUS capability enables the device to interface with any of National Semiconductor's MICROWIRE peripherals (i.e. A/D converters, display drivers, E^{2}PROMs etc.) and with other microcontrollers which support the MICROWIRE interface. It consists of an 8-bit serial shift register (SIO) with serial data input (SI), serial data output (SO) and serial shift clock (SK). *Figure 14* shows a block diagram of the MICROWIRE/PLUS logic.



TL/DD/12859–17

FIGURE 14. MICROWIRE/PLUS Block Diagram

The shift clock can be selected from either an internal source or an external source. Operating the MICROWIRE/PLUS arrangement with the internal clock source is called the Master mode of operation. Similarly, operating the MICROWIRE/PLUS arrangement with an external shift clock is called the Slave mode of operation.

The CNTRL register is used to configure and control the MICROWIRE/PLUS mode. To use the MICROWIRE/PLUS, the MSEL bit in the CNTRL register is set to one. In the master mode, the SK clock rate is selected by the two bits, SL0 and SL1, in the CNTRL register. Table V details the different clock rates that may be selected.

TABLE IV. WATCHDOG Service Actions

Key Data	Window Data	Clock Monitor	Action
Match	Match	Match	Valid Service: Restart Service Window
Don't Care	Mismatch	Don't Care	Error: Generate WATCHDOG Output
Mismatch	Don't Care	Don't Care	Error: Generate WATCHDOG Output
Don't Care	Don't Care	Mismatch	Error: Generate WATCHDOG Output

TABLE V. MICROWIRE/PLUS Master Mode Clock Select

SL1	SL0	SK
0	0	$2 \times t_c$
0	1	$4 \times t_c$
1	x	$8 \times t_c$

where t_c is the instruction cycle clock

MICROWIRE/PLUS (Continued)

MICROWIRE/PLUS OPERATION

Setting the BUSY bit in the PSW register causes the MICROWIRE/PLUS to start shifting the data. It gets reset when eight data bits have been shifted. The user may reset the BUSY bit by software to allow less than 8 bits to shift. If enabled, an interrupt is generated when eight data bits have been shifted. The device may enter the MICROWIRE/PLUS mode either as a Master or as a Slave. *Figure 15* shows how two microcontroller devices and several peripherals may be interconnected using the MICROWIRE/PLUS arrangements.

Warning:

The SIO register should only be loaded when the SK clock is low. Loading the SIO register while the SK clock is high will result in undefined data in the SIO register. SK clock is normally low when not shifting.

Setting the BUSY flag when the input SK clock is high in the MICROWIRE/PLUS slave mode may cause the current SK clock for the SIO shift register to be narrow. For safety, the BUSY flag should only be set when the input SK clock is low.

MICROWIRE/PLUS Master Mode Operation

In the MICROWIRE/PLUS Master mode of operation the shift clock (SK) is generated internally by the device. The MICROWIRE Master always initiates all data exchanges. The MSEL bit in the CNTRL register must be set to enable the SO and SK functions onto the G Port. The SO and SK pins must also be selected as outputs by setting appropriate bits in the Port G configuration register. Table VI summarizes the bit settings required for Master mode of operation.

MICROWIRE/PLUS Slave Mode Operation

In the MICROWIRE/PLUS Slave mode of operation the SK clock is generated by an external source. Setting the MSEL bit in the CNTRL register enables the SO and SK functions onto the G Port. The SK pin must be selected as an input and the SO pin is selected as an output pin by setting and resetting the appropriate bits in the Port G configuration register. Table VI summarizes the settings required to enter the Slave mode of operation.

The user must set the BUSY flag immediately upon entering the Slave mode. This will ensure that all data bits sent by the Master will be shifted properly. After eight clock pulses the BUSY flag will be cleared and the sequence may be repeated.

Alternate SK Phase Operation

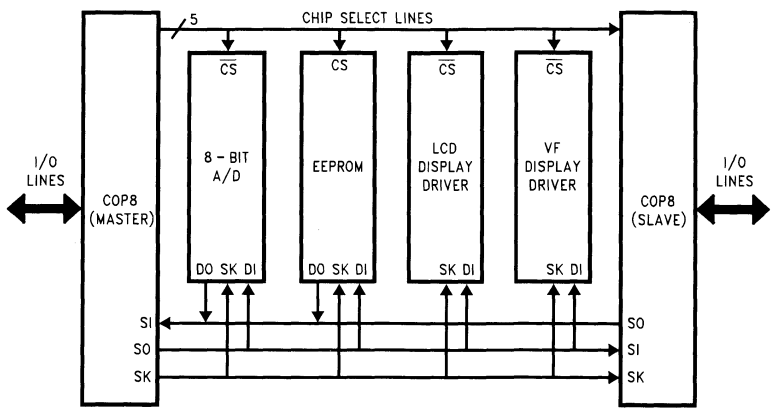
The device allows either the normal SK clock or an alternate phase SK clock to shift data in and out of the SIO register. In both the modes the SK is normally low. In the normal mode data is shifted in on the rising edge of the SK clock and the data is shifted out on the falling edge of the SK clock. The SIO register is shifted on each falling edge of the SK clock. In the alternate SK phase operation, data is shifted in on the falling edge of the SK clock and shifted out on the rising edge of the SK clock.

A control flag, SKSEL, allows either the normal SK clock or the alternate SK clock to be selected. Resetting SKSEL causes the MICROWIRE/PLUS logic to be clocked from the normal SK signal. Setting the SKSEL flag selects the alternate SK clock. The SKSEL is mapped into the G6 configuration bit. The SKSEL flag will power up in the reset condition, selecting the normal SK signal.

TABLE VI

This table assumes that the control flag MSEL is set.

G4 (SO) Config. Bit	G5 (SK) Config. Bit	G4 Fun.	G5 Fun.	Operation
1	1	SO	Int. SK	MICROWIRE/PLUS Master
0	1	TRI-STATE	Int. SK	MICROWIRE/PLUS Master
1	0	SO	Ext. SK	MICROWIRE/PLUS Slave
0	0	TRI-STATE	Ext. SK	MICROWIRE/PLUS Slave



TL/DD/12859-18

FIGURE 15. MICROWIRE/PLUS Application

Memory Map

All RAM, ports and registers (except A and PC) are mapped into data memory address space.

Address S/ADD REG	Contents
0000 to 006F	On-Chip RAM bytes (112 bytes)
0070 to 007F	Unused RAM Address Space (Reads As All Ones)
xx80 to xxAF	Unused RAM Address Space (Reads Undefined Data)
xxB0	Timer T3 Lower Byte
xxB1	Timer T3 Upper Byte
xxB2	Timer T3 Autoload Register T3RA Lower Byte
xxB3	Timer T3 Autoload Register T3RA Upper Byte
xxB4	Timer T3 Autoload Register T3RB Lower Byte
xxB5	Timer T3 Autoload Register T3RB Upper Byte
xxB6	Timer T3 Control Register
xxB7	Comparator Select Register (CMPSL)
xxB8 to xxBF	Reserved
xxC0	Timer T2 Lower Byte
xxC1	Timer T2 Upper Byte
xxC2	Timer T2 Autoload Register T2RA Lower Byte
xxC3	Timer T2 Autoload Register T2RA Upper Byte
xxC4	Timer T2 Autoload Register T2RB Lower Byte
xxC5	Timer T2 Autoload Register T2RB Upper Byte
xxC6	Timer T2 Control Register
xxC7	WATCHDOG Service Register (Reg:WDSVR)
xxC8	MIWU Edge Select Register (Reg:WKEDG)
xxC9	MIWU Enable Register (Reg:WKEN)
xxCA	MIWU Pending Register (Reg:WKPND)
xxCB	Reserved
xxCC	Reserved
xxCD to xxCF	Reserved

Address S/ADD REG	Contents
xxD0	Port L Data Register
xxD1	Port L Configuration Register
xxD2	Port L Input Pins (Read Only)
xxD3	Reserved for Port L
xxD4	Port G Data Register
xxD5	Port G Configuration Register
xxD6	Port G Input Pins (Read Only)
xxD7	Port I Input Pins (Read Only)
xxD8	Port C Data Register
xxD9	Port C Configuration Register
xxDA	Port C Input Pins (Read Only)
xxDB	Reserved for Port C
xxDC	Port D
xxDD to xxDF	Reserved
xxE0 to xxE5	Reserved
xxE6	Timer T1 Autoload Register T1RB Lower Byte
xxE7	Timer T1 Autoload Register T1RB Upper Byte
xxE8	ICNTRL Register
xxE9	MICROWIRE/PLUS Shift Register
xxEA	Timer T1 Lower Byte
xxEB	Timer T1 Upper Byte
xxEC	Timer T1 Autoload Register T1RA Lower Byte
xxED	Timer T1 Autoload Register T1RA Upper Byte
xxEE	CNTRL Control Register
xxEF	PSW Register
xxF0 to FB	On-Chip RAM Mapped as Registers
xxFC	X Register
xxFD	SP Register
xxFE	B Register
xxFF	S Register
0100 to 017F	On-Chip 128 RAM Bytes

Reading memory locations 0070H–007FH (Segment 0) will return all ones. Reading unused memory locations 0080H–00AFH (Segment 0) will return undefined data. Reading memory locations from other unused Segments (i.e., Segment 2, Segment 3, ... etc.) will return all ones.

Addressing Modes

There are ten addressing modes, six for operand addressing and four for transfer of control.

OPERAND ADDRESSING MODES

Register Indirect

This is the "normal" addressing mode. The operand is the data memory addressed by the B pointer or X pointer.

Register Indirect (with auto post increment or decrement of pointer)

This addressing mode is used with the LD and X instructions. The operand is the data memory addressed by the B pointer or X pointer. This is a register indirect mode that automatically post increments or decrements the B or X register after executing the instruction.

Direct

The instruction contains an 8-bit address field that directly points to the data memory for the operand.

Immediate

The instruction contains an 8-bit immediate field as the operand.

Short Immediate

This addressing mode is used with the Load B Immediate instruction. The instruction contains a 4-bit immediate field as the operand.

Indirect

This addressing mode is used with the LAID instruction. The contents of the accumulator are used as a partial address (lower 8 bits of PC) for accessing a data operand from the program memory.

TRANSFER OF CONTROL ADDRESSING MODES

Relative

This mode is used for the JP instruction, with the instruction field being added to the program counter to get the new program location. JP has a range from −31 to +32 to allow a 1-byte relative jump (JP + 1 is implemented by a NOP instruction). There are no "pages" when using JP, since all 15 bits of PC are used.

Absolute

This mode is used with the JMP and JSR instructions, with the instruction field of 12 bits replacing the lower 12 bits of the program counter (PC). This allows jumping to any location in the current 4k program memory segment.

Absolute Long

This mode is used with the JMPL and JSRL instructions, with the instruction field of 15 bits replacing the entire 15 bits of the program counter (PC). This allows jumping to any location up to 32k in the program memory space.

Indirect

This mode is used with the JID instruction. The contents of the accumulator are used as a partial address (lower 8 bits of PC) for accessing a location in the program memory. The contents of this program memory location serve as a partial address (lower 8 bits of PC) for the jump to the next instruction.

The VIS is a special case of the Indirect Transfer of Control addressing mode, where the double byte vector associated with the interrupt is transferred from adjacent addresses in the program memory into the program counter (PC) in order to jump to the associated interrupt service routine.

Instruction Set

Register and Symbol Definition

Registers	
A	8-Bit Accumulator Register
B	8-Bit Address Register
X	8-Bit Address Register
S	8-Bit Segment Register
SP	8-Bit Stack Pointer Register
PC	15-Bit Program Counter Register
PU	Upper 7 Bits of PC
PL	Lower 8 Bits of PC
C	1 Bit of PSW Register for Carry
HC	1 Bit of PSW Register for Half Carry
GIE	1 Bit of PSW Register for Global Interrupt Enable
VU	Interrupt Vector Upper Byte
VL	Interrupt Vector Lower Byte

Symbols	
[B]	Memory Indirectly Addressed by B Register
[X]	Memory Indirectly Addressed by X Register
MD	Direct Addressed Memory
Mem	Direct Addressed Memory or [B]
Meml	Direct Addressed Memory or [B] or Immediate Data
Imm	8-Bit Immediate Data
Reg	Register Memory: Addresses F0 to FF (Includes B, X and SP)
Bit	Bit Number (0 to 7)
←	Loaded with
↔	Exchanged with

Instruction Set (Continued)

INSTRUCTION SET

ADD	A,Meml	ADD	A ← A + Meml
ADC	A,Meml	ADD with Carry	A ← A + Meml + C, C ← Carry HC ← Half Carry
SUBC	A,Meml	Subtract with Carry	A ← A − $\overline{\text{Meml}}$ + C, C ← Carry HC ← Half Carry
AND	A,Meml	Logical AND	A ← A and Meml
ANDSZ	A,Imm	Logical AND Immed., Skip if Zero	Skip next if (A and Imm) = 0
OR	A,Meml	Logical OR	A ← A or Meml
XOR	A,Meml	Logical EXclusive OR	A ← A xor Meml
IFEQ	MD,Imm	IF EQual	Compare MD and Imm, Do next if MD = Imm
IFEQ	A,Meml	IF EQual	Compare A and Meml, Do next if A = Meml
IFNE	A,Meml	IF Not Equal	Compare A and Meml, Do next if A ≠ Meml
IFGT	A,Meml	IF Greater Than	Compare A and Meml, Do next if A > Meml
IFBNE	#	If B Not Equal	Do next if lower 4 bits of B ≠ Imm
DRSZ	Reg	Decrement Reg., Skip if Zero	Reg ← Reg − 1, Skip if Reg = 0
SBIT	#,Mem	Set BIT	1 to bit, Mem (bit = 0 to 7 immediate)
RBIT	#,Mem	Reset BIT	0 to bit, Mem
IFBIT	#,Mem	IF BIT	If bit in A or Mem is true do next instruction
RPND		Reset PeNDing Flag	Reset Software Interrupt Pending Flag
X	A,Mem	EXchange A with Memory	A ⟷ Mem
X	A,[X]	EXchange A with Memory [X]	A ⟷ [X]
LD	A,Meml	LoaD A with Memory	A ← Meml
LD	A,[X]	LoaD A with Memory [X]	A ← [X]
LD	B,Imm	LoaD B with Immed.	B ← Imm
LD	Mem,Imm	LoaD Memory Immed	Mem ← Imm
LD	Reg,Imm	LoaD Register Memory Immed.	Reg ← Imm
X	A, [B ±]	EXchange A with Memory [B]	A ⟷ [B], (B ← B ± 1)
X	A, [X ±]	EXchange A with Memory [X]	A ⟷ [X], (X ← ± 1)
LD	A, [B±]	LoaD A with Memory [B]	A ← [B], (B ← B ± 1)
LD	A, [X±]	LoaD A with Memory [X]	A ← [X], (X ← X ± 1)
LD	[B±],Imm	LoaD Memory [B] Immed.	[B] ← Imm, (B ← B ± 1)
CLR	A	CLeaR A	A ← 0
INC	A	INCrement A	A ← A + 1
DEC	A	DECrement A	A ← A − 1
LAID		Load A InDirect from ROM	A ← ROM (PU,A)
DCOR	A	Decimal CORrect A	A ← BCD correction of A (follows ADC, SUBC)
RRC	A	Rotate A Right thru C	C → A7 → . . . → A0 → C
RLC	A	Rotate A Left thru C	C ← A7 ← . . . ← A0 ← C
SWAP	A	SWAP nibbles of A	A7 . . . A4 ⟷ A3 . . . A0
SC		Set C	C ← 1, HC ← 1
RC		Reset C	C ← 0, HC ← 0
IFC		IF C	IF C is true, do next instruction
IFNC		IF Not C	If C is not true, do next instruction
POP	A	POP the stack into A	SP ← SP + 1, A ← [SP]
PUSH	A	PUSH A onto the stack	[SP] ← A, SP ← SP − 1
VIS		Vector to Interrupt Service Routine	PU ← [VU], PL ← [VL]
JMPL	Addr.	Jump absolute Long	PC ← ii (ii = 15 bits, 0 to 32k)
JMP	Addr.	Jump absolute	PC9 . . . 0 ← i (i = 12 bits)
JP	Disp.	Jump relative short	PC ← PC + r (r is −31 to +32, except 1)
JSRL	Addr.	Jump SubRoutine Long	[SP] ← PL, [SP−1] ← PU,SP−2, PC ← ii
JSR	Addr	Jump SubRoutine	[SP] ← PL, [SP−1] ← PU,SP−2, PC9 . . . 0 ← i
JID		Jump InDirect	PL ← ROM (PU,A)
RET		RETurn from subroutine	SP + 2, PL ← [SP], PU ← [SP−1]
RETSK		RETurn and SKip	SP + 2, PL ← [SP],PU ← [SP−1]
RETI		RETurn from Interrupt	SP + 2, PL ← [SP],PU ← [SP−1],GIE ← 1
INTR		Generate an Interrupt	[SP] ← PL, [SP−1] ← PU, SP−2, PC ← 0FF
NOP		No OPeration	PC ← PC + 1

Instruction Execution Time

Most instructions are single byte (with immediate addressing mode instructions taking two bytes).

Most single byte instructions take one cycle time to execute.

Skipped instructions require x number of cycles to be skipped, where x equals the number of bytes in the skipped instruction opcode.

See the BYTES and CYCLES per INSTRUCTION table for details.

Bytes and Cycles per Instruction

The following table shows the number of bytes and cycles for each instruction in the format of byte/cycle.

Arithmetic and Logic Instructions

	[B]	Direct	Immed.
ADD	1/1	3/4	2/2
ADC	1/1	3/4	2/2
SUBC	1/1	3/4	2/2
AND	1/1	3/4	2/2
OR	1/1	3/4	2/2
XOR	1/1	3/4	2/2
IFEQ	1/1	3/4	2/2
IFNE	1/1	3/4	2/2
IFGT	1/1	3/4	2/2
IFBNE	1/1		
DRSZ		1/3	
SBIT	1/1	3/4	
RBIT	1/1	3/4	
IFBIT	1/1	3/4	

RPND	1/1

Instructions Using A & C

CLRA	1/1
INCA	1/1
DECA	1/1
LAID	1/3
DCOR	1/1
RRCA	1/1
RLCA	1/1
SWAPA	1/1
SC	1/1
RC	1/1
IFC	1/1
IFNC	1/1
PUSHA	1/3
POPA	1/3
ANDSZ	2/2

Transfer of Control Instructions

JMPL	3/4
JMP	2/3
JP	1/3
JSRL	3/5
JSR	2/5
JID	1/3
VIS	1/5
RET	1/5
RETSK	1/5
RETI	1/5
INTR	1/7
NOP	1/1

Memory Transfer Instructions

	Register Indirect		Direct	Immed.	Register Indirect Auto Incr. & Decr.		
	[B]	[X]			[B+, B−]	[X+, X−]	
X A,*	1/1	1/3	2/3		1/2	1/3	
LD A,*	1/1	1/3	2/3	2/2	1/2	1/3	
LD B, Imm				1/1			(IF B < 16)
LD B, Imm				2/2			(IF B > 15)
LD Mem, Imm	2/2		3/3		2/2		
LD Reg, Imm			2/3				
IFEQ MD, Imm			3/3				

* = > Memory location addressed by B or X or directly.

Opcode Table

Upper Nibble																Lower Nibble
F	E	D	C	B	A	9	8	7	6	5	4	3	2	1	0	
JP−15	JP−31	LD 0F0, # i	DRSZ 0F0	RRCA	RC	ADC A,#i	ADC A,[B]	IFBIT 0,[B]	ANDSZ A, #i	LD B,#0F	IFBNE 0	JSR x000−x0FF	JMP x000−x0FF	JP+17	INTR	0
JP−14	JP−30	LD 0F1, # i	DRSZ 0F1	*	SC	SUBC A, #i	SUB A,[B]	IFBIT 1,[B]	*	LD B,#0E	IFBNE 1	JSR x100−x1FF	JMP x100−x1FF	JP+18	JP+2	1
JP−13	JP−29	LD 0F2, # i	DRSZ 0F2	X A,[X+]	X A,[B+]	IFEQ A,#i	IFEQ A,[B]	IFBIT 2,[B]	*	LD B,#0D	IFBNE 2	JSR x200−x2FF	JMP x200−x2FF	JP+19	JP+3	2
JP−12	JP−28	LD 0F3, # i	DRSZ 0F3	X A,[X−]	X A,[B−]	IFGT A,#i	IFGT A,[B]	IFBIT 3,[B]	*	LD B,#0C	IFBNE 3	JSR x300−x3FF	JMP x300−x3FF	JP+20	JP+4	3
JP−11	JP−27	LD 0F4, # i	DRSZ 0F4	VIS	LAID	ADD A,#i	ADD A,[B]	IFBIT 4,[B]	CLRA	LD B,#0B	IFBNE 4	JSR x400−x4FF	JMP x400−x4FF	JP+21	JP+5	4
JP−10	JP−26	LD 0F5, # i	DRSZ 0F5	RPND	JID	AND A,#i	AND A,[B]	IFBIT 5,[B]	SWAPA	LD B,#0A	IFBNE 5	JSR x500−x5FF	JMP x500−x5FF	JP+22	JP+6	5
JP−9	JP−25	LD 0F6, # i	DRSZ 0F6	X A,[X]	X A,[B]	XOR A,#i	XOR A,[B]	IFBIT 6,[B]	DCORA	LD B,#09	IFBNE 6	JSR x600−x6FF	JMP x600−x6FF	JP+23	JP+7	6
JP−8	JP−24	LD 0F7, # i	DRSZ 0F7	*	*	OR A,#i	OR A,[B]	IFBIT 7,[B]	PUSHA	LD B,#08	IFBNE 7	JSR x700−x7FF	JMP x700−x7FF	JP+24	JP+8	7
JP−7	JP−23	LD 0F8, # i	DRSZ 0F8	NOP	RLCA	LD A,#i	IFC	SBIT 0,[B]	RBIT 0,[B]	LD B,#07	IFBNE 8	JSR x800−x8FF	JMP x800−x8FF	JP+25	JP+9	8
JP−6	JP−22	LD 0F9, # i	DRSZ 0F9	IFNE A,[B]	IFEQ Md,#i	IFNE A,#i	IFNC	SBIT 1,[B]	RBIT 1,[B]	LD B,#06	IFBNE 9	JSR x900−x9FF	JMP x900−x9FF	JP+26	JP+10	9
JP−5	JP−21	LD 0FA, # i	DRSZ 0FA	LD A,[X+]	LD A,[B+]	LD [B+],#i	INCA	SBIT 2,[B]	RBIT 2,[B]	LD B,#05	IFBNE 0A	JSR xA00−xAFF	JMP xA00−xAFF	JP+27	JP+11	A
JP−4	JP−20	LD 0FB, # i	DRSZ 0FB	LD A,[X−]	LD A,[B−]	LD [B−],#i	DECA	SBIT 3,[B]	RBIT 3,[B]	LD B,#04	IFBNE 0B	JSR xB00−xBFF	JMP xB00−xBFF	JP+28	JP+12	B
JP−3	JP−19	LD 0FC, # i	DRSZ 0FC	LD Md,#i	JMPL	X A,Md	POPA	SBIT 4,[B]	RBIT 4,[B]	LD B,#03	IFBNE 0C	JSR xC00−xCFF	JMP xC00−xCFF	JP+29	JP+13	C
JP−2	JP−18	LD 0FD, # i	DRSZ 0FD	DIR	JSRL	LD A,Md	RETSK	SBIT 5,[B]	RBIT 5,[B]	LD B,#02	IFBNE 0D	JSR xD00−xDFF	JMP xD00−xDFF	JP+30	JP+14	D
JP−1	JP−17	LD 0FE, # i	DRSZ 0FE	LD A,[X]	LD A,[B]	LD [B],#i	RET	SBIT 6,[B]	RBIT 6,[B]	LD B,#01	IFBNE 0E	JSR xE00−xEFF	JMP xE00−xEFF	JP+31	JP+15	E
JP−0	JP−16	LD 0FF, # i	DRSZ 0FF	*	*	LD B,#i	RETI	SBIT 7,[B]	RBIT 7,[B]	LD B,#00	IFBNE 0F	JSR xF00−xFFF	JMP xF00−xFFF	JP+32	JP+16	F

Where,

i is the immediate data

Md is a directly addressed memory location

* is an unused opcode

Note: The opcode 60 Hex is also the opcode for IFBIT #i,A

Development Support

SUMMARY

- iceMASTER™: IM-COP8/400—Full feature in-circuit emulation for all COP8 products. A full set of COP8 Basic and Feature Family device and package specific probes are available.
- COP8 Debug Module: Moderate cost in-circuit emulation and development programming unit.
- COP8 Evaluation and Programming Unit: EPU-COP888GG—low cost In-circuit simulation and development programming unit.
- Assembler: COP8-DEV-IBMA. A DOS installable cross development Assembler, Linker, Librarian and Utility Software Development Tool Kit.
- C Compiler: COP8C. A DOS installable cross development Software Tool Kit.
- OTP/EPROM Programmer Support: Covering needs from engineering prototype, pilot production to full production environments.

iceMASTER (IM) IN-CIRCUIT EMULATION

The iceMASTER IM-COP8/400 is a full feature, PC based, in-circuit emulation tool developed and marketed by Meta-Link Corporation to support the whole COP8 family of products. National is a resale vendor for these products.

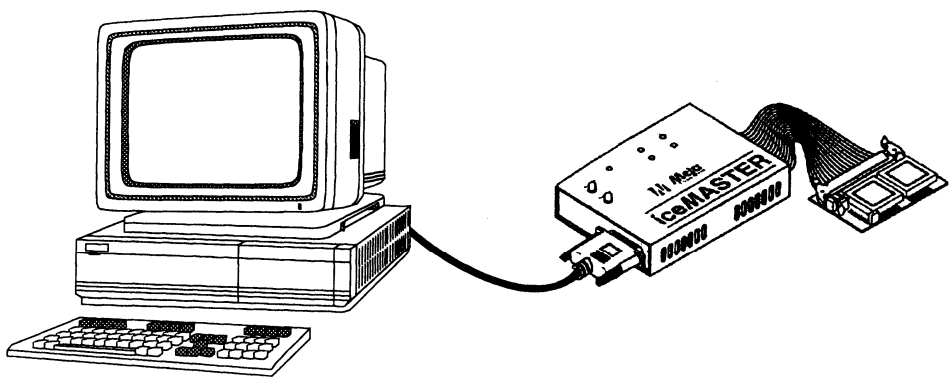
See *Figure 16* for configuration.

The iceMASTER IM-COP8/400 with its device specific COP8 Probe provides a rich feature set for developing, testing and maintaining product:

- Real-time in-circuit emulation; full 2.4V–5.5V operation range, full DC-10 MHz clock. Chip options are programmable or jumper selectable.
- Direct connection to application board by package compatible socket or surface mount assembly.
- Full 32 kbytes of loadable programming space that overlays (replaces) the on-chip ROM or EPROM. On-chip RAM and I/O blocks are used directly or recreated on the probe as necessary.
- Full 4k frame synchronous trace memory. Address, instruction, and 8 unspecified, circuit connectable trace lines. Display can be HLL source (e.g., C source), assembly or mixed.
- A full 64k hardware configurable break, trace on, trace off control, and pass count increment events.
- Tool set integrated interactive symbolic debugger—supports both assembler (COFF) and C Compiler (.COD) linked object formats.
- Real time performance profiling analysis; selectable bucket definition.
- Watch windows, content updated automatically at each execution break.
- Instruction by instruction memory/register changes displayed on source window when in single step operation.
- Single base unit and debugger software reconfigurable to support the entire COP8 family; only the probe personality needs to change. Debugger software is processor customized, and reconfigured from a master model file.
- Processor specific symbolic display of registers and bit level assignments, configured from master model file.
- Halt/Idle mode notification.
- On-line HELP customized to specific processor using master model file.
- Includes a copy of COP8-DEV-IBMA assembler and linker SDK.

IM Order Information

Base Unit	
IM-COP8/400-1	iceMASTER Base Unit, 110V Power Supply
IM-COP8/400-2	iceMASTER Base Unit, 220V Power Supply
iceMASTER Probe	
MHW-884EK28DWPC	28 DIP
MHW-888EK40DWPC	40 DIP
MHW-888EK44PWPC	44 PLCC
28 DIP to 28 SO Adapter	
MHW-SOIC28	28 SO



TL/DD/12859–19

FIGURE 16. COP8 iceMASTER Environment

Development Support (Continued)

iceMASTER DEBUG MODULE (DM)

The iceMASTER Debug Module is a PC based, combination in-circuit emulation tool and COP8 based OTP/EPROM programming tool developed and marketed by MetaLink Corporation to support the whole COP8 family of products. National is a resale vendor for these products.

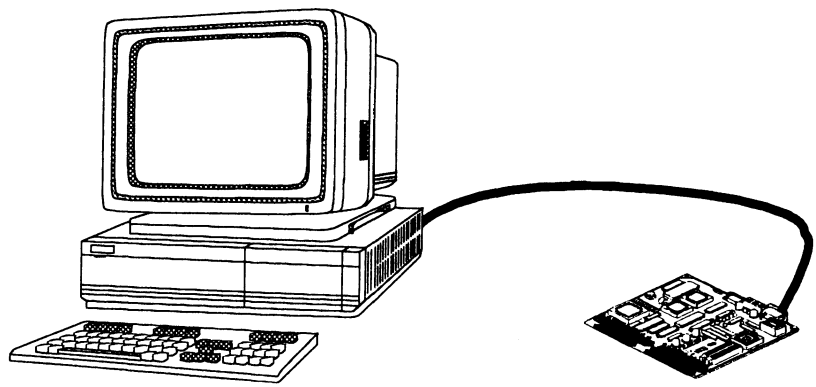
See *Figure 17* for configuration.

The iceMASTER Debug Module is a moderate cost development tool. It has the capability of in-circuit emulation for a specific COP8 microcontroller and in addition serves as a programming tool for COP8 OTP and EPROM product families. Summary of features is as follows:

- Real-time in-circuit emulation; full operating voltage range operation, full DC-10 MHz clock.
- All processor I/O pins can be cabled to an application development board with package compatible cable to socket and surface mount assembly.
- Full 32 kbytes of loadable programming space that overlays (replaces) the on-chip ROM or EPROM. On-chip RAM and I/O blocks are used directly or recreated as necessary.
- 100 frames of synchronous trace memory. The display can be HLL source (C source), assembly or mixed. The most recent history prior to a break is available in the trace memory.
- Configured break points; uses INTR instruction which is modestly intrusive.
- Software—only supported features are selectable.
- Tool set integrated interactive symbolic debugger—supports both assembler (COFF) and C Compiler (.COD) SDK linked object formats.
- Instruction by instruction memory/register changes displayed when in single step operation.
- Debugger software is processor customized, and reconfigured from a master model file.
- Processor specific symbolic display of registers and bit level assignments, configured from master model file.
- Halt/Idle mode notification.
- Programming menu supports full product line of programmable OTP and EPROM COP8 products. Program data is taken directly from the overlay RAM.
- Programming of 44 PLCC and 68 PLCC parts requires external programming adapters.
- Includes wall mount power supply.
- On-board V_{PP} generator from 5V input or connection to external supply supported. Requires V_{PP} level adjustment per the family programming specification (correct level is provided on an on-screen pop-down display).
- On-line HELP customized to specific processor using master model file.
- Includes a copy of COP8-DEV-IBMA assembler and linker SDK.

DM Order Information

Debug Model Unit	
COP8-DM/888EK	
Cable Adapters	
DM-COP8/28D	28 DIP
DM-COP8/40D	40 DIP
DM-COP8/44P	44 PLCC
28 DIP to 28 SO Adapter	
DM-COP8/28D-SO	28 SO



TL/DD/12859–20

FIGURE 17. COP8-DM Environment

Development Support

COP8 ASSEMBLER/LINKER SOFTWARE DEVELOPMENT TOOL KIT

National Semiconductor offers a relocatable COP8 macro cross assembler, linker, librarian and utility software development tool kit. Features are summarized as follows:

- Basic and Feature Family instruction set by "device" type.
- Nested macro capability.
- Extensive set of assembler directives.
- Supported on PC/DOS platform.
- Generates National standard COFF output files.
- Integrated Linker and Librarian.
- Integrated utilities to generate ROM code file outputs.
- DUMPCOFF utility.

This product is integrated as a part of MetaLink tools as a development kit, fully supported by the MetaLink debugger. It may be ordered separately or it is bundled with the MetaLink products at no additional cost.

Order Information

Assembler SDK	
COP8-DEV-IBMA	Assembler SDK on installable 3.5" PC/DOS Floppy Disk Drive format. Periodic upgrades and most recent version is available on National's BBS and Internet.

COP8 C COMPILER

A C Compiler is developed and marketed by Byte Craft Limited. The COP8C compiler is a fully integrated development tool specifically designed to support the compact embedded configuration of the COP8 family of products.

Features are summarized as follows:

- ANSI C with some restrictions and extensions that optimize development for the COP8 embedded application.
- BITS data type extension. Register declaration #pragma with direct bit level definitions.
- C language support for interrupt routines.
- Expert system, rule based code generation and optimization.
- Performs consistency checks against the architectural definitions of the target COP8 device.
- Generates program memory code.
- Supports linking of compiled object or COP8 assembled object formats.

Approved List

Manufacturer	North America	Europe	Asia
BP Microsystems	(800) 225-2102 (713) 688-4600 Fax: (713) 688-0920	+49-8152-4183 +49-8856-932616	+852-234-16611 +852-2710-8121
Data I/O	(800) 426-1045 (206) 881-6444 Fax: (206) 882-1043	+44-0734-440011	Call North America
HI-LO	(510) 623-8860	Call Asia	+886-2-764-0215 Fax: +886-2-756-6403
ICE Technology	(800) 624-8949 (919) 430-7915	+44-1226-767404 Fax: 0-1226-370-434	
MetaLink	(800) 638-2423 (602) 926-0797 Fax: (602) 693-0681	+49-80 9156 96-0 Fax: +49-80 9123 86	+852-737-1800
Systems General	(408) 263-6667	+41-1-9450300	+886-2-917-3005 Fax: +886-2-911-1283
Needhams	(916) 924-8037 Fax: (916) 924-8065		

Development Support (Continued)

- Global optimization of linked code.
- Symbolic debug load format fully sourced level supported by the MetaLink debugger.

OTP/EMULATOR SUPPORT

The COP87L88RK/COP87L84RK devices provide emulation and OTP support for the COP888EK/COP884EK mask programmable devices.

COP8788RK/COP8784RK Ordering Information

Device Number	Clock Option	Package	Emulates
COP87L88RKV-XE	Crystal/HALT En	44 PLCC	COP888EK
COP87L88RKN-XE	Crystal/HALT En	40 DIP	COP888EK
COP87L84RKN-XE	Crystal/HALT En	28 DIP	COP884EK
COP87L84RKM-XE	Crystal/HALT En	28 SO	COP884EK

*Check with the local sales office about the availability.

INDUSTRY WIDE OTP/EPROM PROGRAMMING SUPPORT

Programming support, in addition to the MetaLink development tools, is provided by a full range of independent approved vendors to meet the needs from the engineering laboratory to full production.

AVAILABLE LITERATURE

For more information, please see the COP8 Basic Family User's Manual, Literature Number 620895, COP8 Feature Family User's Manual, Literature Number 620897 and National's Family of 8-bit Microcontrollers COP8 Selection Guide, Literature Number 630009.

DIAL-A-HELPER SERVICE

Dial-A-Helper is a service provided by the Microcontroller Applications group. The Dial-A-Helper is an Electronic Information System that may be accessed as a Bulletin Board System (BBS) via data modem, as an FTP site on the Internet via standard FTP client application or as an FTP site on the Internet using a standard Internet browser such as Netscape or Mosaic.

The Dial-A-Helper system provides access to an automated information storage and retrieval system. The system capabilities include a MESSAGE SECTION (electronic mail, when accessed as a BBS) for communications to and from the Microcontroller Applications Group and a FILE SECTION which consists of several file areas where valuable application software and utilities could be found.

DIAL-A-HELPER BBS via a Standard Modem

Modem: CANADA/U.S.: (800) NSC-MICRO
(800) 672-6427
EUROPE: (+49) 0-8141-351332
Baud: 14.4k
Set-Up: Length: 8-Bit
Parity: None
Stop Bit: 1
Operation: 24 Hours, 7 Days

DIAL-A-HELPER via FTP

ftp nscmicro.nsc.com
user: anonymous
password: username@yourhost.site.domain

DIAL-A-HELPER via a WorldWide Web Browser

ftp://nscmicro.nsc.com

National Semiconductor on the WorldWide Web

See us on the WorldWide Web at: http://www.natsemi.com

CUSTOMER RESPONSE CENTER

Complete product information and technical support is available from National's customer response centers.

CANADA/US:	Tel:	(800) 272-9959
	email:	support @tevm2.nsc.com
EUROPE:	email:	europe.support@nsc.com
	Deutsch Tel:	+49 (0) 180-530 85 85
	English Tel:	+49 (0) 180-532 78 32
	Français Tel:	+49 (0) 180-532 93 58
	Italiano Tel:	+49 (0) 180-534 16 80
JAPAN:	Tel:	+81-043-299-2309
S.E. ASIA:	Beijing Tel:	(+86) 10-6856-8601
	Shanghai Tel:	(+86) 21-6415-4092
	Hong Kong Tel:	(+852) 2737-1600
	Korea Tel:	(+82) 2-3771-6909
	Malaysia Tel:	(+60-4) 644-9061
	Singapore Tel:	(+65) 255-2226
	Taiwan Tel:	+886-2-521-3288
AUSTRALIA:	Tel:	(+61) 3-9558-9999
INDIA:	Tel:	(+91) 80-559-9467

National Semiconductor

PRELIMINARY

COP87L88RG
8-Bit One-Time Programmable (OTP) Microcontroller with 32 Kbytes of Program Memory

General Description

The COP87L88RG is a member of the COP8™ OTP microcontroller family. It is pin and software compatible to the mask ROM COP888GG product family.

(Continued)

Key Features

- Full duplex UART
- Three 16-bit timers, each with two 16-bit registers supporting:
  - — Processor independent PWM mode
  - — External event counter mode
  - — Input capture mode
- 32 kbytes on-board EPROM with security feature

  Note: Mask ROMed devices with equivalent on-chip features and program memory sizes of 4k, 8k, 16k, 20k, and 24k are available (see Table I).
- 512 bytes on-board RAM

Additional Peripheral Features

- Idle timer
- Multi-Input Wake Up (MIWU) with optional interrupts (8)
- Two analog comparators
- WATCHDOG™ and clock monitor logic
- MICROWIRE/PLUS™ serial I/O

I/O Features

- Memory mapped I/O
- Software selectable I/O options (TRI-STATE® output, push-pull output, weak pull-up input, high impedance input
- Schmitt trigger inputs on ports G and L
- Packages:
  - — 44 PLCC with 40 I/O pins
  - — 40 DIP with 36 I/O pins

CPU/Instruction Set Features

- 1 μs instruction cycle time
- Fourteen multi-source vectored interrupts servicing
  - — External interrupt
  - — Idle timer T0
  - — Two timers (each with 2 interrupts)
  - — MICROWIRE/PLUS
  - — Multi-Input Wake Up
  - — Software trap
  - — UART (2)
  - — Default VIS (default interrupt)
- Versatile instruction set with true bit manipulation
- 8-bit Stack Pointer SP (stack in RAM)
- Two 8-bit register indirect data memory pointers (B and X)

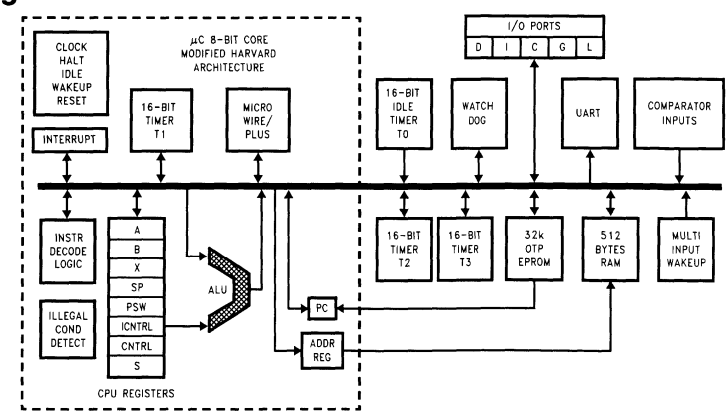
Fully Static CMOS

- Two power saving modes: HALT and IDLE
- Single supply operation: 2.7V–5.5V
- Temperature range: −40°C to +85°C

Development Support

- Emulation device for the COP888EG, COP888GG and COP888HG
- Real time emulation and full program debug offered by MetaLink Development System

Block Diagram



TL/DD12860–1

FIGURE 1. Block Diagram

General Description (Continued)

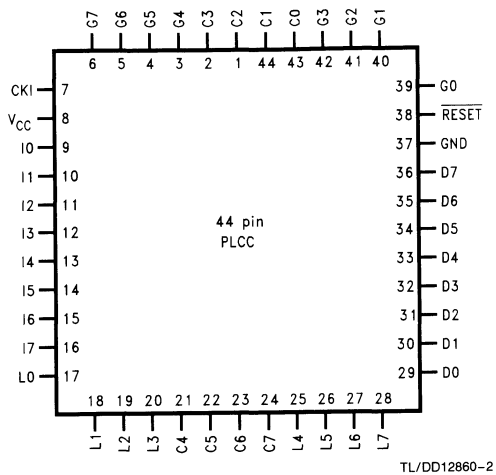
The device is a fully static part, fabricated using double-metal silicon gate microCMOS technology. Features include an 8-bit memory mapped architecture, MICROWIRE/PLUS™ serial I/O, three 16-bit timer/counters supporting three modes (Processor Independent PWM generation, External Event counter, and Input Capture mode capabilities), full duplex UART and two comparators. Each I/O pin has software selectable configurations. The devices operates over a voltage range of 2.7V to 5.5V. High throughput is achieved with an efficient, regular instruction set operating at a maximum rate of 1 μs per instruction.

TABLE I. COP888CG/EG/GG/HG Family Members

Device	ROM/EPROM (Bytes)	RAM (Bytes)	Key Common Features
COP888CG	4k ROM	192	3 Timers
COP888EG	8k ROM	256	UART
COP888GG	16k ROM	512	2 Comparators
COP87L88GG	16k OTP EPROM	512	
COP888HG	20k ROM	512	
COP87L88RG	32k OTP EPROM	512	

Connection Diagrams

Plastic Chip Carrier

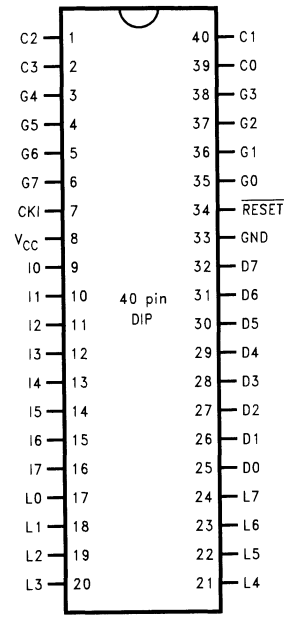


Top View

Order Number COP87L88RGV-XE
See NS Package Number V44A

Note: -X Crystal Oscillator
-E Halt Enable

Dual-In-Line Package



Top View

Order Number COP87L88RGN-XE
See NS Package Number N40A

FIGURE 2. Connection Diagrams

Connection Diagrams (Continued)

Pinouts for 40- and 44-Pin Packages

Port	Type	Alt. Fun	Alt. Fun	40-Pin DIP	44-Pin PLCC
L0	I/O	MIWU		17	17
L1	I/O	MIWU	CKX	18	18
L2	I/O	MIWU	TDX	19	19
L3	I/O	MIWU	RDX	20	20
L4	I/O	MIWU	T2A	21	25
L5	I/O	MIWU	T2B	22	26
L6	I/O	MIWU	T3A	23	27
L7	I/O	MIWU	T3B	24	28
G0	I/O	INT		35	39
G1	WDOUT			36	40
G2	I/O	T1B		37	41
G3	I/O	T1A		38	42
G4	I/O	SO		3	3
G5	I/O	SK		4	4
G6	I	SI		5	5
G7	I/CKO	HALT Restart		6	6
D0	O			25	29
D1	O			26	30
D2	O			27	31
D3	O			28	32
D4	O			29	33
D5	O			30	34
D6	O			31	35
D7	O			32	36
I0	I			9	9
I1	I	COMP1IN−		10	10
I2	I	COMP1IN+		11	11
I3	I	COMP1OUT		12	12
I4	I	COMP2IN−		13	13
I5	I	COMP2IN+		14	14
I6	I	COMP2OUT		15	15
I7	I			16	16
C0	I/O			39	43
C1	I/O			40	44
C2	I/O			1	1
C3	I/O			2	2
C4	I/O				21
C5	I/O				22
C6	I/O				23
C7	I/O				24
V_{CC}				8	8
GND				33	37
CKI				7	7
$\overline{\text{RESET}}$				34	38

Absolute Maximum Ratings (Note)

If Military/Aerospace specified devices are required, please contact the National Semiconductor Sales Office/Distributors for availability and specifications.

Supply Voltage (V_{CC})	7V
Voltage at Any Pin	−0.3V to V_{CC} + 0.3V
Total Current into V_{CC} Pin (Source)	100 mA
Total Current out of GND Pin (Sink)	110 mA
Storage Temperature Range	−65°C to +140°C

Note: *Absolute maximum ratings indicate limits beyond which damage to the device may occur. DC and AC electrical specifications are not ensured when operating the device at absolute maximum ratings.*

DC Electrical Characteristics −40°C ≤ T_A ≤ +85°C unless otherwise specified

Parameter	Conditions	Min	Typ	Max	Units
Operating Voltage		2.7		5.5	V
Power Supply Ripple (Note 1)	Peak-to-Peak			0.1 V_{CC}	V
Supply Current (Note 2)					
CKI = 10 MHz	V_{CC} = 5.5V, t_c = 1 μs			16.5	mA
CKI = 4 MHz	V_{CC} = 4.0V, t_c = 2.5 μs			6.5	mA
HALT Current (Note 3)	V_{CC} = 5.5V, CKI = 0 MHz			12	μA
	V_{CC} = 4.0V, CKI = 0 MHz			8	μA
IDLE Current					
CKI = 10 MHz	V_{CC} = 5.5V, t_c = 1 μs			3.5	mA
CKI = 1 MHz	V_{CC} = 4.0V, t_c = 10 μs			0.7	mA
Input Levels					
$\overline{RESET}$					
Logic High		0.8 V_{CC}			V
Logic Low				0.2 V_{CC}	V
CKI (External and Crystal Osc. Modes)					
Logic High		0.7 V_{CC}			V
Logic Low				0.2 V_{CC}	V
All Other Inputs					
Logic High		0.7 V_{CC}			V
Logic Low				0.2 V_{CC}	V
Hi-Z Input Leakage	V_{CC} = 5.5V	−2		+2	μA
Input Pullup Current	V_{CC} = 5.5V	40		250	μA
G and L Port Input Hysteresis			0.05 V_{CC}	0.35 V_{CC}	V
Output Current Levels					
D Outputs					
Source	V_{CC} = 4.5V, V_{OH} = 3.3V	0.4			mA
Sink (Note 4)	V_{CC} = 4.5V, V_{OL} = 1V	10			mA
All Others					
Source (Weak Pull-Up Mode)	V_{CC} = 4.5V, V_{OH} = 2.7V	10		100	μA
Source (Push-Pull Mode)	V_{CC} = 4.5V, V_{OH} = 3.3V	0.4			mA
Sink (Push-Pull Mode)	V_{CC} = 4.5V, V_{OL} = 0.4V	1.6			mA
TRI-STATE Leakage	V_{CC} = 5.5V	−2		+2	μA
Allowable Sink/Source Current per Pin					
D Outputs (Sink)				15	mA
All others				3	mA
Maximum Input Current without Latchup (Note 5)	T_A = 25°C			±100	mA
RAM Retention Voltage, V_r	500 ns Rise and Fall Time (Min)	2			V
Input Capacitance				7	pF
Load Capacitance on D2				1000	pF

Note 1: Rate of voltage change must be less then 0.5 V/ms.

Note 2: Supply current is measured after running 2000 cycles with a square wave CKI input, CKO open, inputs at rails and outputs open.

Note 3: The HALT mode will stop CKI from oscillating in the RC and the Crystal configurations by bringing CKI high. Test conditions: All inputs tied to V_{CC}, L and G ports in the TRI-STATE mode and tied to ground, all outputs low and tied to ground. The clock monitor is disabled.

Note 4: The user must guarantee that D2 pin does not source more than 10 mA during RESET. If D2 sources more than 10 mA during reset, the device will go into programming mode.

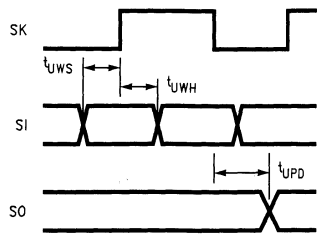
Note 5: Pins G6 and $\overline{RESET}$ are designed with a high voltage input network for factory testing. These pins allow input voltages greater than V_{CC} and the pins will have sink current to V_{CC} when biased at voltages greater than V_{CC} (the pins do not have source current when biased at a voltage below V_{CC}). The effective resistance to V_{CC} is 750Ω (typical). These two pins will not latch up. The voltage at the pins must be limited to less than 14V.

AC Electrical Characteristics $-40°C \le T_A \le +85°C$ unless otherwise specified

Parameter	Conditions	Min	Typ	Max	Units
Instruction Cycle Time (t_c)					
Crystal, Resonator,		1		DC	μs
R/C Oscillator		3		DC	μs
Inputs					
t_{SETUP}		200			ns
t_{HOLD}		60			ns
Output Propagation Delay	$R_L = 2.2k, C_L = 100$ pF				
t_{PD1}, t_{PD0}					
SO, SK				0.7	μs
All Others				1	μs
MICROWIRE™ Setup Time (t_{UWS})		20			ns
MICROWIRE Hold Time (t_{UWH})		56			ns
MICROWIRE Output Propagation Delay (t_{UPD})				220	ns
Input Pulse Width					
Interrupt Input High Time		1			t_c
Interrupt Input Low Time		1			t_c
Timer Input High Time		1			t_c
Timer Input Low Time		1			t_c
Reset Pulse Width		1			μs

Comparators AC and DC Characteristics $V_{CC} = 5V, T_A = 25°C$

Parameter	Conditions	Min	Typ	Max	Units
Input Offset Voltage	$0.4V \le V_{IN} \le V_{CC} - 1.5V$		±10	±25	mV
Input Common Mode Voltage Range		0.4		$V_{CC} - 1.5$	V
Low Level Output Current	$V_{OL} = 0.4V$	1.6			mA
High Level Output Current	$V_{OH} = 4.6V$	1.6			mA
DC Supply Current Per Comparator (When Enabled)				250	μA
Response Time	TBD mV Step, TBD mV Overdrive, 100 pF Load		1		μs



TL/DD12860–5

FIGURE 3. MICROWIRE/PLUS Timing

Pin Descriptions

V_{CC} and GND are the power supply pins.

CKI is the clock input. This can come from an R/C generated oscillator, or a crystal oscillator (in conjunction with CKO). See Oscillator Description section.

$\overline{\text{RESET}}$ is the master reset input. See Reset Description section.

The device contains three bidirectional 8-bit I/O ports (C, G and L), where each individual bit may be independently configured as an input (Schmitt trigger inputs on ports L and G), output or TRI-STATE under program control. Three data memory address locations are allocated for each of these I/O ports. Each I/O port has two associated 8-bit memory mapped registers, the CONFIGURATION register and the output DATA register. A memory mapped address is also reserved for the input pins of each I/O port. (See the memory map for the various addresses associated with the I/O ports.) *Figure 4* shows the I/O port configurations. The DATA and CONFIGURATION registers allow for each port bit to be individually configured under software control as shown below:

CONFIGURATION Register	DATA Register	Port Set-Up
0	0	Hi-Z Input (TRI-STATE Output)
0	1	Input with Weak Pull-Up
1	0	Push-Pull Zero Output
1	1	Push-Pull One Output

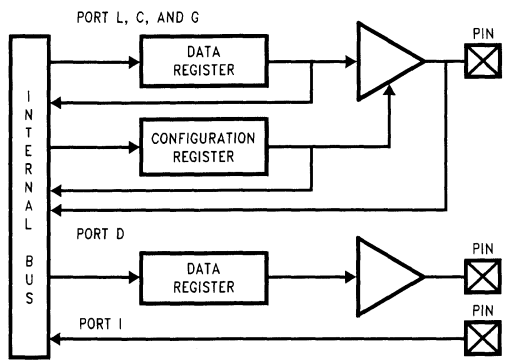
PORT L is an 8-bit I/O port. All L-pins have Schmitt triggers on the inputs.

Port L supports Multi-Input Wake Up (MIWU) on all eight pins. L1 is used for the UART external clock. L2 and L3 are used for the UART transmit and receive. L4 and L5 are used for the timer input functions T2A and T2B. L6 and L7 are used for the timer input functions T3A and T3B.

Port L has the following alternate features:

- L0 MIWU
- L1 MIWU or CKX
- L2 MIWU or TDX
- L3 MIWU or RDX
- L4 MIWU or T2A
- L5 MIWU or T2B
- L6 MIWU or T3A
- L7 MIWU or T3B

Port G is an 8-bit port with 5 I/O pins (G0, G2–G5), an input pin (G6), and two dedicated output pins (G1 and G7). Pins G0 and G2–G6 all have Schmitt Triggers on their inputs. Pin G1 serves as the dedicated WDOUT WATCHDOG output, while pin G7 is either input or output depending on the oscillator mask option selected. With the crystal oscillator option selected, G7 serves as the dedicated output pin for the CKO clock output. With the single-pin R/C oscillator mask option selected, G7 serves as a general purpose input pin but is also used to bring the device out of HALT mode with a low to high transition on G7. There are two registers associated with the G Port, a data register and a configuration register. Therefore, each of the 5 I/O bits (G0, G2–G5) can be individually configured under software control.



TL/DD12860–6

FIGURE 4. I/O Port Configurations

Pin Descriptions (Continued)

Since G6 is an input only pin and G7 is the dedicated CKO clock output pin (crystal clock option) or general purpose input (R/C clock option), the associated bits in the data and configuration registers for G6 and G7 are used for special purpose functions as outlined below. Reading the G6 and G7 data bits will return zeros.

Note that the chip will be placed in the HALT mode by writing a "1" to bit 7 of the Port G Data Register. Similarly the chip will be placed in the IDLE mode by writing a "1" to bit 6 of the Port G Data Register.

Writing a "1" to bit 6 of the Port G Configuration Register enables the MICROWIRE/PLUS to operate with the alternate phase of the SK clock. The G7 configuration bit, if set high, enables the clock start up delay after HALT when the R/C clock configuration is used.

	Config Reg.	Data Reg.
G7	CLKDLY	HALT
G6	Alternate SK	IDLE

Port G has the following alternate features:

- G0 INTR (External Interrupt Input)
- G2 T1B (Timer T1 Capture Input)
- G3 T1A (Timer T1 I/O)
- G4 SO (MICROWIRE Serial Data Output)
- G5 SK (MICROWIRE Serial Clock)
- G6 SI (MICROWIRE Serial Data Input)

Port G has the following dedicated functions:

- G1 WDOUT WATCHDOG and/or Clock Monitor dedicated output
- G7 CKO Oscillator dedicated output or general purpose input

Port C is an 8-bit I/O port. The 40-pin device does not have a full complement of Port C pins. The unavailable pins are not terminated. A read operation for these unterminated pins will return unpredictable values.

PORT I is an eight-bit Hi-Z input port.

Port I1–I3 are used for Comparator 1. Port I4–I6 are used for Comparator 2.

The Port I has the following alternate features.

- I1 COMP1 − IN (Comparator 1 Negative Input)
- I2 COMP1 + IN (Comparator 1 Positive Input)
- I3 COMP1OUT (Comparator 1 Output)
- I4 COMP2 − IN (Comparator 2 Negative Input)
- I5 COMP2 + IN (Comparator 2 Positive Input)
- I6 COMP2OUT (Comparator 2 Output)

Port D is a recreated 8-bit output port that is preset high when $\overline{\text{RESET}}$ goes low. D port recreation is one clock cycle behind normal port timing. The user can tie two or more D port outputs (except D2) together in order to get a higher drive.

Functional Description

The architecture of the device is modified Harvard architecture. With the Harvard architecture, the control store program memory (ROM) is separated from the data store memory (RAM). Both ROM and RAM have their own separate addressing space with separate address buses. The architecture, though based on Harvard architecture, permits transfer of data from ROM to RAM.

CPU REGISTERS

The CPU can do an 8-bit addition, subtraction, logical or shift operation in one instruction (t_C) cycle time.

There are six CPU registers:

A is the 8-bit Accumulator Register

PC is the 15-bit Program Counter Register

PU is the upper 7 bits of the program counter (PC)
PL is the lower 8 bits of the program counter (PC)

B is an 8-bit RAM address pointer, which can be optionally post auto incremented or decremented.

X is an 8-bit alternate RAM address pointer, which can be optionally post auto incremented or decremented.

SP is the 8-bit stack pointer, which points to the subroutine/interrupt stack (in RAM). The SP is initialized to RAM address 06F with reset.

S is the 8-bit Data Segment Address Register used to extend the lower half of the address range (00 to 7F) into 256 data segments of 128 bytes each.

All the CPU registers are memory mapped with the exception of the Accumulator (A) and the Program Counter (PC).

PROGRAM MEMORY

The program memory consists of 32 kbytes of OTP EPROM. These bytes may hold program instructions or constant data (data tables for the LAID instruction, jump vectors for the JID instruction, and interrupt vectors for the VIS instruction). The program memory is addressed by the 15-bit program counter (PC). All interrupts in the devices vector to program memory location 0FF Hex.

The device can be configured to inhibit external reads of the program memory. This is done by programming the Security Byte.

Note: Mask ROMed devices with equivalent on-chip features and program memory sizes of 4k, 8k, 16k, 20k, and 24k are available.

SECURITY FEATURE

The program memory array has an associate Security Byte that is located outside of the program address range. This byte can be addressed only from programming mode by a programmer tool.

Security is an optional feature and can only be asserted after the memory array has been programmed and verified. A secured part will read all 00(hex) by a programmer. The part will fail Blank Check and will fail Verify operations. A Read operaiton will fill the programmer's memory with 00(hex). The Security Byte itself is always readable with value of 00(hex) if unsecure and FF(hex) if secure.

DATA MEMORY

The data memory address space includes the on-chip RAM and data registers, the I/O registers (Configuration, Data and Pin), the control registers, the MICROWIRE/PLUS SIO shift register, and the various registers, and counters associated with the timers (with the exception of the IDLE timer). Data memory is addressed directly by the instruction or indirectly by the B, X, SP pointers and S register.

Functional Description (Continued)

The data memory consists of 512 bytes of RAM. Sixteen bytes of RAM are mapped as "registers" at addresses 0F0 to 0FF Hex. These registers can be loaded immediately, and also decremented and tested with the DRSZ (decrement register and skip if zero) instruction. The memory pointer registers X, SP, B and S are memory mapped into this space at address locations 0FC to 0FF Hex respectively, with the other registers being available for general usage.

The instruction set permits any bit in memory to be set, reset or tested. All I/O and registers (except A and PC) are memory mapped; therefore, I/O bits and register bits can be directly and individually set, reset and tested. The accumulator (A) bits can also be directly and individually tested.

Note: RAM contents are undefined upon power-up.

Data Memory Segment RAM Extension

Data memory address 0FF is used as a memory mapped location for the Data Segment Address Register (S).

The data store memory is either addressed directly by a single byte address within the instruction, or indirectly relative to the reference of the B, X, or SP pointers (each contains a single-byte address). This single-byte address allows an addressing range of 256 locations from 00 to FF hex. The upper bit of this single-byte address divides the data store memory into two separate sections as outlined previously. With the exception of the RAM register memory from address locations 00F0 to 00FF, all RAM memory is memory mapped with the upper bit of the single-byte address being equal to zero. This allows the upper bit of the single-byte address to determine whether or not the base address range (from 0000 to 00FF) is extended. If this upper bit equals one (representing address range 0080 to 00FF), then address extension does not take place. Alternatively, if this upper bit equals zero, then the data segment extension register S is used to extend the base address range (from 0000 to 007F) from XX00 to XX7F, where XX represents the 8 bits from the S register. Thus the 128-byte data segment extensions are located from addresses 0100 to 017F for data segment 1, 0200 to 027F for data segment 2, etc., up to FF00 to FF7F for data segment 255. The base address range from 0000 to 007F represents data segment 0.

Figure 5 illustrates how the S register data memory extension is used in extending the lower half of the base address range (00 to 7F hex) into 256 data segments of 128 bytes each, with a total addressing range of 32 kbytes from XX00 to XX7F. This organization allows a total of 256 data segments of 128 bytes each with an additional upper base segment of 128 bytes. Furthermore, all addressing modes are available for all data segments. The S register must be changed under program control to move from one data segment (128 bytes) to another. However, the upper base segment (containing the 16 memory registers, I/O registers, control registers, etc.) is always available regardless of the contents of the S register, since the upper base segment (address range 0080 to 00FF) is independent of data segment extension.

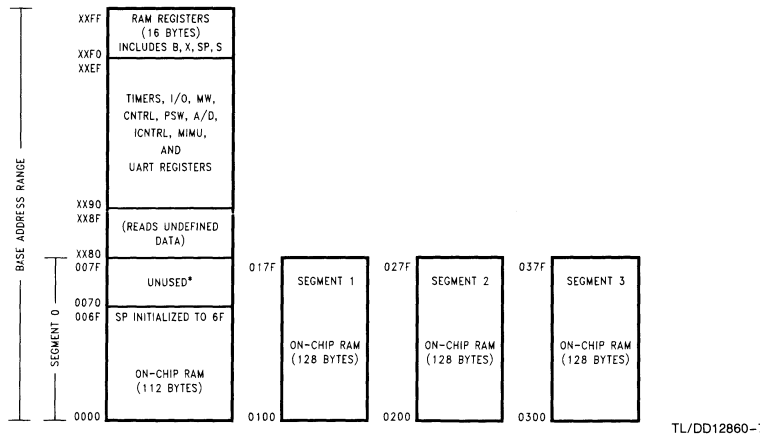
The instructions that utilize the stack pointer (SP) always reference the stack as part of the base segment (Segment 0), regardless of the contents of the S register. The S register is not changed by these instructions. Consequently, the stack (used with subroutine linkage and interrupts) is always located in the base segment. The stack pointer will be initialized to point at data memory location 006F as a result of reset.

The 128 bytes of RAM contained in the base segment are split between the lower and upper base segments. The first 116 bytes of RAM are resident from address 0000 to 006F in the lower base segment, while the remaining 16 bytes of RAM represent the 16 data memory registers located at addresses 00F0 to 00FF of the upper base segment. No RAM is located at the upper sixteen addresses (0070 to 007F) of the lower base segment.

Additional RAM beyond these initial 128 bytes, however, will always be memory mapped in groups of 128 bytes (or less) at the data segment address extensions (XX00 to XX7F) of the lower base segment. The additional 128 bytes of RAM are memory mapped at address locations 0100 to 017F hex.

Reset

The $\overline{\text{RESET}}$ input when pulled low initializes the microcontroller. Initialization will occur whenever the $\overline{\text{RESET}}$ input is pulled low. Upon initialization, the data and configuration registers for ports L, G and C are cleared, resulting in these



TL/DD12860–7

*Reads as all ones.

FIGURE 5. RAM Organization

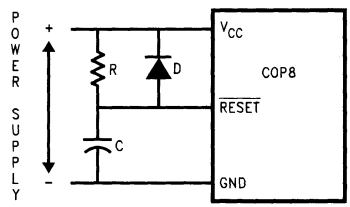
Reset (Continued)

Ports being initialized to the TRI-STATE mode. Pin G1 of the G Port is an exception (as noted below) since pin G1 is dedicated as the WATCHDOG and/or Clock Monitor error output pin. Port D is set high. The PC, PSW, ICNTRL, CNTRL, T2CNTRL and T3CNTRL control registers are cleared. The UART registers PSR, ENU (except that TBMT bit is set), ENUR and ENUI are cleared. The Comparator Select Register is cleared. The S register is initialized to zero. The Multi-Input Wake Up registers WKEN, WKEDG and WKPND are cleared. The stack pointer, SP, is initialized to 6F Hex.

The device comes out of reset with both the WATCHDOG logic and the Clock Monitor detector armed, with the WATCHDOG service window bits set and the Clock Monitor bit set. The WATCHDOG and Clock Monitor circuits are inhibited during reset. The WATCHDOG service window bits being initialized high default to the maximum WATCHDOG service window of 64k t_C clock cycles. The Clock Monitor bit being initialized high will cause a Clock Monitor error following reset if the clock has not reached the minimum specified frequency at the termination of reset. A Clock Monitor error will cause an active low error output on pin G1. This error output will continue until 16 t_C–32 t_C clock cycles following the clock frequency reaching the minimum specified value, at which time the G1 output will enter the TRI-STATE mode.

The external RC network shown in *Figure 6* should be used to ensure that the $\overline{\text{RESET}}$ pin is held low until the power supply to the chip stabilizes.

Note: Continual state of reset will cause the device to draw excessive current.



TL/DD/12860–8

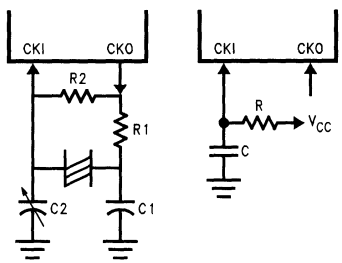
RC > 5 × Power Supply Rise Time

FIGURE 6. Recommended Reset Circuit

Oscillator Circuits

The chip can be driven by a clock input on the CKI input pin which can be between DC and 10 MHz. The CKO output clock is on pin G7 (crystal configuration). The CKI input frequency is divided down by 10 to produce the instruction cycle clock ($1/t_C$).

Figure 7 shows the Crystal and R/C diagrams.



TL/DD12860–9

FIGURE 7. Crystal and R/C Oscillator Diagrams

CRYSTAL OSCILLATOR

CKI and CKO can be connected to make a closed loop crystal (or resonator) controlled oscillator.

Table I shows the component values required for various standard crystal values.

TABLE I. Crystal Oscillator Configuration, T_A = 25°C

R1 (kΩ)	R2 (MΩ)	C1 (pF)	C2 (pF)	CKI Freq (MHz)	Conditions
0	1	30	30–36	10	V_{CC} = 5V
0	1	30	30–36	4	V_{CC} = 5V
0	1	200	100–150	0.455	V_{CC} = 5V

R/C OSCILLATOR

By selecting CKI as a single pin oscillator input, a single pin R/C oscillator circuit can be connected to it. CKO is available as a general purpose input, and/or HALT restart pin.

Table II shows the variation in the oscillator frequencies as functions of the component (R and C) values.

TABLE II. R/C Oscillator Configuration, T_A = 25°C

R (kΩ)	C (pF)	CKI Freq (MHz)	Instr. Cycle (μs)	Conditions
3.3	82	2.2–2.7	3.7–4.6	V_{CC} = 5V
5.6	100	1.1–1.3	7.4–9.0	V_{CC} = 5V
6.8	100	0.9–1.1	8.8–10.8	V_{CC} = 5V

Note: 3k ≤ R ≤ 200k

50 pF ≤ C ≤ 200 pF

Control Registers

CNTRL Register (Address X'00EE)

The Timer1 (T1) and MICROWIRE/PLUS control register contains the following bits:

SL1 & SL0	Select the MICROWIRE/PLUS clock divide by (00 = 2, 01 = 4, 1x = 8)
IEDG	External interrupt edge polarity select (0 = Rising edge, 1 = Falling edge)
MSEL	Selects G5 and G4 as MICROWIRE/PLUS signals SK and SO respectively
T1C0	Timer T1 Start/Stop control in timer modes 1 and 2 Timer T1 Underflow Interrupt Pending Flag in timer mode 3
T1C1	Timer T1 mode control bit
T1C2	Timer T1 mode control bit
T1C3	Timer T1 mode control bit

T1C3	T1C2	T1C1	T1C0	MSEL	IEDG	SL1	SL0

Bit 7 Bit 0

Control Registers (Continued)

PSW Register (Address X'00EF)

The PSW register contains the following select bits:

- GIE Global interrupt enable (enables interrupts)
- EXEN Enable external interrupt
- BUSY MICROWIRE/PLUS busy shifting flag
- EXPND External interrupt pending
- T1ENA Timer T1 Interrupt Enable for Timer Underflow or T1A Input capture edge
- T1PNDA Timer T1 Interrupt Pending Flag (Autoreload RA in mode 1, T1 Underflow in Mode 2, T1A capture edge in mode 3)
- C Carry Flag
- HC Half Carry Flag

HC	C	T1PNDA	T1ENA	EXPND	BUSY	EXEN	GIE

Bit 7 Bit 0

The Half-Carry bit is also affected by all the instructions that affect the Carry flag. The SC (Set Carry) and RC (Reset Carry) instructions will respectively set or clear both the carry flags. In addition to the SC and RC instructions, ADC, SUBC, RRC and RLC instructions affect the carry and Half Carry flags.

ICNTRL Register (Address X'00E8)

The ICNTRL register contains the following bits:

- T1ENB Timer T1 Interrupt Enable for T1B Input capture edge
- T1PNDB Timer T1 Interrupt Pending Flag for T1B capture edge
- WEN Enable MICROWIRE/PLUS interrupt
- WPND MICROWIRE/PLUS interrupt pending
- T0EN Timer T0 Interrupt Enable (Bit 12 toggle)
- T0PND Timer T0 Interrupt pending
- LPEN L Port Interrupt Enable (Multi-Input Wake Up/ Interrupt)

  Bit 7 could be used as a flag

Unused	LPEN	T0PND	T0EN	WPND	WEN	T1PNDB	T1ENB

Bit 7 Bit 0

T2CNTRL Register (Address X'00C6)

The T2CNTRL register contains the following bits:

- T2ENB Timer T2 Interrupt Enable for T2B Input capture edge
- T2PNDB Timer T2 Interrupt Pending Flag for T2B capture edge
- T2ENA Timer T2 Interrupt Enable for Timer Underflow or T2A Input capture edge
- T2PNDA Timer T2 Interrupt Pending Flag (Autoreload RA in mode 1, T2 Underflow in mode 2, T2A capture edge in mode 3)
- T2C0 Timer T2 Start/Stop control in timer modes 1 and 2 Timer T2 Underflow Interrupt Pending Flag in timer mode 3
- T2C1 Timer T2 mode control bit
- T2C2 Timer T2 mode control bit
- T2C3 Timer T2 mode control bit

T2C3	T2C2	T2C1	T2C0	T2PNDA	T2ENA	T2PNDB	T2ENB

Bit 7 Bit 0

T3CNTRL Register (Address X'00B6)

The T3CNTRL register contains the following bits:

- T3ENB Timer T3 Interrupt Enable for T3B
- T3PNDB Timer T3 Interrupt Pending Flag for T3B pin (T3B capture edge)
- T3ENA Timer T3 Interrupt Enable for Timer Underflow or T3A pin
- T3PNDA Timer T3 Interrupt Pending Flag (Autoload RA in mode 1, T3 Underflow in mode 2, T3a capture edge in mode 3)
- T3C0 Timer T3 Start/Stop control in timer modes 1 and 2

  Timer T3 Underflow Interrupt Pending Flag in timer mode 3
- T3C1 Timer T3 mode control bit
- T3C2 Timer T3 mode control bit
- T3C3 Timer T3 mode control bit

T3C3	T3C2	T3C1	T3C0	T3PNDA	T3ENA	T3PNDB	T3ENB

Bit 7 Bit 0

Timers

The device contains a very versatile set of timers (T0, T1, T2, T3). All timers and associated autoreload/capture registers power up containing random data.

TIMER T0 (IDLE TIMER)

The devices support applications that require maintaining real time and low power with the IDLE mode. This IDLE mode support is furnished by the IDLE timer T0, which is a 16-bit timer. The Timer T0 runs continuously at the fixed rate of the instruction cycle clock, t_C. The user cannot read or write to the IDLE Timer T0, which is a count down timer.

The Timer T0 supports the following functions:

Exit out of the Idle Mode (See Idle Mode description)
WATCHDOG logic (See WATCHDOG description)
Start up delay out of the HALT mode

The IDLE Timer T0 can generate an interrupt when the thirteenth bit toggles. This toggle is latched into the T0PND pending flag, and will occur every 4 ms at the maximum clock frequency (t_C = 1 μs). A control flag T0EN allows the interrupt from the thirteenth bit of Timer T0 to be enabled or disabled. Setting T0EN will enable the interrupt, while resetting it will disable the interrupt.

Timers (Continued)

TIMER T1, TIMER T2 AND TIMER T3

The devices have a set of three powerful timer/counter blocks, T1, T2 and T3. The associated features and functioning of a timer block are described by referring to the timer block Tx. Since the three timer blocks, T1, T2 and T3 are identical, all comments are equally applicable to any of the three timer blocks.

Each timer block consists of a 16-bit timer, Tx, and two supporting 16-bit autoreload/capture registers, RxA and RxB. Each timer block has two pins associated with it, TxA and TxB. The pin TxA supports I/O required by the timer block, while the pin TxB is an input to the timer block. The powerful and flexible timer block allows the device to easily perform all timer functions with minimal software overhead. The timer block has three operating modes: Processor Independent PWM mode, External Event Counter mode, and Input Capture mode.

The control bits TxC3, TxC2, and TxC1 allow selection of the different modes of operation.

Mode 1. Processor Independent PWM Mode

As the name suggests, this mode allows the device to generate a PWM signal with very minimal user intervention. The user only has to define the parameters of the PWM signal (ON time and OFF time). Once begun, the timer block will continuously generate the PWM signal completely independent of the microcontroller. The user software services the timer block only when the PWM parameters require updating.

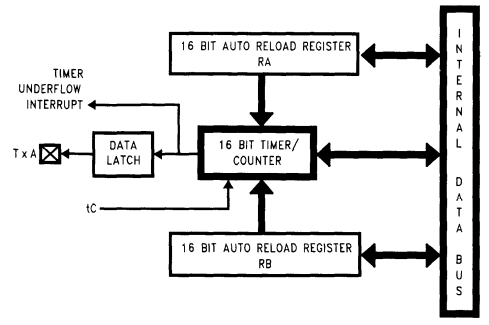
In this mode the timer Tx counts down at a fixed rate of t_C. Upon every underflow the timer is alternately reloaded with the contents of supporting registers, RxA and RxB. The very first underflow of the timer causes the timer to reload from the register RxA. Subsequent underflows cause the timer to be reloaded from the registers alternately beginning with the register RxB.

The Tx Timer control bits, TxC3, TxC2 and TxC1 set up the timer for PWM mode operation.

Figure 8 shows a block diagram of the timer in PWM mode. The underflows can be programmed to toggle the TxA output pin. The underflows can also be programmed to generate interrupts.

Underflows from the timer are alternately latched into two pending flags, TxPNDA and TxPNDB. The user must reset these pending flags under software control. Two control enable flags, TxENA and TxENB, allow the interrupts from the timer underflow to be enabled or disabled. Setting the timer enable flag TxENA will cause an interrupt when a timer underflow causes the RxA register to be reloaded into the timer. Setting the timer enable flag TxENB will cause an interrupt when a timer underflow causes the RxB register to be reloaded into the timer. Resetting the timer enable flags will disable the associated interrupts.

Either or both of the timer underflow interrupts may be enabled. This gives the user the flexibility of interrupting once per PWM period on either the rising or falling edge of the PWM output. Alternatively, the user may choose to interrupt on both edges of the PWM output.



TL/DD12860–10

FIGURE 8. Timer in PWM Mode

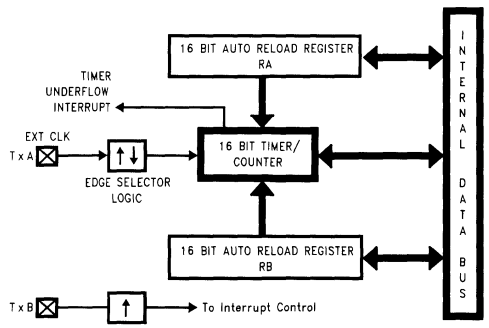
Mode 2. External Event Counter Mode

This mode is quite similar to the processor independent PWM mode described above. The main difference is that the timer, Tx, is clocked by the input signal from the TxA pin. The Tx timer control bits, TxC3, TxC2 and TxC1 allow the timer to be clocked either on a positive or negative edge from the TxA pin. Underflows from the timer are latched into the TxPNDA pending flag. Setting the TxENA control flag will cause an interrupt when the timer underflows.

In this mode the input pin TxB can be used as an independent positive edge sensitive interrupt input if the TxENB control flag is set. The occurrence of a positive edge on the TxB input pin is latched into the TxPNDB flag.

Figure 9 shows a block diagram of the timer in External Event Counter mode.

Note: The PWM output is not available in this mode since the TxA pin is being used as the counter input clock.



TL/DD12860–11

FIGURE 9. Timer in External Event Counter Mode

Mode 3. Input Capture Mode

The device can precisely measure external frequencies or time external events by placing the timer block, Tx, in the input capture mode.

In this mode, the timer Tx is constantly running at the fixed t_C rate. The two registers, RxA and RxB, act as capture registers. Each register acts in conjunction with a pin. The register RxA acts in conjunction with the TxA pin and the register RxB acts in conjunction with the TxB pin.

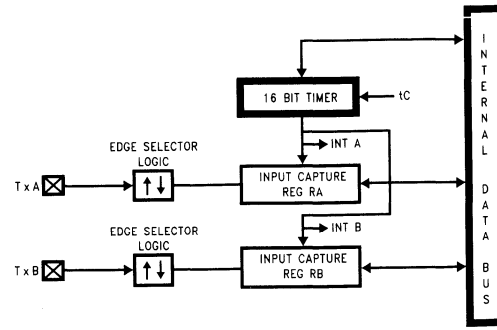
Timers (Continued)

The timer value gets copied over into the register when a trigger event occurs on its corresponding pin. Control bits, TxC3, TxC2 and TxC1, allow the trigger events to be specified either as a positive or a negative edge. The trigger condition for each input pin can be specified independently.

The trigger conditions can also be programmed to generate interrupts. The occurrence of the specified trigger condition on the TxA and TxB pins will be respectively latched into the pending flags, TxPNDA and TxPNDB. The control flag TxENA allows the interrupt on TxA to be either enabled or disabled. Setting the TxENA flag enables interrupts to be generated when the selected trigger condition occurs on the TxA pin. Similarly, the flag TxENB controls the interrupts from the TxB pin.

Underflows from the timer can also be programmed to generate interrupts. Underflows are latched into the timer TxC0 pending flag (the TxC0 control bit serves as the timer underflow interrupt pending flag in the Input Capture mode). Consequently, the TxC0 control bit should be reset when entering the Input Capture mode. The timer underflow interrupt is enabled with the TxENA control flag. When a TxA interrupt occurs in the Input Capture mode, the user must check both the TxPNDA and TxC0 pending flags in order to determine whether a TxA input capture or a timer underflow (or both) caused the interrupt.

Figure 10 shows a block diagram of the timer in Input Capture mode.



TL/DD12860–12

FIGURE 10. Timer in Input Capture Mode

TIMER CONTROL FLAGS

The timers T1, T2 and T3 have indentical control structures. The control bits and their functions are summarized below.

TxC0	Timer Start/Stop control in Modes 1 and 2 (Processor Independent PWM and External Event Counter), where 1 = Start, 0 = Stop Timer Underflow Interrupt Pending Flag in Mode 3 (Input Capture)
TxPNDA	Timer Interrupt Pending Flag
TxPNDB	Timer Interrupt Pending Flag
TxENA	Timer Interrupt Enable Flag
TxENB	Timer Interrupt Enable Flag 1 = Timer Interrupt Enabled 0 = Timer Interrupt Disabled
TxC3	Timer mode control
TxC2	Timer mode control
TxC1	Timer mode control

Timers (Continued)

The timer mode control bits (TxC3, TxC2 and TxC1) are detailed below:

TxC3	TxC2	TxC1	Timer Mode	Interrupt A Source	Interrupt B Source	Timer Counts On
0	0	0	MODE 2 (External Event Counter)	Timer Underflow	Pos. TxB Edge	TxA Pos. Edge
0	0	1	MODE 2 (External Event Counter)	Timer Underflow	Pos. TxB Edge	TxA Neg. Edge
1	0	1	MODE 1 (PWM) TxA Toggle	Autoreload RA	Autoreload RB	t_C
1	0	0	MODE 1 (PWM) No TxA Toggle	Autoreload RA	Autoreload RB	t_C
0	1	0	MODE 3 (Capture) Captures: TxA Pos. Edge TxB Pos. Edge	Pos. TxA Edge or Timer Underflow	Pos. TxB Edge	t_C
1	1	0	MODE 3 (Capture) Captures: TxA Pos. Edge TxB Neg. Edge	Pos. TxA Edge or Timer Underflow	Neg. TxB Edge	t_C
0	1	1	MODE 3 (Capture) Captures: TxA Neg. Edge TxB Pos. Edge	Neg. TxA Edge or Timer Underflow	Pos. TxB Edge	t_C
1	1	1	MODE 3 (Capture) Captures: TxA Neg. Edge TxB Neg. Edge	Neg. TxA Edge or Timer Underflow	Neg. TxB Edge	t_C

Power Save Modes

The devices offer the user two power save modes of operation: HALT and IDLE. In the HALT mode, all microcontroller activities are stopped. In the IDLE mode, the on-board oscillator circuitry the WATCHDOG logic, the Clock Monitor and timer T0 are active but all other microcontroller activities are stopped. In either mode, all on-board RAM, registers, I/O states, and timers (with the exception of T0) are unaltered.

HALT MODE

The devices can be placed in the HALT mode by writing a "1" to the HALT flag (G7 data bit). All microcontroller activities, including the clock and timers, are stopped. The WATCHDOG logic on the device is disabled during the HALT mode. However, the clock monitor circuitry if enabled remains active and will cause the WATCHDOG output pin (WDOUT) to go low. If the HALT mode is used and the user does not want to activate the WDOUT pin, the Clock Monitor should be disabled after the device comes out of reset (resetting the Clock Monitor control bit with the first write to the WDSVR register). In the HALT mode, the power requirements of the device are minimal and the applied voltage (V_{CC}) may be decreased to V_r (V_r = 2.0V) without altering the state of the machine.

The devices support three different ways of exiting the HALT mode. The first method of exiting the HALT mode is with the Multi-Input Wake Up feature on the L port. The second method is with a low to high transition on the CKO (G7) pin. This method precludes the use of the crystal clock configuration (since CKO becomes a dedicated output), and so may be used with an RC clock configuration. The third method of exiting the HALT mode is by pulling the $\overline{\text{RESET}}$ pin low.

Since a crystal or ceramic resonator may be selected as the oscillator, the Wake Up signal is not allowed to start the chip running immediately since crystal oscillators and ceramic resonators have a delayed start up time to reach full amplitude and frequency stability. The IDLE timer is used to generate a fixed delay to ensure that the oscillator has indeed stabilized before allowing instruction execution. In this case, upon detecting a valid Wake Up signal, only the oscillator circuitry is enabled. The IDLE timer is loaded with a value of 256 and is clocked with the t_C instruction cycle clock. The t_C clock is derived by dividing the oscillator clock down by a factor of 10. The Schmitt trigger following the CKI inverter on the chip ensures that the IDLE timer is clocked only when the oscillator has a sufficiently large amplitude to meet the Schmitt trigger specifications. This Schmitt trigger is not part of the oscillator closed loop. The startup timeout from the IDLE timer enables the clock signals to be routed to the rest of the chip.

If an RC clock option is being used, the fixed delay is introduced optionally. A control bit, CLKDLY, mapped as configuration bit G7, controls whether the delay is to be introduced or not. The delay is included if CLKDLY is set, and excluded if CLKDLY is reset. The CLKDLY bit is cleared on reset.

Power Save Modes (Continued)

The WATCHDOG detector circuit is inhibited during the HALT mode. However, the clock monitor circuit if enabled remains active during HALT mode in order to ensure a clock monitor error if the device inadvertently enters the HALT mode as a result of a runaway program or power glitch.

IDLE MODE

The device is placed in the IDLE mode by writing a "1" to the IDLE flag (G6 data bit). In this mode, all activities, except the associated on-board oscillator circuitry, the WATCHDOG logic, the clock monitor and the IDLE Timer T0, are stopped. The power supply requirements of the micro-controller in this mode of operation are typically around 30% of normal power requirement of the microcontroller.

As with the HALT mode, the device can be returned to normal operation with a reset, or with a Multi-Input Wake Up from the L Port. Alternately, the microcontroller resumes normal operation from the IDLE mode when the thirteenth bit (representing 4.096 ms at internal clock frequency of 1 MHz, t_c = 1 μs) of the IDLE Timer toggles.

This toggle condition of the thirteenth bit of the IDLE Timer T0 is latched into the T0PND pending flag.

The user has the option of being interrupted with a transition on the thirteenth bit of the IDLE Timer T0. The interrupt can be enabled or disabled via the T0EN control bit. Setting the T0EN flag enables the interrupt and vice versa.

The user can enter the IDLE mode with the Timer T0 interrupt enabled. In this case, when the T0PND bit gets set, the device will first execute the Timer T0 interrupt service routine and then return to the instruction following the "Enter Idle Mode" instruction.

Alternatively, the user can enter the IDLE mode with the IDLE Timer T0 interrupt disabled. In this case, the device will resume normal operation with the instruction immediately following the "Enter IDLE Mode" instruction.

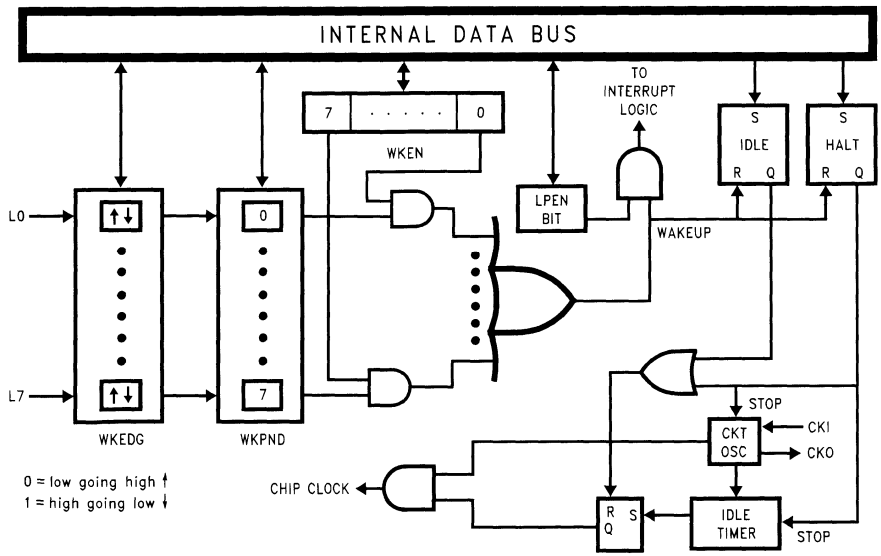
Note: It is necessary to program two NOP instructions following both the set HALT mode and set IDLE mode instructions. These NOP instructions are necessary to allow clock resynchronization following the HALT or IDLE modes.

Due to the on-board 8k EPROM with port recreation logic, the HALT/IDLE current is much higher compared to the equivalent masked port.

Multi-Input Wake Up

The Multi-Input Wake Up feature is ued to return (Wake Up) the device from either the HALT or IDLE modes. Alternately Multi-Input Wake Up/Interrupt feature may also be used to generate up to 8 edge selectable external interrupts.

Figure 11 shows the Multi-Input Wake Up logic. The Multi-Input Wake Up feature utilizes the L Port. The user selects which particular L port bit (or combination of L Port bits) will cause the device to exit the HALT or IDLE modes. The selection is done through the Reg: WKEN. The Reg: WKEN



TL/DD12860–13

FIGURE 11. Multi-Input Wake Up Logic

Multi-Input Wake Up (Continued)

is an 8-bit read/write register, which contains a control bit for every L port bit. Setting a particular WKEN bit enables a Wake Up from the associated L port pin.

The user can select whether the trigger condition on the selected L Port pin is going to be either a positive edge (low to high transition) or a negative edge (high to low transition). This selection is made via the Reg: WKEDG, which is an 8-bit control register with a bit assigned to each L Port pin. Setting the control bit will select the trigger condition to be a negative edge on that particular L Port pin. Resetting the bit selects the trigger condition to be a positive edge. Changing an edge select entails several steps in order to avoid a pseudo Wake Up condition as a result of the edge change. First, the associated WKEN bit should be reset, followed by the edge select change in WKEDG. Next, the associated WKPND bit should be cleared, followed by the associated WKEN bit being re-enabled.

An example may serve to clarify this procedure. Suppose we wish to change the edge select from positive (low going high) to negative (high going low) for L Port bit 5, where bit 5 has previously been enabled for an input interrupt. The program would be as follows:

```
RBIT  5, WKEN
SBIT  5, WKEDG
RBIT  5, WKPND
SBIT  5, WKEN
```

If the L port bits have been used as outputs and then changed to inputs with Multi-Input Wake Up/Interrupt, a safety procedure should also be followed to avoid inherited pseudo wakeup conditions. After the selected L port bits have been changed from output to input but before the associated WKEN bits are enabled, the associated edge select bits in WKEDG should be set or reset for the desired edge selects, followed by the associated WKPND bits being cleared.

This same procedure should be used following reset, since the L port inputs are left floating as a result of reset.

The occurrence of the selected trigger condition for Multi-Input Wake Up is latched into a pending register called WKPND. The respective bits of the WKPND register will be set on the occurrence of the selected trigger edge on the corresponding Port L pin. The user has the responsibility of clearing these pending flags. Since WKPND is a pending register for the occurrence of selected Wake Up conditions, the device will not enter the HALT mode if any Wake Up bit is both enabled and pending. Consequently, the user has the responsibility of clearing the pending flags before attempting to enter the HALT mode.

WKEN, WKPND and WKEDG are all read/write registers, and are cleared at reset.

PORT L INTERRUPTS

Port L provides the user with an additional eight fully selectable, edge sensitive interrupts which are all vectored into the same service subroutine.

The interrupt from Port L shares logic with the wake up circuitry. The register WKEN allows interrupts from Port L to be individually enabled or disabled. The register WKEDG specifies the trigger condition to be either a positive or a negative edge. Finally, the register WKPND latches in the pending trigger conditions.

The GIE (Global Interrupt Enable) bit enables the interrupt function.

A control flag, LPEN, functions as a global interrupt enable for Port L interrupts. Setting the LPEN flag will enable interrupts and vice versa. A separate global pending flag is not needed since the register WKPND is adequate.

Since Port L is also used for waking the device out of the HALT or IDLE modes, the user can elect to exit the HALT or IDLE modes either with or without the interrupt enabled. If he elects to disable the interrupt, then the device will restart execution from the instruction immediately following the instruction that placed the microcontroller in the HALT or IDLE modes. In the other case, the device will first execute the interrupt service routine and then revert to normal operation.

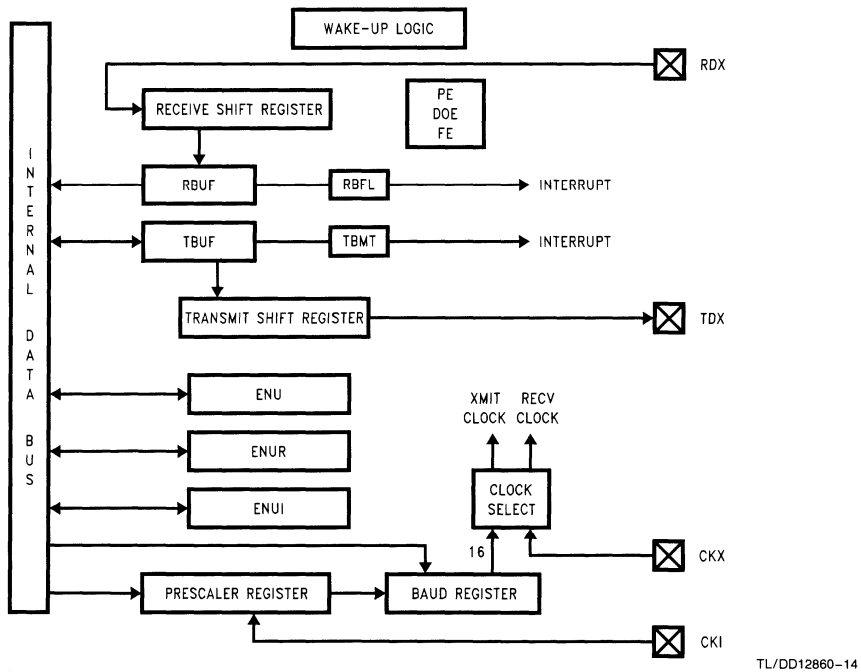
The Wake Up signal will not start the chip running immediately since crystal oscillators or ceramic resonators have a finite start up time. The IDLE Timer (T0) generates a fixed delay to ensure that the oscillator has indeed stabilized before allowing the device to execute instructions. In this case, upon detecting a valid Wake Up signal, only the oscillator circuitry and the IDLE Timer T0 are enabled. The IDLE Timer is loaded with a value of 256 and is clocked from the t_C instruction cycle clock. The t_C clock is derived by dividing down the oscillator clock by a factor of 10. A Schmitt trigger following the CKI on-chip inverter ensures that the IDLE timer is clocked only when the oscillator has a sufficiently large amplitude to meet the Schmitt trigger specifications. This Schmitt trigger is not part of the oscillator closed loop. The startup timeout from the IDLE timer enables the clock signals to be routed to the rest of the chip.

If the RC clock option is used, the fixed delay is under software control. A control flag, CLKDLY, in the G7 configuration bit allows the clock start up delay to be optionally inserted. Setting CLKDLY flag high will cause clock start up delay to be inserted and resetting it will exclude the clock start up delay. The CLKDLY flag is cleared during reset, so the clock start up delay is not present following reset with the RC clock options.

UART

The device contains a full-duplex software programmable UART. The UART *(Figure 12)* consists of a transmit shift register, a receiver shift register and seven addressable registers, as follows: a transmit buffer register (TBUF), a receiver buffer register (RBUF), a UART control and status register (ENU), a UART receive control and status register (ENUR), a UART interrupt and clock source register (ENUI), a prescaler select register (PSR) and baud (BAUD) register. The ENU register contains flags for transmit and receive functions; this register also determines the length of the data frame (7, 8 or 9 bits), the value of the ninth bit in transmission, and parity selection bits. The ENUR register flags framing, data overrun and parity errors while the UART is receiving.

Other functions of the ENUR register include saving the ninth bit received in the data frame, enabling or disabling the UART's attention mode of operation and providing additional receiver/transmitter status information via RCVG and XMTG bits. The determination of an internal or external clock source is done by the ENUI register, as well as selecting the number of stop bits and enabling or disabling transmit and receive interrupts. A control flag in this register can also select the UART mode of operation: asynchronous or synchronous.



TL/DD12860–14

FIGURE 12. UART Block Diagram

UART (Continued)

UART CONTROL AND STATUS REGISTERS

The operation of the UART is programmed through three registers: ENU, ENUR and ENUI. The function of the individual bits in these registers is as follows:

ENU-UART Control and Status Register (Address at 0BA)

PEN	PSEL1	XBIT9/ PSEL0	CHL1	CHL0	ERR	RBFL	TBMT
0RW	0RW	0RW	0RW	0RW	0R	0R	1R

Bit 7 Bit 0

ENUR-UART Receive Control and Status Register (Address at 0BB)

DOE	FE	PE	SPARE	RBIT9	ATTN	XMTG	RCVG
0RD	0RD	0RD	0RW*	0R	0RW	0R	0R

Bit7 Bit0

ENUI-UART Interrupt and Clock Source Register (Address at 0BC)

STP2	STP78	ETDX	SSEL	XRCLK	XTCLK	ERI	ETI
0RW	0RW	0RW	0RW	0RW	0RW	0RW	0RW

Bit7 Bit0

*Bit is not used.

0 Bit is cleared on reset.

1 Bit is set to one on reset.

R Bit is read-only; it cannot be written by software.

RW Bit is read/write.

D Bit is cleared on read; when read by software as a one, it is cleared automatically. Writing to the bit does not affect its state.

DESCRIPTION OF UART REGISTER BITS

ENU—UART CONTROL AND STATUS REGISTER

TBMT: This bit is set when the UART transfers a byte of data from the TBUF register into the TSFT register for transmission. It is automatically reset when software writes into the TBUF register.

RBFL: This bit is set when the UART has received a complete character and has copied it into the RBUF register. It is automatically reset when software reads the character from RBUF.

ERR: This bit is a global UART error flag which gets set if any or a combination of the errors (DOE, FE, PE) occur.

CHL1, CHL0: These bits select the character frame format. Parity is not included and is generated/verified by hardware.

CHL1 = 0, CHL0 = 0 The frame contains eight data bits.
CHL1 = 0, CHL0 = 1 The frame contains seven data bits.
CHL1 = 1, CHL0 = 0 The frame contains nine data bits.
CHL1 = 1, CHL0 = 1 Loopback Mode selected. Transmitter output internally looped back to receiver input. Nine bit framing format is used.

XBIT9/PSEL0: Programs the ninth bit for transmission when the UART is operating with nine data bits per frame. For seven or eight data bits per frame, this bit in conjunction with PSEL1 selects parity.

PSEL1, PSEL0: Parity select bits.

PSEL1 = 0, PSEL0 = 0 Odd Parity (if Parity enabled)
PSEL1 = 0, PSEL0 = 1 Odd Parity (if Parity enabled)
PSEL1 = 1, PSEL0 = 0 Mark(1) (if Parity enabled)
PSEL1 = 1, PSEL1 = 1 Space(0) (if Parity enabled)

PEN: This bit enables/disables Parity (7- and 8-bit modes only).

PEN = 0 Parity disabled.
PEN = 1 Parity enabled.

ENUR—UART RECEIVE CONTROL AND STATUS REGISTER

RCVG: This bit is set high whenever a framing error occurs and goes low when RDX goes high.

XMTG: This bit is set to indicate that the UART is transmitting. It gets reset at the end of the last frame (end of last Stop bit).

ATTN: ATTENTION Mode is enabled while this bit is set. This bit is cleared automatically on receiving a character with data bit nine set.

RBIT9: Contains the ninth data bit received when the UART is operating with nine data bits per frame.

SPARE: Reserved for future use.

PE: Flags a Parity Error.

PE = 0 Indicates no Parity Error has been detected since the last time the ENUR register was read.
PE = 1 Indicates the occurrence of a Parity Error.

FE: Flags a Framing Error.

FE = 0 Indicates no Framing Error has been detected since the last time the ENUR register was read.
FE = 1 Indicates the occurrence of a Framing Error.

DOE: Flags a Data Overrun Error.

DOE = 0 Indicates no Data Overrun Error has been detected since the last time the ENUR register was read.
DOE = 1 Indicates the occurrence of a Data Overrun Error.

ENUI—UART INTERRUPT AND CLOCK SOURCE REGISTER

ETI: This bit enables/disables interrupt from the transmitter section.

ETI = 0 Interrupt from the transmitter is disabled.
ETI = 1 Interrupt from the transmitter is enabled.

ERI: This bit enables/disables interrupt from the receiver section.

ERI = 0 Interrupt from the receiver is disabled.
ERI = 1 Interrupt from the receiver is enabled.

XTCLK: This bit selects the clock source for the transmitter section.

XTCLK = 0 The clock source is selected through the PSR and BAUD registers.
XTCLK = 1 Signal on CKX (L1) pin is used as the clock.

XRCLK: This bit selects the clock source for the receiver section.

XRCLK = 0 The clock source is selected through the PSR and BAUD registers.
XRCLK = 1 Signal on CKX (L1) pin is used as the clock.

SSEL: UART mode select.

SSEL = 0 Asynchronous Mode.
SSEL = 1 Synchronous Mode.

UART (Continued)

ETDX: TDX (UART Transmit Pin) is the alternate function assigned to Port L pin L2; it is selected by setting ETDX bit. To simulate line break generation, software should reset ETDX bit and output logic zero to TDX pin through Port L data and configuration registers.

STP78: This bit is set to program the last Stop bit to be 7/8th of a bit in length.

STP2: This bit programs the number of Stop bits to be transmitted.

STP2 = 0 One Stop bit transmitted.
STP2 = 1 Two Stop bits transmitted.

Associated I/O Pins

Data is transmitted on the TDX pin and received on the RDX pin. TDX is the alternate function assigned to Port L pin L2; it is selected by setting ETDX (in the ENUI register) to one. RDX is an inherent function of Port L pin L3, requiring no setup.

The baud rate clock for the UART can be generated on-chip, or can be taken from an external source. Port L pin L1 (CKX) is the external clock I/O pin. The CKX pin can be either an input or an output, as determined by Port L Configuration and Data registers (Bit 1). As an input, it accepts a clock signal which may be selected to drive the transmitter and/or receiver. As an output, it presents the internal Baud Rate Generator output.

UART Operation

The UART has two modes of operation: asynchronous mode and synchronous mode.

ASYNCHRONOUS MODE

This mode is selected by resetting the SSEL (in the ENUI register) bit to zero. The input frequency to the UART is 16 times the baud rate.

The TSFT and TBUF registers double-buffer data for transmission. While TSFT is shifting out the current character on the TDX pin, the TBUF register may be loaded by software with the next byte to be transmitted. When TSFT finishes transmitting the current character the contents of TBUF are transferred to the TSFT register and the Transmit Buffer Empty Flag (TBMT in the ENU register) is set. The TBMT flag is automatically reset by the UART when software loads a new character into the TBUF register. There is also the XMTG bit which is set to indicate that the UART is transmitting. This bit gets reset at the end of the last frame (end of last Stop bit). TBUF is a read/write register.

The RSFT and RBUF registers double-buffer data being received. The UART receiver continually monitors the signal on the RDX pin for a low level to detect the beginning of a Start bit. Upon sensing this low level, it waits for half a bit time and samples again. If the RDX pin is still low, the receiver considers this to be a valid Start bit, and the remaining bits in the character frame are each sampled a single time, at the mid-bit position. Serial data input on the RDX pin is shifted into the RSFT register. Upon receiving the complete character, the contents of the RSFT register are copied into the RBUF register and the Received Buffer Full Flag (RBFL) is set. RBFL is automatically reset when software reads the character from the RBUF register. RBUF is a read only register. There is also the RCVG bit which is set high when a framing error occurs and goes low once RDX goes high. TBMT, XMTG, RBFL and RCVG are read only bits.

SYNCHRONOUS MODE

In this mode data is transferred synchronously with the clock. Data is transmitted on the rising edge and received on the falling edge of the synchronous clock.

This mode is selected by setting SSEL bit in the ENUI register. The input frequency to the UART is the same as the baud rate.

When an external clock input is selected at the CKX pin, data transmit and receive are performed synchronously with this clock through TDX/RDX pins.

If data transmit and receive are selected with the CKX pin as clock output, the device generates the synchronous clock output at the CKX pin. The internal baud rate generator is used to produce the synchronous clock. Data transmit and receive are performed synchronously with this clock.

FRAMING FORMATS

The UART supports several serial framing formats *(Figure 13)*. The format is selected using control bits in the ENU, ENUR and ENUI registers.

The first format (1, 1a, 1b, 1c) for data transmission (CHL0 = 1, CHL1 = 0) consists of Start bit, seven Data bits (excluding parity) and 7/8, one or two Stop bits. In applications using parity, the parity bit is generated and verified by hardware.

The second format (CHL0 = 0, CHL1 = 0) consists of one Start bit, eight Data bits (excluding parity) and 7/8, one or two Stop bits. Parity bit is generated and verified by hardware.

The third format for transmission (CHL0 = 0, CHL1 = 1) consists of one Start bit, nine Data bits and 7/8, one or two Stop bits. This format also supports the UART "ATTENTION" feature. When operating in this format, all eight bits of TBUF and RBUF are used for data. The ninth data bit is transmitted and received using two bits in the ENU and ENUR registers, called XBIT9 and RBIT9. RBIT9 is a read only bit. Parity is not generated or verified in this mode.

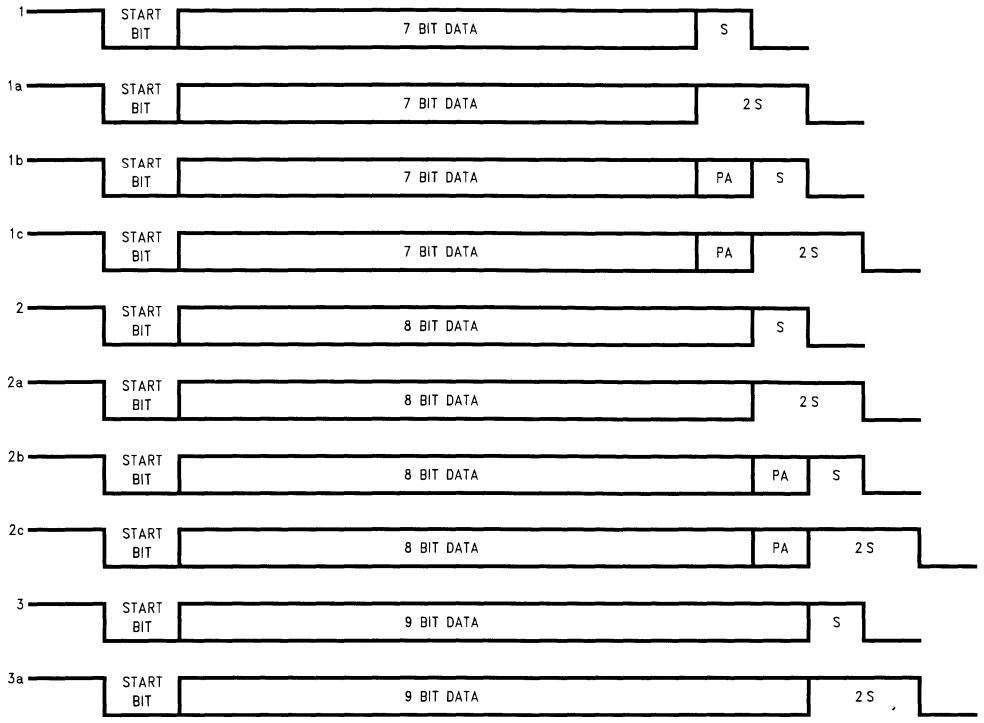
For any of the above framing formats, the last Stop bit can be programmed to be 7/8th of a bit in length. If two Stop bits are selected and the 7/8th bit is set (selected), the second Stop bit will be 7/8th of a bit in length.

The parity is enabled/disabled by PEN bit located in the ENU register. Parity is selected for 7- and 8-bit modes only. If parity is enabled (PEN = 1), the parity selection is then performed by PSEL0 and PSEL1 bits located in the ENU register.

Note that the XBIT9/PSEL0 bit located in the ENU register serves two mutually exclusive functions. This bit programs the ninth bit for transmission when the UART is operating with nine data bits per frame. There is no parity selection in this framing format. For other framing formats XBIT9 is not needed and the bit is PSEL0 used in conjunction with PSEL1 to select parity.

The frame formats for the receiver differ from the transmitter in the number of Stop bits required. The receiver only requires one Stop bit in a frame, regardless of the setting of the Stop bit selection bits in the control register. Note that an implicit assumption is made for full duplex UART operation that the framing formats are the same for the transmitter and receiver.

UART Operation (Continued)



TL/DD12860–15

FIGURE 13. Framing Formats

UART INTERRUPTS

The UART is capable of generating interrupts. Interrupts are generated on Receive Buffer Full and Transmit Buffer Empty. Both interrupts have individual interrupt vectors. Two bytes of program memory space are reserved for each interrupt vector. The two vectors are located at addresses 0xEC to 0xEF Hex in the program memory space. The interrupts can be individually enabled or disabled using Enable Transmit Interrupt (ETI) and Enable Receive Interrupt (ERI) bits in the ENUI register.

The interrupt from the transmitter is set pending, and remains pending, as long as both the TBMT and ETI bits are set. To remove this interrupt, software must either clear the ETI bit or write to the TBUF register (thus clearing the TBMT bit).

The interrupt from the receiver is set pending, and remains pending, as long as both the RBFL and ERI bits are set. To remove this interrupt, software must either clear the ERI bit or read from the RBUF register (thus clearing the RBFL bit).

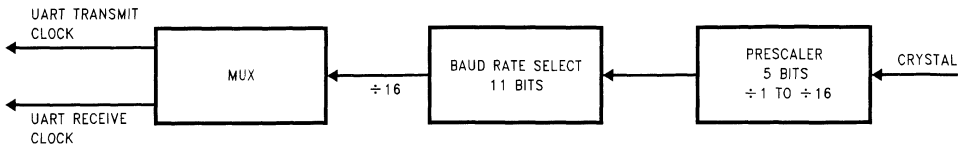
Baud Clock Generation

The clock inputs to the transmitter and receiver sections of the UART can be individually selected to come either from an external source at the CKX pin (port L, pin L1) or from a source selected in the PSR and BAUD registers. Internally, the basic baud clock is created from the oscillator frequency through a two-stage divider chain consisting of a 1–16 (increments of 0.5) prescaler and an 11-bit binary counter. *(Figure 14)* The divide factors are specified through two read/write registers shown in *Figure 15.* Note that the 11-bit Baud Rate Divisor spills over into the Prescaler Select Register (PSR). PSR is cleared upon reset.

As shown in Table III, a Prescaler Factor of 0 corresponds to NO CLOCK. NO CLOCK condition is the UART power down mode where the UART clock is turned off for power saving purpose. The user must also turn the UART clock off when a different baud rate is chosen.

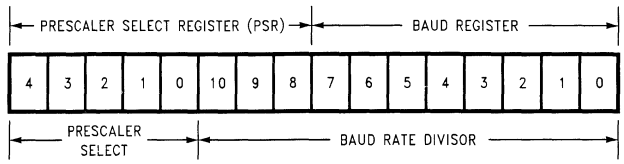
The correspondences between the 5-bit Prescaler Select and Prescaler factors are shown in Table III. There are many ways to calculate the two divisor factors, but one particularly effective method would be to achieve a 1.8432 MHz frequency coming out of the first stage. The 1.8432 MHz prescaler output is then used to drive the software programmable baud rate counter to create a x16 clock for the following baud rates: 110, 134.5, 150, 300, 600, 1200, 1800, 2400, 3600, 4800, 7200, 9600, 19200 and 38400 (Table IV). Other baud rates may be created by using appropriate divisors. The x16 clock is then divided by 16 to provide the rate for the serial shift registers of the transmitter and receiver.

Baud Clock Generation (Continued)



TL/DD12860–16

FIGURE 14. UART BAUD Clock Generation



TL/DD12860–17

FIGURE 15. UART BAUD Clock Divisor Registers

TABLE III. Prescaler Factors

Prescaler Select	Prescaler Factor	Prescaler Select	Prescaler Factor
00000	NO CLOCK	10000	8.5
00001	1	10001	9
00010	1.5	10010	9.5
00011	2	10011	10
00100	2.5	10100	10.5
00101	3	10101	11
00110	3.5	10110	11.5
00111	4	10111	12
01000	4.5	11000	12.5
01001	5	11001	13
01010	5.5	11010	13.5
01011	6	11011	14
01100	6.5	11100	14.5
01101	7	11101	15
01110	7.5	11110	15.5
01111	8	11111	16

TABLE IV. Baud Rate Divisors (1.8432 MHz Prescaler Output)

Baud Rate	Baud Rate Divisor − 1 (N-1)
110 (110.03)	1046
134.5 (134.58)	855
150	767
300	383
600	191
1200	95
1800	63
2400	47
3600	31
4800	23
7200	15
9600	11
19200	5
38400	2

Note: The entries in Table IV assume a prescaler output of 1.8432 MHz. In the asynchronous mode the baud rate could be as high as 625k.

As an example, considering the Asynchronous Mode and a CKI clock of 4.608 MHz, the prescaler factor selected is:

$$4.608/1.8432 = 2.5$$

The 2.5 entry is available in Table III. The 1.8432 MHz prescaler output is then used with proper Baud Rate Divisor (Table II) to obtain different baud rates. For a baud rate of 19200 e.g., the entry in Table IV is 5.

$N - 1 = 5$ ($N - 1$ is the value from Table IV)

$N = 6$ (N is the Baud Rate Divisor)

$$\text{Baud Rate} = 1.8432\ \text{MHz}/(16 \times 6) = 19200$$

The divide by 16 is performed because in the asynchronous mode, the input frequency to the UART is 16 times the baud rate. The equation to calculate baud rates is given below.

The actual Baud Rate may be found from:

$$BR = Fc/(16 \times N \times P)$$

Baud Clock Generation (Continued)

Where:

BR is the Baud Rate

Fc is the CKI frequency

N is the Baud Rate Divisor (Table IV).

P is the Prescaler Divide Factor selected by the value in the Prescaler Select Register (Table III)

Note: In the Synchronous Mode, the divisor 16 is replaced by two.

Example:

Asynchronous Mode:

Crystal Frequency = 5 MHz

Desired baud rate = 9600

Using the above equation N × P can be calculated first.

$$N \times P = (5 \times 10^6)/(16 \times 9600) = 32.552$$

Now 32.552 is divided by each Prescaler Factor (Table III) to obtain a value closest to an integer. This factor happens to be 6.5 (P = 6.5).

$$N = 32.552/6.5 = 5.008 \ (N = 5)$$

The programmed value (from Table IV) should be 4 (N − 1).

Using the above values calculated for N and P:

$$BR = (5 \times 10^6)/(16 \times 5 \times 6.5) = 9615.384$$

$$\% \text{ error} = (9615.385 - 9600)/9600 = 0.16$$

Effect of HALT/IDLE

The UART logic is reinitialized when either the HALT or IDLE modes are entered. This reinitialization sets the TBMT flag and resets all read only bits in the UART control and status registers. Read/Write bits remain unchanged. The Transmit Buffer (TBUF) is not affected, but the Transmit Shift register (TSFT) bits are set to one. The receiver registers RBUF and RSFT are not affected.

The device will exit from the HALT/IDLE modes when the Start bit of a character is detected at the RDX (L3) pin. This feature is obtained by using the Multi-Input Wake Up scheme provided on the device.

Before entering the HALT or IDLE modes the user program must select the Wake Up source to be on the RDX pin. This selection is done by setting bit 3 of WKEN (Wake Up Enable) register. The Wake Up trigger condition is then selected to be high to low transition. This is done via the WKEDG register (Bit 3 is one.)

If the device is halted and crystal oscillator is used, the Wake Up signal will not start the chip running immediately because of the finite start up time requirement of the crystal oscillator. The idle timer (T0) generates a fixed (256 t_c) delay to ensure that the oscillator has indeed stabilized before allowing the device to execute code. The user has to consider this delay when data transfer is expected immediately after exiting the HALT mode.

Diagnostic

Bits CHARL0 and CHARL1 in the ENU register provide a loopback feature for diagnostic testing of the UART. When these bits are set to one, the following occur: The receiver input pin (RDX) is internally connected to the transmitter output pin (TDX); the output of the Transmitter Shift Register is "looped back" into the Receive Shift Register input. In this mode, data that is transmitted is immediately received. This feature allows the processor to verify the transmit and receive data paths of the UART.

Note that the framing format for this mode is the nine bit format; one Start bit, nine data bits, and 7/8, one or two Stop bits. Parity is not generated or verified in this mode.

Attention Mode

The UART Receiver section supports an alternate mode of operation, referred to as ATTENTION Mode. This mode of operation is selected by the ATTN bit in the ENUR register. The data format for transmission must also be selected as having nine Data bits and either 7/8, one or two Stop bits.

The ATTENTION mode of operation is intended for use in networking the device with other processors. Typically in such environments the messages consists of device addresses, indicating which of several destinations should receive them, and the actual data. This Mode supports a scheme in which addresses are flagged by having the ninth bit of the data field set to a 1. If the ninth bit is reset to a zero the byte is a Data byte.

While in ATTENTION mode, the UART monitors the communication flow, but ignores all characters until an address character is received. Upon receiving an address character, the UART signals that the character is ready by setting the RBFL flag, which in turn interrupts the processor if UART Receiver interrupts are enabled. The ATTN bit is also cleared automatically at this point, so that data characters as well as address characters are recognized. Software examines the contents of the RBUF and responds by deciding either to accept the subsequent data stream (by leaving the ATTN bit reset) or to wait until the next address character is seen (by setting the ATTN bit again).

Operation of the UART Transmitter is not affected by selection of this Mode. The value of the ninth bit to be transmitted is programmed by setting XBIT9 appropriately. The value of the ninth bit received is obtained by reading RBIT9. Since this bit is located in ENUR register where the error flags reside, a bit operation on it will reset the error flags.

Comparators

The devices contain two differential comparators, each with a pair of inputs (positive and negative) and an output. Ports I1–I3 and I4–I6 are used for the comparators. The following is the Port I assignment:

- I1 Comparator1 negative input
- I2 Comparator1 positive input
- I3 Comparator1 output
- I4 Comparator2 negative input
- I5 Comparator2 positive input
- I6 Comparator2 output

A Comparator Select Register (CMPSL) is used to enable the comparators, read the outputs of the comparators internally, and enable the outputs of the comparators to the pins. Two control bits (enable and output enable) and one result bit are associated with each comparator. The comparator result bits (CMP1RD and CMP2RD) are read only bits which will read as zero if the associated comparator is not enabled. The Comparator Select Register is cleared with reset, resulting in the comparators being disabled. The comparators should also be disabled before entering either the HALT or IDLE modes in order to save power. The configuration of the CMPSL register is as follows:

Comparators (Continued)

CMPSL REGISTER (ADDRESS X'00B7)

The CMPSL register contains the following bits:

CMP1EN Enable comparator 1

CMP1RD Comparator 1 result (this is a read only bit, which will read as 0 if the comparator is not enabled)

CMP10E Selects pin I3 as comparator 1 output provided that CMPIEN is set to enable the comparator

CMP2EN Enable comparator 2

CMP2RD Comparator 2 result (this is a read only bit, which will read as 0 if the comparator is not enabled)

CMP20E Selects pin I6 as comparator 2 output provided that CMP2EN is set to enable the comparator

Unused	CMP20E	CMP2RD	CMP2EN	CMP10E	CMP1RD	CMP1EN	Unused

Bit 7 Bit 0

Note that the two unused bits of CMPSL may be used as software flags.

Comparator outputs have the same spec as Ports L and G except that the rise and fall times are symmetrical.

Interrupts

The devices support a vectored interrupt scheme. It supports a total of fourteen interrupt sources. The following table lists all the possible device interrupt sources, their arbitration ranking and the memory locations reserved for the interrupt vector for each source.

Two bytes of program memory space are reserved for each interrupt source. All interrupt sources except the software interrupt are maskable. Each of the maskable interrupts have an Enable bit and a Pending bit. A maskable interrupt is active if its associated enable and pending bits are set. If GIE = 1 and an interrupt is active, then the processor will be interrupted as soon as it is ready to start executing an instruction except if the above conditions happen during the Software Trap service routine. This exception is described in the Software Trap sub-section.

The interruption process is accomplished with the INTR instruction (opcode 00), which is jammed inside the Instruction Register and replaces the opcode about to be executed. The following steps are performed for every interrupt:

1. The GIE (Global Interrupt Enable) bit is reset.
2. The address of the instruction about to be executed is pushed into the stack.
3. The PC (Program Counter) branches to address 00FF. This procedure takes 7 t_C cycles to execute.

Arbitration Ranking	Source	Description	Vector Address Hi-Low Byte
(1) Highest	Software	INTR Instruction	0yFE–0yFF
(2)	Reserved	for Future Use	0yFC–0yFD
(3)	External	Pin G0 Edge	0yFA–0yFB
(4)	Timer T0	Underflow	0yF8–0yF9
(5)	Timer T1	T1A/Underflow	0yF6–0yF7
(6)	Timer T1	T1B	0yF4–0yF5
(7)	MICROWIRE/PLUS	BUSY Goes Low	0yF2–0yF3
(8)	Reserved	for Future Use	0yF0–0yF1
(9)	UART	Receive	0yEE–0yEF
(10)	UART	Transmit	0yEC–0yED
(11)	Timer T2	T2A/Underflow	0yEA–0yEB
(12)	Timer T2	T2B	0yE8–0yE9
(13)	Timer T3	T3A/Underflow	0yE6–0yE7
(14)	Timer T3	T3B	0yE4–0yE5
(15)	Port L/Wake Up	Port L Edge	0yE2–0yE3
(16) Lowest	Default	VIS Instr. Execution without Any Interrupts	0yE0–0yE1

y is VIS page, $y \neq 0$.

Interrupts (Continued)

At this time, since GIE = 0, other maskable interrupts are disabled. The user is now free to do whatever context switching is required by saving the context of the machine in the stack with PUSH instructions. The user would then program a VIS (Vector Interrupt Select) instruction in order to branch to the interrupt service routine of the highest priority interrupt enabled and pending at the time of the VIS. Note that this is not necessarily the interrupt that caused the branch to address location 00FF Hex prior to the context switching.

Thus, if an interrupt with a higher rank than the one which caused the interruption becomes active before the decision of which interrupt to service is made by the VIS, then the interrupt with the higher rank will override any lower ones and will be acknowledged. The lower priority interrupt(s) are still pending, however, and will cause another interrupt immediately following the completion of the interrupt service routine associated with the higher priority interrupt just serviced. This lower priority interrupt will occur immediately following the RETI (Return from Interrupt) instruction at the end of the interrupt service routine just completed.

Inside the interrupt service routine, the associated pending bit has to be cleared by software. The RETI (Return from Interrupt) instruction at the end of the interrupt service routine will set the GIE (Global Interrupt Enable) bit, allowing the processor to be interrupted again if another interrupt is active and pending.

The VIS instruction looks at all the active interrupts at the time it is executed and performs an indirect jump to the beginning of the service routine of the one with the highest rank.

The addresses of the different interrupt service routines, called vectors, are chosen by the user and stored in ROM in a table starting at 01E0 (assuming that VIS is located between 00FF and 01DF). The vectors are 15-bit wide and therefore occupy 2 ROM locations.

VIS and the vector table must be located in the same 256-byte block (0y00 to 0yFF) except if VIS is located at the last address of a block. In this case, the table must be in the next block. The vector table cannot be inserted in the first 256-byte block (y ≠ 0).

The vector of the maskable interrupt with the lowest rank is located at 0yE0 (Hi-Order byte) and 0yE1 (Lo-Order byte) and so forth in increasing rank number. The vector of the maskable interrupt with the highest rank is located at 0yFA (Hi-Order byte) and 0yFB (Lo-Order byte).

The Software Trap has the highest rank and its vector is located at 0yFE and 0yFF.

If, by accident, a VIS gets executed and no interrupt is active, then the PC (Program Counter) will branch to a vector located at 0yE0–0yE1.

WARNING

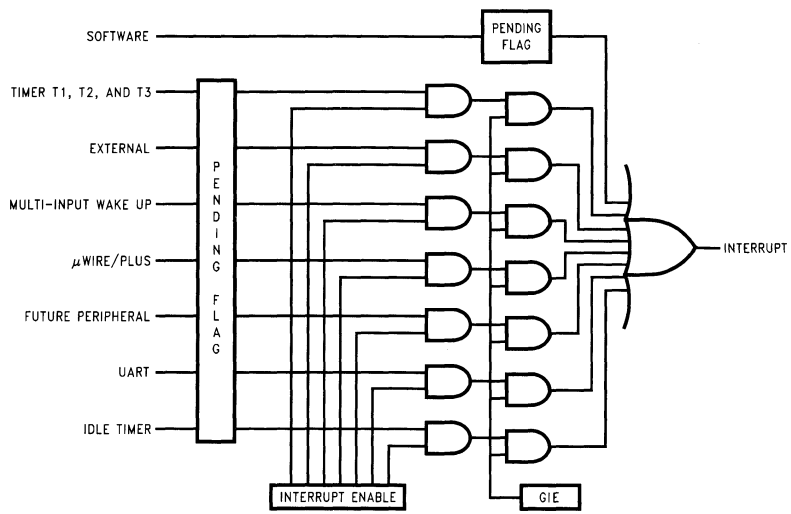
A Default VIS interrupt handler routine must be present. As a minimum, this handler should confirm that the GIE bit is cleared (this indicates that the interrupt sequence has been taken), take care of any required housekeeping, restore context and return. Some sort of Warm Restart procedure should be implemented. These events can occur without any error on the part of the system designer or programmer.

Note: There is always the possibility of an interrupt occurring during an instruction which is attempting to reset the GIE bit or any other interrupt enable bit. If this occurs when a single cycle instruction is being used to reset the interrupt enable bit, the interrupt enable bit will be reset but an interrupt may still occur. This is because interrupt processing is started at the same time as the interrupt bit is being reset. To avoid this scenario, the user should always use a two, three, or four cycle instruction to reset interrupt enable bits.

Figure 16 shows the Interrupt block diagram.

SOFTWARE TRAP

The Software Trap (ST) is a special kind of non-maskable interrupt which occurs when the INTR instruction (used to acknowledge interrupts) is fetched from ROM and placed inside the instruction register. This may happen when the PC is pointing beyond the available ROM address space or when the stack is over-popped.



TL/DD12860–18

FIGURE 16. Interrupt Block Diagram

Interrupts (Continued)

When an ST occurs, the user can re-initialize the stack pointer and do a recovery procedure (similar to reset, but not necessarily containing all of the same initialization procedures) before restarting.

The occurrence of an ST is latched into the ST pending bit. The GIE bit is not affected and the ST pending bit **(not accessible by the user)** is used to inhibit other interrupts and to direct the program to the ST service routine with the VIS instruction. The RPND instruction is used to clear the software interrupt pending bit. This pending bit is also cleared on reset.

The ST has the highest rank among all interrupts.

Nothing (except another ST) can interrupt an ST being serviced.

WATCHDOG

The devices contain a WATCHDOG and clock monitor. The WATCHDOG is designed to detect the user program getting stuck in infinite loops resulting in loss of program control or "runaway" programs. The Clock Monitor is used to detect the absence of a clock or a very slow clock below a specified rate on the CKI pin.

The WATCHDOG consists of two independent logic blocks: WD UPPER and WD LOWER. WD UPPER establishes the upper limit on the service window and WD LOWER defines the lower limit of the service window.

Servicing the WATCHDOG consists of writing a specific value to a WATCHDOG Service Register named WDSVR which is memory mapped in the RAM. This value is composed of three fields, consisting of a 2-bit Window Select, a 5-bit Key Data field, and the 1-bit Clock Monitor Select field. Table V shows the WDSVR register.

The lower limit of the service window is fixed at 2048 instruction cycles. Bits 7 and 6 of the WDSVR register allow the user to pick an upper limit of the service window.

Table VI shows the four possible combinations of lower and upper limits for the WATCHDOG service window. This flexibility in choosing the WATCHDOG service window prevents any undue burden on the user software.

Bits 5, 4, 3, 2 and 1 of the WDSVR register represent the 5-bit Key Data field. The key data is fixed at 01100. Bit 0 of the WDSVR Register is the Clock Monitor Select bit.

TABLE V. WATCHDOG Service Register (WDSVR)

Window Select		Key Data					Clock Monitor
X	X	0	1	1	0	0	Y
7	6	5	4	3	2	1	0

TABLE VI. WATCHDOG Service Window Select

WDSVR Bit 7	WDSVR Bit 6	Service Window (Lower-Upper Limits)
0	0	2k–8k t_C Cycles
0	1	2k–16k t_C Cycles
1	0	2k–32k t_C Cycles
1	1	2k–64k t_C Cycles

Clock Monitor

The Clock Monitor aboard the device can be selected or deselected under program control. The Clock Monitor is guaranteed not to reject the clock if the instruction cycle clock ($1/t_C$) is greater or equal to 10 kHz. This equates to a clock input rate on CKI of greater or equal to 100 kHz.

WATCHDOG Operation

The WATCHDOG and Clock Monitor are disabled during reset. The device comes out of reset with the WATCHDOG armed, the WATCHDOG Window Select bits (bits 6, 7 of the WDSVR Register) set, and the Clock Monitor bit (bit 0 of the WDSVR Register) enabled. Thus, a Clock Monitor error will occur after coming out of reset, if the instruction cycle clock frequency has not reached a minimum specified value, including the case where the oscillator fails to start.

The WDSVR register can be written to only once after reset and the key data (bits 5 through 1 of the WDSVR Register) must match to be a valid write. This write to the WDSVR register involves two irrevocable choices: (i) the selection of the WATCHDOG service window (ii) enabling or disabling of the Clock Monitor. Hence, the first write to WDSVR Register involves selecting or deselecting the Clock Monitor, select the WATCHDOG service window and match the WATCHDOG key data. Subsequent writes to the WDSVR register will compare the value being written by the user to the WATCHDOG service window value and the key data (bits 7 through 1) in the WDSVR Register. Table VII shows the sequence of events that can occur.

The user must service the WATCHDOG at least once before the upper limit of the service window expires. The WATCHDOG may not be serviced more than once in every lower limit of the service window. The user may service the WATCHDOG as many times as wished in the time period between the lower and upper limits of the service window. The first write to the WDSVR Register is also counted as a WATCHDOG service.

The WATCHDOG has an output pin associated with it. This is the WDOUT pin, on pin 1 of the port G. WDOUT is active low. The WDOUT pin is in the high impedance state in the inactive state. Upon triggering the WATCHDOG, the logic will pull the WDOUT (G1) pin low for an additional 16 t_C–32 t_C cycles after the signal level on WDOUT pin goes below the lower Schmitt trigger threshold. After this delay, the device will stop forcing the WDOUT output low.

TABLE VII. WATCHDOG Service Actions

Key Data	Window Data	Clock Monitor	Action
Match	Match	Match	Valid Service: Restart Service Window
Don't Care	Mismatch	Don't Care	Error: Generate WATCHDOG Output
Mismatch	Don't Care	Don't Care	Error: Generate WATCHDOG Output
Don't Care	Don't Care	Mismatch	Error: Generate WATCHDOG Output

WATCHDOG Operation (Continued)

The WATCHDOG service window will restart when the WDOUT pin goes high. It is recommended that the user tie the WDOUT pin back to V_{CC} through a resistor in order to pull WDOUT high.

A WATCHDOG service while the WDOUT signal is active will be ignored. The state of the WDOUT pin is not guaranteed on reset, but if it powers up low then the WATCHDOG will time out and WDOUT will enter high impedance state.

The Clock Monitor forces the G1 pin low upon detecting a clock frequency error. The Clock Monitor error will continue until the clock frequency has reached the minimum specified value, after which the G1 output will enter the high impedance TRI-STATE mode following 16 t_c–32 t_c clock cycles. The Clock Monitor generates a continual Clock Monitor error if the oscillator fails to start, or fails to reach the minimum specified frequency. The specification for the Clock Monitor is as follows:

$1/t_c > 10$ kHz—No clock rejection.

$1/t_c < 10$ Hz—Guaranteed clock rejection.

WATCHDOG AND CLOCK MONITOR SUMMARY

The following salient points regarding the WATCHDOG and CLOCK MONITOR should be noted:

- Both the WATCHDOG and CLOCK MONITOR detector circuits are inhibited during RESET.
- Following RESET, the WATCHDOG and CLOCK MONITOR are both enabled, with the WATCHDOG having the maximum service window selected.
- The WATCHDOG service window and CLOCK MONITOR enable/disable option can only be changed once, during the initial WATCHDOG service following RESET.
- The initial WATCHDOG service must match the key data value in the WATCHDOG Service register WDSVR in order to avoid a WATCHDOG error.
- Subsequent WATCHDOG services must match all three data fields in WDSVR in order to avoid WATCHDOG errors.
- The correct key data value cannot be read from the WATCHDOG Service register WDSVR. Any attempt to read this key data value of 01100 from WDSVR will read as key data value of all 0's.
- The WATCHDOG detector circuit is inhibited during both the HALT and IDLE modes.
- The CLOCK MONITOR detector circuit is active during both the HALT and IDLE modes. Consequently, the COP888 inadvertently entering the HALT mode will be detected as a CLOCK MONITOR error (provided that the CLOCK MONITOR enable option has been selected by the program).
- With the single-pin R/C oscillator mask option selected and the CLKDLY bit reset, the WATCHDOG service window will resume following HALT mode from where it left off before entering the HALT mode.
- With the crystal oscillator mask option selected, or with the single-pin R/C oscillator mask option selected and the CLKDLY bit set, the WATCHDOG service window will be set to its selected value from WDSVR following HALT. Consequently, the WATCHDOG should not be serviced for at least 2048 instruction cycles following HALT, but must be serviced within the selected window to avoid a WATCHDOG error.
- The IDLE timer T0 is not initialized with RESET.
- The user can sync in to the IDLE counter cycle with an IDLE counter (T0) interrupt or by monitoring the T0PND flag. The T0PND flag is set whenever the thirteenth bit of the IDLE counter toggles (every 4096 instruction cycles). The user is responsible for resetting the T0PND flag.
- A hardware WATCHDOG service occurs just as the device exits the IDLE mode. Consequently, the WATCHDOG should not be serviced for at least 2048 instruction cycles following IDLE, but must be serviced within the selected window to avoid a WATCHDOG error.
- Following RESET, the initial WATCHDOG service (where the service window and the CLOCK MONITOR enable/disable must be selected) may be programmed anywhere within the maximum service window (65,536 instruction cycles) initialized by RESET. Note that this initial WATCHDOG service may be programmed within the initial 2048 instruction cycles without causing a WATCHDOG error.

Detection of Illegal Conditions

The device can detect various illegal conditions resulting from coding errors, transient noise, power supply voltage drops, runaway programs, etc.

Reading of undefined ROM gets zeros. The opcode for software interrupt is zero. If the program fetches instructions from undefined ROM, this will force a software interrupt, thus signaling that an illegal condition has occurred.

The subroutine stack grows down for each call (jump to subroutine), interrupt, or PUSH, and grows up for each return or POP. The stack pointer is initialized to RAM location 06F Hex during reset. Consequently, if there are more returns than calls, the stack pointer will point to addresses 070 and 071 Hex (which are undefined RAM). Undefined RAM from addresses 070 to 07F (Segment 0), 140 to 17F (Segment 1), and all other segments (i.e., Segments 3 . . . etc.) is read as all 1's, which in turn will cause the program to return to address 7FFF Hex. This is an undefined ROM location and the instruction fetched (all 0's) from this location will generate a software interrupt signaling an illegal condition.

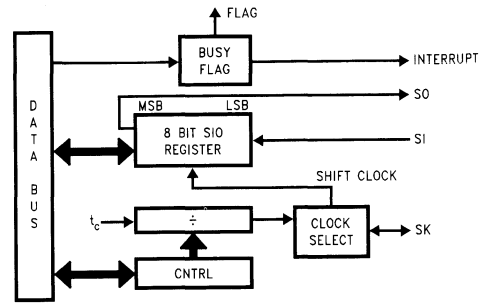
Thus, the chip can detect the following illegal conditions:

1. Executing from undefined ROM
2. Over "POP"ing the stack by having more returns than calls.

When the software interrupt occurs, the user can re-initialize the stack pointer and do a recovery procedure before restarting (this recovery program is probably similar to that following reset, but might not contain the same program initialization procedures). The recovery program should reset the software interrupt pending bit using the RPND instruction.

MICROWIRE/PLUS

MICROWIRE/PLUS is a serial synchronous communications interface. The MICROWIRE/PLUS capability enables the device to interface with any of National Semiconductor's MICROWIRE peripherals (i.e. A/D converters, display drivers, E^2PROMs etc.) and with other microcontrollers which support the MICROWIRE interface. It consists of an 8-bit serial shift register (SIO) with serial data input (SI), serial data output (SO) and serial shift clock (SK). *Figure 17* shows a block diagram of the MICROWIRE/PLUS logic.



TL/DD12860–19

FIGURE 17. MICROWIRE/PLUS Block Diagram

The shift clock can be selected from either an internal source or an external source. Operating the MICROWIRE/PLUS arrangement with the internal clock source is called the Master mode of operation. Similarly, operating the MICROWIRE/PLUS arrangement with an external shift clock is called the Slave mode of operation.

The CNTRL register is used to configure and control the MICROWIRE/PLUS mode. To use the MICROWIRE/PLUS, the MSEL bit in the CNTRL register is set to one. In the master mode, the SK clock rate is selected by the two bits, SL0 and SL1, in the CNTRL register. Table VIII details the different clock rates that may be selected.

TABLE VIII. MICROWIRE/PLUS Master Mode Clock Select

SL1	SL0	SK
0	0	$2 \times t_c$
0	1	$4 \times t_c$
1	x	$8 \times t_c$

Where t_c is the instruction cycle clock

MICROWIRE/PLUS (Continued)

MICROWIRE/PLUS OPERATION

Setting the BUSY bit in the PSW register causes the MICROWIRE/PLUS to start shifting the data. It gets reset when eight data bits have been shifted. The user may reset the BUSY bit by software to allow less than 8 bits to shift. If enabled, an interrupt is generated when eight data bits have been shifted. The device may enter the MICROWIRE/PLUS mode either as a Master or as a Slave. *Figure 18* shows how two devices, microcontrollers and several peripherals may be interconnected using the MICROWIRE/PLUS arrangements.

Warning

The SIO register should only be loaded when the SK clock is low. Loading the SIO register while the SK clock is high will result in undefined data in the SIO register. SK clock is normally low when not shifting.

Setting the BUSY flag when the input SK clock is high in the MICROWIRE/PLUS slave mode may cause the current SK clock for the SIO shift register to be narrow. For safety, the BUSY flag should only be set when the input SK clock is low.

MICROWIRE/PLUS Master Mode Operation

In the MICROWIRE/PLUS Master mode of operation the shift clock (SK) is generated internally by the device. The MICROWIRE Master always initiates all data exchanges. The MSEL bit in the CNTRL register must be set to enable the SO and SK functions onto the G Port. The SO and SK pins must also be selected as outputs by setting appropriate bits in the Port G configuration register. Table IX summarizes the bit settings required for Master mode of operation.

MICROWIRE/PLUS Slave Mode Operation

In the MICROWIRE/PLUS Slave mode of operation the SK clock is generated by an external source. Setting the MSEL bit in the CNTRL register enables the SO and SK functions onto the G Port. The SK pin must be selected as an input and the SO pin is selected as an output pin by setting and resetting the appropriate bit in the Port G configuration register. Table IX summarizes the settings required to enter the Slave mode of operation.

The user must set the BUSY flag immediately upon entering the Slave mode. This will ensure that all data bits sent by the Master will be shifted properly. After eight clock pulses the BUSY flag will be cleared and the sequence may be repeated.

Alternate SK Phase Operation

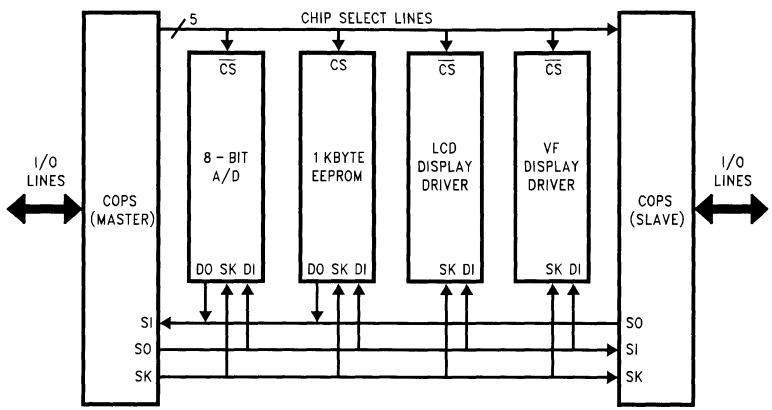
The device allows either the normal SK clock or an alternate phase SK clock to shift data in and out of the SIO register. In both the modes the SK is normally low. In the normal mode data is shifted in on the rising edge of the SK clock and the data is shifted out on the falling edge of the SK clock. The SIO register is shifted on each falling edge of the SK clock. In the alternate SK phase operation, data is shifted in on the falling edge of the SK clock and shifted out on the rising edge of the SK clock.

A control flag, SKSEL, allows either the normal SK clock or the alternate SK clock to be selected. Resetting SKSEL causes the MICROWIRE/PLUS logic to be clocked from the normal SK signal. Setting the SKSEL flag selects the alternate SK clock. The SKSEL is mapped into the G6 configuration bit. The SKSEL flag will power up in the reset condition, selecting the normal SK signal.

TABLE IX. MICROWIRE/PLUS Mode Selection

G4 (SO) Config. Bit	G5 (SK) Config. Bit	G4 Fun.	G5 Fun.	Operation
1	1	SO	Int. SK	MICROWIRE/PLUS Master
0	1	TRI-STATE	Int. SK	MICROWIRE/PLUS Master
1	0	SO	Ext. SK	MICROWIRE/PLUS Slave
0	0	TRI-STATE	Ext. SK	MICROWIRE/PLUS Slave

Note: This table assumes that the control flag MSEL is set.



TL/DD12860-20

FIGURE 18. MICROWIRE/PLUS Application

Memory Map

All RAM, ports and registers (except A and PC) are mapped into data memory address space.

Address S/ADD REG	Contents
0000 to 006F	On-Chip RAM bytes (112 bytes)
0070 to 007F	Unused RAM Address Space (Reads As All Ones)
xx80 to xxAF	Unused RAM Address Space (Reads Undefined Data)
xxB0	Timer T3 Lower Byte
xxB1	Timer T3 Upper Byte
xxB2	Timer T3 Autoload Register T3RA Lower Byte
xxB3	Timer T3 Autoload Register T3RA Upper Byte
xxB4	Timer T3 Autoload Register T3RB Lower Byte
xxB5	Timer T3 Autoload Register T3RB Upper Byte
xxB6	Timer T3 Control Register
xxB7	Comparator Select Register (CMPSL)
xxB8	UART Transmit Buffer (TBUF)
xxB9	UART Receive Buffer (RBUF)
xxBA	UART Control and Status Register (ENU)
xxBB	UART Receive Control and Status Register (ENUR)
xxBC	UART Interrupt and Clock Source Register (ENUI)
xxBD	UART Baud Register (BAUD)
xxBE	UART Prescale Select Register (PSR)
xxBF	Reserved for UART
xxC0	Timer T2 Lower Byte
xxC1	Timer T2 Upper Byte
xxC2	Timer T2 Autoload Register T2RA Lower Byte
xxC3	Timer T2 Autoload Register T2RA Upper Byte
xxC4	Timer T2 Autoload Register T2RB Lower Byte
xxC5	Timer T2 Autoload Register T2RB Upper Byte
xxC6	Timer T2 Control Register
xxC7	WATCHDOG Service Register (Reg:WDSVR)
xxC8	MIWU Edge Select Register (Reg:WKEDG)
xxC9	MIWU Enable Register (Reg:WKEN)
xxCA	MIWU Pending Register (Reg:WKPND)
xxCB	Reserved
xxCC	Reserved
xxCD to xxCF	Reserved

Address S/ADD REG	Contents
xxD0	Port L Data Register
xxD1	Port L Configuration Register
xxD2	Port L Input Pins (Read Only)
xxD3	Reserved for Port L
xxD4	Port G Data Register
xxD5	Port G Configuration Register
xxD6	Port G Input Pins (Read Only)
xxD7	Port I Input Pins (Read Only)
xxD8	Port C Data Register
xxD9	Port C Configuration Register
xxDA	Port C Input Pins (Read Only)
xxDB	Reserved for Port C
xxDC	Port D
xxDD to DF	Reserved for Port D
xxE0 to xxE5	Reserved for EE Control Registers
xxE6	Timer T1 Autoload Register T1RB Lower Byte
xxE7	Timer T1 Autoload Register T1RB Upper Byte
xxE8	ICNTRL Register
xxE9	MICROWIRE/PLUS Shift Register
xxEA	Timer T1 Lower Byte
xxEB	Timer T1 Upper Byte
xxEC	Timer T1 Autoload Register T1RA Lower Byte
xxED	Timer T1 Autoload Register T1RA Upper Byte
xxEE	CNTRL Control Register
xxEF	PSW Register
xxF0 to FB	On-Chip RAM Mapped as Registers
xxFC	X Register
xxFD	SP Register
xxFE	B Register
xxFF	S Register
0100–017F	On-Chip 128 RAM Bytes
0200–027F	On-Chip 128 RAM Bytes
0200–037F	On-Chip 128 RAM Bytes

Note: Reading memory locations 0070H–007FH (Segment 0) will return all ones. Reading unused memory locations 0080H–00AFH (Segment 0) will return undefined data. Reading memory locations from other Segments (i.e., Segment 4, Segment 5, ... etc.) will return all ones.

Addressing Modes

There are ten addressing modes, six for operand addressing and four for transfer of control.

OPERAND ADDRESSING MODES

Register Indirect

This is the "normal" addressing mode. The operand is the data memory addressed by the B pointer or X pointer.

Register Indirect (with auto post increment or decrement of pointer)

This addressing mode is used with the LD and X instructions. The operand is the data memory addressed by the B pointer or X pointer. This is a register indirect mode that automatically post increments or decrements the B or X register after executing the instruction.

Direct

The instruction contains an 8-bit address field that directly points to the data memory for the operand.

Immediate

The instruction contains an 8-bit immediate field as the operand.

Short Immediate

This addressing mode is used with the Load B Immediate instruction. The instruction contains a 4-bit immediate field as the operand.

Indirect

This addressing mode is used with the LAID instruction. The contents of the accumulator are used as a partial address (lower 8 bits of PC) for accessing a data operand from the program memory.

TRANSFER OF CONTROL ADDRESSING MODES

Relative

This mode is used for the JP instruction, with the instruction field being added to the program counter to get the new program location. JP has a range from −31 to +32 to allow a 1-byte relative jump (JP + 1 is implemented by a NOP instruction). There are no "pages" when using JP, since all 15 bits of PC are used.

Absolute

This mode is used with the JMP and JSR instructions, with the instruction field of 12 bits replacing the lower 12 bits of the program counter (PC). This allows jumping to any location in the current 4k program memory segment.

Absolute Long

This mode is used with the JMPL and JSRL instructions, with the instruction field of 15 bits replacing the entire 15 bits of the program counter (PC). This allows jumping to any location up to 32k in the program memory space.

Indirect

This mode is used with the JID instruction. The contents of the accumulator are used as a partial address (lower 8 bits of PC) for accessing a location in the program memory. The contents of this program memory location serve as a partial address (lower 8 bits of PC) for the jump to the next instruction.

Note: The VIS is a special case of the Indirect Transfer of Control addressing mode, where the double byte vector associated with the interrupt is transferred from adjacent addresses in the program memory into the program counter (PC) in order to jump to the associated interrupt service routine.

Instruction Set

Register and Symbol Definition

Registers	
A	8-Bit Accumulator Register
B	8-Bit Address Register
X	8-Bit Address Register
SP	8-Bit Stack Pointer Register
PC	15-Bit Program Counter Register
PU	Upper 7 Bits of PC
PL	Lower 8 Bits of PC
C	1 Bit of PSW Register for Carry
HC	1 Bit of PSW Register for Half Carry
GIE	1 Bit of PSW Register for Global Interrupt Enable
VU	Interrupt Vector Upper Byte
VL	Interrupt Vector Lower Byte

Symbols	
[B]	Memory Indirectly Addressed by B Register
[X]	Memory Indirectly Addressed by X Register
MD	Direct Addressed Memory
Mem	Direct Addressed Memory or [B]
Meml	Direct Addressed Memory or [B] or Immediate Data
Imm	8-Bit Immediate Data
Reg	Register Memory: Addresses F0 to FF (Includes B, X and SP)
Bit	Bit Number (0 to 7)
←	Loaded with
↔	Exchanged with

Instruction Set (Continued)

INSTRUCTION SET

ADD	A,Meml	ADD	A ← A + Meml
ADC	A,Meml	ADD with Carry	A ← A + Meml + C, C ← Carry, HC ← Half Carry
SUBC	A,Meml	Subtract with Carry	A ← A − Meml + C, C ← Carry, HC ← Half Carry
AND	A,Meml	Logical AND	A ← A and $\overline{\text{Meml}}$
ANDSZ	A,Imm	Logical AND Immed., Skip if Zero	Skip next if (A and Imm) = 0
OR	A,Meml	Logical OR	A ← A or Meml
XOR	A,Meml	Logical EXclusive OR	A ← A xor Meml
IFEQ	MD,Imm	IF EQual	Compare MD and Imm, Do next if MD = Imm
IFEQ	A,Meml	IF EQual	Compare A and Meml, Do next if A = Meml
IFNE	A,Meml	IF Not Equal	Compare A and Meml, Do next if A ≠ Meml
IFGT	A,Meml	IF Greater Than	Compare A and Meml, Do next if A > Meml
IFBNE	#	If B Not Equal	Do next if lower 4 bits of B ≠ Imm
DRSZ	Reg	Decrement Reg., Skip if Zero	Reg ← Reg − 1, Skip if Reg = 0
SBIT	#,Mem	Set BIT	1 to bit, Mem (bit = 0 to 7 immediate)
RBIT	#,Mem	Reset BIT	0 to bit, Mem
IFBIT	#,Mem	IF BIT	If bit in A or Mem is true do next instruction
RPND		Reset PeNDing Flag	Reset Software Interrupt Pending Flag
X	A,Mem	EXchange A with Memory	A ⟷ Mem
X	A,[X]	EXchange A with Memory [X]	A ⟷ [X]
LD	A,Meml	LoaD A with Memory	A ← Meml
LD	A,[X]	LoaD A with Memory [X]	A ← [X]
LD	B,Imm	LoaD B with Immed.	B ← Imm
LD	Mem,Imm	LoaD Memory Immed.	Mem ← Imm
LD	Reg,Imm	LoaD Register Memory Immed.	Reg ← Imm
X	A, [B ±]	EXchange A with Memory [B]	A ⟷ [B], (B ← B ±1)
X	A, [X ±]	EXchange A with Memory [X]	A ⟷ [X], (X ← ±1)
LD	A, [B±]	LoaD A with Memory [B]	A ← [B], (B ← B ±1)
LD	A, [X±]	LoaD A with Memory [X]	A ← [X], (X ← X±1)
LD	[B±],Imm	LoaD Memory [B] Immed.	[B] ← Imm, (B ← B±1)
CLR	A	CLeaR A	A ← 0
INC	A	INCrement A	A ← A + 1
DEC	A	DECrementA	A ← A − 1
LAID		Load A InDirect from ROM	A ← ROM (PU,A)
DCOR	A	Decimal CORrect A	A ← BCD correction of A (follows ADC, SUBC)
RRC	A	Rotate A Right thru C	C → A7 → . . . → A0 → C
RLC	A	Rotate A Left thru C	C ← A7 ← . . . ← A0 ← C
SWAP	A	SWAP nibbles of A	A7 . . . A4 ⟷ A3 . . . A0
SC		Set C	C ← 1, HC ← 1
RC		Reset C	C ← 0, HC ← 0
IFC		IF C	IF C is true, do next instruction
IFNC		IF Not C	If C is not true, do next instruction
POP	A	POP the stack into A	SP ← SP + 1, A ← [SP]
PUSH	A	PUSH A onto the stack	[SP] ← A, SP ← SP − 1
VIS		Vector to Interrupt Service Routine	PU ← [VU], PL ← [VL]
JMPL	Addr.	Jump absolute Long	PC ← ii (ii = 15 bits, 0k to 32k)
JMP	Addr.	Jump absolute	PC9 . . . 0 ← i (i = 12 bits)
JP	Disp.	Jump relative short	PC ← PC + r (r is −31 to +32, except 1)
JSRL	Addr	Jump SubRoutine Long	[SP] ← PL, [SP−1] ← PU,SP−2, PC ← ii
JSR	Add.	Jump SubRoutine	[SP] ← PL, [SP−1] ← PU,SP−2, PC9 . . . 0 ← i
JID		Jump InDirect	PL ← ROM (PU,A)
RET		RETurn from subroutine	SP + 2, PL ← [SP], PU ← [SP−1]
RETSK		RETurn and SKip	SP + 2, PL ← [SP],PU ← [SP−1]
RETI		RETurn from Interrupt	SP + 2, PL ← [SP],PU ← [SP−1],GIE ← 1
INTR		Generate an Interrupt	[SP] ← PL, [SP−1] ← PU, SP−2, PC ← 0FF
NOP		No OPeration	PC ← PC + 1

Instruction Execution Time

Most instructions are single byte (with immediate addressing mode instructions taking two bytes).

Most single byte instructions take one cycle time to execute.

Skipped instructions require x number of cycles to be skipped, where x equals the number of bytes in the skipped instruction opcode.

See the BYTES and CYCLES per INSTRUCTION table for details.

Bytes and Cycles per Instruction

The following table shows the number of bytes and cycles for each instruction in the format of byte/cycle.

Logic and Arithmetic Instructions

	[B]	Direct	Immed.
ADD	1/1	3/4	2/2
ADC	1/1	3/4	2/2
SUBC	1/1	3/4	2/2
AND	1/1	3/4	2/2
OR	1/1	3/4	2/2
XOR	1/1	3/4	2/2
IFEQ	1/1	3/4	2/2
IFGT	1/1	3/4	2/2
IFBNE	1/1		
DRSZ		1/3	
SBIT	1/1	3/4	
RBIT	1/1	3/4	
IFBIT	1/1	3/4	

RPND	1/1

Instructions Using A and C

CLRA	1/1
INCA	1/1
DECA	1/1
LAID	1/3
DCORA	1/1
RRCA	1/1
RLCA	1/1
SWAPA	1/1
SC	1/1
RC	1/1
IFC	1/1
IFNC	1/1
PUSHA	1/3
POPA	1/3
ANDSZ	2/2

Transfer of Control Instructions

JMPL	3/4
JMP	2/3
JP	1/3
JSRL	3/5
JSR	2/5
JID	1/3
VIS	1/5
RET	1/5
RETSK	1/5
RETI	1/5
INTR	1/7
NOP	1/1

Memory Transfer Instructions

	Register Indirect		Direct	Immed.	Register Indirect Auto Incr. and Decr.		
	[B]	[X]			[B+, B−]	[X+, X−]	
X A,*	1/1	1/3	2/3		1/2	1/3	
LD A,*	1/1	1/3	2/3	2/2	1/2	1/3	
LD B, Imm				1/1			(IF B < 16)
LD B, Imm				2/3			(IF B > 15)
LD Mem, Imm	2/2	2/2	3/3		2/2		
LD Reg, Imm			2/3				
IFEQ MD, Imm			3/3				

* = > Memory location addressed by B or X or directly.

Opcode Table

UPPER NIBBLE																
F	**E**	**D**	**C**	**B**	**A**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**	
JP −15	JP −31	LD 0F0, #i	DRSZ 0F0	RRCA	RC	ADC A, #i	ADC A,[B]	IFBIT 0,[B]	ANDSZ A, #i	LD B,#0F	IFBNE 0	JSR x000–x0FF	JMP x000–x0FF	JP + 17	INTR	0
JP −14	JP −30	LD 0F1, #i	DRSZ 0F1	*	SC	SUBC A, #i	SUB A,[B]	IFBIT 1,[B]	*	LD B,#0E	IFBNE 1	JSR x100–x1FF	JMP x100–x1FF	JP + 18	JP + 2	1
JP −13	JP −29	LD 0F2, #i	DRSZ 0F2	X A, [X+]	X A, [B+]	IFEQ A, #i	IFEQ A,[B]	IFBIT 2,[B]	*	LD B,#0D	IFBNE 2	JSR x200–x2FF	JMP x200–x2FF	JP + 19	JP + 3	2
JP −12	JP −28	LD 0F3, #i	DRSZ 0F3	X A, [X−]	X A, [B−]	IFGT A, #i	IFGT A,[B]	IFBIT 3,[B]	*	LD B,#0C	IFBNE 3	JSR x300–x3FF	JMP x300–x3FF	JP + 20	JP + 4	3
JP −11	JP −27	LD 0F4, #i	DRSZ 0F4	VIS	LAID	ADD A, #i	ADD A,[B]	IFBIT 4,[B]	CLRA	LD B,#0B	IFBNE 4	JSR x400–x4FF	JMP x400–x4FF	JP + 21	JP + 5	4
JP −10	JP −26	LD 0F5, #i	DRSZ 0F5	RPND	JID	AND A, #i	AND A,[B]	IFBIT 5,[B]	SWAPA	LD B,#0A	IFBNE 5	JSR x500–x5FF	JMP x500–x5FF	JP + 22	JP + 6	5
JP −9	JP −25	LD 0F6, #i	DRSZ 0F6	X A,[X]	X A,[B]	XOR A, #i	XOR A,[B]	IFBIT 6,[B]	DCORA	LD B,#09	IFBNE 6	JSR x600–x6FF	JMP x600–x6FF	JP + 23	JP + 7	6
JP −8	JP −24	LD 0F7, #i	DRSZ 0F7	*	*	OR A,#i	OR A,[B]	IFBIT 7,[B]	PUSHA	LD B,#08	IFBNE 7	JSR x700–x7FF	JMP x700–x7FF	JP + 24	JP + 8	7
JP −7	JP −23	LD 0F8, #i	DRSZ 0F8	NOP	RLCA	LD A,#i	IFC	SBIT 0,[B]	RBIT 0,[B]	LD B,#07	IFBNE 8	JSR x800–x8FF	JMP x800–x8FF	JP + 25	JP + 9	8
JP −6	JP −22	LD 0F9, #i	DRSZ 0F9	IFNE A,[B]	IFEQ Md,#i	IFNE A,#i	IFNC	SBIT 1,[B]	RBIT 1,[B]	LD B,#06	IFBNE 9	JSR x900–x9FF	JMP x900–x9FF	JP + 26	JP + 10	9
JP −5	JP −21	LD 0FA, #i	DRSZ 0FA	LD A, [X+]	LD A, [B+]	LD [B+], #i	INCA	SBIT 2,[B]	RBIT 2,[B]	LD B,#05	IFBNE 0A	JSR xA00–xAFF	JMP xA00–xAFF	JP + 27	JP + 11	A
JP −4	JP −20	LD 0FB, #i	DRSZ 0FB	LD A, [X−]	LD A, [B−]	LD [B−], #i	DECA	SBIT 3,[B]	RBIT 3,[B]	LD B,#04	IFBNE 0B	JSR xB00–xBFF	JMP xB00–xBFF	JP + 28	JP + 12	B
JP −3	JP −19	LD 0FC, #i	DRSZ 0FC	LD Md,#i	JMPL	X A,Md	POPA	SBIT 4,[B]	RBIT 4,[B]	LD B,#03	IFBNE 0C	JSR xC00–xCFF	JMP xC00–xCFF	JP + 29	JP + 13	C
JP −2	JP −18	LD 0FD, #i	DRSZ 0FD	DIR	JSRL	LD A,Md	RETSK	SBIT 5,[B]	RBIT 5,[B]	LD B,#02	IFBNE 0D	JSR xD00–xDFF	JMP xD00–xDFF	JP + 30	JP + 14	D
JP −1	JP −17	LD 0FE, #i	DRSZ 0FE	LD A,[X]	LD A,[B]	LD [B],#i	RET	SBIT 6,[B]	RBIT 6,[B]	LD B,#01	IFBNE 0E	JSR xE00–xEFF	JMP xE00–xEFF	JP + 31	JP + 15	E
JP −0	JP −16	LD 0FF, #i	DRSZ 0FF	*	*	LD B,#i	RETI	SBIT 7,[B]	RBIT 7,[B]	LD B,#00	IFBNE 0F	JSR xF00–xFFF	JMP xF00–xFFF	JP + 32	JP + 16	F

LOWER NIBBLE

where,

i is the immediate data

Md is a directly addressed memory location

* is an unused opcode

Note: The opcode 60 Hex is also the opcode for IFBIT #i,A

Development Support

SUMMARY

- iceMASTER™: IM-COP8/400—Full feature in-circuit emulation for all COP8 products. A full set of COP8 Basic and Feature Family device and package specific probes are available.
- COP8 Debug Module: Moderate cost in-circuit emulation and development programming unit.
- COP8 Evaluation and Programming Unit: EPU-COP888GG—low cost in-circuit simulation and development programming unit.
- Assembler: COP8-DEV-IBMA. A DOS installable cross development Assembler, Linker, Librarian and Utility Software Development Tool Kit.
- C Compiler: COP8C. A DOS installable cross development Software Tool Kit.
- OTP/EPROM Programmer Support: Covering needs from engineering prototype, pilot production to full production environments.

iceMASTER (IM) IN-CIRCUIT EMULATION

The iceMASTER IM-COP8/400 is a full feature, PC based, in-circuit emulation tool developed and marketed by MetaLink Corporation to support the whole COP8 family of products. National is a resale vendor for these products.

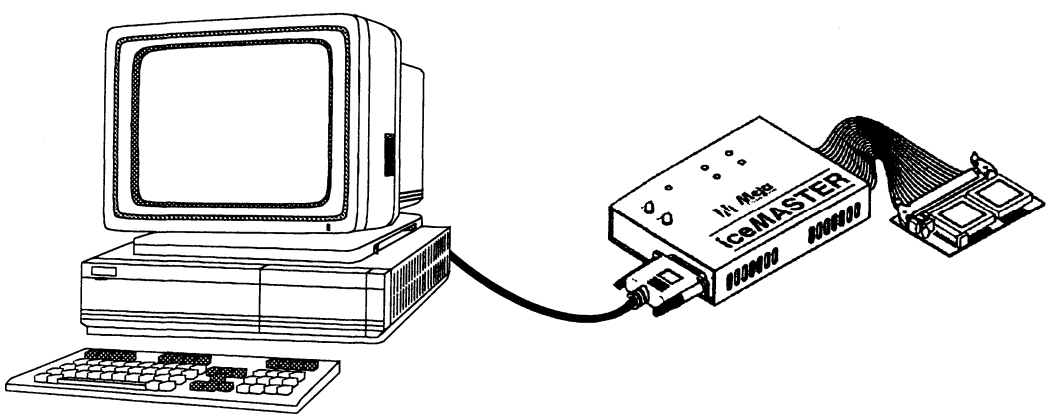
See *Figure 19* for configuration.

The iceMASTER IM-COP8/400 with its device specific COP8 Probe provides a rich feature set for developing, testing and maintaining product:

- Real-time in-circuit emulation; full 2.4V–5.5V operation range, full DC-10 MHz clock. Chip options are programmable or jumper selectable.
- Direct connection to application board by package compatible socket or surface mount assembly.
- Full 32 kbytes of loadable programming space that overlays (replaces) the on-chip ROM or EPROM. On-chip RAM and I/O blocks are used directly or recreated on the probe as necessary.
- Full 4k frame synchronous trace memory. Address, instruction, and 8 unspecified, circuit connectable trace lines. Display can be HLL source (e.g., C source), assembly or mixed.
- A full 64k hardware configurable break, trace on, trace off control, and pass count increment events.
- Tool set integrated interactive symbolic debugger—supports both assembler (COFF) and C Compiler (.COD) linked object formats.
- Real time performance profiling analysis; selectable bucket definition.
- Watch windows, content updated automatically at each execution break.
- Instruction by instruction memory/register changes displayed on source window when in single step operation.
- Single base unit and debugger software reconfigurable to support the entire COP8 family; only the probe personality needs to change. Debugger software is processor customized, and reconfigured from a master model file.
- Processor specific symbolic display of registers and bit level assignments, configured from master model file.
- Halt/Idle mode notification.
- On-line HELP customized to specific processor using master model file.
- Includes a copy of COP8-DEV-IBMA assembler and linker SDK.

IM Order Information

Base Unit	
IM-COP8/400-1	iceMASTER Base Unit, 110V Power Supply
IM-COP8/400-2	iceMASTER Base Unit, 220V Power Supply
iceMASTER Probe	
MHW-888GG40DWPC	40 DIP
MHW-888GG44PWPC	44 PLCC



TL/DD/12860–21

FIGURE 19. COP8 iceMASTER Environment

Development Support (Continued)

iceMASTER DEBUG MODULE (DM)

The iceMASTER Debug Module is a PC based, combination in-circuit emulation tool and COP8 based OTP/EPROM programming tool developed and marketed by MetaLink Corporation to support the whole COP8 family of products. National is a resale vendor for these products.

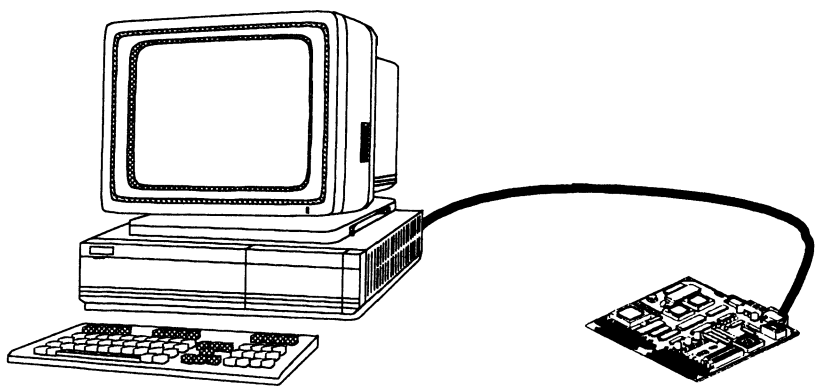
See *Figure 20* for configuration.

The iceMASTER Debug Module is a moderate cost development tool. It has the capability of in-circuit emulation for a specific COP8 microcontroller and in addition serves as a programming tool for COP8 OTP and EPROM product families. Summary of features is as follows:

- Real-time in-circuit emulation; full operating voltage range operation, full DC-10 MHz clock.
- All processor I/O pins can be cabled to an application development board with package compatible cable to socket and surface mount assembly.
- Full 32 kbytes of loadable programming space that overlays (replaces) the on-chip ROM or EPROM. On-chip RAM and I/O blocks are used directly or recreated as necessary.
- 100 frames of synchronous trace memory. The display can be HLL source (C source), assembly or mixed. The most recent history prior to a break is available in the trace memory.
- Configured break points; uses INTR instruction which is modestly intrusive.
- Software—only supported features are selectable.
- Tool set integrated interactive symbolic debugger—supports both assembler (COFF) and C Compiler (.COD) SDK linked object formats.
- Instruction by instruction memory/register changes displayed when in single step operation.
- Debugger software is processor customized, and reconfigured from a master model file.
- Processor specific symbolic display of registers and bit level assignments, configured from master model file.
- Halt/Idle mode notification.
- Programming menu supports full product line of programmable OTP and EPROM COP8 products. Program data is taken directly from the overlay RAM.
- Programming of 44 PLCC and 68 PLCC parts requires external programming adapters.
- Includes wallmount power supply.
- On-board V_{PP} generator from 5V input or connection to external supply supported. Rquires V_{PP} level adjustment per the family programming specification (correct level is provided on an on-screen pop-down display).
- On-line HELP customized to specific processor using master model file.
- Includes a copy of COP8-DEV-IBMA assembler and linker SDK.

DM Order Information

Debug Module Unit	
COP8-DM/888GG	
Cable Adapters	
DM-COP8/40D	40 DIP
DM-COP8/44P	44 PLCC



TL/DD/12860–22

FIGURE 20. COP8-DM Environment

Development Support (Continued)

iceMASTER EVALUATION PROGRAMMING UNIT (EPU)

The iceMASTER EPU-COP888GG is a PC based, in-circuit simulation tool to support the feature family COP8 products.

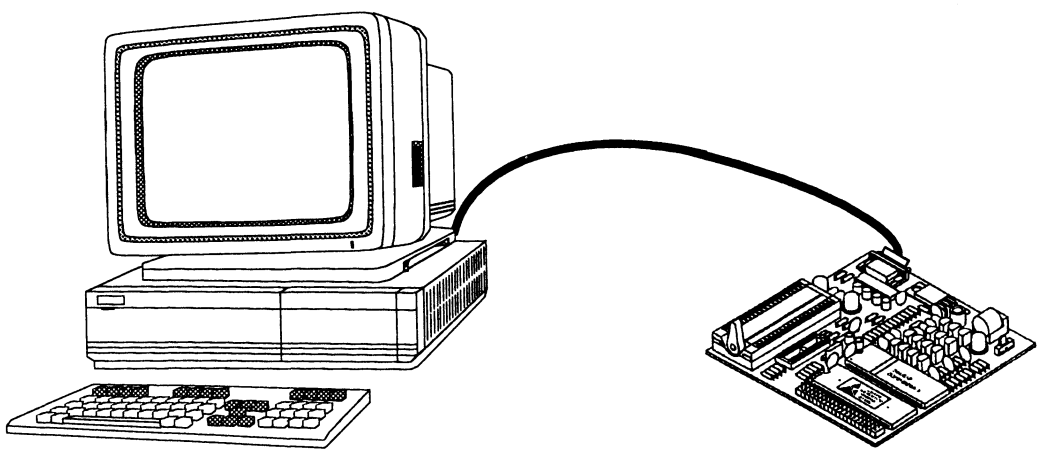
See *Figure 21* for configuration.

The simulation capability is a very low cost means of evaluating the general COP8 architecture. In addition, the EPU has programming capability, with added adapters, for programming the whole COP8 product family of OTP and EPROM products. The product includes the following features:

- Non-real-time in-circuit simulation. Program overlay memory is PC resident; instructions are downloaded over RS-232 as executed. Approximate performance is 20 kHz.
- Includes a 40-pin DIP cable adapter. Other target packages are not supported. All processor I/O pins are cabled to the application development environment.
- Full 32 kbytes of loadable programming space that overlays (replaces) the on-chip ROM or EPROM. On-chip RAM and I/O blocks are used directly or recreated as necessary.
- On-chip timer and WATCHDOG execution are not well synchronized to the instruction simulation.
- 100 frames of synchronous trace memory. The display can be HLL source (e.g., C source), assembly or mixed. The most recent history prior to a break is available in the trace memory.
- Up to eight software configured break points; uses INTR instruction which is modestly intrusive.
- Common look-feel debugger software across all MetaLink products—only supported features are selectable.
- Tool set integrated interactive symbolic debugger—supports both assembler (COFF) and C Compiler (.COD) SDK linked object formats.
- Instruction by instruction memory/register changes displayed when in single step operation.
- Processor specific symbolic display of registers and bit level assignments, configured from master model file.
- Halt/Idle mode notification. Restart requires special handling.
- Programming menu supports full product line of programmable OTP and EPROM COP8 products. Only a 40 ZIF socket is available on the EPU unit. Adapters are available for other part package configurations.
- Integral wall mount power supply provides 5V and develops the required V_{PP} to program parts.
- Includes a copy of COP8-DEV-IBMA assembler, linker SDK.

EPU Order Information

Evaluation Programming Unit	
EPU-COP888GG	Evaluation Programming Unit with debugger and programmer control software with 40 ZIF programming socket.
General Programming Adapters	
COP8-PGMA-DS44P	28 and 20 DIP and SOIC plus 44 PLCC adapter



TL/DD/12860–23

FIGURE 21. EPU-COP8 Tool Environment

Development Support (Continued)

COP8 ASSEMBLER/LINKER SOFTWARE DEVELOPMENT TOOL KIT

National Semiconductor offers a relocateable COP8 macro cross assembler, linker, librarian and utility software development tool kit. Features are summarized as follows:

- Basic and Feature Family instruction set by "device" type.
- Nested macro capability.
- Extensive set of assembler directives.
- Supported on PC/DOS platform.
- Generates National standard COFF output files.
- Integrated Linker and Librarian.
- Integrated utilities to generate ROM code file outputs.
- DUMPCOFF utility.

This product is integrated as a part of MetaLink tools as a development kit, fully supported by the MetaLink debugger. It may be ordered separately or it is bundled with the MetaLink products at no additional cost.

Order Information

Assembler SDK	
COP8-DEV-IBMA	Assembler SDK on installable 3.5" PC/DOS Floppy Disk Drive format. Periodic upgrades and most recent version is available on National's BBS and Internet.

COP8 C COMPILER

A C Compiler is developed and marketed by Byte Craft Limited. The COP8C compiler is a fully integrated development tool specifically designed to support the compact embedded configuration of the COP8 family of products.

Features are summarized as follows:

- ANSI C with some restrictions and extensions that optimize development for the COP8 embedded application.
- BITS data type extension. Register declaration #pragma with direct bit level definitions.
- C language support for interrupt routines.
- Expert system, rule based code generation and optimization.
- Performs consistency checks against the architectural definitions of the target COP8 device.
- Generates program memory code.
- Supports linking of compiled object or COP8 assembled object formats.
- Global optimization of linked code.
- Symbolic debug load format fully sourced level supported by the MetaLink debugger.

INDUSTRY WIDE OTP/EPROM PROGRAMMING SUPPORT

Programming support, in addition to the MetaLink development tools, is provided by a full range of independent approved vendors to meet the needs from the engineering laboratory to full production.

Approved List

Manufacturer	North America	Europe	Asia
BP Microsystems	(800) 225-2102 (713) 688-4600 Fax: (713) 688-0920	+49-8152-4183 +49-8856-932616	+852-234-16611 +852-2710-8121
Data I/O	(800) 426-1045 (206) 881-6444 Fax: (206) 882-1043	+44-0734-440011	Call North America
HI-LO	(510) 623-8860	Call Asia	+886-2-764-0215 Fax: +886-2-756-6403
ICE Technology	(800) 624-8949 (919) 430-7915	+44-1226-767404 Fax: 0-1226-370-434	
MetaLink	(800) 638-2423 (602) 926-0797 Fax: (602) 693-0681	+49-80 9156 96-0 Fax: +49-80 9123 86	+852-737-1800
Systems General	(408) 263-6667	+41-1-9450300	+886-2-9173005 Fax: +886-2-911-1283
Needhams	(916) 924-8037 Fax: (916) 924-8065		

Development Support (Continued)

AVAILABLE LITERATURE

For more information, please see the COP8 Basic Family User's Manual, Literature Number 620895, COP8 Feature Family User's Manual, Literature Number 620897 and National's Family of 8-bit Microcontrollers COP8 Selection Guide, Literature Number 630009.

DIAL-A-HELPER SERVICE

Dial-A-Helper is a service provided by the Microcontroller Applications group. The Dial-A-Helper is an Electronic Information System that may be accessed as a Bulletin Board System (BBS) via data modem, as an FTP site on the Internet via standard FTP client application or as an FTP site on the Internet using a standard Internet browser such as Netscape or Mosaic.

The Dial-A-Helper system provides access to an automated information storage and retrieval system . The system capabilities include a MESSAGE SECTION (electronic mail, when accessed as a BBS) for communications to and from the Microcontroller Applications Group and a FILE SECTION which consists of several file areas where valuable application software and utilities could be found.

DIAL-A-HELPER BBS via Standard Modem

Modem:	CANADA/U.S.:	(800) NSC-MICRO (800) 672-6427	
	EUROPE:	(+49) 0-8141-351332	
	Baud:	14.4k	
	Set-Up:	Length:	8-Bit
		Parity:	None
		Stop Bit:	1
	Operation:	24 Hours, 7 Days	

DIAL-A-HELPER via FTP

ftp nscmicro.nsc.com
user: anonymous
password: username@yourhost.site.domain

DIAL-A-HELPER via WorldWide Web Browser

ftp://nscmicro.nsc.com

National Semiconductor on the WorldWide Web

See us on the WorldWide Web at: http://www.natsemi.com

CUSTOMER RESPONSE CENTER

Complete product information and technical support is available from National's customer response centers.

CANADA/U.S.:	Tel:	(800)272-9959
	email:	support@tevm2.nsc.com
EUROPE:	email:	europe.support@nsc.com
	Deutsch Tel:	+49 (0) 180-530 85 85
	English Tel:	+49 (0) 180-532 78 32
	Français Tel:	+49 (0) 180-532 93 58
	Italiano Tel:	+49 (0) 180-534 16 80
JAPAN:	Tel:	+81-043-299-2309
S.E. ASIA:	Beijing Tel:	(+86) 10-6856-8601
	Shanghai Tel:	(+86) 21-6415-4092
	Hong Kong Tel:	(+852) 2737-1600
	Korea Tel:	(+82) 2-3771-6909
	Malaysia Tel:	(+60-4) 644-9061
	Singapore Tel:	(+65) 255-2226
	Taiwan Tel:	+886-2-521-3288
AUSTRALIA:	Tel:	(+61) 3-9558-9999
INDIA:	Tel:	(+91) 80-559-9467

National Semiconductor

PRELIMINARY

COP87L84RG
8-Bit One-Time Programmable (OTP) Microcontroller with 32 Kbytes of Program Memory

General Description

The COP87L84RG is a member of the COP8™ OTP microcontroller family. It is pin and software compatible to the mask ROM COP888EG product family.

(Continued)

Key Features

- Full duplex UART
- Three 16-bit timers, each with two 16-bit registers supporting:
  - — Processor independent PWM mode
  - — External event counter mode
  - — Input capture mode
- 32 kbytes on-board EPROM with security feature

  Note: Mask ROMed devices with equivalent on-chip features and program memory sizes of 4k and 8k are available (see Table I).
- 512 bytes on-board RAM

Additional Peripheral Features

- Idle timer
- Multi-Input Wake Up (MIWU) with optional interrupts (8)
- One analog comparator
- WATCHDOG™ and clock monitor logic
- MICROWIRE/PLUS™ serial I/O

I/O Features

- Memory mapped I/O
- Software selectable I/O options (TRI-STATE® output, push-pull output, weak pull-up input, high impedance input
- Schmitt trigger inputs on ports G and L
- Packages:
  - — 28 DIP or SO each with 24 I/O pins

CPU/Instruction Set Features

- 1 μs instruction cycle time
- Fourteen multi-source vectored interrupts servicing
  - — External interrupt
  - — Idle timer T0
  - — Two timers (each with 2 interrupts)
  - — MICROWIRE/PLUS
  - — Multi-Input Wake Up
  - — Software trap
  - — UART (2)
  - — Default VIS (default interrupt)
- Versatile instruction set with true bit manipulation
- 8-bit Stack Pointer SP (stack in RAM)
- Two 8-bit register indirect data memory pointers (B and X)

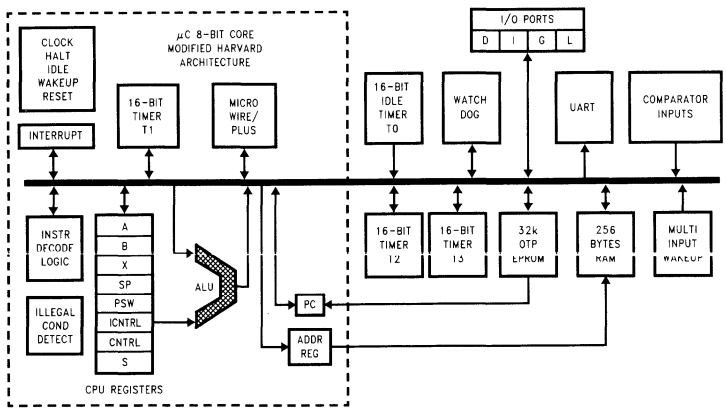
Fully Static CMOS

- Two power saving modes: HALT and IDLE
- Single supply operation: 2.7V–5.5V
- Temperature range: −40°C to +85°C

Development Support

- Emulation device for the COP884EG
- Real time emulation and full program debug offered by MetaLink Development Systems

Block Diagram



TL/DD12872–1

FIGURE 1. Block Diagram

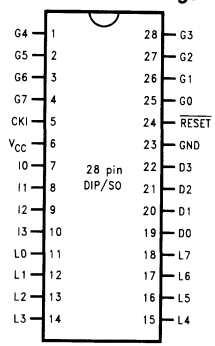
General Description (Continued)

The device is a fully static part, fabricated using double-metal silicon gate microCMOS technology. Features include an 8-bit memory mapped architecture, MICROWIRE/PLUS™ serial I/O, three 16-bit timer/counters supporting three modes (Processor Independent PWM generation, External Event counter, and Input Capture mode capabilities), full duplex UART and two comparators. Each I/O pin has software selectable configurations. The devices operates over a voltage range of 2.7V to 5.5V. High throughput is achieved with an efficient, regular instruction set operating at a maximum rate of 1 μs per instruction.

TABLE I. COP888CG/EG Family Members

Device	ROM/EPROM (Bytes)	RAM (Bytes)	Key Common Features
COP888CG	4k ROM	192	3 Timers
COP888EG	8k ROM	256	UART
COP87L84EG	8k OTP EPROM	256	1 Comparator
COP87L84RG	32k OTP EPROM	256	

Dual-In-Line Package



TL/DD12872–2

Top View

Order Number COP87L84RGM-XE or COP87L84RGN-XE
See NS Package Number M28B or N28B

Note: -X Crystal Oscillator
-E Halt Enable

FIGURE 2. Connection Diagrams

Connection Diagrams (Continued)

Pinouts for 28-Pin Package

Port	Type	Alt. Fun	Alt. Fun	28-Pin DIP/SO
L0	I/O	MIWU		11
L1	I/O	MIWU	CKX	12
L2	I/O	MIWU	TDX	13
L3	I/O	MIWU	RDX	14
L4	I/O	MIWU	T2A	15
L5	I/O	MIWU	T2B	16
L6	I/O	MIWU	T3A	17
L7	I/O	MIWU	T3B	18
G0	I/O	INT		25
G1	WDOUT			26
G2	I/O	T1B		27
G3	I/O	T1A		28
G4	I/O	SO		1
G5	I/O	SK		2
G6	I	SI		3
G7	I/CKO	HALT Restart		4
D0	O			19
D1	O			20
D2	O			21
D3	O			22
I0	I			7
I1	I	COMP1IN−		8
I2	I	COMP1IN+		9
I3	I	COMP1OUT		10
V_{CC}				6
GND				23
CKI				5
$\overline{\text{RESET}}$				24

Absolute Maximum Ratings (Note)

If Military/Aerospace specified devices are required, please contact the National Semiconductor Sales Office/Distributors for availability and specifications.

Supply Voltage (V_{CC})	7V
Voltage at Any Pin	−0.3V to V_{CC} + 0.3V
Total Current into V_{CC} Pin (Source)	100 mA
Total Current out of GND Pin (Sink)	110 mA
Storage Temperature Range	−65°C to +140°C

Note: *Absolute maximum ratings indicate limits beyond which damage to the device may occur. DC and AC electrical specifications are not ensured when operating the device at absolute maximum ratings.*

DC Electrical Characteristics −40°C ≤ T_A ≤ +85°C unless otherwise specified

Parameter	Conditions	Min	Typ	Max	Units
Operating Voltage		2.7		5.5	V
Power Supply Ripple (Note 1)	Peak-to-Peak			0.1 V_{CC}	V
Supply Current (Note 2)					
CKI = 10 MHz	V_{CC} = 5.5V, t_c = 1 μs			16.5	mA
CKI = 4 MHz	V_{CC} = 4.0V, t_c = 2.5 μs			6.5	mA
HALT Current (Note 3)	V_{CC} = 5.5V, CKI = 0 MHz			12	μA
	V_{CC} = 4.0V, CKI = 0 MHz			8	μA
IDLE Current					
CKI = 10 MHz	V_{CC} = 5.5V, t_c = 1 μs			3.5	mA
CKI = 1 MHz	V_{CC} = 4.0V, t_c = 10 μs			0.7	mA
Input Levels					
$\overline{RESET}$					
Logic High		0.8 V_{CC}			V
Logic Low				0.2 V_{CC}	V
CKI (External and Crystal Osc. Modes)					
Logic High		0.7 V_{CC}			V
Logic Low				0.2 V_{CC}	V
All Other Inputs					
Logic High		0.7 V_{CC}			V
Logic Low				0.2 V_{CC}	V
Hi-Z Input Leakage	V_{CC} = 5.5V	−2		+2	μA
Input Pullup Current	V_{CC} = 5.5V	40		250	μA
G and L Port Input Hysteresis			0.05 V_{CC}	0.35 V_{CC}	V
Output Current Levels					
D Outputs					
Source	V_{CC} = 4.5V, V_{OH} = 3.3V	0.4			mA
Sink (Note 4)	V_{CC} = 4.5V, V_{OL} = 1V	10			mA
All Others					
Source (Weak Pull-Up Mode)	V_{CC} = 4.5V, V_{OH} = 2.7V	10		100	μA
Source (Push-Pull Mode)	V_{CC} = 4.5V, V_{OH} = 3.3V	0.4			mA
Sink (Push-Pull Mode)	V_{CC} = 4.5V, V_{OL} = 0.4V	1.6			mA
TRI-STATE Leakage	V_{CC} = 5.5V	−2		+2	μA
Allowable Sink/Source Current per Pin					
D Outputs (Sink)				15	mA
All others				3	mA
Maximum Input Current without Latchup (Note 5)	T_A = 25°C			±100	mA
RAM Retention Voltage, V_r	500 ns Rise and Fall Time (Min)	2			V
Input Capacitance				7	pF
Load Capacitance on D2				1000	pF

Note 1: Rate of voltage change must be less then 0.5 V/ms.

Note 2: Supply current is measured after running 2000 cycles with a square wave CKI input, CKO open, inputs at rails and outputs open.

Note 3: The HALT mode will stop CKI from oscillating in the RC and the Crystal configurations by bringing CKI high. Test conditions: All inputs tied to V_{CC}, L and G ports in the TRI-STATE mode and tied to ground, all outputs low and tied to ground. The clock monitor is disabled.

Note 4: The user must guarantee that D2 pin does not source more than 10 mA during RESET. If D2 sources more than 10 mA during reset, the device will go into programming mode.

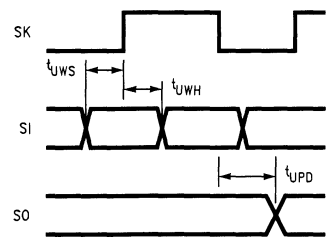
Note 5: Pins G6 and $\overline{RESET}$ are designed with a high voltage input network for factory testing. These pins allow input voltages greater than V_{CC} and the pins will have sink current to V_{CC} when biased at voltages greater than V_{CC} (the pins do not have source current when biased at a voltage below V_{CC}). The effective resistance to V_{CC} is 750Ω (typical). These two pins will not latch up. The voltage at the pins must be limited to less than 14V.

AC Electrical Characteristics $-40°C \leq T_A \leq +85°C$ unless otherwise specified

Parameter	Conditions	Min	Typ	Max	Units
Instruction Cycle Time (t_C)					
Crystal, Resonator,		1		DC	μs
R/C Oscillator		3		DC	μs
Inputs					
t_{SETUP}		200			ns
t_{HOLD}		60			ns
Output Propagation Delay	$R_L = 2.2k, C_L = 100$ pF				
t_{PD1}, t_{PD0}					
SO, SK				0.7	μs
All Others				1	μs
MICROWIRE™ Setup Time (t_{UWS})		20			ns
MICROWIRE Hold Time (t_{UWH})		56			ns
MICROWIRE Output Propagation Delay (t_{UPD})				220	ns
Input Pulse Width					
Interrupt Input High Time		1			t_C
Interrupt Input Low Time		1			t_C
Timer Input High Time		1			t_C
Timer Input Low Time		1			t_C
Reset Pulse Width		1			μs

Comparator AC and DC Characteristics $V_{CC} = 5V, T_A = 25°C$

Parameter	Conditions	Min	Typ	Max	Units
Input Offset Voltage	$0.4V \leq V_{IN} \leq V_{CC} - 1.5V$		±10	±25	mV
Input Common Mode Voltage Range		0.4		$V_{CC} - 1.5$	V
Low Level Output Current	$V_{OL} = 0.4V$	1.6			mA
High Level Output Current	$V_{OH} = 4.6V$	1.6			mA
Comparator DC Supply Current (When Enabled)				250	μA
Response Time	TBD mV Step, TBD mV Overdrive, 100 pF Load		1		μs



TL/DD/12872-3

FIGURE 3. MICROWIRE/PLUS Timing

Pin Descriptions

V_{CC} and GND are the power supply pins.

CKI is the clock input. This can come from an R/C generated oscillator, or a crystal oscillator (in conjunction with CKO). See Oscillator Description section.

$\overline{\text{RESET}}$ is the master reset input. See Reset Description section.

The device contains two bidirectional 8-bit I/O ports (G and L), where each individual bit may be independently configured as an input (Schmitt trigger inputs on ports L and G), output or TRI-STATE under program control. Three data memory address locations are allocated for each of these I/O ports. Each I/O port has two associated 8-bit memory mapped registers, the CONFIGURATION register and the output DATA register. A memory mapped address is also reserved for the input pins of each I/O port. (See the memory map for the various addresses associated with the I/O ports.) *Figure 4* shows the I/O port configurations. The DATA and CONFIGURATION registers allow for each port bit to be individually configured under software control as shown below:

CONFIGURATION Register	DATA Register	Port Set-Up
0	0	Hi-Z Input (TRI-STATE Output)
0	1	Input with Weak Pull-Up
1	0	Push-Pull Zero Output
1	1	Push-Pull One Output

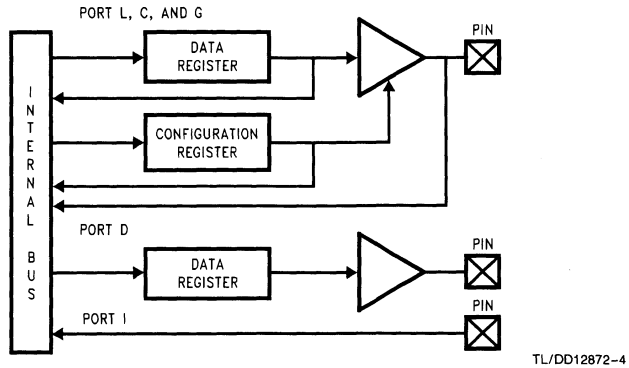
PORT L is an 8-bit I/O port. All L-pins have Schmitt triggers on the inputs.

Port L supports Multi-Input Wake Up (MIWU) on all eight pins. L1 is used for the UART external clock. L2 and L3 are used for the UART transmit and receive. L4 and L5 are used for the timer input functions T2A and T2B. L6 and L7 are used for the timer input functions T3A and T3B.

Port L has the following alternate features:

- L0 MIWU
- L1 MIWU or CKX
- L2 MIWU or TDX
- L3 MIWU or RDX
- L4 MIWU or T2A
- L5 MIWU or T2B
- L6 MIWU or T3A
- L7 MIWU or T3B

Port G is an 8-bit port with 5 I/O pins (G0, G2–G5), an input pin (G6), and two dedicated output pins (G1 and G7). Pins G0 and G2–G6 all have Schmitt Triggers on their inputs. Pin G1 serves as the dedicated WDOUT WATCHDOG output, while pin G7 is either input or output depending on the oscillator mask option selected. With the crystal oscillator option selected, G7 serves as the dedicated output pin for the CKO clock output. With the single-pin R/C oscillator mask option selected, G7 serves as a general purpose input pin but is also used to bring the device out of HALT mode with a low to high transition on G7. There are two registers associated with the G Port, a data register and a configuration register. Therefore, each of the 5 I/O bits (G0, G2–G5) can be individually configured under software control.



TL/DD12872–4

FIGURE 4. I/O Port Configurations

Pin Descriptions (Continued)

Since G6 is an input only pin and G7 is the dedicated CKO clock output pin (crystal clock option) or general purpose input (R/C clock option), the associated bits in the data and configuration registers for G6 and G7 are used for special purpose functions as outlined below. Reading the G6 and G7 data bits will return zeros.

Note that the chip will be placed in the HALT mode by writing a "1" to bit 7 of the Port G Data Register. Similarly the chip will be placed in the IDLE mode by writing a "1" to bit 6 of the Port G Data Register.

Writing a "1" to bit 6 of the Port G Configuration Register enables the MICROWIRE/PLUS to operate with the alternate phase of the SK clock. The G7 configuration bit, if set high, enables the clock start up delay after HALT when the R/C clock configuration is used.

	Config Reg.	Data Reg.
G7	CLKDLY	HALT
G6	Alternate SK	IDLE

Port G has the following alternate features:

- G0 INTR (External Interrupt Input)
- G2 T1B (Timer T1 Capture Input)
- G3 T1A (Timer T1 I/O)
- G4 SO (MICROWIRE Serial Data Output)
- G5 SK (MICROWIRE Serial Clock)
- G6 SI (MICROWIRE Serial Data Input)

Port G has the following dedicated functions:

- G1 WDOUT WATCHDOG and/or Clock Monitor dedicated output
- G7 CKO Oscillator dedicated output or general purpose input

Port I is a four-bit Hi-Z input port.

Port I1–I3 are used for Comparator 1. Port I4–I6 are used for Comparator 2.

The Port I has the following alternate features.

- I1 COMP1 − IN (Comparator 1 Negative Input)
- I2 COMP1 + IN (Comparator 1 Positive Input)
- I3 COMP1OUT (Comparator 1 Output)

Port D is a recreated 4-bit output port that is preset high when $\overline{\text{RESET}}$ goes low. D port recreation is one clock cycle behind normal port timing. The user can tie two or more D port outputs (except D2) together in order to get a higher drive.

Functional Description

The architecture of the device is modified Harvard architecture. With the Harvard architecture, the control store program memory (ROM) is separated from the data store memory (RAM). Both ROM and RAM have their own separate addressing space with separate address buses. The architecture, though based on Harvard architecture, permits transfer of data from ROM to RAM.

CPU REGISTERS

The CPU can do an 8-bit addition, subtraction, logical or shift operation in one instruction (t_C) cycle time.

There are six CPU registers:

A is the 8-bit Accumulator Register

PC is the 15-bit Program Counter Register

PU is the upper 7 bits of the program counter (PC)
PL is the lower 8 bits of the program counter (PC)

B is an 8-bit RAM address pointer, which can be optionally post auto incremented or decremented.

X is an 8-bit alternate RAM address pointer, which can be optionally post auto incremented or decremented.

SP is the 8-bit stack pointer, which points to the subroutine/interrupt stack (in RAM). The SP is initialized to RAM address 06F with reset.

S is the 8-bit Data Segment Address Register used to extend the lower half of the address range (00 to 7F) into 256 data segments of 128 bytes each.

All the CPU registers are memory mapped with the exception of the Accumulator (A) and the Program Counter (PC).

PROGRAM MEMORY

The program memory consists of 32 kbytes of OTP EPROM. These bytes may hold program instructions or constant data (data tables for the LAID instruction, jump vectors for the JID instruction, and interrupt vectors for the VIS instruction). The program memory is addressed by the 15-bit program counter (PC). All interrupts in the devices vector to program memory location 0FF Hex.

The device can be configured to inhibit external reads of the program memory. This is done by programming the Security Byte.

Note: Mask ROMed devices with equivalent on-chip features and program memory sizes of 4k and 8k are available.

SECURITY FEATURE

The program memory array has an associate Security Byte that is located outside of the program address range. This byte can be addressed only from programming mode by a programmer tool.

Security is an optional feature and can only be asserted after the memory array has been programmed and verified. A secured part will read all 00(hex) by a programmer. The part will fail Blank Check and will fail Verify operations. A Read operaiton will fill the programmer's memory with 00(hex). The Security Byte itself is always readable with value of 00(hex) if unsecure and FF(hex) if secure.

DATA MEMORY

The data memory address space includes the on-chip RAM and data registers, the I/O registers (Configuration, Data and Pin), the control registers, the MICROWIRE/PLUS SIO shift register, and the various registers, and counters associated with the timers (with the exception of the IDLE timer). Data memory is addressed directly by the instruction or indirectly by the B, X, SP pointers and S register.

Functional Description (Continued)

The data memory consists of 256 bytes of RAM. Sixteen bytes of RAM are mapped as "registers" at addresses 0F0 to 0FF Hex. These registers can be loaded immediately, and also decremented and tested with the DRSZ (decrement register and skip if zero) instruction. The memory pointer registers X, SP, B and S are memory mapped into this space at address locations 0FC to 0FF Hex respectively, with the other registers being available for general usage.

The instruction set permits any bit in memory to be set, reset or tested. All I/O and registers (except A and PC) are memory mapped; therefore, I/O bits and register bits can be directly and individually set, reset and tested. The accumulator (A) bits can also be directly and individually tested.

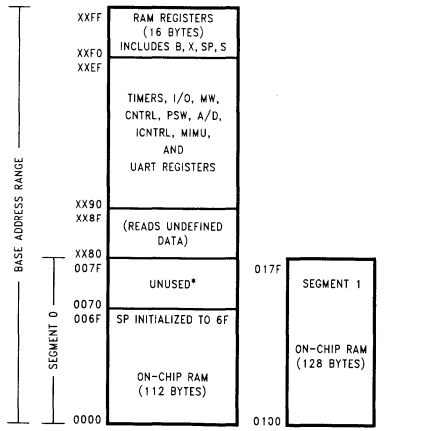
Note: RAM contents are undefined upon power-up.

Data Memory Segment RAM Extension

Data memory address 0FF is used as a memory mapped location for the Data Segment Address Register (S).

The data store memory is either addressed directly by a single byte address within the instruction, or indirectly relative to the reference of the B, X, or SP pointers (each contains a single-byte address). This single-byte address allows an addressing range of 256 locations from 00 to FF hex. The upper bit of this single-byte address divides the data store memory into two separate sections as outlined previously. With the exception of the RAM register memory from address locations 00F0 to 00FF, all RAM memory is memory mapped with the upper bit of the single-byte address being equal to zero. This allows the upper bit of the single-byte address to determine whether or not the base address range (from 0000 to 00FF) is extended. If this upper bit equals one (representing address range 0080 to 00FF), then address extension does not take place. Alternatively, if this upper bit equals zero, then the data segment extension register S is used to extend the base address range (from 0000 to 007F) from XX00 to XX7F, where XX represents the 8 bits from the S register. Thus the 128-byte data segment extensions are located from addresses 0100 to 017F for data segment 1, 0200 to 027F for data segment 2, etc., up to FF00 to FF7F for data segment 255. The base address range from 0000 to 007F represents data segment 0.

Figure 5 illustrates how the S register data memory extension is used in extending the lower half of the base address range (00 to 7F hex) into 256 data segments of 128 bytes each, with a total addressing range of 32 kbytes from XX00 to XX7F. This organization allows a total of 256 data segments of 128 bytes each with an additional upper base segment of 128 bytes. Furthermore, all addressing modes are available for all data segments. The S register must be changed under program control to move from one data segment (128 bytes) to another. However, the upper base segment (containing the 16 memory registers, I/O registers, control registers, etc.) is always available regardless of the contents of the S register, since the upper base segment (address range 0080 to 00FF) is independent of data segment extension.



*Reads as all ones.

FIGURE 5. RAM Organization

The instructions that utilize the stack pointer (SP) always reference the stack as part of the base segment (Segment 0), regardless of the contents of the S register. The S register is not changed by these instructions. Consequently, the stack (used with subroutine linkage and interrupts) is always located in the base segment. The stack pointer will be initialized to point at data memory location 006F as a result of reset.

The 128 bytes of RAM contained in the base segment are split between the lower and upper base segments. The first 116 bytes of RAM are resident from address 0000 to 006F in the lower base segment, while the remaining 16 bytes of RAM represent the 16 data memory registers located at addresses 00F0 to 00FF of the upper base segment. No RAM is located at the upper sixteen addresses (0070 to 007F) of the lower base segment.

Additional RAM beyond these initial 128 bytes, however, will always be memory mapped in groups of 128 bytes (or less) at the data segment address extensions (XX00 to XX7F) of the lower base segment. The additional 128 bytes of RAM are memory mapped at address locations 0100 to 017F hex.

Reset

The $\overline{\text{RESET}}$ input when pulled low initializes the microcontroller. Initialization will occur whenever the $\overline{\text{RESET}}$ input is pulled low. Upon initialization, the data and configuration registers for ports L, G and C are cleared, resulting in these

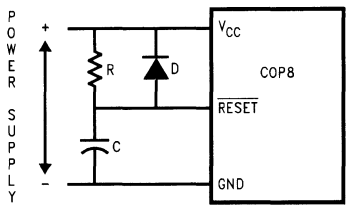
Reset (Continued)

Ports being initialized to the TRI-STATE mode. Pin G1 of the G Port is an exception (as noted below) since pin G1 is dedicated as the WATCHDOG and/or Clock Monitor error output pin. Port D is set high. The PC, PSW, ICNTRL, CNTRL, T2CNTRL and T3CNTRL control registers are cleared. The UART registers PSR, ENU (except that TBMT bit is set), ENUR and ENUI are cleared. The Comparator Select Register is cleared. The S register is initialized to zero. The Multi-Input Wake Up registers WKEN, WKEDG and WKPND are cleared. The stack pointer, SP, is initialized to 6F Hex.

The device comes out of reset with both the WATCHDOG logic and the Clock Monitor detector armed, with the WATCHDOG service window bits set and the Clock Monitor bit set. The WATCHDOG and Clock Monitor circuits are inhibited during reset. The WATCHDOG service window bits being initialized high default to the maximum WATCHDOG service window of 64k t_c clock cycles. The Clock Monitor bit being initialized high will cause a Clock Monitor error following reset if the clock has not reached the minimum specified frequency at the termination of reset. A Clock Monitor error will cause an active low error output on pin G1. This error output will continue until 16 t_c–32 t_c clock cycles following the clock frequency reaching the minimum specified value, at which time the G1 output will enter the TRI-STATE mode.

The external RC network shown in *Figure 6* should be used to ensure that the $\overline{\text{RESET}}$ pin is held low until the power supply to the chip stabilizes.

Note: Continual state of reset will cause the device to draw excessive current.



TL/DD/12872–6

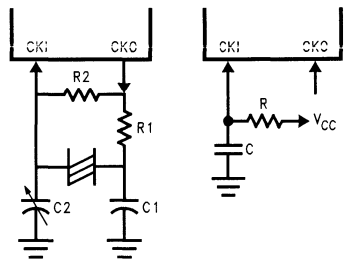
RC > 5 × Power Supply Rise Time

FIGURE 6. Recommended Reset Circuit

Oscillator Circuits

The chip can be driven by a clock input on the CKI input pin which can be between DC and 10 MHz. The CKO output clock is on pin G7 (crystal configuration). The CKI input frequency is divided down by 10 to produce the instruction cycle clock ($1/t_c$).

Figure 7 shows the Crystal and R/C diagrams.



TL/DD12872–7

FIGURE 7. Crystal and R/C Oscillator Diagrams

CRYSTAL OSCILLATOR

CKI and CKO can be connected to make a closed loop crystal (or resonator) controlled oscillator.

Table I shows the component values required for various standard crystal values.

TABLE I. Crystal Oscillator Configuration, T_A = 25°C

R1 (kΩ)	R2 (MΩ)	C1 (pF)	C2 (pF)	CKI Freq (MHz)	Conditions
0	1	30	30–36	10	V_{CC} = 5V
0	1	30	30–36	4	V_{CC} = 5V
0	1	200	100–150	0.455	V_{CC} = 5V

R/C OSCILLATOR

By selecting CKI as a single pin oscillator input, a single pin R/C oscillator circuit can be connected to it. CKO is available as a general purpose input, and/or HALT restart pin.

Table II shows the variation in the oscillator frequencies as functions of the component (R and C) values.

TABLE II. R/C Oscillator Configuration, T_A = 25°C

R (kΩ)	C (pF)	CKI Freq (MHz)	Instr. Cycle (μs)	Conditions
3.3	82	2.2–2.7	3.7–4.6	V_{CC} = 5V
5.6	100	1.1–1.3	7.4–9.0	V_{CC} = 5V
6.8	100	0.9–1.1	8.8–10.8	V_{CC} = 5V

Note: 3k ≤ R ≤ 200k

50 pF ≤ C ≤ 200 pF

Control Registers

CNTRL Register (Address X'00EE)

The Timer1 (T1) and MICROWIRE/PLUS control register contains the following bits:

SL1 & SL0	Select the MICROWIRE/PLUS clock divide by (00 = 2, 01 = 4, 1x = 8)
IEDG	External interrupt edge polarity select (0 = Rising edge, 1 = Falling edge)
MSEL	Selects G5 and G4 as MICROWIRE/PLUS signals SK and SO respectively
T1C0	Timer T1 Start/Stop control in timer modes 1 and 2 Timer T1 Underflow Interrupt Pending Flag in timer mode 3
T1C1	Timer T1 mode control bit
T1C2	Timer T1 mode control bit
T1C3	Timer T1 mode control bit

T1C3	T1C2	T1C1	T1C0	MSEL	IEDG	SL1	SL0

Bit 7 Bit 0

Control Registers (Continued)

PSW Register (Address X'00EF)

The PSW register contains the following select bits:

- GIE Global interrupt enable (enables interrupts)
- EXEN Enable external interrupt
- BUSY MICROWIRE/PLUS busy shifting flag
- EXPND External interrupt pending
- T1ENA Timer T1 Interrupt Enable for Timer Underflow or T1A Input capture edge
- T1PNDA Timer T1 Interrupt Pending Flag (Autoreload RA in mode 1, T1 Underflow in Mode 2, T1A capture edge in mode 3)
- C Carry Flag
- HC Half Carry Flag

HC	C	T1PNDA	T1ENA	EXPND	BUSY	EXEN	GIE

Bit 7 Bit 0

The Half-Carry bit is also affected by all the instructions that affect the Carry flag. The SC (Set Carry) and RC (Reset Carry) instructions will respectively set or clear both the carry flags. In addition to the SC and RC instructions, ADC, SUBC, RRC and RLC instructions affect the carry and Half Carry flags.

ICNTRL Register (Address X'00E8)

The ICNTRL register contains the following bits:

- T1ENB Timer T1 Interrupt Enable for T1B Input capture edge
- T1PNDB Timer T1 Interrupt Pending Flag for T1B capture edge
- WEN Enable MICROWIRE/PLUS interrupt
- WPND MICROWIRE/PLUS interrupt pending
- T0EN Timer T0 Interrupt Enable (Bit 12 toggle)
- T0PND Timer T0 Interrupt pending
- LPEN L Port Interrupt Enable (Multi-Input Wake Up/ Interrupt)

  Bit 7 could be used as a flag

Unused	LPEN	T0PND	T0EN	WPND	WEN	T1PNDB	T1ENB

Bit 7 Bit 0

T2CNTRL Register (Address X'00C6)

The T2CNTRL register contains the following bits:

- T2ENB Timer T2 Interrupt Enable for T2B Input capture edge
- T2PNDB Timer T2 Interrupt Pending Flag for T2B capture edge
- T2ENA Timer T2 Interrupt Enable for Timer Underflow or T2A Input capture edge
- T2PNDA Timer T2 Interrupt Pending Flag (Autoreload RA in mode 1, T2 Underflow in mode 2, T2A capture edge in mode 3)
- T2C0 Timer T2 Start/Stop control in timer modes 1 and 2 Timer T2 Underflow Interrupt Pending Flag in timer mode 3
- T2C1 Timer T2 mode control bit
- T2C2 Timer T2 mode control bit
- T2C3 Timer T2 mode control bit

T2C3	T2C2	T2C1	T2C0	T2PNDA	T2ENA	T2PNDB	T2ENB

Bit 7 Bit 0

T3CNTRL Register (Address X'00B6)

The T3CNTRL register contains the following bits:

- T3ENB Timer T3 Interrupt Enable for T3B
- T3PNDB Timer T3 Interrupt Pending Flag for T3B pin (T3B capture edge)
- T3ENA Timer T3 Interrupt Enable for Timer Underflow or T3A pin
- T3PNDA Timer T3 Interrupt Pending Flag (Autoload RA in mode 1, T3 Underflow in mode 2, T3a capture edge in mode 3)
- T3C0 Timer T3 Start/Stop control in timer modes 1 and 2

  Timer T3 Underflow Interrupt Pending Flag in timer mode 3
- T3C1 Timer T3 mode control bit
- T3C2 Timer T3 mode control bit
- T3C3 Timer T3 mode control bit

T3C3	T3C2	T3C1	T3C0	T3PNDA	T3ENA	T3PNDB	T3ENB

Bit 7 Bit 0

Timers

The device contains a very versatile set of timers (T0, T1, T2, T3). All timers and associated autoreload/capture registers power up containing random data.

TIMER T0 (IDLE TIMER)

The devices support applications that require maintaining real time and low power with the IDLE mode. This IDLE mode support is furnished by the IDLE timer T0, which is a 16-bit timer. The Timer T0 runs continuously at the fixed rate of the instruction cycle clock, t_C. The user cannot read or write to the IDLE Timer T0, which is a count down timer.

The Timer T0 supports the following functions:

Exit out of the Idle Mode (See Idle Mode description)
WATCHDOG logic (See WATCHDOG description)
Start up delay out of the HALT mode

The IDLE Timer T0 can generate an interrupt when the thirteenth bit toggles. This toggle is latched into the T0PND pending flag, and will occur every 4 ms at the maximum clock frequency (t_C = 1 μs). A control flag T0EN allows the interrupt from the thirteenth bit of Timer T0 to be enabled or disabled. Setting T0EN will enable the interrupt, while resetting it will disable the interrupt.

Timers (Continued)

TIMER T1, TIMER T2 AND TIMER T3

The devices have a set of three powerful timer/counter blocks, T1, T2 and T3. The associated features and functioning of a timer block are described by referring to the timer block Tx. Since the three timer blocks, T1, T2 and T3 are identical, all comments are equally applicable to any of the three timer blocks.

Each timer block consists of a 16-bit timer, Tx, and two supporting 16-bit autoreload/capture registers, RxA and RxB. Each timer block has two pins associated with it, TxA and TxB. The pin TxA supports I/O required by the timer block, while the pin TxB is an input to the timer block. The powerful and flexible timer block allows the device to easily perform all timer functions with minimal software overhead. The timer block has three operating modes: Processor Independent PWM mode, External Event Counter mode, and Input Capture mode.

The control bits TxC3, TxC2, and TxC1 allow selection of the different modes of operation.

Mode 1. Processor Independent PWM Mode

As the name suggests, this mode allows the device to generate a PWM signal with very minimal user intervention. The user only has to define the parameters of the PWM signal (ON time and OFF time). Once begun, the timer block will continuously generate the PWM signal completely independent of the microcontroller. The user software services the timer block only when the PWM parameters require updating.

In this mode the timer Tx counts down at a fixed rate of t_C. Upon every underflow the timer is alternately reloaded with the contents of supporting registers, RxA and RxB. The very first underflow of the timer causes the timer to reload from the register RxA. Subsequent underflows cause the timer to be reloaded from the registers alternately beginning with the register RxB.

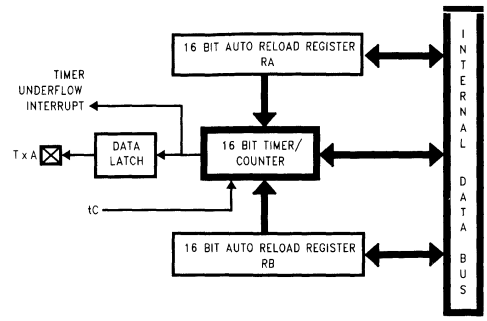
The Tx Timer control bits, TxC3, TxC2 and TxC1 set up the timer for PWM mode operation.

Figure 8 shows a block diagram of the timer in PWM mode.

The underflows can be programmed to toggle the TxA output pin. The underflows can also be programmed to generate interrupts.

Underflows from the timer are alternately latched into two pending flags, TxPNDA and TxPNDB. The user must reset these pending flags under software control. Two control enable flags, TxENA and TxENB, allow the interrupts from the timer underflow to be enabled or disabled. Setting the timer enable flag TxENA will cause an interrupt when a timer underflow causes the RxA register to be reloaded into the timer. Setting the timer enable flag TxENB will cause an interrupt when a timer underflow causes the RxB register to be reloaded into the timer. Resetting the timer enable flags will disable the associated interrupts.

Either or both of the timer underflow interrupts may be enabled. This gives the user the flexibility of interrupting once per PWM period on either the rising or falling edge of the PWM output. Alternatively, the user may choose to interrupt on both edges of the PWM output.



TL/DD12872-8

FIGURE 8. Timer in PWM Mode

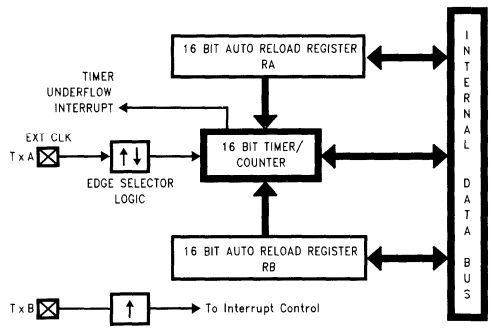
Mode 2. External Event Counter Mode

This mode is quite similar to the processor independent PWM mode described above. The main difference is that the timer, Tx, is clocked by the input signal from the TxA pin. The Tx timer control bits, TxC3, TxC2 and TxC1 allow the timer to be clocked either on a positive or negative edge from the TxA pin. Underflows from the timer are latched into the TxPNDA pending flag. Setting the TxENA control flag will cause an interrupt when the timer underflows.

In this mode the input pin TxB can be used as an independent positive edge sensitive interrupt input if the TxENB control flag is set. The occurrence of a positive edge on the TxB input pin is latched into the TxPNDB flag.

Figure 9 shows a block diagram of the timer in External Event Counter mode.

Note: The PWM output is not available in this mode since the TxA pin is being used as the counter input clock.



TL/DD12872-9

FIGURE 9. Timer in External Event Counter Mode

Mode 3. Input Capture Mode

The device can precisely measure external frequencies or time external events by placing the timer block, Tx, in the input capture mode.

In this mode, the timer Tx is constantly running at the fixed t_C rate. The two registers, RxA and RxB, act as capture registers. Each register acts in conjunction with a pin. The register RxA acts in conjunction with the TxA pin and the register RxB acts in conjunction with the TxB pin.

Timers (Continued)

The timer value gets copied over into the register when a trigger event occurs on its corresponding pin. Control bits, TxC3, TxC2 and TxC1, allow the trigger events to be specified either as a positive or a negative edge. The trigger condition for each input pin can be specified independently.

The trigger conditions can also be programmed to generate interrupts. The occurrence of the specified trigger condition on the TxA and TxB pins will be respectively latched into the pending flags, TxPNDA and TxPNDB. The control flag TxENA allows the interrupt on TxA to be either enabled or disabled. Setting the TxENA flag enables interrupts to be generated when the selected trigger condition occurs on the TxA pin. Similarly, the flag TxENB controls the interrupts from the TxB pin.

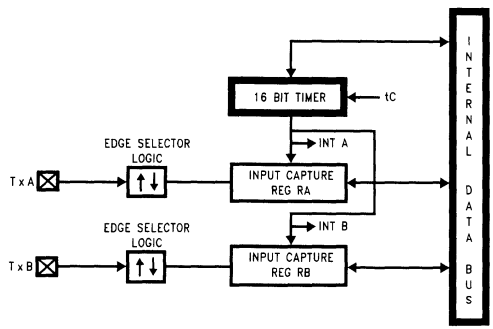
Underflows from the timer can also be programmed to generate interrupts. Underflows are latched into the timer TxC0 pending flag (the TxC0 control bit serves as the timer underflow interrupt pending flag in the Input Capture mode). Consequently, the TxC0 control bit should be reset when entering the Input Capture mode. The timer underflow interrupt is enabled with the TxENA control flag. When a TxA interrupt occurs in the Input Capture mode, the user must check both the TxPNDA and TxC0 pending flags in order to determine whether a TxA input capture or a timer underflow (or both) caused the interrupt.

Figure 10 shows a block diagram of the timer in Input Capture mode.

TIMER CONTROL FLAGS

The timers T1, T2 and T3 have indentical control structures. The control bits and their functions are summarized below.

TxC0	Timer Start/Stop control in Modes 1 and 2 (Processor Independent PWM and External Event Counter), where 1 = Start, 0 = Stop Timer Underflow Interrupt Pending Flag in Mode 3 (Input Capture)
TxPNDA	Timer Interrupt Pending Flag
TxPNDB	Timer Interrupt Pending Flag
TxENA	Timer Interrupt Enable Flag
TxENB	Timer Interrupt Enable Flag 1 = Timer Interrupt Enabled 0 = Timer Interrupt Disabled
TxC3	Timer mode control
TxC2	Timer mode control
TxC1	Timer mode control



TL/DD12872–10

FIGURE 10. Timer in Input Capture Mode

Timers (Continued)

The timer mode control bits (TxC3, TxC2 and TxC1) are detailed below:

TxC3	TxC2	TxC1	Timer Mode	Interrupt A Source	Interrupt B Source	Timer Counts On
0	0	0	MODE 2 (External Event Counter)	Timer Underflow	Pos. TxB Edge	TxA Pos. Edge
0	0	1	MODE 2 (External Event Counter)	Timer Underflow	Pos. TxB Edge	TxA Neg. Edge
1	0	1	MODE 1 (PWM) TxA Toggle	Autoreload RA	Autoreload RB	t_C
1	0	0	MODE 1 (PWM) No TxA Toggle	Autoreload RA	Autoreload RB	t_C
0	1	0	MODE 3 (Capture) Captures: TxA Pos. Edge TxB Pos. Edge	Pos. TxA Edge or Timer Underflow	Pos. TxB Edge	t_C
1	1	0	MODE 3 (Capture) Captures: TxA Pos. Edge TxB Neg. Edge	Pos. TxA Edge or Timer Underflow	Neg. TxB Edge	t_C
0	1	1	MODE 3 (Capture) Captures: TxA Neg. Edge TxB Pos. Edge	Neg. TxA Edge or Timer Underflow	Pos. TxB Edge	t_C
1	1	1	MODE 3 (Capture) Captures: TxA Neg. Edge TxB Neg. Edge	Neg. TxA Edge or Timer Underflow	Neg. TxB Edge	t_C

Power Save Modes

The devices offer the user two power save modes of operation: HALT and IDLE. In the HALT mode, all microcontroller activities are stopped. In the IDLE mode, the on-board oscillator circuitry the WATCHDOG logic, the Clock Monitor and timer T0 are active but all other microcontroller activities are stopped. In either mode, all on-board RAM, registers, I/O states, and timers (with the exception of T0) are unaltered.

HALT MODE

The devices can be placed in the HALT mode by writing a "1" to the HALT flag (G7 data bit). All microcontroller activities, including the clock and timers, are stopped. The WATCHDOG logic on the device is disabled during the HALT mode. However, the clock monitor circuitry if enabled remains active and will cause the WATCHDOG output pin (WDOUT) to go low. If the HALT mode is used and the user does not want to activate the WDOUT pin, the Clock Monitor should be disabled after the device comes out of reset (resetting the Clock Monitor control bit with the first write to the WDSVR register). In the HALT mode, the power requirements of the device are minimal and the applied voltage (V_{CC}) may be decreased to V_r (V_r = 2.0V) without altering the state of the machine.

The devices support three different ways of exiting the HALT mode. The first method of exiting the HALT mode is with the Multi-Input Wake Up feature on the L port. The second method is with a low to high transition on the CKO (G7) pin. This method precludes the use of the crystal clock configuration (since CKO becomes a dedicated output), and so may be used with an RC clock configuration. The third method of exiting the HALT mode is by pulling the $\overline{\text{RESET}}$ pin low.

Since a crystal or ceramic resonator may be selected as the oscillator, the Wake Up signal is not allowed to start the chip running immediately since crystal oscillators and ceramic resonators have a delayed start up time to reach full amplitude and frequency stability. The IDLE timer is used to generate a fixed delay to ensure that the oscillator has indeed stabilized before allowing instruction execution. In this case, upon detecting a valid Wake Up signal, only the oscillator circuitry is enabled. The IDLE timer is loaded with a value of 256 and is clocked with the t_C instruction cycle clock. The t_C clock is derived by dividing the oscillator clock down by a factor of 10. The Schmitt trigger following the CKI inverter on the chip ensures that the IDLE timer is clocked only when the oscillator has a sufficiently large amplitude to meet the Schmitt trigger specifications. This Schmitt trigger is not part of the oscillator closed loop. The startup timeout from the IDLE timer enables the clock signals to be routed to the rest of the chip.

If an RC clock option is being used, the fixed delay is introduced optionally. A control bit, CLKDLY, mapped as configuration bit G7, controls whether the delay is to be introduced or not. The delay is included if CLKDLY is set, and excluded if CLKDLY is reset. The CLKDLY bit is cleared on reset.

Power Save Modes (Continued)

The WATCHDOG detector circuit is inhibited during the HALT mode. However, the clock monitor circuit if enabled remains active during HALT mode in order to ensure a clock monitor error if the device inadvertently enters the HALT mode as a result of a runaway program or power glitch.

IDLE MODE

The device is placed in the IDLE mode by writing a "1" to the IDLE flag (G6 data bit). In this mode, all activities, except the associated on-board oscillator circuitry, the WATCHDOG logic, the clock monitor and the IDLE Timer T0, are stopped. The power supply requirements of the micro-controller in this mode of operation are typically around 30% of normal power requirement of the microcontroller.

As with the HALT mode, the device can be returned to normal operation with a reset, or with a Multi-Input Wake Up from the L Port. Alternately, the microcontroller resumes normal operation from the IDLE mode when the thirteenth bit (representing 4.096 ms at internal clock frequency of 1 MHz, $t_C = 1\ \mu s$) of the IDLE Timer toggles.

This toggle condition of the thirteenth bit of the IDLE Timer T0 is latched into the T0PND pending flag.

The user has the option of being interrupted with a transition on the thirteenth bit of the IDLE Timer T0. The interrupt can be enabled or disabled via the T0EN control bit. Setting the T0EN flag enables the interrupt and vice versa.

The user can enter the IDLE mode with the Timer T0 interrupt enabled. In this case, when the T0PND bit gets set, the device will first execute the Timer T0 interrupt service routine and then return to the instruction following the "Enter Idle Mode" instruction.

Alternatively, the user can enter the IDLE mode with the IDLE Timer T0 interrupt disabled. In this case, the device will resume normal operation with the instruction immediately following the "Enter IDLE Mode" instruction.

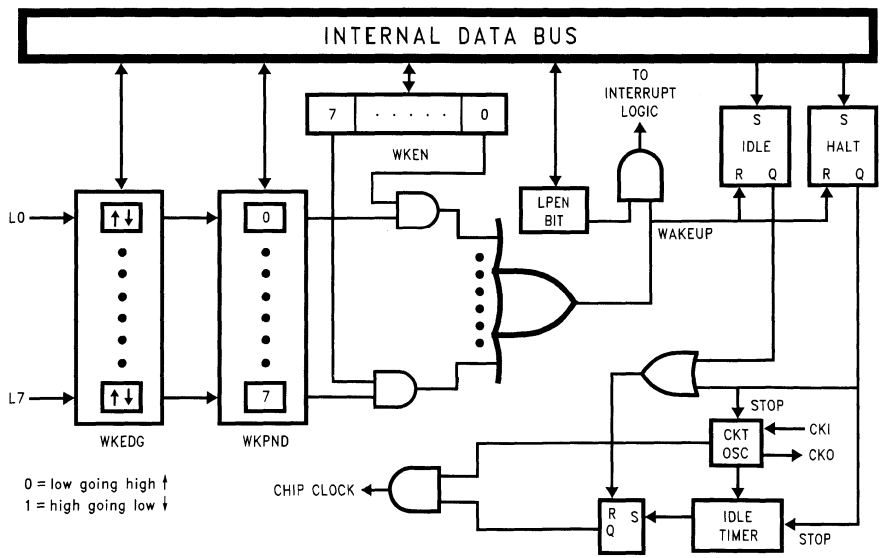
Note: It is necessary to program two NOP instructions following both the set HALT mode and set IDLE mode instructions. These NOP instructions are necessary to allow clock resynchronization following the HALT or IDLE modes.

Due to the on-board 8k EPROM with port recreation logic, the HALT/IDLE current is much higher compared to the equivalent masked port.

Multi-Input Wake Up

The Multi-Input Wake Up feature is ued to return (Wake Up) the device from either the HALT or IDLE modes. Alternately Multi-Input Wake Up/Interrupt feature may also be used to generate up to 8 edge selectable external interrupts.

Figure 11 shows the Multi-Input Wake Up logic. The Multi-Input Wake Up feature utilizes the L Port. The user selects which particular L port bit (or combination of L Port bits) will cause the device to exit the HALT or IDLE modes. The selection is done through the Reg: WKEN. The Reg: WKEN



TL/DD12872–11

FIGURE 11. Multi-Input Wake Up Logic

Multi-Input Wake Up (Continued)

is an 8-bit read/write register, which contains a control bit for every L port bit. Setting a particular WKEN bit enables a Wake Up from the associated L port pin.

The user can select whether the trigger condition on the selected L Port pin is going to be either a positive edge (low to high transition) or a negative edge (high to low transition). This selection is made via the Reg: WKEDG, which is an 8-bit control register with a bit assigned to each L Port pin. Setting the control bit will select the trigger condition to be a negative edge on that particular L Port pin. Resetting the bit selects the trigger condition to be a positive edge. Changing an edge select entails several steps in order to avoid a pseudo Wake Up condition as a result of the edge change. First, the associated WKEN bit should be reset, followed by the edge select change in WKEDG. Next, the associated WKPND bit should be cleared, followed by the associated WKEN bit being re-enabled.

An example may serve to clarify this procedure. Suppose we wish to change the edge select from positive (low going high) to negative (high going low) for L Port bit 5, where bit 5 has previously been enabled for an input interrupt. The program would be as follows:

```
RBIT  5, WKEN
SBIT  5, WKEDG
RBIT  5, WKPND
SBIT  5, WKEN
```

If the L port bits have been used as outputs and then changed to inputs with Multi-Input Wake Up/Interrupt, a safety procedure should also be followed to avoid inherited pseudo wakeup conditions. After the selected L port bits have been changed from output to input but before the associated WKEN bits are enabled, the associated edge select bits in WKEDG should be set or reset for the desired edge selects, followed by the associated WKPND bits being cleared.

This same procedure should be used following reset, since the L port inputs are left floating as a result of reset.

The occurrence of the selected trigger condition for Multi-Input Wake Up is latched into a pending register called WKPND. The respective bits of the WKPND register will be set on the occurrence of the selected trigger edge on the corresponding Port L pin. The user has the responsibility of clearing these pending flags. Since WKPND is a pending register for the occurrence of selected Wake Up conditions, the device will not enter the HALT mode if any Wake Up bit is both enabled and pending. Consequently, the user has the responsibility of clearing the pending flags before attempting to enter the HALT mode.

WKEN, WKPND and WKEDG are all read/write registers, and are cleared at reset.

PORT L INTERRUPTS

Port L provides the user with an additional eight fully selectable, edge sensitive interrupts which are all vectored into the same service subroutine.

The interrupt from Port L shares logic with the wake up circuitry. The register WKEN allows interrupts from Port L to be individually enabled or disabled. The register WKEDG specifies the trigger condition to be either a positive or a negative edge. Finally, the register WKPND latches in the pending trigger conditions.

The GIE (Global Interrupt Enable) bit enables the interrupt function.

A control flag, LPEN, functions as a global interrupt enable for Port L interrupts. Setting the LPEN flag will enable interrupts and vice versa. A separate global pending flag is not needed since the register WKPND is adequate.

Since Port L is also used for waking the device out of the HALT or IDLE modes, the user can elect to exit the HALT or IDLE modes either with or without the interrupt enabled. If he elects to disable the interrupt, then the device will restart execution from the instruction immediately following the instruction that placed the microcontroller in the HALT or IDLE modes. In the other case, the device will first execute the interrupt service routine and then revert to normal operation.

The Wake Up signal will not start the chip running immediately since crystal oscillators or ceramic resonators have a finite start up time. The IDLE Timer (T0) generates a fixed delay to ensure that the oscillator has indeed stabilized before allowing the device to execute instructions. In this case, upon detecting a valid Wake Up signal, only the oscillator circuitry and the IDLE Timer T0 are enabled. The IDLE Timer is loaded with a value of 256 and is clocked from the t_C instruction cycle clock. The t_C clock is derived by dividing down the oscillator clock by a factor of 10. A Schmitt trigger following the CKI on-chip inverter ensures that the IDLE timer is clocked only when the oscillator has a sufficiently large amplitude to meet the Schmitt trigger specifications. This Schmitt trigger is not part of the oscillator closed loop. The startup timeout from the IDLE timer enables the clock signals to be routed to the rest of the chip.

If the RC clock option is used, the fixed delay is under software control. A control flag, CLKDLY, in the G7 configuration bit allows the clock start up delay to be optionally inserted. Setting CLKDLY flag high will cause clock start up delay to be inserted and resetting it will exclude the clock start up delay. The CLKDLY flag is cleared during reset, so the clock start up delay is not present following reset with the RC clock options.

UART

The device contains a full-duplex software programmable UART. The UART *(Figure 12)* consists of a transmit shift register, a receiver shift register and seven addressable registers, as follows: a transmit buffer register (TBUF), a receiver buffer register (RBUF), a UART control and status register (ENU), a UART receive control and status register (ENUR), a UART interrupt and clock source register (ENUI), a prescaler select register (PSR) and baud (BAUD) register. The ENU register contains flags for transmit and receive functions; this register also determines the length of the data frame (7, 8 or 9 bits), the value of the ninth bit in transmission, and parity selection bits. The ENUR register flags framing, data overrun and parity errors while the UART is receiving.

Other functions of the ENUR register include saving the ninth bit received in the data frame, enabling or disabling the UART's attention mode of operation and providing additional receiver/transmitter status information via RCVG and XMTG bits. The determination of an internal or external clock source is done by the ENUI register, as well as selecting the number of stop bits and enabling or disabling transmit and receive interrupts. A control flag in this register can also select the UART mode of operation: asynchronous or synchronous.

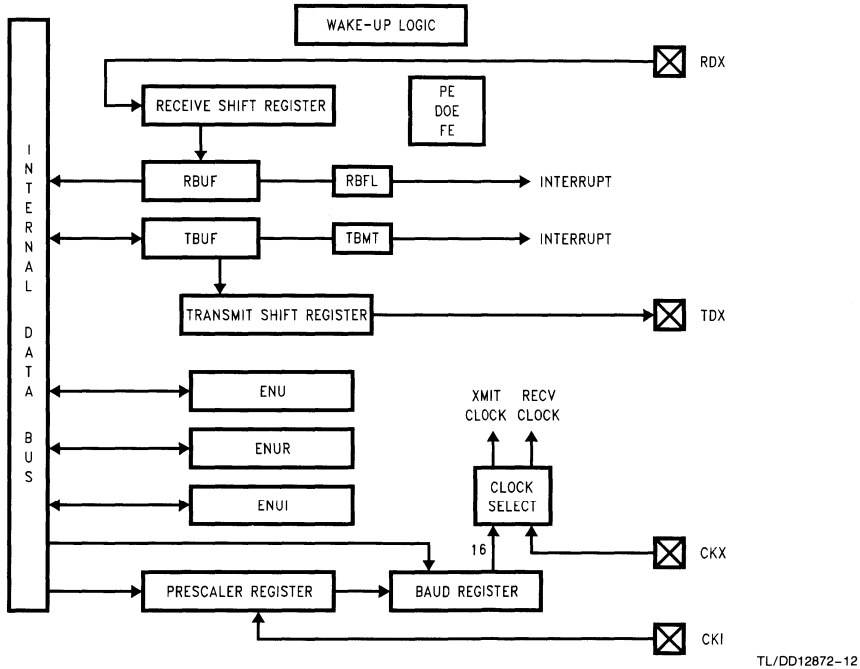


FIGURE 12. UART Block Diagram

UART (Continued)

UART CONTROL AND STATUS REGISTERS

The operation of the UART is programmed through three registers: ENU, ENUR and ENUI. The function of the individual bits in these registers is as follows:

ENU-UART Control and Status Register (Address at 0BA)

PEN	PSEL1	XBIT9/ PSEL0	CHL1	CHL0	ERR	RBFL	TBMT
0RW	0RW	0RW	0RW	0RW	0R	0R	1R

Bit 7 Bit 0

ENUR-UART Receive Control and Status Register (Address at 0BB)

DOE	FE	PE	SPARE	RBIT9	ATTN	XMTG	RCVG
0RD	0RD	0RD	0RW*	0R	0RW	0R	0R

Bit7 Bit0

ENUI-UART Interrupt and Clock Source Register (Address at 0BC)

STP2	STP78	ETDX	SSEL	XRCLK	XTCLK	ERI	ETI
0RW	0RW	0RW	0RW	0RW	0RW	0RW	0RW

Bit7 Bit0

*Bit is not used.

0 Bit is cleared on reset.

1 Bit is set to one on reset.

R Bit is read-only; it cannot be written by software.

RW Bit is read/write.

D Bit is cleared on read; when read by software as a one, it is cleared automatically. Writing to the bit does not affect its state.

DESCRIPTION OF UART REGISTER BITS

ENU—UART CONTROL AND STATUS REGISTER

TBMT: This bit is set when the UART transfers a byte of data from the TBUF register into the TSFT register for transmission. It is automatically reset when software writes into the TBUF register.

RBFL: This bit is set when the UART has received a complete character and has copied it into the RBUF register. It is automatically reset when software reads the character from RBUF.

ERR: This bit is a global UART error flag which gets set if any or a combination of the errors (DOE, FE, PE) occur.

CHL1, CHL0: These bits select the character frame format. Parity is not included and is generated/verified by hardware.

CHL1 = 0, CHL0 = 0 The frame contains eight data bits.
CHL1 = 0, CHL0 = 1 The frame contains seven data bits.
CHL1 = 1, CHL0 = 0 The frame contains nine data bits.
CHL1 = 1, CHL0 = 1 Loopback Mode selected. Transmitter output internally looped back to receiver input. Nine bit framing format is used.

XBIT9/PSEL0: Programs the ninth bit for transmission when the UART is operating with nine data bits per frame. For seven or eight data bits per frame, this bit in conjunction with PSEL1 selects parity.

PSEL1, PSEL0: Parity select bits.

PSEL1 = 0, PSEL0 = 0 Odd Parity (if Parity enabled)
PSEL1 = 0, PSEL0 = 1 Odd Parity (if Parity enabled)
PSEL1 = 1, PSEL0 = 0 Mark(1) (if Parity enabled)
PSEL1 = 1, PSEL1 = 1 Space(0) (if Parity enabled)

PEN: This bit enables/disables Parity (7- and 8-bit modes only).

PEN = 0 Parity disabled.
PEN = 1 Parity enabled.

ENUR—UART RECEIVE CONTROL AND STATUS REGISTER

RCVG: This bit is set high whenever a framing error occurs and goes low when RDX goes high.

XMTG: This bit is set to indicate that the UART is transmitting. It gets reset at the end of the last frame (end of last Stop bit).

ATTN: ATTENTION Mode is enabled while this bit is set. This bit is cleared automatically on receiving a character with data bit nine set.

RBIT9: Contains the ninth data bit received when the UART is operating with nine data bits per frame.

SPARE: Reserved for future use.

PE: Flags a Parity Error.

PE = 0 Indicates no Parity Error has been detected since the last time the ENUR register was read.
PE = 1 Indicates the occurrence of a Parity Error.

FE: Flags a Framing Error.

FE = 0 Indicates no Framing Error has been detected since the last time the ENUR register was read.
FE = 1 Indicates the occurrence of a Framing Error.

DOE: Flags a Data Overrun Error.

DOE = 0 Indicates no Data Overrun Error has been detected since the last time the ENUR register was read.
DOE = 1 Indicates the occurrence of a Data Overrun Error.

ENUI—UART INTERRUPT AND CLOCK SOURCE REGISTER

ETI: This bit enables/disables interrupt from the transmitter section.

ETI = 0 Interrupt from the transmitter is disabled.
ETI = 1 Interrupt from the transmitter is enabled.

ERI: This bit enables/disables interrupt from the receiver section.

ERI = 0 Interrupt from the receiver is disabled.
ERI = 1 Interrupt from the receiver is enabled.

XTCLK: This bit selects the clock source for the transmitter section.

XTCLK = 0 The clock source is selected through the PSR and BAUD registers.
XTCLK = 1 Signal on CKX (L1) pin is used as the clock.

XRCLK: This bit selects the clock source for the receiver section.

XRCLK = 0 The clock source is selected through the PSR and BAUD registers.
XRCLK = 1 Signal on CKX (L1) pin is used as the clock.

SSEL: UART mode select.

SSEL = 0 Asynchronous Mode.
SSEL = 1 Synchronous Mode.

UART (Continued)

ETDX: TDX (UART Transmit Pin) is the alternate function assigned to Port L pin L2; it is selected by setting ETDX bit. To simulate line break generation, software should reset ETDX bit and output logic zero to TDX pin through Port L data and configuration registers.

STP78: This bit is set to program the last Stop bit to be 7/8th of a bit in length.

STP2: This bit programs the number of Stop bits to be transmitted.

STP2 = 0 One Stop bit transmitted.
STP2 = 1 Two Stop bits transmitted.

Associated I/O Pins

Data is transmitted on the TDX pin and received on the RDX pin. TDX is the alternate function assigned to Port L pin L2; it is selected by setting ETDX (in the ENUI register) to one. RDX is an inherent function of Port L pin L3, requiring no setup.

The baud rate clock for the UART can be generated on-chip, or can be taken from an external source. Port L pin L1 (CKX) is the external clock I/O pin. The CKX pin can be either an input or an output, as determined by Port L Configuration and Data registers (Bit 1). As an input, it accepts a clock signal which may be selected to drive the transmitter and/or receiver. As an output, it presents the internal Baud Rate Generator output.

UART Operation

The UART has two modes of operation: asynchronous mode and synchronous mode.

ASYNCHRONOUS MODE

This mode is selected by resetting the SSEL (in the ENUI register) bit to zero. The input frequency to the UART is 16 times the baud rate.

The TSFT and TBUF registers double-buffer data for transmission. While TSFT is shifting out the current character on the TDX pin, the TBUF register may be loaded by software with the next byte to be transmitted. When TSFT finishes transmitting the current character the contents of TBUF are transferred to the TSFT register and the Transmit Buffer Empty Flag (TBMT in the ENU register) is set. The TBMT flag is automatically reset by the UART when software loads a new character into the TBUF register. There is also the XMTG bit which is set to indicate that the UART is transmitting. This bit gets reset at the end of the last frame (end of last Stop bit). TBUF is a read/write register.

The RSFT and RBUF registers double-buffer data being received. The UART receiver continually monitors the signal on the RDX pin for a low level to detect the beginning of a Start bit. Upon sensing this low level, it waits for half a bit time and samples again. If the RDX pin is still low, the receiver considers this to be a valid Start bit, and the remaining bits in the character frame are each sampled a single time, at the mid-bit position. Serial data input on the RDX pin is shifted into the RSFT register. Upon receiving the complete character, the contents of the RSFT register are copied into the RBUF register and the Received Buffer Full Flag (RBFL) is set. RBFL is automatically reset when software reads the character from the RBUF register. RBUF is a read only register. There is also the RCVG bit which is set high when a framing error occurs and goes low once RDX goes high. TBMT, XMTG, RBFL and RCVG are read only bits.

SYNCHRONOUS MODE

In this mode data is transferred synchronously with the clock. Data is transmitted on the rising edge and received on the falling edge of the synchronous clock.

This mode is selected by setting SSEL bit in the ENUI register. The input frequency to the UART is the same as the baud rate.

When an external clock input is selected at the CKX pin, data transmit and receive are performed synchronously with this clock through TDX/RDX pins.

If data transmit and receive are selected with the CKX pin as clock output, the device generates the synchronous clock output at the CKX pin. The internal baud rate generator is used to produce the synchronous clock. Data transmit and receive are performed synchronously with this clock.

FRAMING FORMATS

The UART supports several serial framing formats *(Figure 13).* The format is selected using control bits in the ENU, ENUR and ENUI registers.

The first format (1, 1a, 1b, 1c) for data transmission (CHL0 = 1, CHL1 = 0) consists of Start bit, seven Data bits (excluding parity) and 7/8, one or two Stop bits. In applications using parity, the parity bit is generated and verified by hardware.

The second format (CHL0 = 0, CHL1 = 0) consists of one Start bit, eight Data bits (excluding parity) and 7/8, one or two Stop bits. Parity bit is generated and verified by hardware.

The third format for transmission (CHL0 = 0, CHL1 = 1) consists of one Start bit, nine Data bits and 7/8, one or two Stop bits. This format also supports the UART "ATTENTION" feature. When operating in this format, all eight bits of TBUF and RBUF are used for data. The ninth data bit is transmitted and received using two bits in the ENU and ENUR registers, called XBIT9 and RBIT9. RBIT9 is a read only bit. Parity is not generated or verified in this mode.

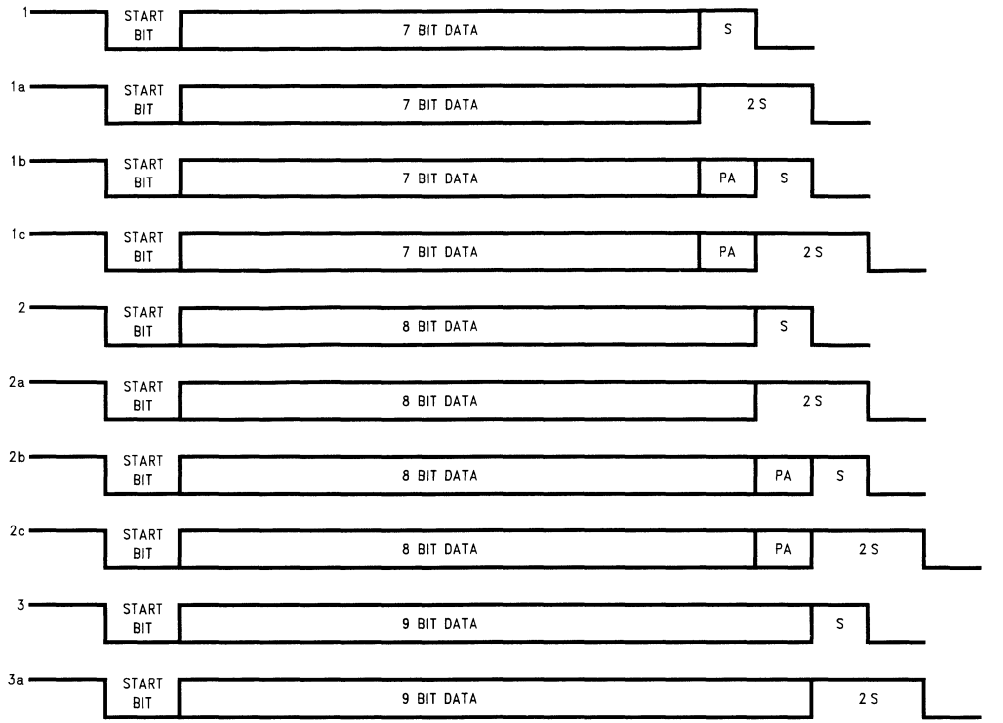
For any of the above framing formats, the last Stop bit can be programmed to be 7/8th of a bit in length. If two Stop bits are selected and the 7/8th bit is set (selected), the second Stop bit will be 7/8th of a bit in length.

The parity is enabled/disabled by PEN bit located in the ENU register. Parity is selected for 7- and 8-bit modes only. If parity is enabled (PEN = 1), the parity selection is then performed by PSEL0 and PSEL1 bits located in the ENU register.

Note that the XBIT9/PSEL0 bit located in the ENU register serves two mutually exclusive functions. This bit programs the ninth bit for transmission when the UART is operating with nine data bits per frame. There is no parity selection in this framing format. For other framing formats XBIT9 is not needed and the bit is PSEL0 used in conjunction with PSEL1 to select parity.

The frame formats for the receiver differ from the transmitter in the number of Stop bits required. The receiver only requires one Stop bit in a frame, regardless of the setting of the Stop bit selection bits in the control register. Note that an implicit assumption is made for full duplex UART operation that the framing formats are the same for the transmitter and receiver.

UART Operation (Continued)



TL/DD12872–13

FIGURE 13. Framing Formats

UART INTERRUPTS

The UART is capable of generating interrupts. Interrupts are generated on Receive Buffer Full and Transmit Buffer Empty. Both interrupts have individual interrupt vectors. Two bytes of program memory space are reserved for each interrupt vector. The two vectors are located at addresses 0xEC to 0xEF Hex in the program memory space. The interrupts can be individually enabled or disabled using Enable Transmit Interrupt (ETI) and Enable Receive Interrupt (ERI) bits in the ENUI register.

The interrupt from the transmitter is set pending, and remains pending, as long as both the TBMT and ETI bits are set. To remove this interrupt, software must either clear the ETI bit or write to the TBUF register (thus clearing the TBMT bit).

The interrupt from the receiver is set pending, and remains pending, as long as both the RBFL and ERI bits are set. To remove this interrupt, software must either clear the ERI bit or read from the RBUF register (thus clearing the RBFL bit).

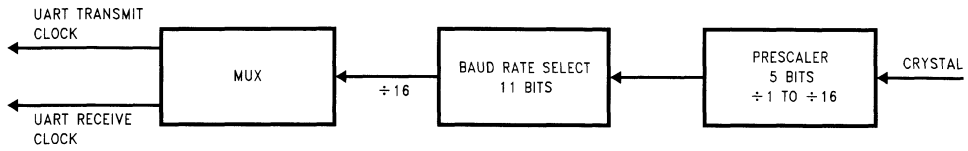
Baud Clock Generation

The clock inputs to the transmitter and receiver sections of the UART can be individually selected to come either from an external source at the CKX pin (port L, pin L1) or from a source selected in the PSR and BAUD registers. Internally, the basic baud clock is created from the oscillator frequency through a two-stage divider chain consisting of a 1–16 (increments of 0.5) prescaler and an 11-bit binary counter. *(Figure 14)* The divide factors are specified through two read/write registers shown in *Figure 15*. Note that the 11-bit Baud Rate Divisor spills over into the Prescaler Select Register (PSR). PSR is cleared upon reset.

As shown in Table III, a Prescaler Factor of 0 corresponds to NO CLOCK. NO CLOCK condition is the UART power down mode where the UART clock is turned off for power saving purpose. The user must also turn the UART clock off when a different baud rate is chosen.

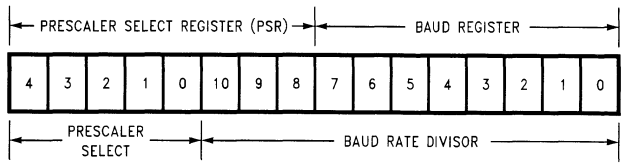
The correspondences between the 5-bit Prescaler Select and Prescaler factors are shown in Table III. There are many ways to calculate the two divisor factors, but one particularly effective method would be to achieve a 1.8432 MHz frequency coming out of the first stage. The 1.8432 MHz prescaler output is then used to drive the software programmable baud rate counter to create a x16 clock for the following baud rates: 110, 134.5, 150, 300, 600, 1200, 1800, 2400, 3600, 4800, 7200, 9600, 19200 and 38400 (Table IV). Other baud rates may be created by using appropriate divisors. The x16 clock is then divided by 16 to provide the rate for the serial shift registers of the transmitter and receiver.

Baud Clock Generation (Continued)



TL/DD12872-14

FIGURE 14. UART BAUD Clock Generation



TL/DD12872-15

FIGURE 15. UART BAUD Clock Divisor Registers

TABLE III. Prescaler Factors

Prescaler Select	Prescaler Factor	Prescaler Select	Prescaler Factor
00000	NO CLOCK	10000	8.5
00001	1	10001	9
00010	1.5	10010	9.5
00011	2	10011	10
00100	2.5	10100	10.5
00101	3	10101	11
00110	3.5	10110	11.5
00111	4	10111	12
01000	4.5	11000	12.5
01001	5	11001	13
01010	5.5	11010	13.5
01011	6	11011	14
01100	6.5	11100	14.5
01101	7	11101	15
01110	7.5	11110	15.5
01111	8	11111	16

TABLE IV. Baud Rate Divisors (1.8432 MHz Prescaler Output)

Baud Rate	Baud Rate Divisor − 1 (N-1)
110 (110.03)	1046
134.5 (134.58)	855
150	767
300	383
600	191
1200	95
1800	63
2400	47
3600	31
4800	23
7200	15
9600	11
19200	5
38400	2

Note: The entries in Table IV assume a prescaler output of 1.8432 MHz. In the asynchronous mode the baud rate could be as high as 625k.

As an example, considering the Asynchronous Mode and a CKI clock of 4.608 MHz, the prescaler factor selected is:

$$4.608/1.8432 = 2.5$$

The 2.5 entry is available in Table III. The 1.8432 MHz prescaler output is then used with proper Baud Rate Divisor (Table II) to obtain different baud rates. For a baud rate of 19200 e.g., the entry in Table IV is 5.

$$N - 1 = 5 \text{ (N} - 1 \text{ is the value from Table IV)}$$

$$N = 6 \text{ (N is the Baud Rate Divisor)}$$

$$\text{Baud Rate} = 1.8432 \text{ MHz}/(16 \times 6) = 19200$$

The divide by 16 is performed because in the asynchronous mode, the input frequency to the UART is 16 times the baud rate. The equation to calculate baud rates is given below.

The actual Baud Rate may be found from:

$$BR = Fc/(16 \times N \times P)$$

Baud Clock Generation (Continued)

Where:

BR is the Baud Rate

Fc is the CKI frequency

N is the Baud Rate Divisor (Table IV).

P is the Prescaler Divide Factor selected by the value in the Prescaler Select Register (Table III)

Note: In the Synchronous Mode, the divisor 16 is replaced by two.

Example:

Asynchronous Mode:

Crystal Frequency = 5 MHz

Desired baud rate = 9600

Using the above equation N × P can be calculated first.

$$N \times P = (5 \times 10^6)/(16 \times 9600) = 32.552$$

Now 32.552 is divided by each Prescaler Factor (Table III) to obtain a value closest to an integer. This factor happens to be 6.5 (P = 6.5).

$$N = 32.552/6.5 = 5.008 \ (N = 5)$$

The programmed value (from Table IV) should be 4 (N − 1).

Using the above values calculated for N and P:

$$BR = (5 \times 10^6)/(16 \times 5 \times 6.5) = 9615.384$$

$$\% \text{ error} = (9615.385 - 9600)/9600 = 0.16$$

Effect of HALT/IDLE

The UART logic is reinitialized when either the HALT or IDLE modes are entered. This reinitialization sets the TBMT flag and resets all read only bits in the UART control and status registers. Read/Write bits remain unchanged. The Transmit Buffer (TBUF) is not affected, but the Transmit Shift register (TSFT) bits are set to one. The receiver registers RBUF and RSFT are not affected.

The device will exit from the HALT/IDLE modes when the Start bit of a character is detected at the RDX (L3) pin. This feature is obtained by using the Multi-Input Wake Up scheme provided on the device.

Before entering the HALT or IDLE modes the user program must select the Wake Up source to be on the RDX pin. This selection is done by setting bit 3 of WKEN (Wake Up Enable) register. The Wake Up trigger condition is then selected to be high to low transition. This is done via the WKEDG register (Bit 3 is one.)

If the device is halted and crystal oscillator is used, the Wake Up signal will not start the chip running immediately because of the finite start up time requirement of the crystal oscillator. The idle timer (T0) generates a fixed (256 t_c) delay to ensure that the oscillator has indeed stabilized before allowing the device to execute code. The user has to consider this delay when data transfer is expected immediately after exiting the HALT mode.

Diagnostic

Bits CHARL0 and CHARL1 in the ENU register provide a loopback feature for diagnostic testing of the UART. When these bits are set to one, the following occur: The receiver input pin (RDX) is internally connected to the transmitter output pin (TDX); the output of the Transmitter Shift Register is "looped back" into the Receive Shift Register input. In this mode, data that is transmitted is immediately received. This feature allows the processor to verify the transmit and receive data paths of the UART.

Note that the framing format for this mode is the nine bit format; one Start bit, nine data bits, and 7/8, one or two Stop bits. Parity is not generated or verified in this mode.

Attention Mode

The UART Receiver section supports an alternate mode of operation, referred to as ATTENTION Mode. This mode of operation is selected by the ATTN bit in the ENUR register. The data format for transmission must also be selected as having nine Data bits and either 7/8, one or two Stop bits.

The ATTENTION mode of operation is intended for use in networking the device with other processors. Typically in such environments the messages consists of device addresses, indicating which of several destinations should receive them, and the actual data. This Mode supports a scheme in which addresses are flagged by having the ninth bit of the data field set to a 1. If the ninth bit is reset to a zero the byte is a Data byte.

While in ATTENTION mode, the UART monitors the communication flow, but ignores all characters until an address character is received. Upon receiving an address character, the UART signals that the character is ready by setting the RBFL flag, which in turn interrupts the processor if UART Receiver interrupts are enabled. The ATTN bit is also cleared automatically at this point, so that data characters as well as address characters are recognized. Software examines the contents of the RBUF and responds by deciding either to accept the subsequent data stream (by leaving the ATTN bit reset) or to wait until the next address character is seen (by setting the ATTN bit again).

Operation of the UART Transmitter is not affected by selection of this Mode. The value of the ninth bit to be transmitted is programmed by setting XBIT9 appropriately. The value of the ninth bit received is obtained by reading RBIT9. Since this bit is located in ENUR register where the error flags reside, a bit operation on it will reset the error flags.

Comparators

The device contains one differential comparator, with a pair of inputs (positive and negative) and an output. Ports I1–I3 is used for the comparator. The following is the Port I assignment:

- I1 Comparator negative input
- I2 Comparator positive input
- I3 Comparator output

A Comparator Select Register (CMPSL) is used to enable the comparator, read the output of the comparator internally, and enable the output of the comparator to the pins. Two control bits (enable and output enable) and one result bit are associated with the comparator. The comparator result bits (CMP1RD) is read only bits which will read as zero if the associated comparator is not enabled. The Comparator Select Register is cleared with reset, resulting in the comparator being disabled. The comparator should also be disabled before entering either the HALT or IDLE modes in order to save power. The configuration of the CMPSL register is as follows:

Comparators (Continued)

CMPSL REGISTER (ADDRESS X'00B7)

The CMPSL register contains the following bits:

CMP1EN Enable comparator

CMP1RD Comparator result (this is a read only bit, which will read as 0 if the comparator is not enabled)

CMP10E Selects pin I3 as comparator output provided that CMPIEN is set to enable the comparator

Unused	Unused	Unused	Unused	CMP10E	CMP1RD	CMP1EN	Unused

Bit 7 Bit 0

Note that the two unused bits of CMPSL may be used as software flags.

Comparator output has the same spec as Ports L and G except that the rise and fall times are symmetrical.

Interrupts

The devices support a vectored interrupt scheme. It supports a total of fourteen interrupt sources. The following table lists all the possible device interrupt sources, their arbitration ranking and the memory locations reserved for the interrupt vector for each source.

Two bytes of program memory space are reserved for each interrupt source. All interrupt sources except the software interrupt are maskable. Each of the maskable interrupts have an Enable bit and a Pending bit. A maskable interrupt is active if its associated enable and pending bits are set. If GIE = 1 and an interrupt is active, then the processor will be interrupted as soon as it is ready to start executing an instruction except if the above conditions happen during the Software Trap service routine. This exception is described in the Software Trap sub-section.

The interruption process is accomplished with the INTR instruction (opcode 00), which is jammed inside the Instruction Register and replaces the opcode about to be executed. The following steps are performed for every interrupt:

1. The GIE (Global Interrupt Enable) bit is reset.
2. The address of the instruction about to be executed is pushed into the stack.
3. The PC (Program Counter) branches to address 00FF. This procedure takes 7 t_C cycles to execute.

Arbitration Ranking	Source	Description	Vector Address Hi-Low Byte
(1) Highest	Software	INTR Instruction	0yFE–0yFF
(2)	Reserved	for Future Use	0yFC–0yFD
(3)	External	Pin G0 Edge	0yFA–0yFB
(4)	Timer T0	Underflow	0yF8–0yF9
(5)	Timer T1	T1A/Underflow	0yF6–0yF7
(6)	Timer T1	T1B	0yF4–0yF5
(7)	MICROWIRE/PLUS	BUSY Goes Low	0yF2–0yF3
(8)	Reserved	for Future Use	0yF0–0yF1
(9)	UART	Receive	0yEE–0yEF
(10)	UART	Transmit	0yEC–0yED
(11)	Timer T2	T2A/Underflow	0yEA–0yEB
(12)	Timer T2	T2B	0yE8–0yE9
(13)	Timer T3	T3A/Underflow	0yE6–0yE7
(14)	Timer T3	T3B	0yE4–0yE5
(15)	Port L/Wake Up	Port L Edge	0yE2–0yE3
(16) Lowest	Default	VIS Instr. Execution without Any Interrupts	0yE0–0yE1

y is VIS page, y ≠ 0.

Interrupts (Continued)

At this time, since GIE = 0, other maskable interrupts are disabled. The user is now free to do whatever context switching is required by saving the context of the machine in the stack with PUSH instructions. The user would then program a VIS (Vector Interrupt Select) instruction in order to branch to the interrupt service routine of the highest priority interrupt enabled and pending at the time of the VIS. Note that this is not necessarily the interrupt that caused the branch to address location 00FF Hex prior to the context switching.

Thus, if an interrupt with a higher rank than the one which caused the interruption becomes active before the decision of which interrupt to service is made by the VIS, then the interrupt with the higher rank will override any lower ones and will be acknowledged. The lower priority interrupt(s) are still pending, however, and will cause another interrupt immediately following the completion of the interrupt service routine associated with the higher priority interrupt just serviced. This lower priority interrupt will occur immediately following the RETI (Return from Interrupt) instruction at the end of the interrupt service routine just completed.

Inside the interrupt service routine, the associated pending bit has to be cleared by software. The RETI (Return from Interrupt) instruction at the end of the interrupt service routine will set the GIE (Global Interrupt Enable) bit, allowing the processor to be interrupted again if another interrupt is active and pending.

The VIS instruction looks at all the active interrupts at the time it is executed and performs an indirect jump to the beginning of the service routine of the one with the highest rank.

The addresses of the different interrupt service routines, called vectors, are chosen by the user and stored in ROM in a table starting at 01E0 (assuming that VIS is located between 00FF and 01DF). The vectors are 15-bit wide and therefore occupy 2 ROM locations.

VIS and the vector table must be located in the same 256-byte block (0y00 to 0yFF) except if VIS is located at the last address of a block. In this case, the table must be in the next block. The vector table cannot be inserted in the first 256-byte block (y ≠ 0).

The vector of the maskable interrupt with the lowest rank is located at 0yE0 (Hi-Order byte) and 0yE1 (Lo-Order byte) and so forth in increasing rank number. The vector of the maskable interrupt with the highest rank is located at 0yFA (Hi-Order byte) and 0yFB (Lo-Order byte).

The Software Trap has the highest rank and its vector is located at 0yFE and 0yFF.

If, by accident, a VIS gets executed and no interrupt is active, then the PC (Program Counter) will branch to a vector located at 0yE0–0yE1.

WARNING

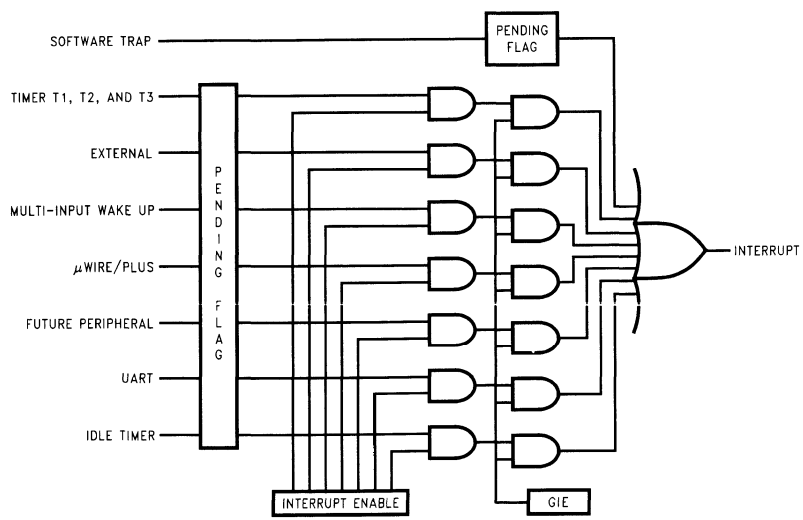
A Default VIS interrupt handler routine must be present. As a minimum, this handler should confirm that the GIE bit is cleared (this indicates that the interrupt sequence has been taken), take care of any required housekeeping, restore context and return. Some sort of Warm Restart procedure should be implemented. These events can occur without any error on the part of the system designer or programmer.

Note: There is always the possibility of an interrupt occurring during an instruction which is attempting to reset the GIE bit or any other interrupt enable bit. If this occurs when a single cycle instruction is being used to reset the interrupt enable bit, the interrupt enable bit will be reset but an interrupt may still occur. This is because interrupt processing is started at the same time as the interrupt bit is being reset. To avoid this scenario, the user should always use a two, three, or four cycle instruction to reset interrupt enable bits.

Figure 16 shows the Interrupt block diagram.

SOFTWARE TRAP

The Software Trap (ST) is a special kind of non-maskable interrupt which occurs when the INTR instruction (used to acknowledge interrupts) is fetched from ROM and placed inside the instruction register. This may happen when the PC is pointing beyond the available ROM address space or when the stack is over-popped.



TL/DD12872–16

FIGURE 16. Interrupt Block Diagram

Interrupts (Continued)

When an ST occurs, the user can re-initialize the stack pointer and do a recovery procedure (similar to reset, but not necessarily containing all of the same initialization procedures) before restarting.

The occurrence of an ST is latched into the ST pending bit. The GIE bit is not affected and the ST pending bit **(not accessible by the user)** is used to inhibit other interrupts and to direct the program to the ST service routine with the VIS instruction. The RPND instruction is used to clear the software interrupt pending bit. This pending bit is also cleared on reset.

The ST has the highest rank among all interrupts.

Nothing (except another ST) can interrupt an ST being serviced.

WATCHDOG

The devices contain a WATCHDOG and clock monitor. The WATCHDOG is designed to detect the user program getting stuck in infinite loops resulting in loss of program control or "runaway" programs. The Clock Monitor is used to detect the absence of a clock or a very slow clock below a specified rate on the CKI pin.

The WATCHDOG consists of two independent logic blocks: WD UPPER and WD LOWER. WD UPPER establishes the upper limit on the service window and WD LOWER defines the lower limit of the service window.

Servicing the WATCHDOG consists of writing a specific value to a WATCHDOG Service Register named WDSVR which is memory mapped in the RAM. This value is composed of three fields, consisting of a 2-bit Window Select, a 5-bit Key Data field, and the 1-bit Clock Monitor Select field. Table V shows the WDSVR register.

The lower limit of the service window is fixed at 2048 instruction cycles. Bits 7 and 6 of the WDSVR register allow the user to pick an upper limit of the service window.

Table VI shows the four possible combinations of lower and upper limits for the WATCHDOG service window. This flexibility in choosing the WATCHDOG service window prevents any undue burden on the user software.

Bits 5, 4, 3, 2 and 1 of the WDSVR register represent the 5-bit Key Data field. The key data is fixed at 01100. Bit 0 of the WDSVR Register is the Clock Monitor Select bit.

TABLE V. WATCHDOG Service Register (WDSVR)

Window Select		Key Data					Clock Monitor
X	X	0	1	1	0	0	Y
7	6	5	4	3	2	1	0

TABLE VI. WATCHDOG Service Window Select

WDSVR Bit 7	WDSVR Bit 6	Service Window (Lower-Upper Limits)
0	0	2k–8k t_C Cycles
0	1	2k–16k t_C Cycles
1	0	2k–32k t_C Cycles
1	1	2k–64k t_C Cycles

Clock Monitor

The Clock Monitor aboard the device can be selected or deselected under program control. The Clock Monitor is guaranteed not to reject the clock if the instruction cycle clock ($1/t_C$) is greater or equal to 10 kHz. This equates to a clock input rate on CKI of greater or equal to 100 kHz.

WATCHDOG Operation

The WATCHDOG and Clock Monitor are disabled during reset. The device comes out of reset with the WATCHDOG armed, the WATCHDOG Window Select bits (bits 6, 7 of the WDSVR Register) set, and the Clock Monitor bit (bit 0 of the WDSVR Register) enabled. Thus, a Clock Monitor error will occur after coming out of reset, if the instruction cycle clock frequency has not reached a minimum specified value, including the case where the oscillator fails to start.

The WDSVR register can be written to only once after reset and the key data (bits 5 through 1 of the WDSVR Register) must match to be a valid write. This write to the WDSVR register involves two irrevocable choices: (i) the selection of the WATCHDOG service window (ii) enabling or disabling of the Clock Monitor. Hence, the first write to WDSVR Register involves selecting or deselecting the Clock Monitor, select the WATCHDOG service window and match the WATCHDOG key data. Subsequent writes to the WDSVR register will compare the value being written by the user to the WATCHDOG service window value and the key data (bits 7 through 1) in the WDSVR Register. Table VII shows the sequence of events that can occur.

The user must service the WATCHDOG at least once before the upper limit of the service window expires. The WATCHDOG may not be serviced more than once in every lower limit of the service window. The user may service the WATCHDOG as many times as wished in the time period between the lower and upper limits of the service window. The first write to the WDSVR Register is also counted as a WATCHDOG service.

The WATCHDOG has an output pin associated with it. This is the WDOUT pin, on pin 1 of the port G. WDOUT is active low. The WDOUT pin is in the high impedance state in the inactive state. Upon triggering the WATCHDOG, the logic will pull the WDOUT (G1) pin low for an additional 16 t_C–32 t_C cycles after the signal level on WDOUT pin goes below the lower Schmitt trigger threshold. After this delay, the device will stop forcing the WDOUT output low.

TABLE VII. WATCHDOG Service Actions

Key Data	Window Data	Clock Monitor	Action
Match	Match	Match	Valid Service: Restart Service Window
Don't Care	Mismatch	Don't Care	Error: Generate WATCHDOG Output
Mismatch	Don't Care	Don't Care	Error: Generate WATCHDOG Output
Don't Care	Don't Care	Mismatch	Error: Generate WATCHDOG Output

WATCHDOG Operation (Continued)

The WATCHDOG service window will restart when the WDOUT pin goes high. It is recommended that the user tie the WDOUT pin back to V_{CC} through a resistor in order to pull WDOUT high.

A WATCHDOG service while the WDOUT signal is active will be ignored. The state of the WDOUT pin is not guaranteed on reset, but if it powers up low then the WATCHDOG will time out and WDOUT will enter high impedance state.

The Clock Monitor forces the G1 pin low upon detecting a clock frequency error. The Clock Monitor error will continue until the clock frequency has reached the minimum specified value, after which the G1 output will enter the high impedance TRI-STATE mode following 16 t_c–32 t_c clock cycles. The Clock Monitor generates a continual Clock Monitor error if the oscillator fails to start, or fails to reach the minimum specified frequency. The specification for the Clock Monitor is as follows:

$1/t_c > 10$ kHz—No clock rejection.

$1/t_c < 10$ Hz—Guaranteed clock rejection.

WATCHDOG AND CLOCK MONITOR SUMMARY

The following salient points regarding the WATCHDOG and CLOCK MONITOR should be noted:

- Both the WATCHDOG and CLOCK MONITOR detector circuits are inhibited during RESET.
- Following RESET, the WATCHDOG and CLOCK MONITOR are both enabled, with the WATCHDOG having the maximum service window selected.
- The WATCHDOG service window and CLOCK MONITOR enable/disable option can only be changed once, during the initial WATCHDOG service following RESET.
- The initial WATCHDOG service must match the key data value in the WATCHDOG Service register WDSVR in order to avoid a WATCHDOG error.
- Subsequent WATCHDOG services must match all three data fields in WDSVR in order to avoid WATCHDOG errors.
- The correct key data value cannot be read from the WATCHDOG Service register WDSVR. Any attempt to read this key data value of 01100 from WDSVR will read as key data value of all 0's.
- The WATCHDOG detector circuit is inhibited during both the HALT and IDLE modes.
- The CLOCK MONITOR detector circuit is active during both the HALT and IDLE modes. Consequently, the COP888 inadvertently entering the HALT mode will be detected as a CLOCK MONITOR error (provided that the CLOCK MONITOR enable option has been selected by the program).
- With the single-pin R/C oscillator mask option selected and the CLKDLY bit reset, the WATCHDOG service window will resume following HALT mode from where it left off before entering the HALT mode.
- With the crystal oscillator mask option selected, or with the single-pin R/C oscillator mask option selected and the CLKDLY bit set, the WATCHDOG service window will be set to its selected value from WDSVR following HALT. Consequently, the WATCHDOG should not be serviced for at least 2048 instruction cycles following HALT, but must be serviced within the selected window to avoid a WATCHDOG error.
- The IDLE timer T0 is not initialized with RESET.
- The user can sync in to the IDLE counter cycle with an IDLE counter (T0) interrupt or by monitoring the T0PND flag. The T0PND flag is set whenever the thirteenth bit of the IDLE counter toggles (every 4096 instruction cycles). The user is responsible for resetting the T0PND flag.
- A hardware WATCHDOG service occurs just as the device exits the IDLE mode. Consequently, the WATCHDOG should not be serviced for at least 2048 instruction cycles following IDLE, but must be serviced within the selected window to avoid a WATCHDOG error.
- Following RESET, the initial WATCHDOG service (where the service window and the CLOCK MONITOR enable/disable must be selected) may be programmed anywhere within the maximum service window (65,536 instruction cycles) initialized by RESET. Note that this initial WATCHDOG service may be programmed within the initial 2048 instruction cycles without causing a WATCHDOG error.

Detection of Illegal Conditions

The device can detect various illegal conditions resulting from coding errors, transient noise, power supply voltage drops, runaway programs, etc.

Reading of undefined ROM gets zeros. The opcode for software interrupt is zero. If the program fetches instructions from undefined ROM, this will force a software interrupt, thus signaling that an illegal condition has occurred.

The subroutine stack grows down for each call (jump to subroutine), interrupt, or PUSH, and grows up for each return or POP. The stack pointer is initialized to RAM location 06F Hex during reset. Consequently, if there are more returns than calls, the stack pointer will point to addresses 070 and 071 Hex (which are undefined RAM). Undefined RAM from addresses 070 to 07F (Segment 0), 140 to 17F (Segment 1), and all other segments (i.e., Segments 3 . . . etc.) is read as all 1's, which in turn will cause the program to return to address 7FFF Hex. This is an undefined ROM location and the instruction fetched (all 0's) from this location will generate a software interrupt signaling an illegal condition.

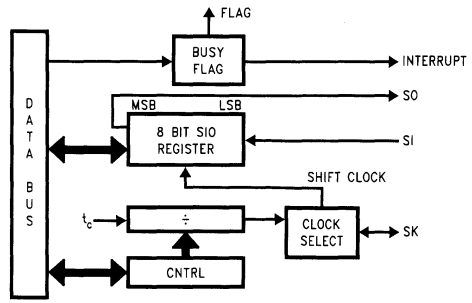
Thus, the chip can detect the following illegal conditions:

1. Executing from undefined ROM
2. Over "POP"ing the stack by having more returns than calls.

When the software interrupt occurs, the user can re-initialize the stack pointer and do a recovery procedure before restarting (this recovery program is probably similar to that following reset, but might not contain the same program initialization procedures). The recovery program should reset the software interrupt pending bit using the RPND instruction.

MICROWIRE/PLUS

MICROWIRE/PLUS is a serial synchronous communications interface. The MICROWIRE/PLUS capability enables the device to interface with any of National Semiconductor's MICROWIRE peripherals (i.e. A/D converters, display drivers, E^2PROMs etc.) and with other microcontrollers which support the MICROWIRE interface. It consists of an 8-bit serial shift register (SIO) with serial data input (SI), serial data output (SO) and serial shift clock (SK). *Figure 17* shows a block diagram of the MICROWIRE/PLUS logic.



TL/DD12872–17

FIGURE 17. MICROWIRE/PLUS Block Diagram

The shift clock can be selected from either an internal source or an external source. Operating the MICROWIRE/PLUS arrangement with the internal clock source is called the Master mode of operation. Similarly, operating the MICROWIRE/PLUS arrangement with an external shift clock is called the Slave mode of operation.

The CNTRL register is used to configure and control the MICROWIRE/PLUS mode. To use the MICROWIRE/PLUS, the MSEL bit in the CNTRL register is set to one. In the master mode, the SK clock rate is selected by the two bits, SL0 and SL1, in the CNTRL register. Table VIII details the different clock rates that may be selected.

TABLE VIII. MICROWIRE/PLUS Master Mode Clock Select

SL1	SL0	SK
0	0	$2 \times t_c$
0	1	$4 \times t_c$
1	x	$8 \times t_c$

Where t_c is the instruction cycle clock

MICROWIRE/PLUS (Continued)

MICROWIRE/PLUS OPERATION

Setting the BUSY bit in the PSW register causes the MICROWIRE/PLUS to start shifting the data. It gets reset when eight data bits have been shifted. The user may reset the BUSY bit by software to allow less than 8 bits to shift. If enabled, an interrupt is generated when eight data bits have been shifted. The device may enter the MICROWIRE/PLUS mode either as a Master or as a Slave. *Figure 18* shows how two devices, microcontrollers and several peripherals may be interconnected using the MICROWIRE/PLUS arrangements.

Warning

The SIO register should only be loaded when the SK clock is low. Loading the SIO register while the SK clock is high will result in undefined data in the SIO register. SK clock is normally low when not shifting.

Setting the BUSY flag when the input SK clock is high in the MICROWIRE/PLUS slave mode may cause the current SK clock for the SIO shift register to be narrow. For safety, the BUSY flag should only be set when the input SK clock is low.

MICROWIRE/PLUS Master Mode Operation

In the MICROWIRE/PLUS Master mode of operation the shift clock (SK) is generated internally by the device. The MICROWIRE Master always initiates all data exchanges. The MSEL bit in the CNTRL register must be set to enable the SO and SK functions onto the G Port. The SO and SK pins must also be selected as outputs by setting appropriate bits in the Port G configuration register. Table IX summarizes the bit settings required for Master mode of operation.

MICROWIRE/PLUS Slave Mode Operation

In the MICROWIRE/PLUS Slave mode of operation the SK clock is generated by an external source. Setting the MSEL bit in the CNTRL register enables the SO and SK functions onto the G Port. The SK pin must be selected as an input and the SO pin is selected as an output pin by setting and resetting the appropriate bit in the Port G configuration register. Table IX summarizes the settings required to enter the Slave mode of operation.

The user must set the BUSY flag immediately upon entering the Slave mode. This will ensure that all data bits sent by the Master will be shifted properly. After eight clock pulses the BUSY flag will be cleared and the sequence may be repeated.

Alternate SK Phase Operation

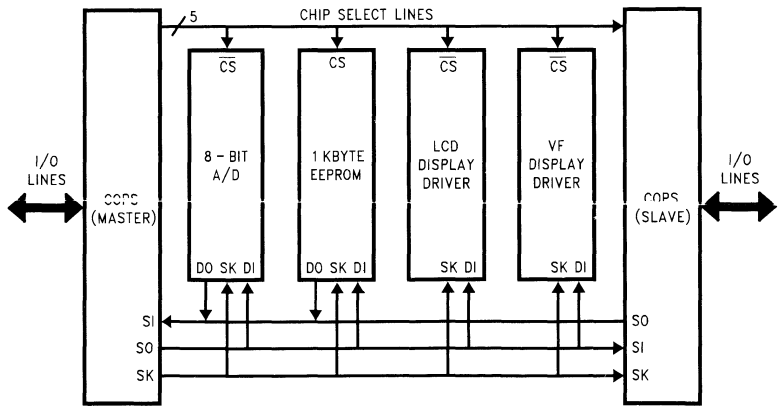
The device allows either the normal SK clock or an alternate phase SK clock to shift data in and out of the SIO register. In both the modes the SK is normally low. In the normal mode data is shifted in on the rising edge of the SK clock and the data is shifted out on the falling edge of the SK clock. The SIO register is shifted on each falling edge of the SK clock. In the alternate SK phase operation, data is shifted in on the falling edge of the SK clock and shifted out on the rising edge of the SK clock.

A control flag, SKSEL, allows either the normal SK clock or the alternate SK clock to be selected. Resetting SKSEL causes the MICROWIRE/PLUS logic to be clocked from the normal SK signal. Setting the SKSEL flag selects the alternate SK clock. The SKSEL is mapped into the G6 configuration bit. The SKSEL flag will power up in the reset condition, selecting the normal SK signal.

TABLE IX. MICROWIRE/PLUS Mode Selection

G4 (SO) Config. Bit	G5 (SK) Config. Bit	G4 Fun.	G5 Fun.	Operation
1	1	SO	Int. SK	MICROWIRE/PLUS Master
0	1	TRI-STATE	Int. SK	MICROWIRE/PLUS Master
1	0	SO	Ext. SK	MICROWIRE/PLUS Slave
0	0	TRI-STATE	Ext. SK	MICROWIRE/PLUS Slave

Note: This table assumes that the control flag MSEL is set.



TL/DD12872-18

FIGURE 18. MICROWIRE/PLUS Application

Memory Map

All RAM, ports and registers (except A and PC) are mapped into data memory address space.

Address S/ADD REG	Contents
0000 to 006F	On-Chip RAM bytes (112 bytes)
0070 to 007F	Unused RAM Address Space (Reads As All Ones)
xx80 to xxAF	Unused RAM Address Space (Reads Undefined Data)
xxB0	Timer T3 Lower Byte
xxB1	Timer T3 Upper Byte
xxB2	Timer T3 Autoload Register T3RA Lower Byte
xxB3	Timer T3 Autoload Register T3RA Upper Byte
xxB4	Timer T3 Autoload Register T3RB Lower Byte
xxB5	Timer T3 Autoload Register T3RB Upper Byte
xxB6	Timer T3 Control Register
xxB7	Comparator Select Register (CMPSL)
xxB8	UART Transmit Buffer (TBUF)
xxB9	UART Receive Buffer (RBUF)
xxBA	UART Control and Status Register (ENU)
xxBB	UART Receive Control and Status Register (ENUR)
xxBC	UART Interrupt and Clock Source Register (ENUI)
xxBD	UART Baud Register (BAUD)
xxBE	UART Prescale Select Register (PSR)
xxBF	Reserved for UART
xxC0	Timer T2 Lower Byte
xxC1	Timer T2 Upper Byte
xxC2	Timer T2 Autoload Register T2RA Lower Byte
xxC3	Timer T2 Autoload Register T2RA Upper Byte
xxC4	Timer T2 Autoload Register T2RB Lower Byte
xxC5	Timer T2 Autoload Register T2RB Upper Byte
xxC6	Timer T2 Control Register
xxC7	WATCHDOG Service Register (Reg:WDSVR)
xxC8	MIWU Edge Select Register (Reg:WKEDG)
xxC9	MIWU Enable Register (Reg:WKEN)
xxCA	MIWU Pending Register (Reg:WKPND)
xxCB	Reserved
xxCC	Reserved
xxCD to xxCF	Reserved

Address S/ADD REG	Contents
xxD0	Port L Data Register
xxD1	Port L Configuration Register
xxD2	Port L Input Pins (Read Only)
xxD3	Reserved for Port L
xxD4	Port G Data Register
xxD5	Port G Configuration Register
xxD6	Port G Input Pins (Read Only)
xxD7	Port I Input Pins (Read Only)
xxD8	Reserved
xxD9	Reserved
xxDA	Reserved
xxDB	Reserved
xxDC	Port D
xxDD to DF	Reserved for Port D
xxE0 to xxE5	Reserved for EE Control Registers
xxE6	Timer T1 Autoload Register T1RB Lower Byte
xxE7	Timer T1 Autoload Register T1RB Upper Byte
xxE8	ICNTRL Register
xxE9	MICROWIRE/PLUS Shift Register
xxEA	Timer T1 Lower Byte
xxEB	Timer T1 Upper Byte
xxEC	Timer T1 Autoload Register T1RA Lower Byte
xxED	Timer T1 Autoload Register T1RA Upper Byte
xxEE	CNTRL Control Register
xxEF	PSW Register
xxF0 to FB	On-Chip RAM Mapped as Registers
xxFC	X Register
xxFD	SP Register
xxFE	B Register
xxFF	S Register
0100–017F	On-Chip 128 RAM Bytes

Note: Reading memory locations 0070H–007FH (Segment 0) will return all ones. Reading unused memory locations 0080H–00AFH (Segment 0) will return undefined data. Reading memory locations from other Segments (i.e., Segment 2, Segment 3, ... etc.) will return all ones.

Addressing Modes

There are ten addressing modes, six for operand addressing and four for transfer of control.

OPERAND ADDRESSING MODES

Register Indirect

This is the "normal" addressing mode. The operand is the data memory addressed by the B pointer or X pointer.

Register Indirect (with auto post increment or decrement of pointer)

This addressing mode is used with the LD and X instructions. The operand is the data memory addressed by the B pointer or X pointer. This is a register indirect mode that automatically post increments or decrements the B or X register after executing the instruction.

Direct

The instruction contains an 8-bit address field that directly points to the data memory for the operand.

Immediate

The instruction contains an 8-bit immediate field as the operand.

Short Immediate

This addressing mode is used with the Load B Immediate instruction. The instruction contains a 4-bit immediate field as the operand.

Indirect

This addressing mode is used with the LAID instruction. The contents of the accumulator are used as a partial address (lower 8 bits of PC) for accessing a data operand from the program memory.

TRANSFER OF CONTROL ADDRESSING MODES

Relative

This mode is used for the JP instruction, with the instruction field being added to the program counter to get the new program location. JP has a range from −31 to +32 to allow a 1-byte relative jump (JP + 1 is implemented by a NOP instruction). There are no "pages" when using JP, since all 15 bits of PC are used.

Absolute

This mode is used with the JMP and JSR instructions, with the instruction field of 12 bits replacing the lower 12 bits of the program counter (PC). This allows jumping to any location in the current 4k program memory segment.

Absolute Long

This mode is used with the JMPL and JSRL instructions, with the instruction field of 15 bits replacing the entire 15 bits of the program counter (PC). This allows jumping to any location up to 32k in the program memory space.

Indirect

This mode is used with the JID instruction. The contents of the accumulator are used as a partial address (lower 8 bits of PC) for accessing a location in the program memory. The contents of this program memory location serve as a partial address (lower 8 bits of PC) for the jump to the next instruction.

Note: The VIS is a special case of the Indirect Transfer of Control addressing mode, where the double byte vector associated with the interrupt is transferred from adjacent addresses in the program memory into the program counter (PC) in order to jump to the associated interrupt service routine.

Instruction Set

Register and Symbol Definition

Registers	
A	8-Bit Accumulator Register
B	8-Bit Address Register
X	8-Bit Address Register
SP	8-Bit Stack Pointer Register
PC	15-Bit Program Counter Register
PU	Upper 7 Bits of PC
PL	Lower 8 Bits of PC
C	1 Bit of PSW Register for Carry
HC	1 Bit of PSW Register for Half Carry
GIE	1 Bit of PSW Register for Global Interrupt Enable
VU	Interrupt Vector Upper Byte
VL	Interrupt Vector Lower Byte

Symbols	
[B]	Memory Indirectly Addressed by B Register
[X]	Memory Indirectly Addressed by X Register
MD	Direct Addressed Memory
Mem	Direct Addressed Memory or [B]
Meml	Direct Addressed Memory or [B] or Immediate Data
Imm	8-Bit Immediate Data
Reg	Register Memory: Addresses F0 to FF (Includes B, X and SP)
Bit	Bit Number (0 to 7)
←	Loaded with
⟷	Exchanged with

Instruction Set (Continued)

INSTRUCTION SET

ADD	A,Meml	ADD	A ← A + Meml
ADC	A,Meml	ADD with Carry	A ← A + Meml + C, C ← Carry, HC ← Half Carry
SUBC	A,Meml	Subtract with Carry	A ← A − Meml + C, C ← Carry, HC ← Half Carry
AND	A,Meml	Logical AND	A ← A and $\overline{\text{Meml}}$
ANDSZ	A,Imm	Logical AND Immed., Skip if Zero	Skip next if (A and Imm) = 0
OR	A,Meml	Logical OR	A ← A or Meml
XOR	A,Meml	Logical EXclusive OR	A ← A xor Meml
IFEQ	MD,Imm	IF EQual	Compare MD and Imm, Do next if MD = Imm
IFEQ	A,Meml	IF EQual	Compare A and Meml, Do next if A = Meml
IFNE	A,Meml	IF Not Equal	Compare A and Meml, Do next if A ≠ Meml
IFGT	A,Meml	IF Greater Than	Compare A and Meml, Do next if A > Meml
IFBNE	#	If B Not Equal	Do next if lower 4 bits of B ≠ Imm
DRSZ	Reg	Decrement Reg., Skip if Zero	Reg ← Reg − 1, Skip if Reg = 0
SBIT	#,Mem	Set BIT	1 to bit, Mem (bit = 0 to 7 immediate)
RBIT	#,Mem	Reset BIT	0 to bit, Mem
IFBIT	#,Mem	IF BIT	If bit in A or Mem is true do next instruction
RPND		Reset PeNDing Flag	Reset Software Interrupt Pending Flag
X	A,Mem	EXchange A with Memory	A ⟷ Mem
X	A,[X]	EXchange A with Memory [X]	A ⟷ [X]
LD	A,Meml	LoaD A with Memory	A ← Meml
LD	A,[X]	LoaD A with Memory [X]	A ← [X]
LD	B,Imm	LoaD B with Immed.	B ← Imm
LD	Mem,Imm	LoaD Memory Immed.	Mem ← Imm
LD	Reg,Imm	LoaD Register Memory Immed.	Reg ← Imm
X	A, [B ±]	EXchange A with Memory [B]	A ⟷ [B], (B ← B ±1)
X	A, [X ±]	EXchange A with Memory [X]	A ⟷ [X], (X ← ±1)
LD	A, [B±]	LoaD A with Memory [B]	A ← [B], (B ← B ±1)
LD	A, [X±]	LoaD A with Memory [X]	A ← [X], (X ← X±1)
LD	[B±],Imm	LoaD Memory [B] Immed.	[B] ← Imm, (B ← B±1)
CLR	A	CLeaR A	A ← 0
INC	A	INCrement A	A ← A + 1
DEC	A	DECrementA	A ← A − 1
LAID		Load A InDirect from ROM	A ← ROM (PU,A)
DCOR	A	Decimal CORrect A	A ← BCD correction of A (follows ADC, SUBC)
RRC	A	Rotate A Right thru C	C → A7 → . . . → A0 → C
RLC	A	Rotate A Left thru C	C ← A7 ← . . . ← A0 ← C
SWAP	A	SWAP nibbles of A	A7 . . . A4 ⟷ A3 . . . A0
SC		Set C	C ← 1, HC ← 1
RC		Reset C	C ← 0, HC ← 0
IFC		IF C	IF C is true, do next instruction
IFNC		IF Not C	If C is not true, do next instruction
POP	A	POP the stack into A	SP ← SP + 1, A ← [SP]
PUSH	A	PUSH A onto the stack	[SP] ← A, SP ← SP − 1
VIS		Vector to Interrupt Service Routine	PU ← [VU], PL ← [VL]
JMPL	Addr.	Jump absolute Long	PC ← ii (ii = 15 bits, 0k to 32k)
JMP	Addr.	Jump absolute	PC9 . . . 0 ← i (i = 12 bits)
JP	Disp.	Jump relative short	PC ← PC + r (r is −31 to +32, except 1)
JSRL	Addr.	Jump SubRoutine Long	[SP] ← PL, [SP−1] ← PU,SP−2, PC ← ii
JSR	Add.	Jump SubRoutine	[SP] ← PL, [SP−1] ← PU,SP−2, PC9 . . . 0 ← i
JID		Jump InDirect	PL ← ROM (PU,A)
RET		RETurn from subroutine	SP + 2, PL ← [SP], PU ← [SP−1]
RETSK		RETurn and SKip	SP + 2, PL ← [SP],PU ← [SP−1]
RETI		RETurn from Interrupt	SP + 2, PL ← [SP],PU ← [SP−1],GIE ← 1
INTR		Generate an Interrupt	[SP] ← PL, [SP−1] ← PU, SP−2, PC ← 0FF
NOP		No OPeration	PC ← PC + 1

Instruction Execution Time

Most instructions are single byte (with immediate addressing mode instructions taking two bytes).

Most single byte instructions take one cycle time to execute.

Skipped instructions require x number of cycles to be skipped, where x equals the number of bytes in the skipped instruction opcode.

See the BYTES and CYCLES per INSTRUCTION table for details.

Bytes and Cycles per Instruction

The following table shows the number of bytes and cycles for each instruction in the format of byte/cycle.

Logic and Arithmetic Instructions

	[B]	Direct	Immed.
ADD	1/1	3/4	2/2
ADC	1/1	3/4	2/2
SUBC	1/1	3/4	2/2
AND	1/1	3/4	2/2
OR	1/1	3/4	2/2
XOR	1/1	3/4	2/2
IFEQ	1/1	3/4	2/2
IFGT	1/1	3/4	2/2
IFBNE	1/1		
DRSZ		1/3	
SBIT	1/1	3/4	
RBIT	1/1	3/4	
IFBIT	1/1	3/4	

RPND	1/1

Instructions Using A and C

CLRA	1/1
INCA	1/1
DECA	1/1
LAID	1/3
DCORA	1/1
RRCA	1/1
RLCA	1/1
SWAPA	1/1
SC	1/1
RC	1/1
IFC	1/1
IFNC	1/1
PUSHA	1/3
POPA	1/3
ANDSZ	2/2

Transfer of Control Instructions

JMPL	3/4
JMP	2/3
JP	1/3
JSRL	3/5
JSR	2/5
JID	1/3
VIS	1/5
RET	1/5
RETSK	1/5
RETI	1/5
INTR	1/7
NOP	1/1

Memory Transfer Instructions

	Register Indirect		Direct	Immed.	Register Indirect Auto Incr. and Decr.		
	[B]	[X]			[B+, B−]	[X+, X−]	
X A,*	1/1	1/3	2/3		1/2	1/3	
LD A,*	1/1	1/3	2/3	2/2	1/2	1/3	
LD B, Imm				1/1			(IF B < 16)
LD B, Imm				2/3			(IF B > 15)
LD Mem, Imm	2/2	2/2	3/3		2/2		
LD Reg, Imm			2/3				
IFEQ MD, Imm			3/3				

* = > Memory location addressed by B or X or directly.

COP87L88EG/COP87L84EG Opcode Table

UPPER NIBBLE																LOWER NIBBLE
F	E	D	C	B	A	9	8	7	6	5	4	3	2	1	0	
JP −15	JP −31	LD 0F0, #i	DRSZ 0F0	RRCA	RC	ADC A, #i	ADC A,[B]	IFBIT 0,[B]	ANDSZ A, #i	LD B,#0F	IFBNE 0	JSR x000−x0FF	JMP x000−x0FF	JP + 17	INTR	0
JP −14	JP −30	LD 0F1, #i	DRSZ 0F1	*	SC	SUBC A, #i	SUB A,[B]	IFBIT 1,[B]	*	LD B,#0E	IFBNE 1	JSR x100−x1FF	JMP x100−x1FF	JP + 18	JP + 2	1
JP −13	JP −29	LD 0F2, #i	DRSZ 0F2	X A, [X+]	X A, [B+]	IFEQ A, #i	IFEQ A,[B]	IFBIT 2,[B]	*	LD B,#0D	IFBNE 2	JSR x200−x2FF	JMP x200−x2FF	JP + 19	JP + 3	2
JP −12	JP −28	LD 0F3, #i	DRSZ 0F3	X A, [X−]	X A, [B−]	IFGT A, #i	IFGT A,[B]	IFBIT 3,[B]	*	LD B,#0C	IFBNE 3	JSR x300−x3FF	JMP x300−x3FF	JP + 20	JP + 4	3
JP −11	JP −27	LD 0F4, #i	DRSZ 0F4	VIS	LAID	ADD A, #i	ADD A,[B]	IFBIT 4,[B]	CLRA	LD B,#0B	IFBNE 4	JSR x400−x4FF	JMP x400−x4FF	JP + 21	JP + 5	4
JP −10	JP −26	LD 0F5, #i	DRSZ 0F5	RPND	JID	AND A, #i	AND A,[B]	IFBIT 5,[B]	SWAPA	LD B,#0A	IFBNE 5	JSR x500−x5FF	JMP x500−x5FF	JP + 22	JP + 6	5
JP −9	JP −25	LD 0F6, #i	DRSZ 0F6	X A,[X]	X A,[B]	XOR A, #i	XOR A,[B]	IFBIT 6,[B]	DCORA	LD B,#09	IFBNE 6	JSR x600−x6FF	JMP x600−x6FF	JP + 23	JP + 7	6
JP −8	JP −24	LD 0F7, #i	DRSZ 0F7	*	*	OR A,#i	OR A,[B]	IFBIT 7,[B]	PUSHA	LD B,#08	IFBNE 7	JSR x700−x7FF	JMP x700−x7FF	JP + 24	JP + 8	7
JP −7	JP −23	LD 0F8, #i	DRSZ 0F8	NOP	RLCA	LD A,#i	IFC	SBIT 0,[B]	RBIT 0,[B]	LD B,#07	IFBNE 8	JSR x800−x8FF	JMP x800−x8FF	JP + 25	JP + 9	8
JP −6	JP −22	LD 0F9, #i	DRSZ 0F9	IFNE A,[B]	IFEQ Md,#i	IFNE A,#i	IFNC	SBIT 1,[B]	RBIT 1,[B]	LD B,#06	IFBNE 9	JSR x900−x9FF	JMP x900−x9FF	JP + 26	JP + 10	9
JP −5	JP −21	LD 0FA, #i	DRSZ 0FA	LD A, [X+]	LD A, [B+]	LD [B+], #i	INCA	SBIT 2,[B]	RBIT 2,[B]	LD B,#05	IFBNE 0A	JSR xA00−xAFF	JMP xA00−xAFF	JP + 27	JP + 11	A
JP −4	JP −20	LD 0FB, #i	DRSZ 0FB	LD A, [X−]	LD A, [B−]	LD [B−], #i	DECA	SBIT 3,[B]	RBIT 3,[B]	LD B,#04	IFBNE 0B	JSR xB00−xBFF	JMP xB00−xBFF	JP + 28	JP + 12	B
JP −3	JP −19	LD 0FC, #i	DRSZ 0FC	LD Md,#i	JMPL	X A,Md	POPA	SBIT 4,[B]	RBIT 4,[B]	LD B,#03	IFBNE 0C	JSR xC00−xCFF	JMP xC00−xCFF	JP + 29	JP + 13	C
JP −2	JP −18	LD 0FD, #i	DRSZ 0FD	DIR	JSRL	LD A,Md	RETSK	SBIT 5,[B]	RBIT 5,[B]	LD B,#02	IFBNE 0D	JSR xD00−xDFF	JMP xD00−xDFF	JP + 30	JP + 14	D
JP −1	JP −17	LD 0FE, #i	DRSZ 0FE	LD A,[X]	LD A,[B]	LD [B],#i	RET	SBIT 6,[B]	RBIT 6,[B]	LD B,#01	IFBNE 0E	JSR xE00−xEFF	JMP xE00−xEFF	JP + 31	JP + 15	E
JP −0	JP −16	LD 0FF, #i	DRSZ 0FF	*	*	LD B,#i	RETI	SBIT 7,[B]	RBIT 7,[B]	LD B,#00	IFBNE 0F	JSR xF00−xFFF	JMP xF00−xFFF	JP + 32	JP + 16	F

where,

i is the immediate data

Md is a directly addressed memory location

* is an unused opcode

Note: The opcode 60 Hex is also the opcode for IFBIT #i,A

Development Support

SUMMARY

- iceMASTER™: IM-COP8/400—Full feature in-circuit emulation for all COP8 products. A full set of COP8 Basic and Feature Family device and package specific probes are available.
- COP8 Debug Module: Moderate cost in-circuit emulation and development programming unit.
- COP8 Evaluation and Programming Unit: EPU-COP888GG—low cost in-circuit simulation and development programming unit.
- Assembler: COP8-DEV-IBMA. A DOS installable cross development Assembler, Linker, Librarian and Utility Software Development Tool Kit.
- C Compiler: COP8C. A DOS installable cross development Software Tool Kit.
- OTP/EPROM Programmer Support: Covering needs from engineering prototype, pilot production to full production environments.

iceMASTER (IM) IN-CIRCUIT EMULATION

The iceMASTER IM-COP8/400 is a full feature, PC based, in-circuit emulation tool developed and marketed by MetaLink Corporation to support the whole COP8 family of products. National is a resale vendor for these products.

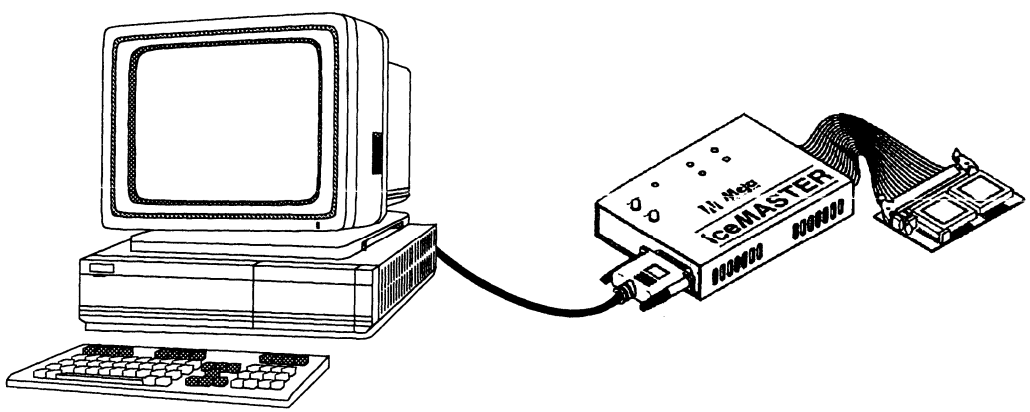
See *Figure 19* for configuration.

The iceMASTER IM-COP8/400 with its device specific COP8 Probe provides a rich feature set for developing, testing and maintaining product:

- Real-time in-circuit emulation; full 2.4V–5.5V operation range, full DC-10 MHz clock. Chip options are programmable or jumper selectable.
- Direct connection to application board by package compatible socket or surface mount assembly.
- Full 32 kbytes of loadable programming space that overlays (replaces) the on-chip ROM or EPROM. On-chip RAM and I/O blocks are used directly or recreated on the probe as necessary.
- Full 4k frame synchronous trace memory. Address, instruction, and 8 unspecified, circuit connectable trace lines. Display can be HLL source (e.g., C source), assembly or mixed.
- A full 64k hardware configurable break, trace on, trace off control, and pass count increment events.
- Tool set integrated interactive symbolic debugger—supports both assembler (COFF) and C Compiler (.COD) linked object formats.
- Real time performance profiling analysis; selectable bucket definition.
- Watch windows, content updated automatically at each execution break.
- Instruction by instruction memory/register changes displayed on source window when in single step operation.
- Single base unit and debugger software reconfigurable to support the entire COP8 family; only the probe personality needs to change. Debugger software is processor customized, and reconfigured from a master model file.
- Processor specific symbolic display of registers and bit level assignments, configured from master model file.
- Halt/Idle mode notification.
- On-line HELP customized to specific processor using master model file.
- Includes a copy of COP8-DEV-IBMA assembler and linker SDK.

IM Order Information

Base Unit	
IM-COP8/400-1	iceMASTER Base Unit, 110V Power Supply
IM-COP8/400-2	iceMASTER Base Unit, 220V Power Supply
iceMASTER Probe	
MHW-888GG28DWPC	28 DIP
28 DIP to 28 SO Adapter	
MHW-SOIC28	28 SO



TL/DD/12872–19

FIGURE 19. COP8 iceMASTER Environment

Development Support (Continued)

iceMASTER DEBUG MODULE (DM)

The iceMASTER Debug Module is a PC based, combination in-circuit emulation tool and COP8 based OTP/EPROM programming tool developed and marketed by MetaLink Corporation to support the whole COP8 family of products. National is a resale vendor for these products.

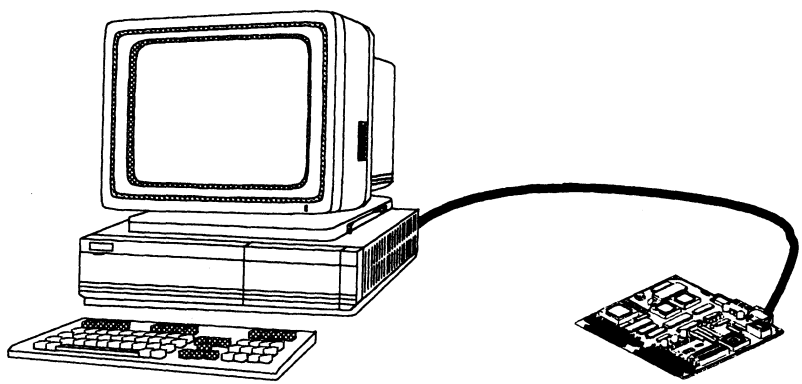
See *Figure 20* for configuration.

The iceMASTER Debug Module is a moderate cost development tool. It has the capability of in-circuit emulation for a specific COP8 microcontroller and in addition serves as a programming tool for COP8 OTP and EPROM product families. Summary of features is as follows:

- Real-time in-circuit emulation; full operating voltage range operation, full DC-10 MHz clock.
- All processor I/O pins can be cabled to an application development board with package compatible cable to socket and surface mount assembly.
- Full 32 kbytes of loadable programming space that overlays (replaces) the on-chip ROM or EPROM. On-chip RAM and I/O blocks are used directly or recreated as necessary.
- 100 frames of synchronous trace memory. The display can be HLL source (C source), assembly or mixed. The most recent history prior to a break is available in the trace memory.
- Configured break points; uses INTR instruction which is modestly intrusive.
- Software—only supported features are selectable.
- Tool set integrated interactive symbolic debugger—supports both assembler (COFF) and C Compiler (.COD) SDK linked object formats.
- Instruction by instruction memory/register changes displayed when in single step operation.
- Debugger software is processor customized, and reconfigured from a master model file.
- Processor specific symbolic display of registers and bit level assignments, configured from master model file.
- Halt/Idle mode notification.
- Programming menu supports full product line of programmable OTP and EPROM COP8 products. Program data is taken directly from the overlay RAM.
- Programming of 44 PLCC and 68 PLCC parts requires external programming adapters.
- Includes wallmount power supply.
- On-board V_{PP} generator from 5V input or connection to external supply supported. Rquires V_{PP} level adjustment per the family programming specification (correct level is provided on an on-screen pop-down display).
- On-line HELP customized to specific processor using master model file.
- Includes a copy of COP8-DEV-IBMA assembler and linker SDK.

DM Order Information

Debug Module Unit	
COP8-DM/888GG	
Cable Adapter	
DM-COP8/28D	28 DIP
28 DIP to 28 SO Adapter	
DM-COP8/28D-SO	28 SO



TL/DD/12872–20

FIGURE 20. COP8-DM Environment

Development Support (Continued)

iceMASTER EVALUATION PROGRAMMING UNIT (EPU)

The iceMASTER EPU-COP888GG is a PC based, in-circuit simulation tool to support the feature family COP8 products.

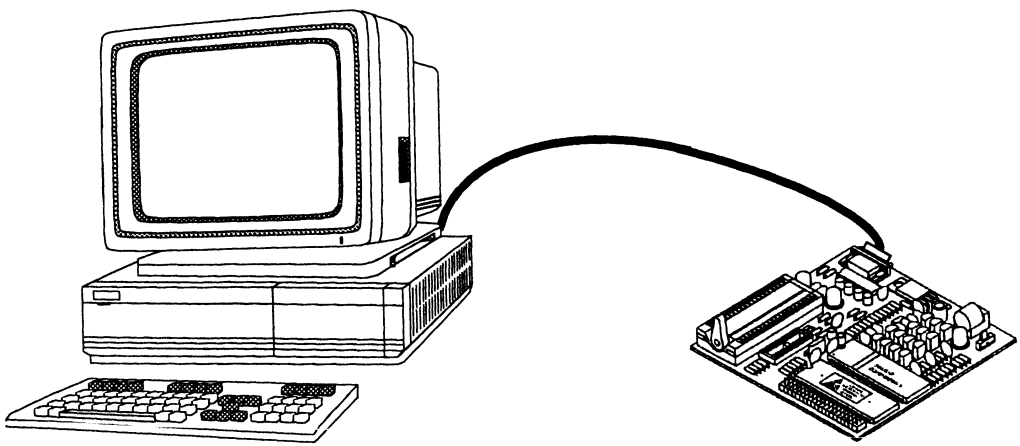
See *Figure 21* for configuration.

The simulation capability is a very low cost means of evaluating the general COP8 architecture. In addition, the EPU has programming capability, with added adapters, for programming the whole COP8 product family of OTP and EPROM products. The product includes the following features:

- Non-real-time in-circuit simulation. Program overlay memory is PC resident; instructions are downloaded over RS-232 as executed. Approximate performance is 20 kHz.
- Includes a 40-pin DIP cable adapter. Other target packages are not supported. All processor I/O pins are cabled to the application development environment.
- Full 32 kbytes of loadable programming space that overlays (replaces) the on-chip ROM or EPROM. On-chip RAM and I/O blocks are used directly or recreated as necessary.
- On-chip timer and WATCHDOG execution are not well synchronized to the instruction simulation.
- 100 frames of synchronous trace memory. The display can be HLL source (e.g., C source), assembly or mixed. The most recent history prior to a break is available in the trace memory.
- Up to eight software configured break points; uses INTR instruction which is modestly intrusive.
- Common look-feel debugger software across all MetaLink products—only supported features are selectable.
- Tool set integrated interactive symbolic debugger—supports both assembler (COFF) and C Compiler (.COD) SDK linked object formats.
- Instruction by instruction memory/register changes displayed when in single step operation.
- Processor specific symbolic display of registers and bit level assignments, configured from master model file.
- Halt/Idle mode notification. Restart requires special handling.
- Programming menu supports full product line of programmable OTP and EPROM COP8 products. Only a 40 ZIF socket is available on the EPU unit. Adapters are available for other part package configurations.
- Integral wall mount power supply provides 5V and develops the required V_{PP} to program parts.
- Includes a copy of COP8-DEV-IBMA assembler, linker SDK.

EPU Order Information

Evaluation Programming Unit	
EPU-COP888GG	Evaluation Programming Unit with debugger and programmer control software with 40 ZIF programming socket.
General Programming Adapters	
COP8-PGMA-DS44P	28 and 20 DIP and SOIC plus 44 PLCC adapter



TL/DD/12872-21

FIGURE 21. EPU-COP8 Tool Environment

Development Support (Continued)

COP8 ASSEMBLER/LINKER SOFTWARE DEVELOPMENT TOOL KIT

National Semiconductor offers a relocateable COP8 macro cross assembler, linker, librarian and utility software development tool kit. Features are summarized as follows:

- Basic and Feature Family instruction set by "device" type.
- Nested macro capability.
- Extensive set of assembler directives.
- Supported on PC/DOS platform.
- Generates National standard COFF output files.
- Integrated Linker and Librarian.
- Integrated utilities to generate ROM code file outputs.
- DUMPCOFF utility.

This product is integrated as a part of MetaLink tools as a development kit, fully supported by the MetaLink debugger. It may be ordered separately or it is bundled with the MetaLink products at no additional cost.

Order Information

Assembler SDK	
COP8-DEV-IBMA	Assembler SDK on installable 3.5" PC/DOS Floppy Disk Drive format. Periodic upgrades and most recent version is available on National's BBS and Internet.

COP8 C COMPILER

A C Compiler is developed and marketed by Byte Craft Limited. The COP8C compiler is a fully integrated development tool specifically designed to support the compact embedded configuration of the COP8 family of products.

Features are summarized as follows:

- ANSI C with some restrictions and extensions that optimize development for the COP8 embedded application.
- BITS data type extension. Register declaration #pragma with direct bit level definitions.
- C language support for interrupt routines.
- Expert system, rule based code generation and optimization.
- Performs consistency checks against the architectural definitions of the target COP8 device.
- Generates program memory code.
- Supports linking of compiled object or COP8 assembled object formats.
- Global optimization of linked code.
- Symbolic debug load format fully sourced level supported by the MetaLink debugger.

INDUSTRY WIDE OTP/EPROM PROGRAMMING SUPPORT

Programming support, in addition to the MetaLink development tools, is provided by a full range of independent approved vendors to meet the needs from the engineering laboratory to full production.

Approved List

Manufacturer	North America	Europe	Asia
BP Microsystems	(800) 225-2102 (713) 688-4600 Fax: (713) 688-0920	+49-8152-4183 +49-8856-932616	+852-234-16611 +852-2710-8121
Data I/O	(800) 426-1045 (206) 881-6444 Fax: (206) 882-1043	+44-0734-440011	Call North America
HI-LO	(510) 623-8860	Call Asia	+886-2-764-0215 Fax: +886-2-756-6403
ICE Technology	(800) 624-8949 (919) 430-7915	+44-1226-767404 Fax: 0-1226-370-434	
MetaLink	(800) 638-2423 (602) 926-0797 Fax: (602) 693-0681	+49-80 9156 96-0 Fax: +49-80 9123 86	+852-737-1800
Systems General	(408) 263-6667	+41-1-9450300	+886-2-917-3005 Fax: +886-2-911-1283
Needhams	(916) 924-8037 Fax: (916) 924-8065		

Development Support (Continued)

AVAILABLE LITERATURE

For more information, please see the COP8 Basic Family User's Manual, Literature Number 620895, COP8 Feature Family User's Manual, Literature Number 620897 and National's Family of 8-bit Microcontrollers COP8 Selection Guide, Literature Number 630009.

DIAL-A-HELPER SERVICE

Dial-A-Helper is a service provided by the Microcontroller Applications group. The Dial-A-Helper is an Electronic Information System that may be accessed as a Bulletin Board System (BBS) via data modem, as an FTP site on the Internet via standard FTP client application or as an FTP site on the Internet using a standard Internet browser such as Netscape or Mosaic.

The Dial-A-Helper system provides access to an automated information storage and retrieval system . The system capabilities include a MESSAGE SECTION (electronic mail, when accessed as a BBS) for communications to and from the Microcontroller Applications Group and a FILE SECTION which consists of several file areas where valuable application software and utilities could be found.

DIAL-A-HELPER BBS via Standard Modem

Modem:	CANADA/U.S.:	(800) NSC-MICRO (800) 672-6427
	EUROPE:	(+49) 0-8141-351332
	Baud:	14.4k
	Set-Up:	Length: 8-Bit Parity: None Stop Bit: 1
	Operation:	24 Hours, 7 Days

DIAL-A-HELPER via FTP

ftp nscmicro.nsc.com
user: anonymous
password: username@yourhost.site.domain

DIAL-A-HELPER via WorldWide Web Browser

ftp://nscmicro.nsc.com

National Semiconductor on the WorldWide Web

See us on the WorldWide Web at: http://www.natsemi.com

CUSTOMER RESPONSE CENTER

Complete product information and technical support is available from National's customer response centers.

CANADA/U.S.:	Tel:	(800)272-9959
	email:	support@tevm2.nsc.com
EUROPE:	email:	europe.support@nsc.com
	Deutsch Tel:	+49 (0) 180-530 85 85
	English Tel:	+49 (0) 180-532 78 32
	Français Tel:	+49 (0) 180-532 93 58
	Italiano Tel:	+49 (0) 180-534 16 80
JAPAN:	Tel:	+81-043-299-2309
S.E. ASIA:	Beijing Tel:	(+86) 10-6856-8601
	Shanghai Tel:	(+86) 21-6415-4092
	Hong Kong Tel:	(+852) 2737-1600
	Korea Tel:	(+82) 2-3771-6909
	Malaysia Tel:	(+60-4) 644-9061
	Singapore Tel:	(+65) 255-2226
	Taiwan Tel:	+886-2-521-3288
AUSTRALIA:	Tel:	(+61) 3-9558-9999
INDIA:	Tel:	(+91) 80-559-9467

National Semiconductor

ADVANCE INFORMATION

COP87L88RD
8-Bit One-Time Programmable (OTP) Microcontroller with A/D Converter and 32 kbytes of Program Memory

General Description

The COP87L88RD is a member of the COP8™ 8-bit OTP microcontroller family. It is pin and software compatible to the mask ROM COP888GD product family.

It is a fully static part, fabricated using double-metal silicon gate microCMOS technology. Features include an 8-bit memory mapped architecture, MICROWIRE/PLUS serial I/O, three 16-bit timer/counters supporting three modes (Processor Independent PWM generation, External Event counter, and Input Capture mode capabilities), 8 channel analog to digital converter, and two power saving modes (HALT and IDLE), both with a multi-sourced wakeup/interrupt capability. This multi-sourced interrupt capability may also be used independent of the HALT or IDLE modes. The device includes an IDLE Timer with 5 selectable interrupt periods which can be used to wake the device from IDLE mode. Each I/O pin has software selectable configurations. The devices operate over a voltage range of 2.7V to 5.5V. High throughput is achieved with an efficient, regular instruction set operating at a maximum rate of 1 μs per instruction.

Key Features

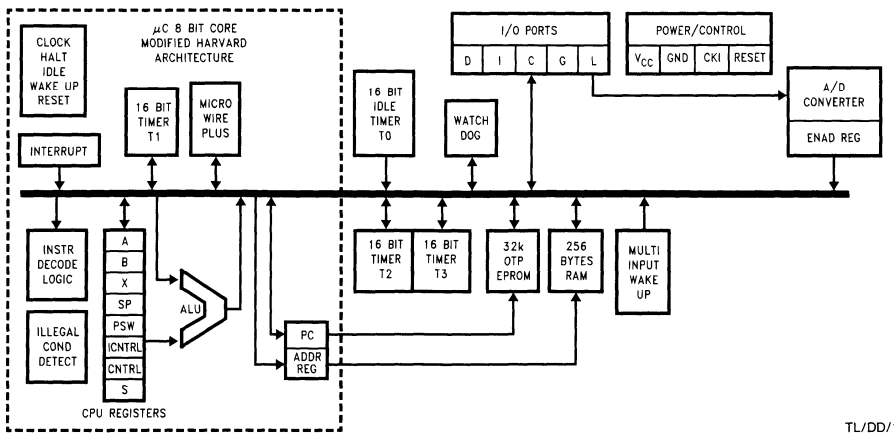
- 8-channel A/D converter with prescaler and both differential and single ended modes
- Idle Timer with 5 selectable Wake-Up periods
- Three 16-bit timers, each with two 16-bit registers supporting:
  - — Processor Independent PWM mode
  - — External Event counter mode
  - — Input Capture mode
- 32 kbytes on-board OTP EPROM with security feature

Note: Mask ROMed devices with equivalent on-chip features and program memory sizes of 16k is available.

- 256 bytes on-board RAM

(Continued)

Block Diagram



TL/DD/12854–1

FIGURE 1. Block Diagram

Additional Peripheral Features

- Multi-Input Wakeup (MIWU) with optional interrupts (8)
- WATCHDOG™ and clock monitor Logic
- MICROWIRE/PLUS™ serial I/O

I/O Features

- Memory mapped I/O
- Software selectable I/O options (TRI-STATE® Output, Push-Pull Output, Weak Pull Up Input, High Impedance Input)
- Schmitt trigger inputs on ports G and L
- Package:
  - 44 PLCC with 40 I/O pins
  - 40 DIP with 36 I/O pins

CPU/Instruction Set Features

- 1 μs instruction cycle time
- Twelve multi-source vectored interrupts servicing
  - External Interrupt
  - Idle Timer T0
  - Three Timers (each with 2 Interrupts)
  - MICROWIRE/PLUS
  - Multi-Input Wake Up
  - Software Trap
  - Default VIS (default interrupt)
- Versatile and easy to use instruction set
- 8-bit Stack Pointer (SP)—stack in RAM
- Two 8-bit Register Indirect Data Memory Pointers (B and X)

Fully Static CMOS

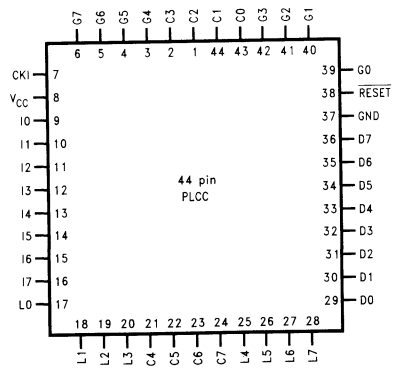
- Two power saving modes: HALT and IDLE
- Single supply operation: 2.7V to 5.5V
- Temperature range: −40°C to +85°C

Development Support

- Emulation device for COP888GD
- Real time emulation and full program debug offered by MetaLink's Development System

Connection Diagrams

Plastic Chip Carrier



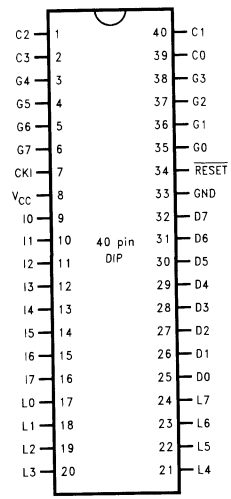
TL/DD/12854–2

Top View

Order Number COP87L88RDV-XE
See NS Plastic Chip Package Number V44A

Note: -X Crystal Oscillator
E Halt Enable

Dual-In-Line Package



TL/DD/12854–3

Top View

Order Number COP87L88RDN-XE
See NS Molded Package Number N40A

FIGURE 2. Connection Diagrams

National Semiconductor

PRELIMINARY

COP87L88RW
8-Bit One-Time Programmable (OTP) Microcontroller with Pulse Train Generators and Capture Modules

General Description

The COP87L88RW is a member of the COP8™ 8-bit OTP microcontroller family. It is pin and software compatible to the mask ROM COP888GW product family. (Continued)

Key Features

- Multiply/divide functions
- Full duplex UART
- Four pulse train generators with 16-bit prescalers
- Two 16-bit input capture modules with 8-bit prescalers
- Two 16-bit timers, each with two 16-bit registers supporting
  - — Processor independent PWM mode
  - — External event counter mode
  - — Input capture mode
- 32 kbytes on-board OTP EPROM with security feature

Note: Mask ROMed devices with equivalent on-chip features and program memory sizes of 16k is available.

- 512 bytes on-board RAM

Additional Peripheral Features

- Idle Timer
- Multi-Input Wake-Up (MIWU) with optional interrupts (8)
- WATCHDOG™ and clock monitor logic
- MICROWIRE/PLUS™ serial I/O

I/O Features

- Memory mapped I/O
- Software selectable I/O options
  - — TRI-STATE® output
  - — Push-pull output
  - — Weak pull-up input
  - — High impedance input
- Schmitt trigger inputs on ports G and L
- Package: 68 PLCC with I/O pins

CPU/Instruction Set Features

- 1 μs instruction cycle time
- Fourteen multi-source vectored interrupts servicing
  - — External interrupt
  - — Idle timer T0
  - — Two timers (each with 2 interrupts)
  - — MICROWIRE/PLUS
  - — Multi-Input Wake-Up
  - — Software trap
  - — UART (2)
  - — Default VIS
  - — Capture timers
  - — Counters (one vector for all four counters)
- Versatile and easy to use instruction set
- 8-bit Stack Pointer (SP)—stack in RAM
- Two 8-bit register indirect data memory pointers (B and X)

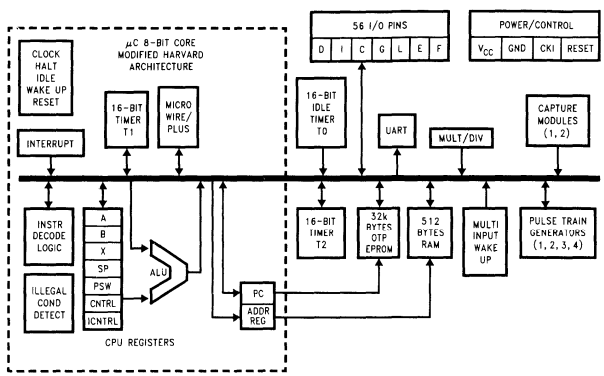
Fully Static CMOS

- Two power saving modes: HALT and IDLE
- Single supply operation: 2.7V to 5.5V
- Temperature range: −40°C to +85°C

Development Support

- Emulation device for the COP888GW
- Real time emulation and full program debug offered by MetaLink's Development System

Block Diagram



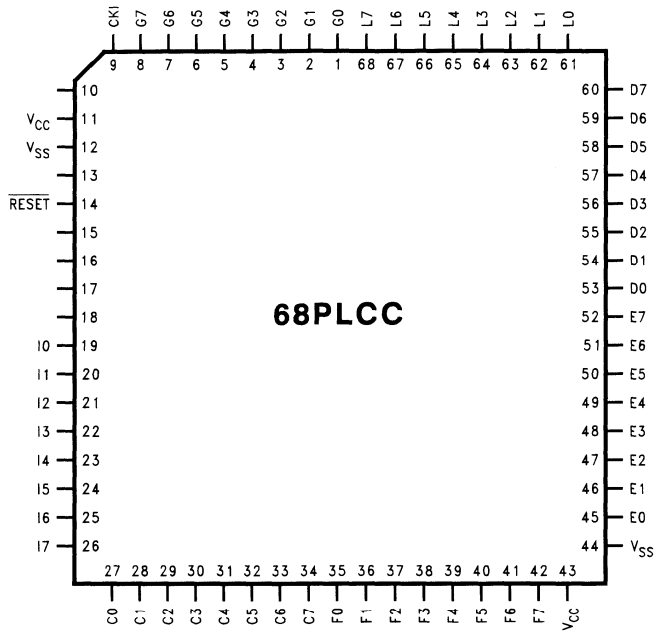
TL/DD/12855–1

FIGURE 1. COP87L88RW Block Diagram

General Description (Continued)

It is a fully static part, fabricated using double-metal silicon gate microCMOS technology. Features include an 8-bit memory mapped architecture, MICROWIRE/PLUS serial I/O, two 16-bit timer/counters supporting three modes (Processor Independent PWM generation, External Event counter and Input Capture mode capabilities), four independent 16-bit pulse train generators with 16-bit prescalers, two independent 16-bit input capture modules with 8-bit prescalers, multiply and divide functions, full duplex UART, and two power savings modes (HALT and IDLE), both with a multi-sourced wake up/interrupt capability. This multi-sourced interrupt capability may also be used independent of the HALT or IDLE modes. Each I/O pin has software selectable configurations. The devices operate over a voltage range of 2.7V--5.5V. High throughput is achieved with an efficient, regular instruction set operating at a maximum of 1 μs per instruction rate.

Connection Diagram



TL/DD/12855-2

Top View

Note: -X Crystal Oscillator
-E Halt Enable

Order Number COP87L88RWV-XE
See NS Plastic Chip Package Number V68A

FIGURE 2. Connection Diagram

Absolute Maximum Ratings (Note)

Supply Voltage (V_{CC})	7V
Voltage at Any Pin	−0.3V to V_{CC} +0.3V
Total Current into V_{CC} Pin (Source)	100 mA
Total Current out of GND Pin (Sink)	110 mA
Storage Temperature Range	−65°C to +150°C

Note: *Absolute maximum ratings indicate limits beyond which damage to the device may occur. DC and AC electrical specifications are not ensured when operating the device at absolute maximum ratings.*

DC Electrical Characteristics −40°C ≤ T_A ≤ +85°C unless otherwise specified

Parameter	Conditions	Min	Typ	Max	Units
Operating Voltage		2.7		5.5	V
Power Supply Ripple (Note 1)	Peak-to-Peak			0.1 V_{CC}	V
Supply Current (Note 2)					
CKI = 10 MHz	V_{CC} = 5.5V, t_c = 1 μs			14	mA
HALT Current (Note 3)	V_{CC} = 5.5V, CKI = 0 MHz			12	μA
IDLE Current					
CKI = 10 MHz	V_{CC} = 5.5V			1.7	mA
Input Levels (V_{IH}, V_{IL})					
$\overline{\text{RESET}}$, CKI					
Logic High		0.8 V_{CC}			V
Logic Low				0.2 V_{CC}	V
All Other Inputs					
Logic High		0.7 V_{CC}			V
Logic Low				0.2 V_{CC}	V
Hi-Z Input Leakage	V_{CC} = 5.5V	−2		+2	μA
Input Pullup Current	V_{CC} = 5.5V, V_{IN} = 0V	40		−250	μA
G Port Input Hysteresis	(Note 6)		0.05 V_{CC}	0.35 V_{CC}	V
Output Current Levels					
D Outputs					
Source	V_{CC} = 4.5V, V_{OH} = 3.3V	−0.4			mA
Sink (Note 4)	V_{CC} = 4.5V, V_{OL} = 1V	10			mA
All Others					
Source (Weak Pull-Up Mode)	V_{CC} = 4.5V, V_{OH} = 2.7V	−10		−100	μA
Source (Push-Pull Mode)	V_{CC} = 4.5V, V_{OH} = 3.3V	−0.4			mA
Sink (Push-Pull Mode)	V_{CC} = 4.5V, V_{OL} = 0.4V	1.6			mA
TRI-STATE Leakage	V_{CC} = 5.5V	−2		+2	μA
Allowable Sink/Source Current per Pin					
D Outputs (Sink)				15	mA
All others				3	mA
Maximum Input Current without Latchup (Note 5, 7)	Room Temp			±200	mA
RAM Retention Voltage, V_r (Note 6)	500 ns Rise and Fall Time (min)	2			V
Input Capacitance	(Note 7)			7	pF
Load Capacitance on D2	(Note 7)			1000	pF

AC Electrical Characteristics $-40°C \leq T_A \leq +85°C$ unless otherwise specified

Parameter	Conditions	Min	Typ	Max	Units
Instruction Cycle Time (t_c)					
Crystal, Resonator		1.0		DC	μs
Ceramic					
Inputs					
t_{SETUP}	$V_{CC} \geq 4.5V$	200			ns
t_{HOLD}	$V_{CC} \geq 4.5V$	60			ns
Output Propagation Delay (Note 9)	$R_L = 2.2k, C_L = 100$ pF				
t_{PD1}, t_{PD0}					
SO, SK	$V_{CC} \geq 4.5V$			0.7	μs
All Others	$V_{CC} \geq 4.5V$			1	μs
MICROWIRE™ Setup Time (t_{UWS}) (Note 7)	$V_{CC} \geq 4.5V$	20			
MICROWIRE Hold Time (t_{UWH}) (Note 7)	$V_{CC} \geq 4.5V$	56			ns
MICROWIRE Output Propagation Delay (t_{UPD})	$V_{CC} \geq 4.5V$			220	
Input Pulse Width (Note 8)					
Interrupt Input High Time		1			
Interrupt Input Low Time		1			t_c
Timer 1, 2 Input High Time		1			
Timer 1, 2 Input Low Time		1			
Capture Timer High Time		1			CKI
Capture Timer Low Time		1			CKI
Reset Pause Width		1			μs

Note 1: Maximum rate of voltage change to be defined.

Note 2: Supply current is measured after running 2000 cydes with a square wave CKI input, CKO open, inputs at rails and outputs open.

Note 3: The HALT mode will stop CKI from oscillatng. Test conditions: All inputs tied to V_{CC}, L, C, E, F, and G port I/O's configured as outputs and programmed low and not driving a load; D outputs programmed low and not driving a load. Parameter refers to HALT mode entered via setting bit 7 of the G Port data register. Part will pull up CKI during HALT in crystal clock mode.

Note 4: The user must guarantee that D2 pin does not source more than 10 mA during RESET. If D2 sources more than 10 mA during reset, the device will go into programming mode.

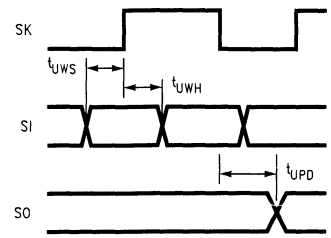
Note 5: Pins G6 and $\overline{\text{RESET}}$ are designed with a high voltage input network. These pins allow input voltages greater than V_{CC} and the pins will have sink current to V_{CC} when biased at voltages greater than V_{CC} (the pins do not have source current when biased at a voltage below V_{CC}.) The effective resistance to V_{CC} is 750Ω (typical). These two pins will not latch up. The voltage at the pins must be limited to less than 14V. WARNING: Voltages in excess of 14V will cause damage to the pins. This warning excludes ESD transients.

Note 6: Condition and parameter valid only for part in HALT mode.

Note 7: Parameter characterized but not tested.

Note 8: t_c = Instruction Cycle Time

Note 9: The output propagation delay is referenced to the end of the instruction cycle where the output change occurs.



TL/DD/12855–3

FIGURE 3. MICROWIRE/PLUS Timing

Pin Descriptions

V_{CC} and GND are the power supply pins. All V_{CC} and GND pins must be connected.

CKI is the clock input. This comes from a crystal oscillator (in conjunction with CKO). See Oscillator Description section.

$\overline{\text{RESET}}$ is the master reset input. See Reset description section.

The device contains five bidirectional 8-bit I/O ports (C, E, F, G and L), where each individual bit may be independently configured as an input (Schmitt trigger inputs on ports L and G), output or TRI-STATE under program control. Three data memory address locations are allocated for each of these I/O ports. Each I/O port has two associated 8-bit memory mapped registers, the CONFIGURATION register and the output DATA register. A memory mapped address is also reserved for the input pins of each I/O port. (See the memory map for the various addresses associated with the I/O ports.) *Figure 4* shows the I/O port configurations. The DATA and CONFIGURATION registers allow for each port bit to be individually configured under software control as shown below:

Configuration Register	Data Register	Port Set-Up
0	0	Hi-Z Input (TRI-STATE Output)
0	1	Input with Weak Pull-Up
1	0	Push-Pull Zero Output
1	1	Push-Pull One Output

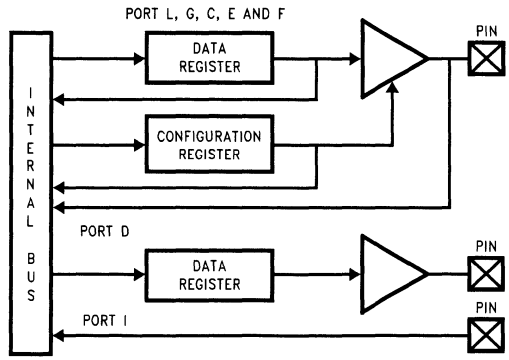
PORT L is an 8-bit I/O port. All L-pins have Schmitt triggers on the inputs.

The Port L supports Multi-Input Wake Up on all eight pins. L1 is used for the UART external clock. L2 and L3 are used for the UART transmit and receive. L4 and L5 are used for the timer input functions T2A and T2B. L6 and L7 are used for the capture timer input functions CAP1 and CAP2.

The Port L has the following alternate features:

- L0 MIWU
- L1 MIWU or CKX
- L2 MIWU or TDX
- L3 MIWU or RDX
- L4 MIWU or T2A
- L5 MIWU or T2B
- L6 MIWU or CAP1
- L7 MIWU or CAP2

Port G is an 8-bit port with 6 I/O pins (G0–G5), an input pin (G6), and a dedicated output pin (G7). Pins G0–G6 all have Schmitt Triggers on their inputs. Pin G7 serves as the dedicated output pin for the CKO clock output. There are two registers associated with the G Port, a data register and a configuration register. Therefore, each of the 6 I/O bits (G0–G5) can be individually configured under software control.



TL/DD/12855–4

FIGURE 4. I/O Port Configurations

Pin Descriptions (Continued)

Since G6 is an input only pin and G7 is dedicated CKO clock output pin, the associated bits in the data and configuration registers for G6 and G7 are used for special purpose functions as outlined below. Reading the G6 and G7 data bits will return zeros.

Note that the chip will be placed in the HALT mode by writing a "1" to bit 7 of the Port G Data Register. Similarly the chip will be placed in the IDLE mode by writing a "1" to bit 6 of the Port G Data Register.

Writing a "1" to bit 6 of the Port G Configuration Register enables the MICROWIRE/PLUS to operate with the alternate phase of the SK clock.

	Config Reg.	Data Reg.
G7	Not Used	HALT
G6	Alternate SK	IDLE

Port G has the following alternate features:

- G0 INTR (External Interrupt Input)
- G2 T1B (Timer T1 Capture Input)
- G3 T1A (Timer T1 I/O)
- G4 SO (MICROWIRE Serial Data Output)
- G5 SK (MICROWIRE Serial Clock)
- G6 SI (MICROWIRE Serial Data Input)

Port G has the following dedicated functions:

- G7 CKO Oscillator dedicated output

Ports C and F are 8-bit I/O ports.

Port E is an 8-bit I/O port. It has the following alternate features:

- E0 CT1 (Output for counter1, Pulse Train Generator)
- E1 CT2 (Output for counter2, Pulse Train Generator)
- E2 CT3 (Output for counter3, Pulse Train Generator)
- E3 CT4 (Output for counter4, Pulse Train Generator)

Port I is an eight-bit Hi-Z input port.

Port D is an 8-bit output port that is preset high when $\overline{\text{RESET}}$ goes low. The user can tie two or more D port outputs (except D2) together in order to get a higher drive.

Note: Care must be exercised with the D2 pin operation. At RESET, the external loads on this pin must ensure that the output voltages stay above 0.8 V_{CC} to prevent the chip from entering special modes. Also keep the external loading on D2 to <1000 pF.

Functional Description

The architecture of the device is modified Harvard architecture. With the Harvard architecture, the control store program memory (ROM) is separated from the data store memory (RAM). Both ROM and RAM have their own separate addressing space with separate address buses. The architecture, though based on Harvard architecture, permits transfer of data from ROM to RAM.

CPU REGISTERS

The CPU can do an 8-bit addition, subtraction, logical or shift operation in one instruction (t_c) cycle time.

There are six CPU registers:

A is the 8-bit Accumulator Register

PC is the 15-bit Program Counter Register

PU is the upper 7 bits of the program counter (PC)

PL is the lower 8 bits of the program counter (PC)

B is an 8-bit RAM address pointer, which can be optionally post auto incremented or decremented.

X is an 8-bit alternate RAM address pointer, which can be optionally post auto incremented or decremented.

SP is the 8-bit stack pointer, which points to the subroutine/interrupt stack (in RAM). The SP is initialized to RAM address 06F with reset.

S is the 8-bit Data Segment Address Register used to extend the lower half of the address range (00 to 7F) into 256 data segments of 128 bytes each.

All the CPU registers are memory mapped with the exception of the Accumulator (A) and the Program Counter (PC).

PROGRAM MEMORY

The program memory consists of 32 kbytes of OTP EPROM. These bytes may hold program instructions or constant data (data tables for the LAID instruction, jump vectors for the JID instruction, and interrupt vectors for the VIS instruction). The program memory is addressed by the 15-bit program counter (PC). All interrupts in the devices Vector to program memory location 0FF Hex.

The device can be configured to inhibit external reads of the program memory. This is done by programming the Security Byte.

Note: Mask ROMed devices with equivalent on-chip features and program memory sizes of 16k is available.

SECURITY FEATURE

The program memory array has an associate Security Byte that is located outside of the program address range. This byte can be addressed only from programming mode by a programmer tool.

Security is an optional feature and can only be asserted after the memory array has been programmed and verified. A secured part will read all 00(hex) by a programmer. The part will fail Blank Check and will fail Verify operations. A Read operation will fill the programmer's memory with 00(hex). The Security Byte itself is always readable with value of 00(hex) if unsecure and FF(hex) if secure.

DATA MEMORY

The data memory address space includes the on-chip RAM and data registers, the I/O registers (Configuration, Data and Pin), the control registers, the MICROWIRE/PLUS SIO shift register, and the various registers, and counters associated with the timers (with the exception of the IDLE timer). Data memory is addressed directly by the instruction or indirectly by the B, X, SP pointers and S register.

The data memory consists of 512 bytes of RAM. Sixteen bytes of RAM are mapped as "registers" at addresses 0F0 to 0FF Hex. These registers can be loaded immediately, and also decremented and tested with the DRSZ (decrement register and skip if zero) instruction. The memory pointer registers X, SP, B and S are memory mapped into this space at address locations 0FC to 0FF Hex respectively, with the other registers being available for general usage.

The instruction set permits any bit in memory to be set, reset or tested. All I/O and registers (except A and PC) are memory mapped; therefore, I/O bits and register bits can be directly and individually set, reset and tested. The accumulator (A) bits can also be directly and individually tested.

Note: RAM contents are undefined upon power-up.

Data Memory Segment RAM Extension

Data memory address 0FF is used as a memory mapped location for the Data Segment Address Register (S).

Data Memory Segment RAM Extension (Continued)

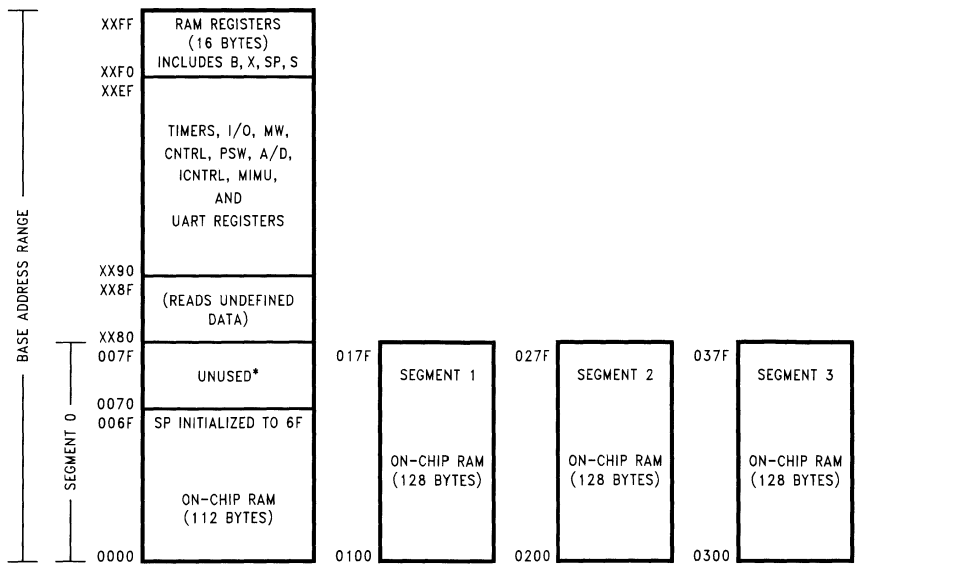
The data store memory is either addressed directly by a single-byte address within the instruction, or indirectly relative to the reference of the B, X, or SP pointers (each contains a single-byte address). This single-byte address allows an addressing range of 256 locations from 00 to FF hex. The upper bit of this single-byte address divides the data store memory into two separate sections as outlined previously. With the exception of the RAM register memory from address locations 00F0 to 00FF, all RAM memory is memory mapped with the upper bit of the single-byte address being equal to zero. This allows the upper bit of the single-byte address to determine whether or not the base address range (from 0000 to 00FF) is extended. If this upper bit equals one (representing address range 0080 to 00FF), then address extension does not take place. Alternatively, if this upper bit equals zero, then the data segment extension register S is used to extend the base address range (from 0000 to 007F) from XX00 to XX7F, where XX represents the 8 bits from the S register. Thus the 128-byte data segment extensions are located from addresses 0100 to 017F for data segment 1, 0200 to 027F for data segment 2, etc., up to FF00 to FF7F for data segment 255. The base address range from 0000 to 007F represents data segment 0.

Figure 5 illustrates how the S register data memory extension is used in extending the lower half of the base address range (00 to 7F hex) into 256 data segments of 128 bytes each, with a total addressing range of 32 kbytes from XX00 to XX7F. This organization allows a total of 256 data segments of 128-bytes each with an additional upper base segment of 128 bytes. Furthermore, all addressing modes are available for all data segments. The S register must be changed under program control to move from one data segment (128 bytes) to another. However, the upper base segment (containing the 16 memory registers, I/O registers, control registers, etc.) is always available regardless of the contents of the S register, since the upper base segment (address range 0080 to 00FF) is independent of data segment extension.

The instructions that utilize the stack pointer (SP) always reference the stack as part of the base segment (Segment 0), regardless of the contents of the S register. The S register is not changed by these instructions. Consequently, the stack (used with subroutine linkage and interrupts) is always located in the base segment. The stack pointer will be initialized to point at data memory location 006F as a result of reset.

The 128 bytes of RAM contained in the base segment are split between the lower and upper base segments. The first 112 bytes of RAM are resident from address 0000 to 006F in the lower base segment, while the remaining 16 bytes of RAM represent the 16 data memory registers located at addresses 00F0 to 00FF of the upper base segment. No RAM is located at the upper sixteen addresses (0070 to 007F) of the lower base segment.

Additional RAM beyond these initial 128 bytes, however, will always be memory mapped in groups of 128 bytes (or less) at the data segment address extensions (XX00 to XX7F) of the lower base segment. The additional 384 bytes of RAM in this device are memory mapped at address locations 0100 to 017F° 0200 to 027F, and 0300 to 037F hex.



*Reads as all ones.

FIGURE 5. RAM Organization

Reset

This device enters a reset state immediately upon detecting a logic low on the $\overline{\text{RESET}}$ pin. The $\overline{\text{RESET}}$ pin must be held low for a minimum of one instruction cycle to guarantee a valid reset. During power-up initialization, the user must insure that the $\overline{\text{RESET}}$ pin is held low until this device is within the specified V_{CC} voltage. An R/C circuit on the $\overline{\text{RESET}}$ pin with a delay 5 times (5x) greater than the power supply rise time is recommended.

When the $\overline{\text{RESET}}$ input goes low, the I/O ports are initialized immediately, with any observed delay being only propagation delay. When the $\overline{\text{RESET}}$ pin goes high, this device comes out of the reset state synchronously. This device will be running within two instruction cycles of the $\overline{\text{RESET}}$ pin going high.

$\overline{\text{RESET}}$ may also be used to exit this device from the HALT mode.

Some registers are reset to a known state, whereas other registers and RAM are "unchanged" by reset. When the controller goes into reset state while it is performing a write operation to one of these registers or RAM that are "unchanged" by reset, the register or RAM value will become unknown (i.e. not unchanged). This is because the write operation is terminated prematurely by reset and the results become uncertain. These registers and RAM locations are unchanged by reset only if they are not written to when the controller resets.

The following initializations occur with $\overline{\text{RESET}}$:

Port L: TRI-STATE

Port C: TRI-STATE

Port G: TRI-STATE

Port E: TRI-STATE

Port F: TRI-STATE

Port D: HIGH

PC: CLEARED

PSW, CNTRL and ICNTRL registers: CLEARED

SIOR:

UNAFFECTED after RESET with power already applied

RANDOM after RESET at power-on

T1CNTRL: CLEARED

T2CNTRL: CLEARED

TxRA, TxRB: RANDOM

CCMR1, CCMR2: CLEARED

CM1PSC, CM1CRL, CM1CRH, CM2PSC, CM2CRL, and CM2CRH:

UNAFFECTED after RESET with power already applied

RANDOM after RESET at power-on

CCR1 and CCR2: CLEARED

CxPRH. CxPRL, CxCTH, and CxCTL:

RANDOM after RESET at power-on

PSR, ENUR and ENUI: CLEARED

ENU: CLEARED except Bit 1 (TBMT) = 1

Accumulator, Timer 1 and Timer 2:

RANDOM after RESET with crystal clock option (power already applied)

UNAFFECTED after RESET with RC clock option (power already applied)

RANDOM after RESET at power-on

MDCR: CLEARED

MDR1, MDR2, MDR3, MDR4, MDR5: RANDOM

WKEN, WKEDG: CLEARED

WKPND: RANDOM

S Register: CLEARED

SP (Stack Pointer): Loaded with 6F Hex

B and X Pointers:

UNAFFECTED after RESET with power already applied

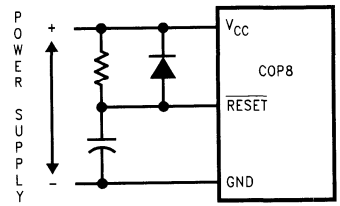
RANDOM after RESET at power-on

RAM:

UNAFFECTED after RESET with power already applied

RANDOM after RESET at power-on

The external RC network shown in *Figure 6* should be used to ensure that the $\overline{\text{RESET}}$ pin is held low until the power supply to the chip stabilizes.



TL/DD/12855–6

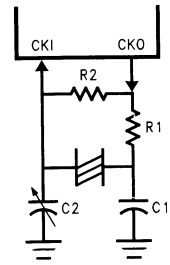
RC > 5 × POWER SUPPLY RISE TIME

FIGURE 6. Recommended Reset Circuit

Oscillator Circuits

The chip can be driven by a clock input on the CKI input pin which can be between DC and 10 MHz. The CKO output clock is on pin G7 (crystal configuration), The CKI input frequency is divided down by 10 to produce the instruction cycle clock (t_c).

Figure 7 shows the Crystal diagram



TL/DD/12855–7

FIGURE 7. Crystal Diagram

CRYSTAL OSCILLATOR

CKI and CKO can be connected to make a closed loop crystal (or resonator) controlled oscillator.

Oscillator Circuits (Continued)

Table I shows the component values required for various standard crystal values.

TABLE I. Crystal Oscillator Configuration, T_A = 25°C

R1 (kΩ)	R2 (MΩ)	C1 (pF)	C2 (pF)	CKI Freq (MHz)	Conditions
0	1	30	30–36	10	V_{CC} = 5V
0	1	30	30–36	4	V_{CC} = 5V
0	1	200	100–150	0.455	V_{CC} = 5V

Control Registers

CNTRL Register (Address X'00EE)

The Timer1 (T1) and MICROWIRE/PLUS control register contains the following bits:

SL1 & SL0 Select the MICROWIRE/PLUS clock divide by (00 = 2, 01 = 4, 1x = 8)

IEDG External interrupt edge polarity select (0 = Rising edge, 1 = Falling edge)

MSEL Selects G5 and G4 as MICROWIRE/PLUS signals SK and SO respectively

T1C0 Timer T1 Start/Stop control in timer modes 1 and 2
T1 Underflow Interrupt Pending Flag in timer mode 3

T1C1 Timer T1 mode control bit

T1C2 Timer T1 mode control bit

T1C3 Timer T1 mode control bit

T1C3	T1C2	T1C1	T1C0	MSEL	IEDG	SL1	SL0

Bit 7 Bit 0

PSW Register (Address X'00EF)

The PSW register contains the following select bits:

GIE Global interrupt enable (enables interrupts)

EXEN Enable external interrupt

BUSY MICROWIRE/PLUS busy shifting flag

EXPND External interrupt pending

T1ENA Timer T1 Interrupt Enable for Timer Underflow or T1A Input capture edge

T1PNDA Timer T1 Interrupt Pending Flag (Autoreload RA in mode 1, T1 Underflow in Mode 2, T1A capture edge in mode 3)

C Carry Flag

HC Half Carry Flag

HC	C	T1PNDA	T1ENA	EXPND	BUSY	EXEN	GIE

Bit 7 Bit 0

The Half-Carry flag is also affected by all the instructions that affect the Carry flag. The SC (Set Carry) and RC (Reset Carry) instructions will respectively set or clear both the carry flags. In addition to the SC and RC instructions, ADC, SUBC, RRC and RLC instructions affect the Carry and Half Carry flags.

ICNTRL Register (Address X'00E8)

The ICNTRL register contains the following bits:

T1ENB Timer T1 Interrupt Enable for T1B Input capture edge

T1PNDB Timer T1 Interrupt Pending Flag for T1B capture edge

μWEN Enable MICROWIRE/PLUS interrupt

μWPND MICROWIRE/PLUS interrupt pending

T0EN Timer T0 Interrupt Enable (Bit 12 toggle)

T0PND Timer T0 Interrupt pending

LPEN L Port Interrupt Enable (Multi-Input Wake up/Interrupt)

Bit 7 could be used as a flag

Unused	LPEN	T0PND	T0EN	μWPND	μWEN	T1PNDB	T1ENB

Bit 7 Bit 0

T2CNTRL Register (Address X'00C6)

The T2CNTRL register contains the following bits:

T2ENB Timer T2 Interrupt Enable for T2B Input capture edge

T2PNDB Timer T2 Interrupt Pending Flag for T2B capture edge

T2ENA Timer T2 Interrupt Enable for Timer Underflow or T2A Input capture edge

T2PNDA Timer T2 Interrupt Pending Flag (Auto reload RA in mode 1, T2 Underflow in mode 2, T2A capture edge in mode 3)

T2C0 Timer T2 Start/Stop control in timer modes 1 and 2 Timer T2 Underflow Interrupt Pending Flag in timer mode 3

T2C1 Timer T2 mode control bit

T2C2 Timer T2 mode control bit

T2C3 Timer T2 mode control bit

T2C3	T2C2	T2C1	T2C0	T2PNDA	T2ENA	T2PNDB	T2ENB

Bit 7 Bit 0

Timers

The device contains a very versatile set of timers (T0, T1, T2). All timers and associated autoreload/capture registers power up containing random data.

TIMER T0 (IDLE TIMER)

The device supports applications that require maintaining real time and low power with the IDLE mode. This IDLE mode support is furnished by the IDLE timer T0, which is a 16-bit timer. The Timer T0 runs continuously at the fixed rate of the instruction cycle clock, t_C. The user cannot read or write to the IDLE Timer T0, which is a count down timer.

The Timer T0 supports the following functions:

- Exit out of the Idle Mode (See Idle Mode description)
- Start up delay out of the HALT mode

The IDLE Timer T0 can generate an interrupt when the thirteenth bit toggles. This toggle is latched into the T0PND pending flag, and will occur every 4 ms at the maximum clock frequency (t_C = 1 μs). A control flag T0EN allows the interrupt from the thirteenth bit of Timer T0 to be enabled or disabled. Setting T0EN will enable the interrupt, while resetting it will disable the interrupt.

TIMER T1 AND TIMER T2

The device has a set of two powerful timer/counter blocks, T1 and T2. The associated features and functioning of a timer block are described by referring to the timer block Tx. Since the two timer blocks, T1 and T2 are identical, all comments are equally applicable to either of the two timer blocks.

Each timer block consists of a 16-bit timer, Tx, and two supporting 16-bit autoreload/capture registers, RxA and RxB. Each timer block has two pins associated with it, TxA and TxB. The pin TxA supports I/O required by the timer block, while the pin TxB is an input to the timer block. The powerful and flexible timer block allows the device to easily perform all timer functions with minimal software overhead. The timer block has three operating modes: Processor Independent PWM mode, External Event Counter mode, and Input Capture mode.

The control bits TxC3, TxC2, and TxC1 allow selection of the different modes of operation.

Mode 1. Processor Independent PWM Mode

As the name suggests, this mode allows the device to generate a PWM signal with very minimal user intervention. The user only has to define the parameters of the PWM signal (ON time and OFF time). Once begun, the timer block will continuously generate the PWM signal completely independent of the microcontroller. The user software services the timer block only when the PWM parameters require updating.

In this mode the timer Tx counts down at a fixed rate of tc. Upon every underflow the timer is alternately reloaded with the contents of supporting registers, RxA and RxB. The very first underflow of the timer causes the timer to reload from the register RxA. Subsequent underflows cause the timer to be reloaded from the registers alternately beginning with the register RxB.

The Tx Timer control bits, TxC3, TxC2 and TxC1 set up the timer for PWM mode operation.

Figure 8 shows a block diagram of the timer in PWM mode.

The underflows can be programmed to toggle the TxA output pin. The underflows can also be programmed to generate interrupts.

Underflows from the timer are alternately latched into two pending flags, TxPNDA and TxPNDB. The user must reset these pending flags under software control. Two control enable flags, TxENA and TxENB, allow the interrupts from the timer underflow to be enabled or disabled. Setting the timer enable flag TxENA will cause an interrupt when a timer underflow causes the RxA register to be reloaded into the timer. Setting the timer enable flag TxENB will cause an interrupt when a timer underflow causes the RxB register to be reloaded into the timer. Resetting the timer enable flags will disable the associated interrupts.

Either or both of the timer underflow interrupts may be enabled. This gives the user the flexibility of interrupting once per PWM period on either the rising or falling edge of the PWM output. Alternatively, the user may choose to interrupt on both edges of the PWM output.

Mode 2. External Event Counter Mode

This mode is quite similar to the processor independent PWM mode described above. The main difference is that the timer, Tx, is clocked by the input signal from the TxA pin. The Tx timer control bits, TxC3, TxC2 and TxC1 allow the timer to be clocked either on a positive or negative edge from the TxA pin. Underflows from the timer are latched into the TxPNDA pending flag. Setting the TxENA control flag will cause an interrupt when the timer underflows.

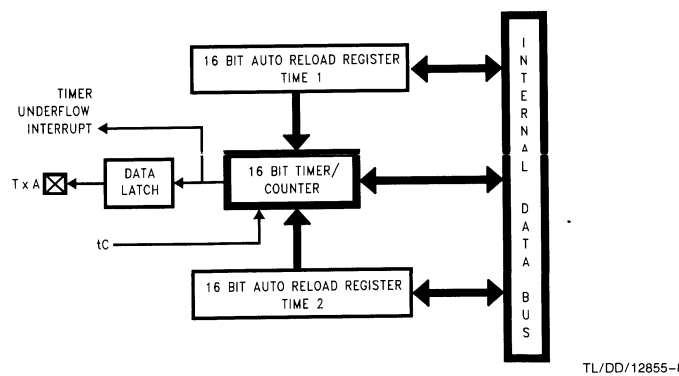
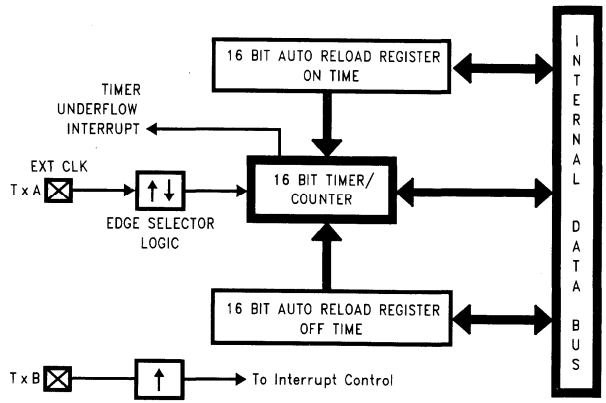


FIGURE 8. Timer in PWM Mode

Timers (Continued)



TL/DD/12855–9

FIGURE 9. Timer in External Event Counter Mode

In this mode the input pin TxB can be used as an independent positive edge sensitive interrupt input if the TxENB control flag is set. The occurrence of a positive edge on the TxB input pin is latched into the TxPNDB flag.

Figure 9 shows a block diagram of the timer in External Event Counter mode.

Note: The PWM output is not available in this mode since the TxA pin is being used as the counter input clock.

Mode 3. Input Capture Mode

The device can precisely measure external frequencies or time external events by placing the timer block, Tx, in the input capture mode.

In this mode, the timer Tx is constantly running at the fixed t_C rate. The two registers, RxA and RxB, act as capture registers. Each register acts in conjunction with a pin. The register RxA acts in conjunction with the TxA pin and the register RxB acts in conjunction with the TxB pin.

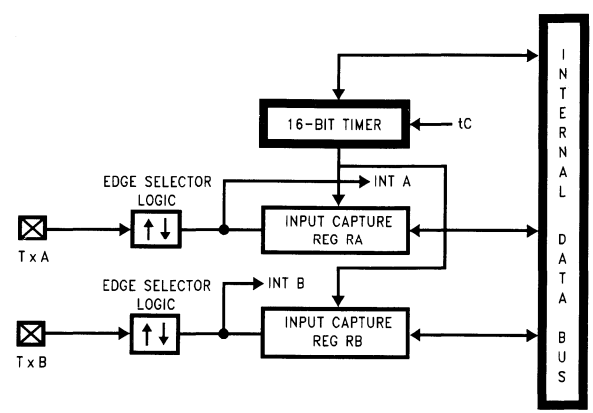
The timer value gets copied over into the register when a trigger event occurs on its corresponding pin. Control bits, TxC3, TxC2 and TxC1, allow the trigger events to be specified either as a positive or a negative edge. The trigger condition for each input pin can be specified independently.

The trigger conditions can also be programmed to generate interrupts. The occurrence of the specified trigger condition on the TxA and TxB pins will be respectively latched into the pending flags, TxPNDA and TxPNDB.

The control flag TxENA allows the interrupt on TxA to be either enabled or disabled. Setting the TxENA flag enables interrupts to be generated when the selected trigger condition occurs on the TxA pin. Similarly, the flag TxENB controls the interrupts from the TxB pin.

Underflows from the timer can also be programmed to generate interrupts. Underflows are latched into the timer TxC0 pending flag (the TxC0 control bit serves as the timer underflow interrupt pending flag in the Input Capture mode). Consequently, the TxC0 control bit should be reset when entering the Input Capture mode. The timer underflow interrupt is enabled with the TxENA control flag. When a TxA interrupt occurs in the Input Capture mode, the user must check both the TxPNDA and TxC0 pending flags in order to determine whether a TxA input capture or a timer underflow (or both) caused the interrupt.

Figure 10 shows a block diagram of the timer in Input Capture mode.



TL/DD/12855–10

FIGURE 10. Timer in Input Capture Mode

Timers (Continued)

TIMER CONTROL FLAGS

The timers T1 and T2 have identical control structures. The control bits and their functions are summarized below.

TxC0 Timer Start/Stop control in Modes 1 and 2 (Processor Independent PWM and External Event Counter), where 1 = Start, 0 = Stop Timer Underflow Interrupt Pending Flag in Mode 3 (Input Capture)

TxPNDA Timer Interrupt Pending Flag

TxPNDB Timer Interrupt Pending Flag

TxENA Timer Interrupt Enable Flag

TxENB Timer Interrupt Enable Flag
1 = Timer Interrupt Enabled
0 = Timer Interrupt Disabled

TxC3 Timer Mode Control

TxC2 Timer Mode Control

TxC1 Timer Mode Control

Capture Timer

This device contains two independent capture timers, Capture Timer 1 and Capture Timer 2. Each capture timer contains an 8-bit programmable prescaler register, a 16-bit down counter, a 16-bit input capture register, and capture edge select logic. The 16-bit down counter is clocked at a specific frequency determined by the value loaded into the prescaler register. A selected positive or negative edge transition on the capture input causes the contents of the down counter to be latched into the capture register. The values captured in the registers reflect the elapsed time between two positive or two negative transitions on the capture input. The time between a positive and negative edge (a pulse width) may be measured if the selected capture edge is switched after the first edge is captured. Each capture timer may be stopped/started under software control, and each capture timer may be configured to interrupt the microcontroller on an underflow or input capture.

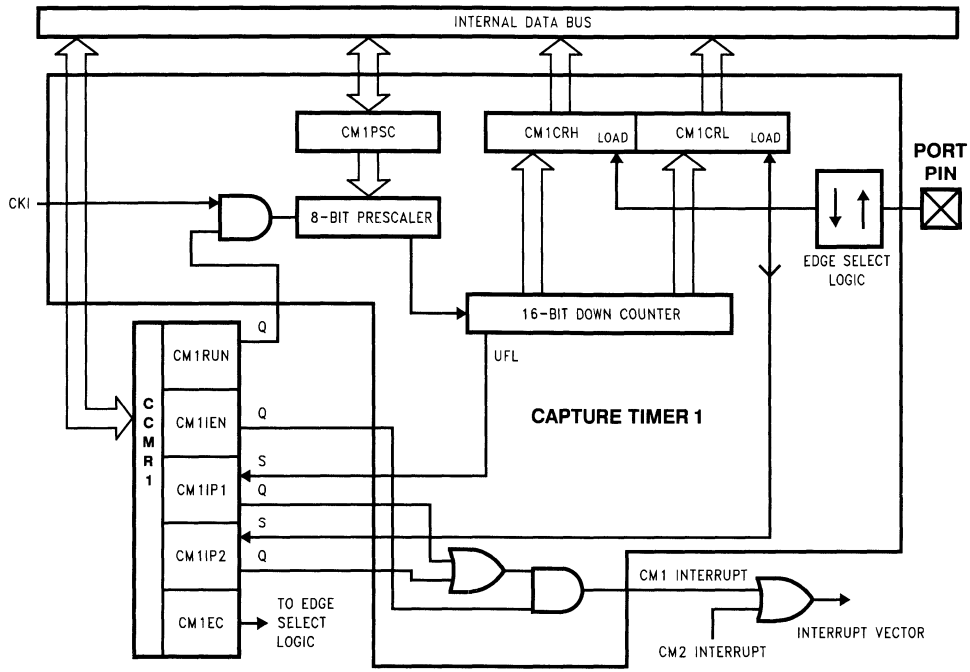
Figure 11 shows the capture timer 1 block diagram.

The timer mode control bits (TxC3, TxC2 and TxC1) are detailed below:

TABLE II. Timer Mode Control

TxC3	TxC2	TxC1	Timer Mode	Interrupt A Source	Interrupt B Source	Timer Counts On
0	0	0	MODE 2 (External Event Counter)	Timer Underflow	Positive TxB Edge	TxA Positive Edge
0	0	1	MODE 2 (External Event Counter)	Timer Underflow	Positive TxB Edge	TxA Negative Edge
1	0	1	MODE 1 (PWM) TxA Toggle	Autoreload RA	Autoreload RB	t_C
1	0	0	MODE 1 (PWM) No TxA Toggle	Autoreload RA	Autoreload RB	t_C
0	1	0	MODE 3 (Capture) Captures: TxA Positive Edge TxB Positive Edge	Positive TxA Edge or Timer Underflow	Positive TxB Edge	t_C
1	1	0	MODE 3 (Capture) Captures: TxA Positive Edge TxB Negative Edge	Positive TxA Edge or Timer Underflow	Negative TxB Edge	t_C
0	1	1	MODE 3 (Capture) Captures: TxA Negative Edge TxB Positive Edge	Negative TxA Edge or Timer Underflow	Positive TxB Edge	t_C
1	1	1	MODE 3 (Capture) Captures: TxA Negative Edge TxB Negative Edge	Negative TxA Edge or Timer Underflow	Negative TxB Edge	t_C

Timers (Continued)



TL/DD/12855–11

FIGURE 11. Capture Timer 1 Block Diagram

The registers shown in the block diagram include those for Capture Timer 1 (CM1), as well as, the capture timer 1 control register. These registers are read/writable (with the exception of the capture registers, which are read-only) and may be accessed through the data memory address/data bus. The registers are designated as:

CM1PSC Capture Timer 1 Prescaler (8-bit)

CM1CRL Capture Timer 1 Capture Register (Low-byte), read-only

CM1CRH Capture Timer 1 Capture Register (High-byte), read-only

CM2PSC Capture Timer 2 Prescaler (8-bit)

CM2CRL Capture Timer 2 Capture Register (Low-byte), read-only

CM2CRH Capture Timer 2 Capture Register (High-byte), read-only

CCMR1 Control Register for Capture Timer 1

CCMR2 Control Register for Capture Timer 2

CONTROL REGISTER BITS

The control bits for Capture Timer 1 (CM1) and Capture Timer 2 (CM2) are contained in CCMR1 and CCMR2.

The CCMR1 Register Bits are:

CM1RUN CM1 start/stop control bit (1 = start; 0 = stop)

CM1IEN CM1 interrupt enable control bit (1 = enable IRQ)

CM1IP1 CM1 interrupt pending bit 1 (1 = CM1 underflowed)

CM1IP2 CM1 interrupt pending bit 2 (1 = CM1 captured)

CM1EC Select the active edge for capture on CM1 (0 = rising, 1 = falling)

CM1TM CM1 test mode control bit (1 = special test path in test mode. This bit is reserved during normal operation, and must never be set to one.)

CM1 TM	un- used	un- used	CM1 EC	CM1 IP2	CM1 IP1	CM1 IEN	CM1 RUN
Bit 7							Bit 0

All interrupt pending bits must be reset by software.

Timers (Continued)

The CCMR2 Register Bits are:

CM2RUN CM2 start/stop control bit (1 start; 0 = stop)

CM2IEN CM2 interrupt enable control bit (1 = enable IRQ)

CM2IP1 CM2 interrupt pending bit 1 (1 = CM2 underflowed)

CM2IP2 CM2 interrupt pending bit 2 (1 = CM2 captured)

CM2EC Select the active edge for capture on CM2 (0 = rising, 1 = falling)

CM2TM CM2 test mode control bit (1 = special test path in test mode. This bit is reserved during normal operation, and must never be set to one.)

CM2 TM	un- used	un- used	CM2 EC	CM2 IP2	CM2 IP1	CM2 IEN	CM2 RUN

Bit 7 Bit 0

All interrupt pending bits must be reset by software.

FUNCTIONAL DESCRIPTION

The capture timer is used to determine the time between events, where an event is simply a selected edge transition on the capture input. The resolution of the time measurement is dependent on the frequency at which the down counter is clocked. The value loaded into the prescaler controls this frequency.

The prescaler is clocked by CKI, while the down counter is clocked on every underflow of the prescaler. This means the prescaler simply divides the CKI clock before it is fed into the down counter. The prescaler register must be loaded with a value corresponding to the CKI divisor needed to produce the desired down counter clock. The appropriate prescaler value can be determined using the following equation:

Down Counter Clock Frequency = CKI/(CMxPSC + 1)

The capture input signal is set up by configuring the port pin associated with the capture timer as an input. The edge select bit for the capture input is then set or reset according to the desired transition. If the pin is configured as an input, the appropriate external transition will cause a capture. If the pin is configured as an output, toggling the data register bit will cause a capture. If interrupts are used, the capture timer interrupt pending bits are cleared and the capture timer interrupt enable bit is set. Both interrupt sources, down counter underflow and input capture edge, are enabled/disabled with the same CMxIEN bit. The GIE bit must also be set to enable interrupts. The interrupt signals from the two capture timers are gated to a single 16-bit interrupt vector located at addresses 0xE6 and 0xE7.

The capture timer is started by writing a "1" to the capture timer start/stop bit. Setting this bit also enables the port pin to be the capture input to the capture timer. The internal prescaler is loaded with the contents of the prescaler register, and begins counting down. Setting the start/stop bit also loads the down counter with 0FFFF Hex. The prescaler is clocked by CKI. An underflow of the prescaler decrements the 16-bit down counter, and reloads the value from the prescaler register into the prescaler. Each additional underflow of the prescaler decrements the down counter, and reloads the prescaler from the prescaler register.

If a selected edge transition on the input capture pin occurs, the contents of the down counter are immediately latched into the capture register, the down counter is re-initialized to 0FFFF Hex, and the capture input pending flag is set. The prescaler counter is not loaded. (In order for an input transition to be guaranteed recognized, the signal on the capture input pin must have a low pulse width and a high pulse width of at least one CKI period.) If interrupts are enabled, the capture timer generates an interrupt. The prescaler and down counter continue to operate until a reset condition occurs or the capture timer start/stop bit is reset. The user must process capture interrupts faster than the capture input frequency, otherwise input captures may be lost or erroneous values may be read.

If the down counter underflows (changes state from 0000 to FFFF) before a capture input is detected, the underflow interrupt pending flag is set. If interrupts are enabled, the capture timer generates an interrupt.

The capture timer may be stopped at any time under software control by resetting the capture timer start/stop bit. A capture may occur before the start/stop bit is physically cleared, due to the fully asynchronous nature of the input capture signal. The user must ensure that the software handles this situation correctly. If the user wishes to process this capture and interrupts are being used, the capture timer interrupts should not be disabled prior to stopping the timer. If interrupts are not being used, the user should poll the capture timer pending bits after stopping the timer. If the user wishes to ignore this capture and interrupts are being used, the capture timer interrupt service routine should check that the timer is still running prior to processing capture interrupts. If the user is polling the pending flags, these flags should be cleared after the timer is stopped. The contents of the prescaler and down counter remain unchanged while the capture timer is stopped. The capture edge detect logic is disabled, and no capture takes place even if an external capture signal occurs. The capture timer may be restarted under software control by writing a "1" to the start/stop bit. This causes the prescaler and down counter to be re-initialized. The prescaler is loaded from the prescaler register, and the down counter is loaded with 0FFFF Hex.

RESET STATE

A reset signal applied to the counter block during normal operation has the following effects:

- Clear CCMR1 register
- Clear CCMR2 register
- CM1PSC, CMICRL, CM1CRH, CM2PSC, CM2CRL and CM2CRH are unaffected. (At power-on, the contents of these registers are undefined.)

The bi-directional port pins are initialized during reset as HI-Z inputs. Setting the start/stop bits connects the pins to the capture timers.

Timers (Continued)

INITIALIZATION

The user should perform the following initialization prior to starting the capture timer:

1. Reset the CMxRUN bit
2. Configure the corresponding Port bits as inputs
3. Set the edge control bits CMxEC
4. Reset CMxIP1 (CMxIP1 = 0)
5. Reset CMxIP2 (CMxIP2 = 0)
6. Load the 8-bit prescaler register CMxPSC with the desired value (from 0 to 255)
7. Set CMxIEN (if interrupts are to be used)
8. Set the Global Interrupt Enable (GIE) bit (if interrupts are to be used)
9. Set CMxRUN bit to start the capture timer

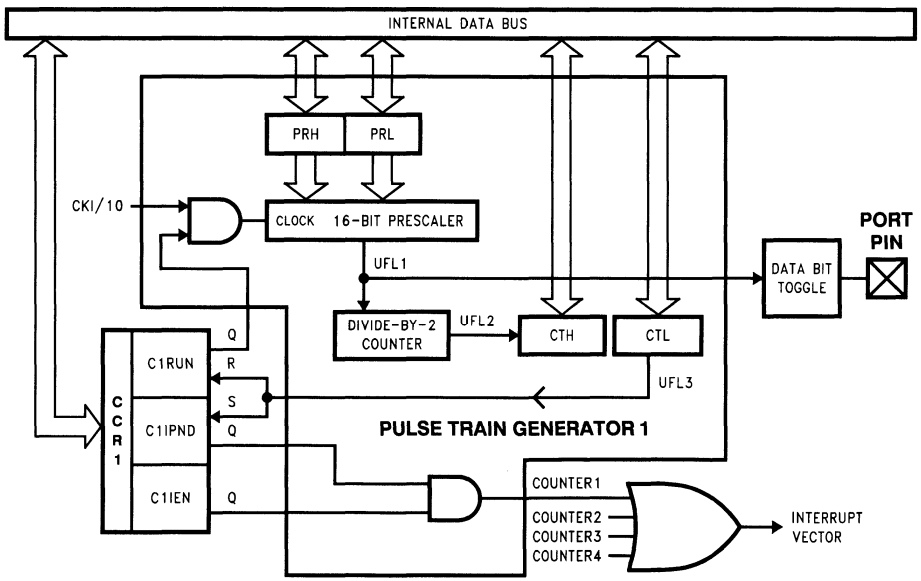
WARNING

In order to avoid erroneous interrupts, the capture timer interrupts must be disabled prior to setting/resetting the capture edge control bits (CMxEC). In addition, after selecting the interrupt edge, the pending flags must be reset before the capture interrupts are enabled or re-enabled. If the initialization sequence outlined above is followed each time the user alters the edge control bits, the user is guaranteed to avoid erroneous interrupts.

Pulse Train Generators

This device contains four independent pulse train generators. Each individual generator is controlled by a corresponding 16-bit counter. Each counter has a 16-bit prescaler and a 16-bit count register. Each counter may be configured to output a selected number of 50% duty cycle pulses. The contents of the prescaler determine the width of the output pulses, and the value of the count register determines the number of pulses. Each counter may be stopped/started under software control, and each counter may be configured to interrupt the microcontroller on an underflow.

Figure 12 shows the pulse train generator 1 block diagram.



TL/DD/12855–12

FIGURE 12. Pulse Train Generator 1 Block Diagram

Pulse Train Generators (Continued)

The four 8-bit registers shown in each individual counter in the block diagram constitute a 16-bit prescaler and a 16-bit count register. These registers are all read/writable and may be accessed through the data memory address/data bus. The registers are designated as:

CxPRL Low-byte of the Prescaler

CxPRH High-byte of the Prescaler

CxCTL Low-byte of the Count Register

CxCTH High-byte of the Count Register

CONTROL REGISTER BITS

The control bits for Counter 1 and Counter 2 are contained in the CCR1 register. The CCR1 Register bits are:

C1RUN COUNTER1 start/stop control bit (1 = start; 0 = stop)

C1IEN COUNTER1 interrupt enable control bit (1 = enable IRQ)

C1IPND COUNTER1 interrupt pending bit (1 = counter 1 underflowed)

C1TM COUNTER1 test mode control bit (1 = special test path in test mode. This bit is reserved during normal operation, and must never be set to one.)

C2RUN COUNTER2 start/stop control bit (1 = start; 0 = stop)

C2IEN COUNTER2 interrupt enable control bit (1 = enable IRQ)

C2IPND COUNTER2 interrupt pending bit (1 = counter 2 underflowed)

C2TM COUNTER2 test mode control bit (1 = special test path. This bit is reserved during normal operation, and must never be set to one.)

All interrupt pending bits must be reset by software.

C2TM	C2 IPND	C2 IEN	C2 RUN	C1TM	C1 IPND	C1 IEN	C1 RUN

Bit 7 Bit 0

The control bits for Counter 3 and Counter 4 are contained in the CCR2 register. The CCR2 Register bits are:

C3RUN COUNTER3 start stop control bit (1 = start; 0 = stop)

C3IEN COUNTER3 interrupt enable control bit (1 = enable IRQ)

C3IPND COUNTER3 interrupt pending Bit (1 = counter 3 underflowed)

C3TM COUNTER3 test mode control bit (1 = special test path. This bit is reserved during normal operation, and must never be set to one.)

C4RUN COUNTER4 start/stop control bit (1 = start; 0 = stop)

C4IEN COUNTER4 interrupt enable control bit (1 = enable IRQ)

C4IPND COUNTER4 interrupt pending bit (1 = counter 4 underflowed

C4TM COUNTER4 test mode control bit (1 = special test path. This bit is reserved during normal operation, and must never be set to one.)

C4TM	C4 IPND	C4 IEN	C4 RUN	C3TM	C3 IPND	C3 IEN	C3 RUN

Bit 7 Bit 0

All interrupt pending bits must be reset by software.

FUNCTIONAL DESCRIPTION

The pulse train generator may be used to produce a series of output pulses of a given width. The high/low time of a pulse is determined by the contents of the prescaler. The number of pulses in a series is determined by the contents of the count register.

The prescaler is loaded with a value corresponding to the desired width of the output pulse (t_w). The high time and low time of the output signal are each equal to t_w, therefore the output signal produced has a 50% duty cycle and a period equal to 2 * t_w. The appropriate prescaler value can be determined using the following equation:

$$t_w = [(PRH * 256) + PRL + 1] * t_c$$

Since PRH and PRL are both 8-bit registers, this equation allows a maximum t_w of 65536 t_c and a minimum t_w of one t_c. The internal prescaler is automatically loaded from PRH and PRL when the counter start/stop bit is set.

The count register is loaded with a value corresponding to the desired number of output pulses. The appropriate count value is calculated with the following equation:

$$\text{Number of Pulses} = CTH * 256 + CTL + 1$$

The port pin associated with the counter OUT signal is configured in software as an output, and preset to the desired start logic level. If interrupts are to be used, the counter interrupt pending bit is cleared and the interrupt enable bit is set. The GIE bit must also be set to enable interrupts. The interrupt signals from the four counters are gated to a single interrupt vector located at addresses 0xF0–0xF1.

The counter is started by writing a "1" to the counter start/stop bit. This resets the divide-by-2 counter which produces the clock signal for the counter register from the prescaler underflow (See *Figure 12*). It also reloads the internal prescaler and starts the prescaler counting down on the next rising edge of t_c. The prescaler is clocked on the rising edge of t_c to ensure synchronization. Each subsequent rising edge of t_c causes the prescaler to be decremented. When the prescaler underflows, UFL1 is generated (see *Figure 13*). This signal causes the port pin to toggle. In addition, the internal prescaler is reloaded with the value from the PRH and PRL registers. Each additional underflow of the prescaler causes the port pin to toggle and reloads the internal prescaler.

Every second underflow of the prescaler generates the signal UFL2. (UFL2 occurs at half the frequency of UFL1, or once per output pulse.) This signal, UFL2, decrements the count register. Therefore, the count registers are decremented once per output pulse.

Pulse Train Generators (Continued)

The underflow of the counter register produces the signal UFL3. This signal stops the counter by resetting the counter start/stop bit, and sets the counter interrupt pending flag. If the counter interrupt is enabled, an interrupt occurs.

The counter may be stopped at any time under software control by resetting the counter start/stop bit. The contents of the count register and the output on the associated port pin are frozen. The counter may be restarted under software control by setting the start/stop bit. The internal prescaler is automatically reloaded from PRH and PRL when the counter start/stop bit is set, therefore a full width pulse will be generated before the output is toggled. The user may also choose to alter the logic level on the port pin before restarting. This is done by initializing the associated port pin data register bit. A counter underflow may occur before the start/stop bit is physically cleared by software. The user must ensure that the software handles this situation correctly. If the user wishes to process this underflow and interrupts are being used, the counter interrupts should not be disabled prior to stopping the timer. If interrupts are not being used, the user should poll the counter pending bits after stopping the timer. If the user wishes to ignore this underflow and interrupts are being used, the counter interrupt should be disabled prior to stopping the timer. If the user is polling the pending flags, these flags should be cleared after the timer is stopped.

If the default level of the output pin is high (associated port data register bit is set to "1") and the counter is stopped during a low level, the low level becomes the default level. The software must reinitialize the port pin to a high level before restarting if necessary. The programmer may also have to adjust the counter value.

RESET STATE

A reset signal applied to the pulse train generator block during normal operation has the following effects:

- Counting stops immediately
- Interrupt enable bit is reset to zero
- Counter start/stop bit is reset to zero
- Interrupt pending bit is reset to zero
- Test mode control bit is reset to zero
- PRL, PRH, CTL and CTH are unaffected (At power-on reset, the contents of the prescaler and count register are undefined.)
- Divide-by-2 counter is reset
- The bi-directional port pins are initialized during reset as HI-Z inputs. The appropriate bits must be initialized as outputs, in order to route the Counter OUT signals to the port pins.

INITIALIZATION

The user should perform the following initialization prior to starting the counter:

1. Load PRL register
2. Load PRH register
3. Load CTL register
4. Load CTH register
5. Reset CxIPND bit
6. Set CxIEN (if interrupt is to be used)
7. Configure the associated port bit as an output (if OUT is to be used)
8. Set the Global Interrupt Enable (GIE) bit (if interrupt is to be used)
9. Set CxRUN bit to start counter

Multiply/Divide

This device contains a multiply/divide block. This block supports a 1 byte x 2 bytes (3 bytes result) multiply or a 3 bytes/2 bytes (2 bytes result) divide operation. The multiply or divide operation is executed by setting control bits located in the multiply/divide control register. The multiply or divide operands must be placed into the appropriate memory mapped locations before the operation is initiated.

TABLE III. Multiply/Divide Registers

Register Name (Address)	Multiplication Assignment		Division Assignment	
	Before Operation	After Operation	Before Operation	After Operation
MDR1 (xx98)	Unused	Unchanged	Low byte of dividend	Low byte of result
MDR2 (xx99)	Multiplier	Low byte of result	Middle byte of dividend	High byte of result
MDR3 (xx9A)		Middle byte of result	High byte of dividend	Undefined
MDR4 (xx9B)	Low byte of multiplicand	High byte of result	Low byte of divisor	Low byte of divisor
MDR5 (xx9C)	High byte of multiplicand	Unchanged	High byte of divisor	High byte of divisor

Multiply/Divide (Continued)

CONTROL REGISTER BITS

The Multiply/Divide control register (MDCR) is located at address xx9D. It has the following bit assignments:

MULT Start Multiplication Operation (1 = start)

DIV Start Division Operation (1 = start)

DIVOVF Division Overflow (if the result of a division is greater than 16 bits or the user attempted to divide by zero; 1 = error)

Rsvd	Rsvd	Rsvd	Rsvd	Rsvd	DIV OVF	DIV	MULT

Bit 7 Bit 0

After the appropriate MDR registers are loaded, the MULT and DIV start bits are set by the user to start a multiply or divide operation. The division operation has priority, if both bits are set simultaneously. The MULT and DIV bits are BOTH automatically cleared by hardware at the end of a divide or multiply operation. Each division operation causes the DIVOVF flag to be set/reset as appropriate. The DIVOVF flag is cleared following a multiplication operation. DIVOVF is a read-only bit. The MULT and DIV bits are read/writable. Bits 3–7 in MDCR should not be used, as the MULT and DIV operations will change their values.

MULTIPLY/DIVIDE OPERATION

For the multiply operation, the multiplicand is placed at addresses xx9B and xx9C. The multiplier is placed at address xx99. For the divide operation, the dividend is placed at addresses xx98 to xx9A and the divisor is placed at addresses xx9B to xx9C. In both operations, all operands are interpreted as unsigned values. The divide or multiply operation is started by setting the appropriate MDCR bit. If both the MULT and DIV bits are set, the microcontroller performs a divide operation. (The user is not required to read or clear the DIVOVF error bit prior to beginning a new multiply/divide operation. This bit is ignored during subsequent operations. However, the next divide operation will overwrite the error flag as appropriate, and the next multiply operation will clear it.)

The multiply operation requires 1 instruction cycle to complete. The divide operation requires 2 instruction cycles to complete. A divide by zero or a division which produces an overflow requires only 1 instruction cycle to execute. The MDR1 through MDR5 registers and the MDCR register can not be read from or written to during a multiply or divide operation. Any attempt to write into these registers will be ignored. Any attempt to read these registers will return undefined data.

The result of a multiply is placed in addresses xx99-xx9B. The result of a divide is placed in addresses xx98-xx99. If a division by zero is attempted or if the resulting quotient of a divide operation is more than 16 bits long, then the DIVOVF bit is set in the multiply/divide control register. The dividend and the divisor are left unchanged. The divide operation always causes the DIVOVF flag to be set or reset as appropriate. The DIVOVF flag is cleared following a multiply operation.

RESET STATE

A reset signal applied to the device during normal operation has the following affects:

MDCR is cleared, and any operation in progress is stopped.

MDR1 through MDR5 are undefined.

Power Save Modes

The device offers the user two power save modes of operation: HALT and IDLE. In the HALT mode, all microcontroller activities are stopped. In the IDLE mode, the on-board oscillator circuitry and timer T0 are active but all other microcontroller activities are stopped. In either mode, all on-board RAM, registers, I/O states, and timers (with the exception of T0) are unaltered.

HALT MODE

The device can be placed in the HALT mode by writing a "1" to the HALT flag (G7 data bit). All microcontroller activities, including the clock and timers, are stopped. In the HALT mode, the power requirements of the device are minimal and the applied voltage (V_{CC}) may be decreased to V_r (V_r = 2.0V) without altering the state of lhe machine.

The device supports two different ways of exiting the HALT mode. The first method of exiting the HALT mode is with the Multi-Input Wakeup feature on the L port. The second method of exiting the HALT mode is by pulling the $\overline{\text{RESET}}$ pin low.

Since a crystal or ceramic resonator may be selected as the oscillator, the Wakeup signal is not allowed to start the chip running immediately since crystal oscillators and ceramic resonators have a delayed start up time to reach full amplitude and frequency stability. The IDLE timer is used to generate a fixed delay to ensure that the oscillator has indeed stabilized before allowing instruction execution. In this case, upon detecting a valid Wakeup signal, only the oscillator circuitry is enabled. The IDLE timer is loaded with a value of 256 and is clocked with the t_c instruction cycle clock. The t_c clock is derived by dividing the oscillator clock down by a factor of 10. The Schmitt trigger following the CKI inverter on the chip ensures that the IDLE timer is clocked only when the oscillator has a sufficiently large amplitude to meet the Schmitt trigger specifications. This Schmitt trigger is not part of the oscillator closed loop. The startup timeout from the IDLE timer enables the clock signals to be routed to the rest of the chip.

The devices have two mask options associated with the HALT mode. The first mask option enables the HALT mode feature, while the second mask option disables the HALT mode. With the HALT mode enable mask option, the device will enter and exit the HALT mode as described above. With the HALT disable mask option, the device cannot be placed in the HALT mode (writing a "1" to the HALT flag will have no effect, the HALT flag will remain "0").

IDLE MODE

The device is placed in the IDLE mode by writing a "1" to the IDLE flag (G6 data bit). In this mode, all activities, except the associated on-board oscillator circuitry and the IDLE Timer T0, are stopped.

Power Save Modes (Continued)

As with the HALT mode, the device can be returned to normal operation with a reset, or with a Multi-Input Wake up from the L Port. Alternately, the microcontroller resumes normal operation from the IDLE mode when the thirteenth bit (representing 4.096 ms at internal clock frequency of 10 MHz, $t_C = 1\ \mu s$) of the IDLE Timer toggles.

This toggle condition of the thirteenth bit of the IDLE Timer T0 is latched into the T0PND pending flag.

The user has the option of being interrupted with a transition on the thirteenth bit of the IDLE Timer T0. The interrupt can be enabled or disabled via the T0EN control bit. Setting the T0EN flag enables the interrupt and vice versa.

The user can enter the IDLE mode with the Timer T0 interrupt enabled. In this case, when the T0PND bit gets set, the device will first execute the Timer T0 interrupt service routine and then return to the instruction following the "Enter Idle Mode" instruction.

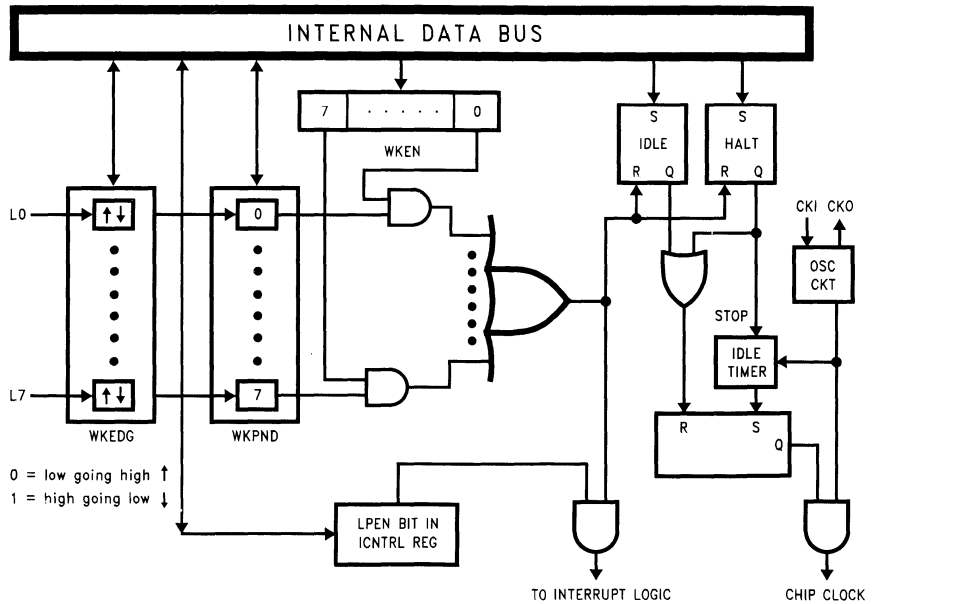
Alternatively, the user can enter the IDLE mode with the IDLE Timer T0 interrupt disabled. In this case, the device will resume normal operation with the instruction immediately following the "Enter IDLE Mode" instruction.

Note: It is necessary to program two NOP instructions following both the set HALT mode and set IDLE mode instructions. These NOP instructions are necessary to allow clock resynchronization following the HALT or IDLE modes.

Multi-Input Wakeup

The Multi-Input Wake Up feature is used to return (wake up) the device from either the HALT or IDLE modes. Alternately Multi-Input Wake Up/Interrupt feature may also be used to generate up to 8 edge selectable external interrupts.

Figure 13 shows the Multi-Input Wake Up logic.



TL/DD/12855–13

FIGURE 13. Multi-Input Wake Up Logic

Multi-Input Wakeup (Continued)

The Multi-Input Wake Up feature utilizes the L Port. The user selects which particular L port bit (or combination of L Port bits) will cause the device to exit the HALT or IDLE modes. The selection is done through the register WKEN. The register WKEN is an 8-bit read/write register, which contains a control bit for every L port bit. Setting a particular WKEN bit enables a Wake Up from the associated L port pin.

The user can select whether the trigger condition on the selected L Port pin is going to be either a positive edge (low to high transition) or a negative edge (high to low transition). This selection is made via the register WKEDG, which is an 8-bit control register with a bit assigned to each L Port pin. Setting the control bit will select the trigger condition to be a negative edge on that particular L Port pin. Resetting the bit selects the trigger condition to be a positive edge. Changing an edge select entails several steps in order to avoid a Wake Up condition as a result of the edge change. First, the associated WKEN bit should be reset, followed by the edge select change in WKEDG. Next, the associated WKPND bit should be cleared, followed by the associated WKEN bit being reenabled.

An example may serve to clarify this procedure. Suppose we wish to change the edge select from positive (low going high) to negative (high going low) for L Port bit 5, where bit 5 has previously been enabled for an input interrupt. The program would be as follows:

```
RBIT   5,   WKEN
SBIT   5,   WKEDG
RBIT   5,   WKPND
SB1T   5,   WKEN
```

If the L port bits have been used as outputs and then changed to inputs with Multi-Input Wake Up/Interrupt, a safety procedure should also be followed to avoid wakeup conditions. After the selected L port bits have been changed from output to input but before the associated WKEN bits are enabled, the associated edge select bits in WKEDG should be set or reset for the desired edge selects, followed by the associated WKPND bits being cleared,

This same procedure should be used following reset, since the L port inputs are left floating as a result of reset.

The occurrence of the selected trigger condition for Multi-Input Wake Up is latched into a pending register called WKPND. The respective bits of the WKPND register will be set on the occurrence of the selected trigger edge on the corresponding Port L pin. The user has the responsibility of clearing these pending flags. Since WKPND is a pending register for the occurrence of selected wake up conditions, the device will not enter the HALT mode if any Wake Up bit is both enabled and pending. Consequently, the user must clear the pending flags before attempting to enter the HALT mode.

WKEN, WKPND and WKEDG are all read/write registers, and are cleared at reset.

PORT L INTERRUPTS

Port L provides the user with an additional eight fully selectable, edge sensitive interrupts which are all vectored into the same service subroutine.

The interrupt from Port L shares logic with the wake up circuitry. The register WKEN allows interrupts from Port L to be individually enabled or disabled. The register WKEDG specifies the trigger condition to be either a positive or a negative edge. Finally, the register WKPND latches in the pending trigger conditions.

The GIE (Global Interrupt Enable) bit enables the interrupt function.

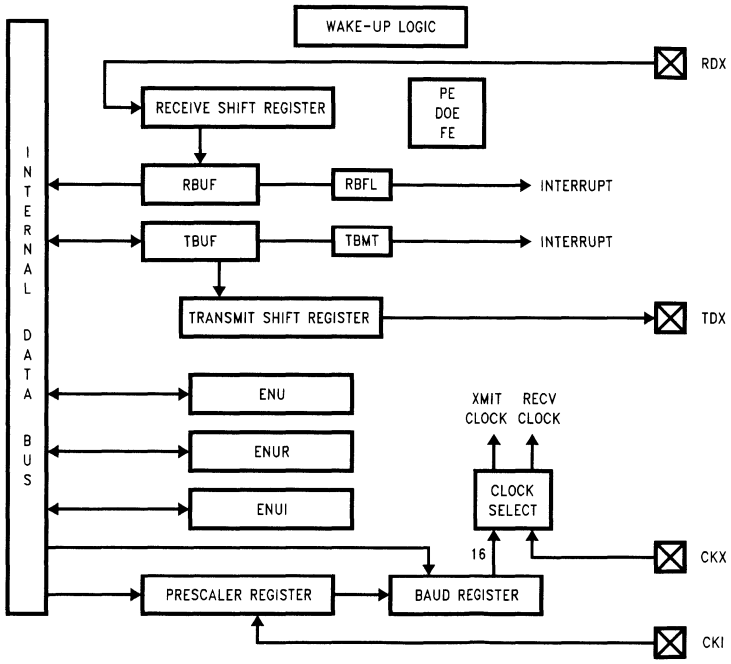
A control flag, LPEN, functions as a global interrupt enable for Port L interrupts. Setting the LPEN flag will enable interrupts and vice versa. A separate global pending flag is not needed since the register WKPND is adequate.

Since Port L is also used for waking the device out of the HALT or IDLE modes, the user can elect to exit the HALT or IDLE modes either with or without the interrupt enabled. If he elects to disable the interrupt, then the device will restart execution from the instruction immediately following the instruction that placed the microcontroller in the HALT or IDLE modes. In the other case, the device will first execute the interrupt service routine and then revert to normal operation. (See HALT MODE for clock option wake up information.)

UART

The device contains a full-duplex software programmable UART. The UART (*Figure 14*) consists of a transmit shift register, a receive shift register and seven addressable registers, as follows: a transmit buffer register (TBUF), a receiver buffer register (RBUF), a UART control and status register (ENU), a UART receive control and status register (ENUR), a UART interrupt and clock source register (ENUI), a prescaler select register (PSR) and baud (BAUD) register. The ENU register contains flags for transmit and receive functions; this register also determines the length of the data frame (7, 8 or 9 bits), the value of the ninth bit in transmission, and parity selection bits. The ENUR register flags framing, data overrun and parity errors while the UART is receiving.

Other functions of the ENUR register include saving the ninth bit received in the data frame, enabling or disabling the UART's attention mode of operation and providing additional receiver/transmitter status information via RCVG and XMTG bits. The determination of an internal or external clock source is done by the ENUI register, as well as selecting the number of stop bits and enabling or disabling transmit and receive interrupts. A control flag in this register can also select the UART mode of operation: asynchronous or synchronous.



TL/DD/12855-14

FIGURE 14. UART Block Diagram

UART (Continued)

UART CONTROL AND STATUS REGISTERS

The operation of the UART is programmed through three registers: ENU, ENUR and ENUI. The function of the individual bits in these registers is as follows:

ENU-UART Control and Status Register (Address at 0BA)

PEN	PSEL1	XBIT9/ PSEL0	CHL1	CHL0	ERR	RBFL	TBMT
0RW	0RW	0RW	0RW	0RW	0R	0R	IR

Bit 7 Bit 0

ENUR-UART Receive Control and Status Register (Address at 0BB)

DOE	FE	PE	SPARE	RBIT9	ATTN	XMTG	RCVG
0RD	0RD	0RD	0RW*	0R	0RW	0R	0R

Bit 7 Bit 0

ENUI-UART Interrupt and Clock Source Register (Address at 0BC)

STP2	STP78	ETDX	SSEL	XRCLK	XTCLK	ERI	ETI
0RW	0RW	0RW	0RW	0RW	0RW	0RW	0RW

Bit 7 Bit 0

* Bit is not used.

0 Bit is cleared on reset.

1 Bit is set to one on reset.

R Bit is read-only; it cannot be written by software.

RW Bit is read/write.

D Bit is cleared on read; when read by software as a one, it is cleared automatically. Writing to the bit does not affect its state.

DESCRIPTION OF UART REGISTER BITS

ENU—UART CONTROL AND STATUS REGISTER

TBMT: This bit is set when the UART transfers a byte of data from the TBUF register into the TSFT register for transmission. It is automatically reset when software writes into the TBUF register.

RBFL: This bit is set when the UART has received a complete character and has copied it into the RBUF register. It is automatically reset when software reads the character from RBUF.

ERR: This bit is a global UART error flag which gets set if any or a combination of the errors (DOE, FE, PE) occur.

CHL1, CHL0: These bits select the character frame format. Parity is not included and is generated/verified by hardware.

CHL1 = 0, CHL0 = 0 The frame contains eight data bits.

CHL1 = 0, CHL0 = 1 The frame continues seven data bits.

CHL1 = 1, CHL0 = 0 The frame continues nine data bits.

CHL1 = 1, CHL0 = 1 Loopback Mode selected. Transmitter output internally looped back to receiver input. Nine bit framing format is used.

XBIT9/PSEL0: Programs the ninth bit for transmission when the UART is operating with nine data bits per frame. For seven or eight data bits per frame, this bit in conjunction with PSEL1 selects parity.

PSEL1, PSEL0: Parity select bits.

PSEL1 = 0, PSEL0 = 0 Odd Parity (if Parity enabled)

PSEL1 = 0, PSEL1 = 1 Odd Parity (if Parity enabled)

PSEL1 = 1, PSEL0 = 0 Mark(1) (if Parity enabled)

PSEL1 = 1, PSEL1 = 1 Space(0) (if Parity enabled)

PEN: This bit enables/disables Parity (7- and 8-bit modes only).

PEN = 0 Parity disabled.

PEN = 1 Parity enabled.

ENUR—UART RECEIVE CONTROL AND STATUS REGISTER

RCVG: This bit is set high whenever a framing error occurs and goes low when RDX goes high.

XMTG: This bit is set to indicate that the UART is transmitting. It gets reset at the end of the last frame (end of last Stop bit).

ATTN: ATTENTION Mode is enabled while this bit is set. This bit is cleared automatically on receiving a character with data bit nine set.

RBIT9: Contains the ninth data bit received when the UART is operating with nine data bits per frame.

SPARE: Reserved for future use.

PE: Flags a Parity Error.

PE = 0 Indicates no Parity Error has been detected since the last time the ENUR register was read.

PE = 1 Indicates the occurrence of a Parity Error.

FE: Flags a Framing Error.

FE = 0 Indicates no Framing Error has been detected since the last time the ENUR register was read.

FE = 1 Indicates the occurrence of a Framing Error.

DOE: Flags a Data Overrun Error.

DOE = 0 Indicates no Data Overrun Error has been detected since the last time the ENUR register was read.

DOE = 1 Indicates the occurrence of a Data Overrun Error.

ENUI—UART INTERRUPT AND CLOCK SOURCE REGISTER

ETI: This bit enables/disables interrupt from the transmitter section.

ETI = 0 Interrupt from the transmitter is disabled.

ETI = 1 Interrupt from the transmitter is enabled.

UART (Continued)

ERI: This bit enables/disables interrupt from the receiver section.

ERI = 0 Interrupt from the receiver is disabled.

ERI = 1 Interrupt from the receiver is enabled.

XTCLK: This bit selects the clock source for the transmitter section.

XTCLK = 0 The clock source is selected through the PSR and BAUD registers.

XTCLK = 1 Signal on CKX (L1) pin is used as the clock.

XRCLK: This bit selects the clock source for the receiver section.

XRCLK = 0 The clock source is selected through the PSR and BAUD registers.

XRCLK = 1 Signal on CKX (L1) pin is used as the clock.

SSEL: UART mode select.

SSEL = 0 Asynchronous Mode.

SSEL = 1 Synchronous Mode.

ETDX: TDX (UART Transmit Pin) is the alternate function assigned to Port L pin L2; it is selected by setting ETDX bit. To simulate line break generation, software should reset ETDX bit and output logic zero to TDX pin through Port L data and configuration registers.

STP78: This bit is set to program the last Stop bit to be 7/8th of a bit in length.

STP2: This bit programs the number of Stop bits to be transmitted.

STP2 = 0 One Stop bit transmitted.

STP2 = 1 Two Stop bits transmitted.

Associated I/O Pins

Data is transmitted on the TDX pin and received on the RDX pin. TDX is the alternate function assigned to Port L pin L2; it is selected by setting ETDX (in the ENUI register) to one. RDX is an inherent function of Port L pin L3, requiring no setup.

The baud rate clock for the UART can be generated on-chip, or can be taken from an external source. Port L pin L1 (CKX) is the external clock I/O pin. The CKX pin can be either an input or an output, as determined by Port L Configuration and Data registers (Bit 1). As an input, it accepts a clock signal which may be selected to drive the transmitter and/or receiver. As an output, it presents the internal Baud Rate Generator output.

UART Operation

The UART has two modes of operation: asynchronous mode and synchronous mode.

ASYNCHRONOUS MODE

This mode is selected by resetting the SSEL (in the ENUI register) bit to zero. The input frequency to the UART is 16 times the baud rate.

The TSFT and TBUF registers double-buffer data for transmission. While TSFT is shifting out the current character on the TDX pin, the TBUF register may be loaded by software with the next byte to be transmitted. When TSFT finishes transmitting the current character the contents of TBUF are transferred to the TSFT register and the Transmit Buffer Empty Flag (TBMT in the ENU register) is set. The TBMT flag is automatically reset by the UART when software loads a new character into the TBUF register. There is also the XMTG bit which is set to indicate that the UART is transmitting. This bit gets reset at the end of the last frame (end of last Stop bit). TBUF is a read/write register.

The RSFT and RBUF registers double-buffer data being received. The UART receiver continually monitors the signal on the RDX pin for a low level to detect the beginning of a Start bit. Upon sensing this low level, it waits for half a bit time and samples again. If the RDX pin is still low, the receiver considers this to be a valid Start bit, and the remaining bits in the character frame are each sampled a single time, at the mid-bit position. Serial data input on the RDX pin is shifted into the RSFT register. Upon receiving the complete character, the contents of the RSFT register are copied into the RBUF register and the Received Buffer Full Flag (RBFL) is set. RBFL is automatically reset when software reads the character from the RBUF register. RBUF is a read only register. There is also the RCVG bit which is set high when a framing error occurs and goes low once RDX goes high. TBMT, XMTG, RBFL and RCVG are read only bits.

SYNCHRONOUS MODE

In this mode data is transferred synchronously with the clock. Data is transmitted on the rising edge and received on the falling edge of the synchronous clock.

This mode is selected by setting SSEL bit in the ENUI register. The input frequency to the UART is the same as the baud rate.

When an external clock input is selected at the CKX pin, data transmit and receive are performed synchronously with this clock through TDX/RDX pins.

If data transmit and receive are selected with the CKX pin as clock output, the device generates the synchronous clock output at the CKX pin. The internal baud rate generator is used to produce the synchronous clock. Data transmit and receive are performed synchronously with this clock.

FRAMING FORMATS

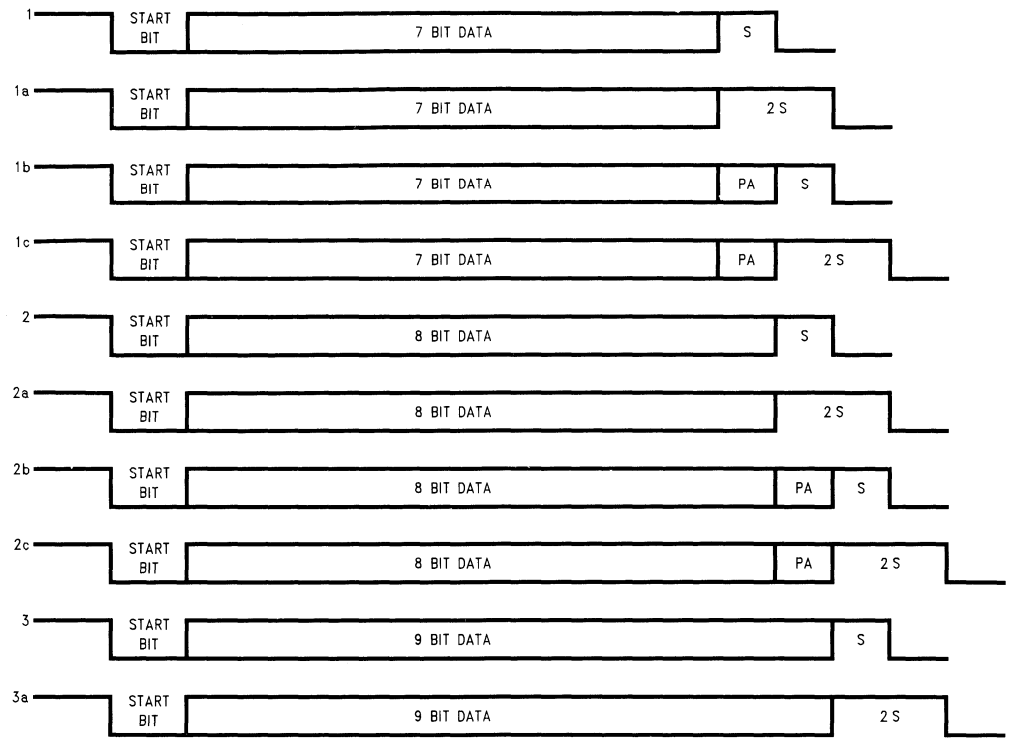
The UART supports several serial framing formats *(Figure 16)*. The format is selected using control bits in the ENU, ENUR and ENUI registers.

The first format (1,1a, 1b, 1c) for data transmission (CHL0 = 1, CHL1 = 0) consists of Start bit, seven Data bits (excluding parity) and 7/8, one or two Stop bits. In applications using parity, the parity bit is generated and verified by hardware.

The second format (CHL0 = 0, CHL1 = 0) consists of one Start bit, eight Data bits (excluding parity) and 7/8, one or two Stop bits. Parity bit is generated and verified by hardware.

The third format for transmission (CHL0 = 0, CHL1 = 1) consists of one Start bit, nine Data bits and 7/8, one or two Stop bits. This format also supports the UART "ATTENTION" feature. When operating in this format, all eight bits of TBUF and RBUF are used for data. The ninth data bit is transmitted and received using two bits in the ENU and ENUR registers, called XBIT9 and RBIT9. RBIT9 is a read only bit. Parity is not generated or verified in this mode.

UART Operation (Continued)



TL/DD/12855-15

FIGURE 15. Framing Formats

For any of the above framing formats, the last Stop bit can be programmed to be 7/8th of a bit in length. If two Stop bits are selected and the 7/8th bit is set (selected), the second Stop bit will be 7/8th of a bit in length.

The parity is enabled/disabled by PEN bit located in the ENU register. Parity is selected for 7- and 8-bit modes only. If parity is enabled (PEN = 1), the parity selection is then performed by PSEL0 and PSEL1 bits located in the ENU register.

Note that the XBIT9/PSEL0 bit located in the ENU register serves two mutually exclusive functions. This bit programs the ninth bit for transmission when the UART is operating with nine data bits per frame. There is no parity selection in this framing format. For other framing formats XBIT9 is not needed and the bit is PSEL0 used in conjunction with PSEL1 to select parity.

The frame formats for the receiver differ from the transmitter in the number of Stop bits required. The receiver only requires one Stop bit in a frame, regardless of the setting of the Stop bit selection bits in the control register. Note that an implicit assumption is made for full duplex UART operation that the framing formats are the same for the transmitter and receiver.

UART INTERRUPTS

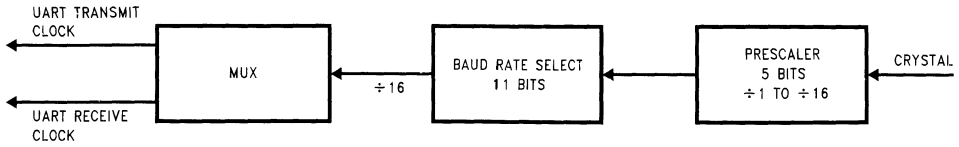
The UART is capable of generating interrupts. Interrupts are generated on Receive Buffer Full and Transmit Buffer Empty. Both interrupts have individual interrupt vectors. Two bytes of program memory space are reserved for each interrupt vector. The two vectors are located at addresses 0xEC to 0xEF Hex in the program memory space. The interrupts can be individually enabled or disabled using Enable Transmit Interrupt (ETI) and Enable Receive Interrupt (ERI) bits in the ENUI register.

The interrupt from the Transmitter is set pending, and remains pending, as long as both the TBMT and ETI bits are set. To remove this interrupt, software must either clear the ETI bit or write to the TBUF register (thus clearing the TBMT bit).

The interrupt from the receiver is set pending, and remains pending, as long as both the RBFL and ERI bits are set. To remove this interrupt, software must either clear the ERI bit or read from the RBUF register (thus clearing the RBFL bit).

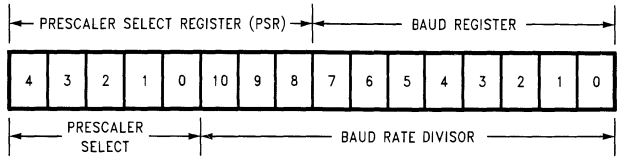
Baud Clock Generation

The clock inputs to the transmitter and receiver sections of the UART can be individually selected to come either from an external source at the CKX pin (port L, pin L1) or from a source selected in the PSR and BAUD registers. Internally, the basic baud clock is created from the oscillator frequency through a two-stage divider chain consisting of a 1–16 (increments of 0.5) prescaler and an 11-bit binary counter *(Figure 16)*.The divide factors are specified through two read/write registers shown in *Figure 17*. Note that the 11-bit Baud Rate Divisor spills over into the Prescaler Select Register (PSR). PSR is cleared upon reset.



TL/DD/12855–16

FIGURE 16. UART BAUD Clock Generation



TL/DD/12855–17

FIGURE 17. UART BAUD Clock Divisor Registers

Baud Clock Generation (Continued)

As shown in Table V, a Prescaler Factor of 0 corresponds to NO CLOCK. This condition is the UART power down mode where the UART clock is turned off for power saving purpose. The user must also turn the UART clock off when a different baud rate is chosen.

The correspondences between the 5-bit Prescaler Select and Prescaler factors are shown in Table V. There are many ways to calculate the two divisor factors, but one particularly effective method would be to achieve a 1.8432 MHz frequency coming out of the first stage. The 1.8432 MHz prescaler output is then used to drive the software programmable baud rate counter to create a 16x clock for the following baud rates: 110, 134.5, 150, 300, 600, 1200, 1800, 2400, 3600, 4800, 7200, 9600, 19200 and 38400 (Table IV). Other baud rates may be created by using appropriate divisors. The 16x clock is then divided by 16 to provide the rate for the serial shift registers of the transmitter and receivers.

TABLE IV. Baud Rate Divisors (1.8432 MHz Prescaler Output)

Baud Rate	Baud Rate Divisor − 1 (N-1)
110 (110.03)	1046
134.5 (134.58)	855
150	767
300	383
600	191
1200	95
1800	63
2400	47
3600	31
4800	23
7200	15
9600	11
19200	5
38400	2

Note: The entries in Table IV assume a prescaler output of 1.8432 MHz. In asynchronous mode the baud rate could be as high as 625k.

TABLE V. Prescaler Factors

Prescaler Select	Prescaler Factor
00000	NO CLOCK
00001	1
00010	1.5
00011	2
00100	2.5
00101	3
00110	3.5
00111	4
01000	4.5
01001	5
01010	5.5
01011	6
01100	6.5
01101	7
01110	7.5
01111	8
10000	8.5
10001	9
10010	9.5
10011	10
10100	10.5
10101	11
10110	11.5
10111	12
11000	12.5
11001	13
11010	13.5
11011	14
11100	14.5
11101	15
11110	15.5
11111	16

Baud Clock Generation (Continued)

As an example, considering Asynchronous Mode and a CKI clock of 4.608 MHz, the prescaler factor selected is:

$$4.608/1.8432 = 2.5$$

The 2.5 entry is available in Table V. The 1.8432 MHz prescaler output is then used with proper Baud Rate Divisor (Table V) to obtain different baud rates. For a baud rate of 19200 e.g., the entry in Table IV is 5.

$$N - 1 = 5 \text{ (N} - 1 \text{ is the value from Table IV)}$$

$$N = 6 \text{ (N is the Baud Rate Divisor)}$$

$$\text{Baud Rate} = 1.8432 \text{ MHz}/(16 \times 6) = 19200$$

The divide by 16 is performed because in the asynchronous mode, the input frequency to the UART is 16 times the baud rate. The equation to calculate baud rates is given below.

The actual Baud Rate may be found from:

$$BR = Fc/(16 \times N \times P)$$

Where:

BR is the Baud Rate

Fc is the CKI frequency

N is the Baud Rate Divisor (Table IV).

P is the Prescaler Divide Factor selected by the value in the Prescaler Select Register (Table V)

Note: In the Synchronous Mode, the divisor 16 is replaced by two.

Example:

Asynchronous Mode:

Crystal Frequency = 5 MHz

Desired baud rate = 9600

Using the above equation N × P can be calculated first.

$$N \times P = (5 \times 106)/(16 \times 9600) = 32.552$$

Now 32.552 is divided by each Prescaler Factor (Table V) to obtain a value closest to an integer. This factor happens to be 6.5 (P = 6.5).

$$N = 32.552/6.5 = 5.008 \text{ (N} = 5)$$

The programmed value (from Table IV) should be 4 (N − 1).

Using the above values calculated for N and P:

$$BR = (5 \times 106)/(16 \times 5 \times 6.5) = 9615.384$$

$$\% \text{ error} = (9615.385 - 9600)/9600 = 0.16$$

Effect of HALT/IDLE

The UART logic is reinitialized when either the HALT or IDLE modes are entered. This reinitialization sets the TBMT flag and resets all read only bits in the UART control and status registers. Read/Write bits remain unchanged. The Transmit Buffer (TBUF) is not affected, but the Transmit Shift register (TSFT) bits are set to one. The receiver registers RBUF and RSFT are not affected.

The device will exit from the HALT/IDLE modes when the Start bit of a character is detected at the RDX (L3) pin. This feature is obtained by using the Multi-Input Wakeup scheme provided on the device.

Before entering the HALT or IDLE modes the user program must select the Wakeup source to be on the RDX pin. This selection is done by setting bit 3 of WKEN (Wakeup Enable) register. The Wakeup trigger condition is then selected to be high to low transition. This is done via the WKEDG register (Bit 3 is one).

If the device is halted and crystal oscillator is used, the Wake Up signal will not start the chip running immediately because of the finite start up time requirement of the crystal oscillator. The idle timer (T0) generates a fixed (256 t_C) delay to ensure that the oscillator has indeed stabilized before allowing the device to execute code. The user has to consider this delay when data transfer is expected immediately after exiting the HALT mode.

Diagnostic

Bits CHARL0 and CHARL1 in the ENU register provide a loopback feature for diagnostic testing of the UART. When these bits are set to one, the following occur: The receiver input pin (RDX) is internally connected to the transmitter output pin (TDX); the output of the Transmitter Shift Register is "looped back" into the Receive Shift Register input. In this mode, data that is transmitted is immediately received. This feature allows the processor to verify the transmit and receive data paths of the UART.

Note that the framing format for this mode is the nine bit format; one Start bit, nine data bits, and 7/8, one or two Stop bits. Parity is not generated or verified in this mode.

Attention Mode

The UART Receiver section supports an alternate mode of operation, referred to as ATTENTION Mode. This mode of operation is selected by the ATTN bit in the ENUR register. The data format for transmission must also be selected as having nine Data bits and either 7/8, one or two Stop bits.

The ATTENTION mode of operation is intended for use in networking the device with other processors. Typically in such environments the messages consists of device addresses, indicating which of several destinations should receive them, and the actual data. This Mode supports a scheme in which addresses are flagged by having the ninth bit of the data field set to a 1. If the ninth bit is reset to a zero the byte is a Data byte.

While in ATTENTION mode, the UART monitors the communication flow, but ignores all characters until an address character is received. Upon receiving an address character, the UART signals that the character is ready by setting the RBFL flag, which in turn interrupts the processor if UART Receiver interrupts are enabled. The ATTN bit is also cleared automatically at this point, so that data characters as well as address characters are recognized. Software examines the contents of the RBUF and responds by deciding either to accept the subsequent data stream (by leaving the ATTN bit reset) or to wait until the next address character is seen (by setting the ATTN bit again).

Operation of the UART Transmitter is not affected by selection of this Mode. The value of the ninth bit to be transmitted is programmed by setting XBIT9 appropriately. The value of the ninth bit received is obtained by reading RBIT9. Since this bit is located in ENUR register where the error flags reside, a bit operation on it will reset the error flags.

Interrupts

The device supports a vectored interrupt scheme. It supports a total of fourteen interrupt sources. Table VI lists all the possible device interrupt sources, their arbitration rankings and the memory locations reserved for the interrupt vector for each source.

Two bytes of program memory space are reserved for each interrupt source. All interrupt sources except the software interrupt are maskable. Each of the maskable interrupts have an Enable bit and one or more Pending bits. A maskable interrupt is active if its associated enable and pending bits are set. If GIE = 1 and an interrupt is active, then the processor will be interrupted as soon as it is ready to start executing an instruction except if the above conditions happen during the Software Trap service routine. This exception is described in the Software Trap sub-section.

The interruption process is accomplished with the INTR instruction (opcode 00), which is jammed inside the Instruction Register and replaces the opcode about to be executed. The following steps are performed for every interrupt:

1. The GIE (Global Interrupt Enable) bit is reset.
2. The address of the instruction about to be executed is pushed into the stack.
3. The PC (Program Counter) branches to address 00FF. This procedure takes 7 t_c cycles to execute.

At this time, since GIE = 0, other maskable interrupts are disabled. The user is now free to do whatever context switching is required by saving the context of the machine in the stack with PUSH instructions. The user would then program a VIS (Vector Interrupt Select) instruction in order to branch to the interrupt service routine of the highest priority interrupt enabled and pending at the time of the VIS. Note that this is not necessarily the interrupt that caused the branch to address location 00FF Hex prior to the context switching.

Thus, if an interrupt with a higher rank than the one which caused the interruption becomes active before the decision of which interrupt to service is made by the VIS, then the interrupt with the higher rank will override any lower ones and will be acknowledged. The lower priority interrupt(s) are still pending, however, and will cause another interrupt immediately following the completion of the interrupt service routine associated with the higher priority interrupt just serviced. This lower priority interrupt will occur immediately following the RETI (Return from Interrupt) instruction at the end of the interrupt service routine just completed.

Inside the interrupt service routine, the associated pending bit has to be cleared by software. The RETI (Return from Interrupt) instruction at the end of the interrupt service routine will set the GIE (Global Interrupt Enable) bit, allowing the processor to be interrupted again if another interrupt is active and pending.

The VIS instruction looks at all the active interrupts at the time it is executed and performs an indirect jump to the beginning of the service routine of the one with the highest rank.

The addresses of the different interrupt service routines, called vectors, are chosen by the user and stored in ROM in a table starting at 01E0 (assuming that VIS is located between 00FF and 01DF). The vectors are 15-bit wide and therefore occupy 2 ROM locations.

TABLE VI. Interrupt Vector Table

ARBITRATION RANKING	SOURCE DESCRIPTION		VECTOR* ADDRESS (Hi-Low Byte)
(1) Highest	Software		0yFE–0yFF
(2)	Reserved		0yFC–0yFD
(3)	External	G0	0yFA-0yFB
(4)	Timer T0	Underflow	0yF8–0yF9
(5)	Timer T1	T1A/Underflow	0yF6–0yF7
(6)	Timer T1	T1B	0yF4-0yF5
(7)	MICROWIRE/PLUS	Busy Low	0yF2–0yF3
(8)	Counters		0yF0–0yF1
(9)	UART	Receive	0yEE–0yEF
(10)	UART	Transmit	0yEC–0yED
(11)	Timer T2	T2A/Underflow	0yEA–0yEB
(12)	Timer T2	T2B	0yE8–0yE9
(13)	Capture Timer 1 and 2		0yE6–0yE7
(14)	Unused		0yE4–0yE5
(15)	Port L/Wakeup		0yE2–0yE3
(16) Lowest	Default VIS	Reserved	0yE0–0yE1

* y is a variable which represents the VIS block. VIS and the vector table must be located in the same 256-byte block except if VIS is located at the last address of a block, In this case, the table must be in the next block.

Interrupts (Continued)

VIS and the vector table must be located in the same 256-byte block (0y00 to 0yFF) except if VIS is located at the last address of a block. In this case, the table must be in the next block. The vector table cannot be inserted in the first 256-byte block (y ≠ 0).

The vector of the maskable interrupt with the lowest rank is located at 0yE0 (Hi-Order byte) and 0yE1 (Lo-Order byte) and so forth in increasing rank number. The vector of the maskable interrupt with the highest rank is located at 0yFA (Hi-Order byte) and 0yFB (Lo-Order byte).

The Software Trap has the highest rank and its vector is located at 0yFE and 0yFF.

If, by accident, a VIS gets executed and no interrupt is active, then the PC (Program Counter) will branch to a vector located at 0yE0–0yE1.

WARNING

A Default VIS interrupt handler routine must be present. As a minimum, this handler should confirm that the GIE bit is cleared (this indicates that the interrupt sequence has been taken), take care of any required housekeeping, restore ocntext and return. Some sort of Warm Restart procedure should be implemented. These events can occur without any error on the part of the system designer or programmer.

Note: There is always the possibility of an interrupt occurring during an instruction which is attempting to reset the GIE bit or any other interrupt enable bit. If this occurs when a single cycle instruction is being used to reset the interrupt enable bit, the interrupt enable bit will be reset but an interrupt may still occur. This is because interrupt processing is started at the same time as the interrupt bit is being reset. To avoid this scenario, the user should always use a two, three, or four cycle instruction to reset interrupt enable bits.

Figure 18 shows the Interrupt block diagram.

SOFTWARE TRAP

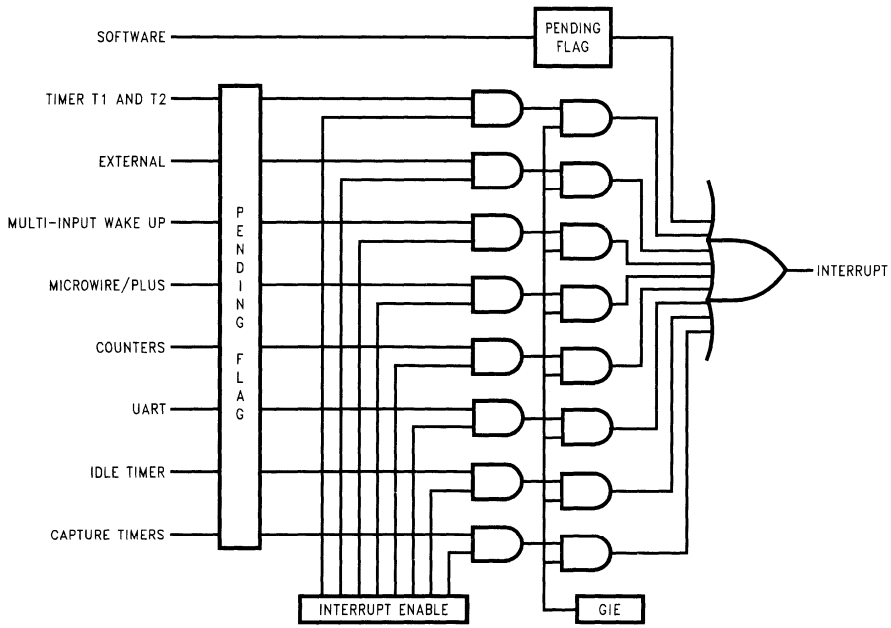
The Software Trap (ST) is a special kind of non-maskable interrupt which occurs when the INTR instruction (used to acknowledge interrupts) is fetched from ROM and placed inside the instruction register. This may happen when the PC is pointing beyond the available ROM address space or when the stack is over-popped.

When an ST occurs, the user can re-initialize the stack pointer and do a recovery procedure (similar to reset, but not necessarily containing all of the same initialization procedures) before restarting.

The occurrence of an ST is latched into the ST pending bit. The GIE bit is not affected and the ST pending bit (**not accessible by the user**) is used to inhibit other interrupts and to direct the program to the ST service routine with the VIS instruction. The RPND instruction is used to clear the software interrupt pending bit. This pending bit is also cleared on reset.

The ST has the highest rank among all interrupts.

Nothing (except another ST) can interrupt an ST being serviced.



TL/DD/12855–18

FIGURE 18. Interrupt Block Diagram

Detection of Illegal Conditions

The device can detect various illegal conditions resulting from coding errors, transient noise, power supply voltage drops, runaway programs, etc.

Reading of undefined ROM gets zeroes. The opcode for software interrupt is 00. If the program fetches instructions from undefined ROM, this will force a software interrupt, thus signaling that an illegal condition has occurred.

The subroutine stack grows down for each call (jump to subroutine), interrupt, or PUSH, and grows up for each return or POR The stack pointer is initialized to RAM location 06F Hex during reset. Consequently, if there are more returns than calls, the stack pointer will point to addresses 070 and 071 Hex (which are undefined RAM). Undefined RAM from addresses 070 to 07F (Segment 0), 140 to 17F (Segment 1), and all other segments (i.e., Segments 3... etc.) is read as all 1's, which in turn will cause the program to return to address 7FFF Hex. This is an undefined ROM location and the instruction fetched (all 0's) from this location will generate a software interrupt signaling an illegal condition.

Thus, the chip can detect the following illegal conditions:

1. Executing from undefined ROM
2. Over "POP"ing the stack by having more returns than calls.

When the software interrupt occurs, the user can re-initialize the stack pointer and do a recovery procedure before restarting (this recovery program is probably similar to that following reset, but might not contain the same program initialization procedures). The recovery program should reset the software interrupt pending bit using the RPND instruction.

MICROWIRE/PLUS

MICROWIRE/PLUS is a serial synchronous communications interface. The MICROWIRE/PLUS capability enables the device to interface with any of National Semiconductor's MICROWIRE peripherals (i.e., A/D converters, display drivers, E2PROMs etc.) and with other microcontrollers which support the MICROWIRE interface. It consists of an 8-bit serial shift register (SIO) with serial data input (SI), serial data output (SO) and serial shift clock (SK). *Figure 19* shows a block diagram of the MICROWIRE/PLUS logic.

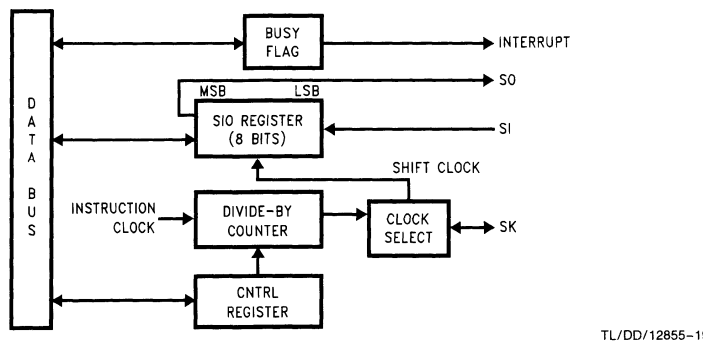
The shift clock can be selected from either an internal source or an external source. Operating the MICROWIRE/PLUS arrangement with the internal clock source is called the Master mode of operation. Similarly, operating the MICROWIRE/PLUS arrangement with an external shift clock is called the Slave mode of operation.

The CNTRL register is used to configure and control the MICROWIRE/PLUS mode. To use the MICROWIRE/PLUS, the MSEL bit in the CNTRL register is set to one. In the master mode, the SK clock rate is selected by the two bits, SL0 and SL1, in the CNTRL register. Table VII details the different clock rates that may be selected.

TABLE VII. MICROWIRE/PLUS Master Mode Clock Select

SL1	SL0	SK Period
0	0	$2 \times t_c$
0	1	$4 \times t_c$
1	x	$8 \times t_c$

Where t_c is the instruction cycle clock



TL/DD/12855-19

FIGURE 19. MICROWIRE/PLUS Block Diagram

MICROWIRE/PLUS (Continued)

MICROWIRE/PLUS OPERATION

Setting the BUSY bit in the PSW register causes the MICROWIRE/PLUS to start shifting the data. It gets reset when eight data bits have been shifted. The user may reset the BUSY bit by software to allow less than 8 bits to shift. If enabled, an interrupt is generated when eight data bits have been shifted. The device may enter the MICROWIRE/PLUS mode either as a Master or as a Slave. *Figure 20* shows how two devices, microcontrollers and several peripherals may be interconnected using the MICROWIRE/PLUS arrangements.

Warning:

The SIO register should only be loaded when the SK clock is low. Loading the SIO register while the SK clock is high will result in undefined data in the SIO register. SK clock is normally low when not shifting.

Setting the BUSY flag when the input SK clock is high in the MICROWIRE/PLUS slave mode may cause the current SK clock for the SIO shift register to be narrow. For safety, the BUSY flag should only be set when the input SK clock is low.

MICROWIRE/PLUS Master Mode Operation

In the MICROWIRE/PLUS Master mode of operation the shift clock (SK) is generated internally by the device. The MICROWIRE Master always initiates all data exchanges. The MSEL bit in the CNTRL register must be set to enable the SO and SK functions onto the G Port. The SO and SK pins must also be selected as outputs by setting appropriate bits in the Port G configuration register. Table VIII summarizes the bit settings required for Master mode of operation.

MICROWIRE/PLUS Slave Mode Operation

In the MICROWIRE/PLUS Slave mode of operation the SK clock is generated by an external source. Setting the MSEL bit in the CNTRL register enables the SO and SK functions onto the G Port. The SK pin must be selected as an input and the SO pin is selected as an output pin by setting and resetting the appropriate bits in the Port G configuration register. Table VIII summarizes the settings required to enter the Slave mode of operation.

This table assumes that the control flag MSEL is set.

TABLE VIII. MICROWIRE Mode Settings

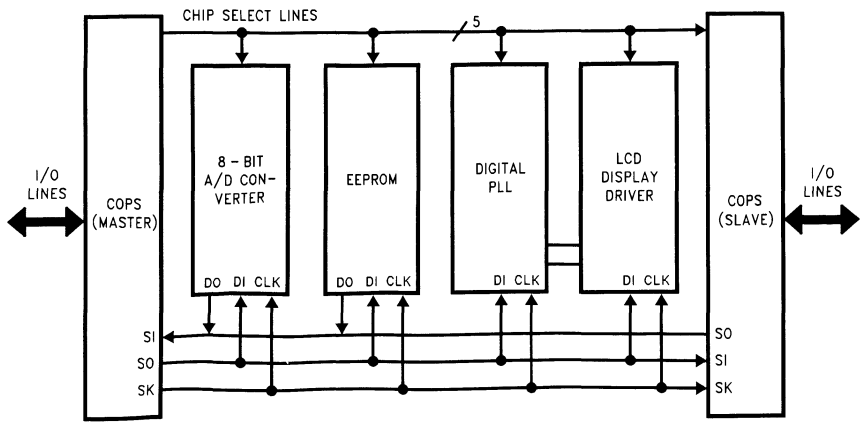
G4 (SO) Config. Bit	G5 (SK) Config. Bit	G4 Fun.	G5 Fun.	Operation
1	1	SO	Int. SK	MICROWIRE/PLUS Master
0	1	TRI-STATE	Int. SK	MICROWIRE/PLUS Master
1	0	SO	Ext. SK	MICROWIRE/PLUS Slave
0	0	TRI-STATE	Ext. SK	MICROWIRE/PLUS Slave

The user must set the BUSY flag immediately upon entering the Slave mode. This will ensure that all data bits sent by the Master will be shifted properly. After eight clock pulses the BUSY flag will be cleared and the sequence may be repeated.

Alternate SK Phase Operation

The device allows either the normal SK clock or an alternate phase SK clock to shift data in and out of the SIO register. in both the modes the SK is normally low. In the normal mode data is shifted in on the rising edge of the SK clock and the data is shifted out on the falling edge of the SK clock. The SIO register is shifted on each falling edge of the SK clock. In the alternate SK phase operation, data is shifted in on the falling edge of the SK clock and shifted out on the rising edge of the SK clock.

A control flag, SKSEL, allows either the normal SK clock or the alternate SK clock to be selected. Resetting SKSEL causes the MICROWIRE/PLUS logic to be clocked from the normal SK signal. Setting the SKSEL flag selects the alternate SK clock. The SKSEL is mapped into the G6 configuration bit. The SKSEL flag will power up in the reset condition, selecting the normal SK signal.



TL/DD/12855-20

FIGURE 20. MICROWIRE/PLUS Application

Memory Map

All RAM, ports and registers (except A and PC) are mapped into data memory address space.

ADDRESS S/ADD REG	CONTENTS
0000 to 006F	112 On-Chip RAM Bytes
0070 to 007F	Unused RAM Address Space (reads as all 1's)
xx80 to xx8F	Unused RAM Address Space (reads undefined data)
xx90	Port E Data Register
xx91	Port E Configuration Register
xx92	Port E Input Pins (read only)
xx93	Reserved
xx94	Port F Data Register
xx95	Port F Configuration Register
xx96	Port F Input Pins (read only)
xx97	Reserved
xx98	Dividend or Result Byte (MDR1)
xx99	Dividend/Multiplier or Result Byte (MDR2)
xx9A	Dividend/Result Byte or Undefined (MDR3)
xx9B	Divisor/Multiplicand or Result Byte (MDR4)
xx9C	Divisor or Multiplicand Byte(MDR5)
xx9D	Multiply/Divide Control Register (MDCR)
xx9E	Counter Control 1 Register (CCR1)
xx9F	Counter Control 2 Register (CCR2)
xxA0	Counter 1 Prescaler Lower Byte (C1PRL)
xxA1	Counter 1 Prescaler Upper Byte (C1PRH)
xxA2	Counter 1 Count Register Lower Byte (C1CTL)
xxA3	Counter 1 Count Register Upper Byte (C1CTH)
xxA4	Counter 2 Prescaler Lower Byte (C2PRL)
xxA5	Counter 2 Prescaler Upper Byte (C2PRH)
xxA6	Counter 2 Count Register Lower Byte (C2CTL)
xxA7	Counter 2 Count Register Upper Byte (C2CTH)
xxA8	Counter 3 Prescaler Lower Byte (C3PRL)
xxA9	Counter 3 Prescaler Upper Byte (C3PRH)
xxAA	Counter 3 Count Register Lower Byte (C3CTL)
xxAB	Counter 3 Count Register Upper Byte (C3CTH)
xxAC	Counter 4 Prescaler Lower Byte (C4PRL)
xxAD	Counter 4 Prescaler Upper Byte (C4PRH)
xxAE	Counter 4 Count Register Lower Byte (C4CTL)
xxAF	Counter 4 Count Register Upper Byte (C4CTH)
xxB0	Capture Timer 1 Prescaler Register (CM1 PSC)
xxB1	Capture Timer 1 Lower Byte (CM1CRL) Read-Only
xxB2	Capture Timer 1 Upper Byte (CM1CRH) Read-Only
xxB3	Capture Timer 2 Prescaler Register (CM2PSC)
xxB4	Capture Timer 2 Lower Byte (CM2CRL) Read-Only
xxB5	Capture Timer 2 Upper Byte (CM2CRH) Read-Only
xxB6	Capture Timer 1 Control Register (CCMR1)
xxB7	Capture Timer 2 Control Register (CCMR2)
xxB8	UART Transmit Buffer (TBUF)
xxB9	UART Receive Buffer (RBUF)
xxBA	UART Control and Status Register (ENU)

Memory Map (Continued)

ADDRESS S/ADD REG	CONTENTS
xxBB	UART Receive Control and Status Register (ENUR)
xxBC	UART Interrupt and Clock Source Register (ENUI)
xxBD	UART Baud Register (BAUD)
xxBE	UART Prescaler Select Register (PSR)
xxBF	Reserved for UART
xxC0	Timer T2 Lower Byte
xxC1	Timer T2 Upper Byte
xxC2	Timer T2 Autoload Register T2RA Lower Byte
xxC3	Timer T2 Autoload Register T2RA Upper Byte
xxC4	Timer T2 Autoload Register T2RB Lower Byte
xxC5	Timer T2 Autoload Register T2RB Upper Byte
xxC6	Timer T2 Control Register
xxC7	Reserved
xxC8	MIWU Edge Select Register (WKEDG)
xxC9	MIWU Enable Register (WKEN)
xxCA	MIWU Pending Register (WKPND)
xxCB	Reserved
xxCC	Reserved
xxCD to xxCF	Reserved
xxD0	Port L Data Register
xxD1	Port L Configuration Register
xxD2	Port L Input Pins (Read Only)
xxD3	Reserved for Port L
xxD4	Port G Data Register
xxD5	Port G Configuration Register
xxD6	Port G Input Pins (Read Only)
xxD7	Port I Input Pins (Read Only)
xxD8	Port C Data Register
xxD9	Port C Configuration Register
xxDA	Port C Input Pins (Read Only)
xxDB	Reserved for Port C
xxDC	Port D
xxDD to xxDF	Reserved for Port D
xxE0 to xxE5	Reserved for EE Control Registers
xxE6	Timer T1 Autoload Register T1RB Lower Byte
xxE7	Timer T1 Autoload Register T1RB Upper Byte
xxE8	ICNTRL Register
xxE9	MICROWIRE Shift Register
xxEA	Timer T1 Lower Byte
xxEB	Timer T1 Upper Byte
xxEC	Timer T1 Autoload Register T1RA Lower Byte
xxED	Timer T1 Autoload Register T1RA Upper Byte
xxEE	CNTRL Control Register
xxEF	PSW Register
xxF0 to xxFB	On-chip RAM Mapped as Registers
xxFC	X Register
xxFD	SP Register
xxFE	B Register
xxFF	S Register
0100 to 017F 0200 to 027F 0300 to 037F	On Chip RAM Bytes (384 Bytes)

Reading memory locations 0070H-007FH (Segment 0) will return all ones. Reading unused memory locations between 0080H-00F0 Hex (Segment 0) will return undefined data. Reading memory locations from other segments (i.e., segment 4, segment 5, etc.) will return all ones.

Addressing Modes

There are ten addressing modes, six for operand addressing and four for transfer of control.

OPERAND ADDRESSING MODES

Register Indirect

This is the "normal" addressing mode. The operand is the data memory addressed by the B pointer or X pointer.

Register Indirect (with auto post Increment or decrement of pointer)

This addressing mode is used with the LD and X instructions. The operand is the data memory addressed by the B pointer or X pointer. This is a register indirect mode that automatically post increments or decrements the B or X register after executing the instruction.

Direct

The instruction contains an 8-bit address field that directly points to the data memory for the operand.

Immediate

The instruction contains an 8-bit immediate field as the operand.

Short Immediate

This addressing mode is used with the Load B Immediate instruction. The instruction contains a 4-bit immediate field as the operand.

Indirect

This addressing mode is used with the LAID instruction. The contents of the accumulator are used as a partial address (lower 8 bits of PC) for accessing a data operand from the program memory.

TRANSFER OF CONTROL ADDRESSING MODES

Relative

This mode is used for the JP instruction, with the instruction field being added to the program counter to get the new program location. JP has a range from −31 to +32 to allow a 1-byte relative jump (JP + 1 is implemented by a NOP instruction). There are no "pages" when using JP, since all 15 bits of PC are used.

Absolute

This mode is used with the JMP and JSR instructions, with the instruction field of I 2 bits replacing the lower 12 bits of the program counter (PC). This allows jumping to any location in the current 4k program memory segment.

Absolute Long

This mode is used with the JMPL and JSRL instructions, with the instruction field of 15 bits replacing the entire 15 bits of the program counter (PC). This allows jumping to any location up to 32k in the program memory space.

Indirect

This mode is used with the JID instruction. The contents of the accumulator are used as a partial address (lower 8 bits of PC) for accessing a location in the program memory. The contents of this program memory location serve as a partial address (lower 8 bits of PC) for the jump to the next instruction.

Note: The VIS is a special case of the Indirect Transfer of Control addressing mode, where the double byte vector associated with the interrupt is transferred from adjacent addresses in the program memory into the program counter (PC) in order to jump to the associated interrupt service routine.

Instruction Set

Register and Symbol Definition

Registers	
A	8-Bit Accumulator Register
B	8-Bit Address Register
X	8-Bit Address Register
SP	8-Bit Stack Pointer Register
PC	15-Bit Program Counter Register
PU	Upper 7 Bits of PC
PL	Lower 8 Bits of PC
C	1 Bit of PSW Register for Carry
HC	1 Bit of PSW Register for Half Carry
GIE	1 Bit of PSW Register for Global Interrupt Enable
VU	Interrupt Vector Upper Byte
VL	Interrupt Vector Lower Byte

Symbols	
[B]	Memory Indirectly Addressed by B Register
[X]	Memory Indirectly Addressed by X Register
MD	Direct Addressed Memory
Mem	Direct Addressed Memory or [B]
Meml	Direct Addressed Memory or [B] or Immediate Data
Imm	8-Bit Immediate Data
Reg	Register Memory: Addresses F0 to FF (Includes B, X and SP)
Bit	Bit Number (0 to 7)
←	Loaded with
↔	Exchanged with

INSTRUCTION SET

ADD	A,Meml	ADD	A ← A + Meml
ADC	A,Meml	ADD with Carry	A ← A + Meml + C, C ← Carry, HC ← Half Carry
SUBC	A,Meml	Subtract with Carry	A ← A − Meml + C, C ← Carry, HC ← Half Carry
AND	A,Meml	Logical AND	A ← A and $\overline{\text{Meml}}$
ANDSZ	A,Imm	Logical AND Immed., Skip if Zero	Skip next if (A and Imm) = 0
OR	A,Meml	Logical OR	A ← A or Meml
XOR	A,Meml	Logical EXclusive OR	A ← A xor Meml
IFEQ	MD,Imm	IF EQual	Compare MD and Imm, Do next if MD = Imm
IFEQ	A,Meml	IF EQual	Compare A and Meml, Do next if A = Meml
IFNE	A,Meml	IF Not Equal	Compare A and Meml, Do next if A ≠ Meml
IFGT	A,Meml	IF Greater Than	Compare A and Meml, Do next if A > Meml
IFBNE	#	IF B Not Equal	Do next if lower 4 bits of B ≠ Imm
DRSZ	Reg	Decrement Reg., Skip if Zero	Reg ← Reg − 1, Skip if Reg = 0
SBIT	#,Mem	Set BIT	1 to bit, Mem (bit = 0 to 7 immediate)
RBIT	#,Mem	Reset BIT	0 to bit, Mem
IFBIT	#,Mem	IF BIT	If bit #, A or Mem is true do next instruction
RPND		Reset PeNDing Flag	Reset Software Interrupt Pending Flag
X	A,Mem	EXchange A with Memory	A ⟷ Mem
X	A,[X]	EXchange A with Memory [X]	A ⟷ [X]
LD	A,Meml	LoaD A with Memory	A ← Meml
LD	A,[X]	LoaD A with Memory [X]	A ← [X]
LD	B, Imm	LoaD B with Immed.	B ← Imm
LD	Mem, Imm	LoaD Memory Immed.	Mem ← Imm
LD	Reg, Imm	LoaD Register Memory Immed.	Reg ← Imm
X	A, [B±]	EXchange A with Memory [B]	A ⟷ [B], (B ← B ± 1)
X	A, [X±]	EXchange A with Memory [X]	A ⟷ [X], (X ← X ± 1)
LD	A, [B±]	LoaD A with Memory [B]	A ← [B], (B ← B ± 1)
LD	A, [X±]	LoaD A with Memory [X]	A ← [X], (X ← X ± 1)
LD	[B±],Imm	LoaD Memory [B] Immed.	[B] ← Imm, (B ← B ± 1)
CLR	A	CLeaR A	A ← 0
INC	A	INCrement A	A ← A + 1
DEC	A	DECrement A	A ← A − 1
LAID		Load A InDirect from ROM	A ← ROM (PU, A)
DCOR	A	Decimal CORrect A	A ← BCD correction of A (follows ADC, SUBC)
RRC	A	Rotate A Right thru C	C → A7 → . . . → A0 → C
RLC	A	Rotate A Left thru C	C ← A7 ← . . . ← A0 ← C
SWAP	A	SWAP nibbles of A	A7 . . . A4 ⟷ A3 . . . A0
SC		Set C	C ← 1, HC ← 1
RC		Reset C	C ← 0, HC ← 0
IFC		IF C	If C is true, do next instruction
IFNC		IF Not C	If C is not true, do next instruction
POP	A	POP the stack into A	SP ← SP + 1, A ← [SP]
PUSH	A	PUSH A onto the stack	[SP] ← A, SP ← SP − 1
VIS		Vector to Interrupt Service Routine	PU ← [VU], PL ← [VL]
JMPL	Addr.	Jump absolute Long	PC ← ii (ii = 15 bits, 0 to 32k)
JMP	Addr.	Jump absolute	PC9 . . . 0 ← i (i = 12 bits)
JP	Disp.	Jump relative short	PC ← PC + r (r is −31 to +32, except 1)
JSRL	Addr.	Jump SubRoutine Long	[SP] ← PL, [SP − 1] ← PU, SP − 2, PC ← ii
JSR	Addr	Jump SubRoutine	[SP] ← PL, [SP − 1] ← PU, SP − 2, PC9 . . . 0 ← i
JID		Jump InDirect	PL ← ROM (PU, A)
RET		RETurn from subroutine	SP + 2, PL ← [SP], PU ← [SP − 1]
RETSK		RETurn and SKip	SP + 2, PL ← [SP], PU ← [SP − 1], skip next instruction
RETI		RETurn from Interrupt	SP + 2, PL ← [SP], PU ← [SP − 1], GIE ← 1
INTR		Generate an Interrupt	[SP] ← PL, [SP − 1] ← PU, SP − 2, PC ← 0FF
NOP		No OPeration	PC ← PC + 1

Instruction Execution Time

Most instructions are single byte (with immediate addressing mode instructions taking two bytes).

Most single byte instructions take one cycle time to execute.

Skipped instructions require x number of cycles to be skipped, where x equals the number of bytes in the skipped instruction opcode.

See the BYTES and CYCLES per INSTRUCTION table for details.

Bytes and Cycles per Instruction

The following table shows the number of bytes and cycles for each instruction in the format of byte/cycle.

Arithmetic and Logic Instructions

	[B]	Direct	Immed.
ADD	1/1	3/4	2/2
ADC	1/1	3/4	2/2
SUBC	1/1	3/4	2/2
AND	1/1	3/4	2/2
OR	1/1	3/4	2/2
XOR	1/1	3/4	2/2
IFEQ	1/1	3/4	2/2
IFGT	1/1	3/4	2/2
IFBNE	1/1		
DRSZ		1/3	
SBIT	1/1	3/4	
RBIT	1/1	3/4	
IFBIT	1/1	3/4	

RPND	1/1

Instructions Using A & C

CLRA	1/1
INCA	1/1
DECA	1/1
LAID	1/3
DCORA	1/1
RRCA	1/1
RLCA	1/1
SWAPA	1/1
SC	1/1
RC	1/1
IFC	1/1
IFNC	1/1
PUSHA	1/3
POPA	1/3
ANDSZ	2/2

Transfer of Control Instructions

JMPL	3/4
JMP	2/3
JP	1/3
JSRL	3/5
JSR	2/5
JID	1/3
VIS	1/5
RET	1/5
RETSK	1/5
RETI	1/5
INTR	1/7
NOP	1/1

Memory Transfer Instructions

	Register Indirect		Direct	Immed.	Register Indirect Auto Incr. and Decr.		
	[B]	[X]			[B+, B−]	[X+, X−]	
X A, *	1/1	1/3	2/3		1/2	1/3	
LD A, *	1/1	1/3	2/3	2/2	1/2	1/3	
LD B, Imm				1/1			(IF B < 16)
LD B, Imm				2/2			(IF B > 15)
LD Mem, Imm	2/2		3/3		2/2		
LD Reg, Imm			2/3				
IFEQ MD, Imm			3/3				

Note: * = > Memory location addressed by B or X or directly.

Opcode List

Bits 7–4

F	E	D	C	B	A	9	8	7	6	5	4	3	2	1	0	
JP −15	JP −31	LD 0F0, #i	DRSZ 0F0	RRCA	RC	ADC A, #i	ADC A,[B]	IFBIT 0,[B]	ANDSZ A, #i	LD B, 0F	IFBNE 0	JSR x000–x0FF	JMP x000–x0FF	JP +17	INTR	0
JP −14	JP −30	LD 0F1, #i	DRSZ 0F1	*	SC	SUBC A, #i	SUB A,[B]	IFBIT 1,[B]	*	LD B, 0E	IFBNE 1	JSR x100–x1FF	JMP x100–x1FF	JP +18	JP + 2	1
JP −13	JP −29	LD 0F2, #i	DRSZ 0F2	X A, [X+]	X A, [B+]	IFEQ A, #i	IFEQ A,[B]	IFBIT 2,[B]	*	LD B, 0D	IFBNE 2	JSR x200–x2FF	JMP x200–x2FF	JP +19	JP + 3	2
JP −12	JP −28	LD 0F3, #i	DRSZ 0F3	X A, [X−]	X A, [B−]	IFGT A, #i	IFGT A,[B]	IFBIT 3,[B]	*	LD B, 0C	IFBNE 3	JSR x300–x3FF	JMP x300–x3FF	JP +20	JP + 4	3
JP −11	JP −27	LD 0F4, #i	DRSZ 0F4	VIS	LAID	ADD A, #i	ADD A,[B]	IFBIT 4,[B]	CLRA	LD B, 0B	IFBNE 4	JSR x400–x4FF	JMP x400–x4FF	JP +21	JP + 5	4
JP −10	JP −26	LD 0F5, #i	DRSZ 0F5	RPND	JID	AND A, #i	AND A,[B]	IFBIT 5,[B]	SWAPA	LD B, 0A	IFBNE 5	JSR x500–x5FF	JMP x500–x5FF	JP +22	JP + 6	5
JP −9	JP −25	LD 0F6, #i	DRSZ 0F6	X A,[X]	X A,[B]	XOR A, #i	XOR A,[B]	IFBIT 6,[B]	DCORA	LD B, 9	IFBNE 6	JSR x600–x6FF	JMP x600–x6FF	JP +23	JP + 7	6
JP −8	JP −24	LD 0F7, #i	DRSZ 0F7	*	*	OR A, #i	OR A,[B]	IFBIT 7,[B]	PUSHA	LD B, 8	IFBNE 7	JSR x700–x7FF	JMP x700–x7FF	JP +24	JP + 8	7
JP −7	JP −23	LD 0F8, #i	DRSZ 0F8	NOP	RLCA	LD A, #i	IFC	SBIT 0,[B]	RBIT 0,[B]	LD B, 7	IFBNE 8	JSR x800–x8FF	JMP x800–x8FF	JP +25	JP + 9	8
JP −6	JP −22	LD 0F9, #i	DRSZ 0F9	IFNE A,[B]	IFEQ Md, #i	IFNE A, #i	IFNC	SBIT 1,[B]	RBIT 1,[B]	LD B, 6	IFBNE 9	JSR x900–x9FF	JMP x900–x9FF	JP +26	JP + 10	9
JP −5	JP −21	LD 0FA, #i	DRSZ 0FA	LD A, [X+]	LD A, [B+]	LD [B+], #i	INCA	SBIT 2,[B]	RBIT 2,[B]	LD B, 5	IFBNE 0A	JSR xA00–xAFF	JMP xA00–xAFF	JP +27	JP + 11	A
JP −4	JP −20	LD 0FB, #i	DRSZ 0FB	LD A, [X−]	LD A, [B−]	LD [B−], #i	DECA	SBIT 3,[B]	RBIT 3,[B]	LD B, 4	IFBNE 0B	JSR xB00–xBFF	JMP xB00–xBFF	JP +28	JP + 12	B
JP −3	JP −19	LD 0FC, #i	DRSZ 0FC	LD Md, #i	JMPL	X A,Md	POPA	SBIT 4,[B]	RBIT 4,[B]	LD B, 3	IFBNE 0C	JSR xC00–xCFF	JMP xC00–xCFF	JP +29	JP + 13	C
JP −2	JP −18	LD 0FD, #i	DRSZ 0FD	DIR	JSRL	LD A,Md	RETSK	SBIT 5,[B]	RBIT 5,[B]	LD B, 2	IFBNE 0D	JSR xD00–xDFF	JMP xD00–xDFF	JP +30	JP + 14	D
JP −1	JP −17	LD 0FE, #i	DRSZ 0FE	LD A,[X]	LD A,[B]	LD [B], #i	RET	SBIT 6,[B]	RBIT 6,[B]	LD B, 1	IFBNE 0E	JSR xE00–xEFF	JMP xE00–xEFF	JP +31	JP + 15	E
JP −0	JP −16	LD 0FF, #i	DRSZ 0FF	*	*	LD B, #i	RETI	SBIT 7,[B]	RBIT 7,[B]	LD B, 0	IFBNE 0F	JSR xF00–xFFF	JMP xF00–xFFF	JP +32	JP + 16	F

Bits 3–0

Where,

#i is the immediate data

Md is a directly addressed memory location

* is an unused opcode

Note: The opcode 60 Hex is also the opcode for IFBIT #i,A.

Development Support

SUMMARY

- iceMASTER™: IM-COP8/400—Full feature in-circuit emulation for all COP8 products. A full set of COP8 Basic and Feature Family device and package specific probes are available.
- COP8 Debug Module: Moderate cost in-circuit emulation and development programming unit.
- COP8 Evaluation and Programming Unit: EPU-COP888GG—low cost In-circuit simulation and development programming unit.
- Assembler: COP-8-DEV-IBMA. A DOS installable cross development Assembler, Linker, Librarian and Utility Software Development Tool Kit.
- C Compiler: COP8C. A DOS installable cross development Software Tool Kit.
- OTP/EPROM Programmer Support: Covering needs from engineering prototype, pilot production to full production environments.

IceMASTER IN-CIRCUIT EMULATION

The iceMASTER IM-COP8/400 is a full feature, PC based, in-circuit emulation tool developed and marketed by MetaLink Corporation to support the whole COP8 family of products. National is a resale vendor for these products.

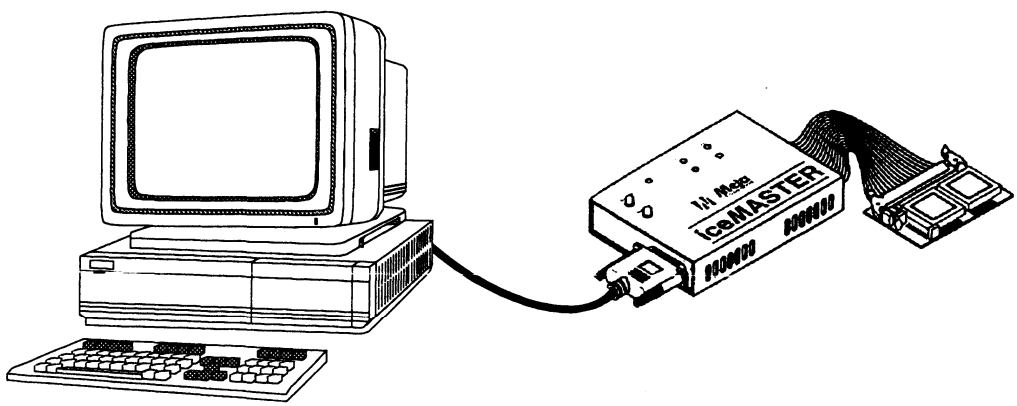
See *Figure 21* for configuration.

The iceMASTER IM-COP8/400 with its device specific COP8 Probe provides a rich feature set for developing, testing and maintaining product:

- Real-time in-circuit emulation; full 2.4V–5.5V operation range, full DC-10 MHz clock. Chip options are programmable or jumper selectable.
- Direct connection to application board by package compatible socket or surface mount assembly.
- Full 32 kbytes of loadable programming space that overlays (replaces) the on-chip ROM or EPROM. On-chip RAM and I/O blocks are used directly or recreated on the probe as necessary.
- Full 4k frame synchronous trace memory. Address, instruction, and 8 unspecified, circuit connectable trace lines. Display can be HLL source (e.g., C source), assembly or mixed.
- A full 64k hardware configurable break, trace on, trace off control, and pass count increment events.
- Tool set integrated interactive symbolic debugger—supports both assembler (COFF) and C Compiler (.COD) linked object formats.
- Real time performance profiling analysis; selectable bucket definition.
- Watch windows, content updated automatically at each execution break.
- Instruction by instruction memory/register changes displayed on source window when in single step operation.
- Single base unit and debugger software reconfigurable to support the entire COP8 family; only the probe personality needs to change. Debugger software is processor customized, and reconfigured from a master model file.
- Processor specific symbolic display of registers and bit level assignments, configured from master model file.
- Halt/Idle mode notification.
- On-line HELP customized to specific processor using master model file.
- Includes a copy of COP8-DEV-IBMA assembler and linker SDK.

Order Information

Base Unit	
IM-COP8/400-1	iceMASTER Base Unit, 110V Power Supply
IM-COP8/400-2	iceMASTER Base Unit, 220V Power Supply
iceMASTER Probe	
MHW-888RW68PWPC	68 PLCC



TL/DD/12855–21

FIGURE 21. COP8 iceMASTER Environment

Development Support (Continued)

IceMASTER DEBUG MODULE

The iceMASTER Debug Module is a PC based, combination in-circuit emulation tool and COP8 based OTP/EPROM programming tool developed and marketed by MetaLink Corporation to support the whole COP8 family of products. National is a resale vendor for these products.

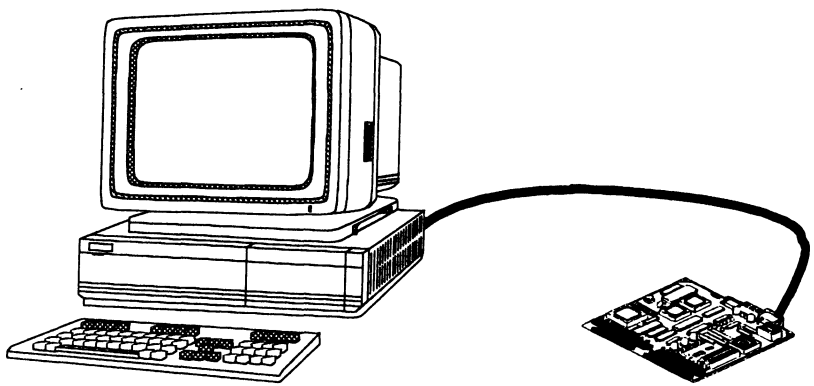
See *Figure 22* for configuration.

The iceMASTER Debug Module is a moderate cost development tool. It has the capability of in-circuit emulation for a specific COP8 microcontroller and in addition serves as a programming tool for COP8 OTP and EPROM product families. Summary of features is as follows:

- Real-time in-circuit emulation; full operating voltage range operation, full DC-10 MHz clock.
- All processor I/O pins can be cabled to an application development board with package compatible cable to socket and surface mount assembly.
- Full 32 kbytes of loadable programming space that overlays (replaces) the on-chip ROM or EPROM. On-chip RAM and I/O blocks are used directly or recreated as necessary.
- 100 frames of synchronous trace memory. The display can be HLL source (C source), assembly or mixed. The most recent history prior to a break is available in the trace memory.
- Configured break points; uses INTR instruction which is modestly intrusive.
- Software—only supported features are selectable.
- Tool set integrated interactive symbolic debugger—supports both assembler (COFF) and C Compiler (.COD) SDK linked object formats.
- Instruction by instruction memory/register changes displayed when in single step operation.
- Debugger software is processor customized, and reconfigured from a master model file.
- Processor specific symbolic display of registers and bit level assignments, configured from master model file.
- Halt/Idle mode notification.
- Programming menu supports full product line of programmable OTP and EPROM COP8 products. Program data is taken directly from the overlay RAM.
- Programming of 44 PLCC and 68 PLCC parts requires external programming adapters.
- Includes wall mount power supply.
- On-board V_{PP} generator from 5V input or connection to external supply supported. Requires V_{PP} level adjustment per the family programming specification (correct level is provided on an on-screen pop-down display). On-line HELP customized to specific processor using master model file.
- On-Line HELP customized to specific processor using master model file.
- Includes a copy of COP8-DEV-IBMA assembler and linker SDK.

Order Information

Debug Module Unit	
COP8-DM/888RW	
Cable Adapters	
DM-COP8/68P	68 PLCC



TL/DD/12855–22

FIGURE 22. COP8-DM Environment

Development Support (Continued)

COP8 ASSEMBLER/LINKER SOFTWARE DEVELOPMENT TOOL KIT

National Semiconductor offers a relocatable COP8 macro cross assembler, linker, librarian and utility software development tool kit. Features are summarized as follows:

- Basic and Feature Family instruction set by "device" type.
- Nested macro capability.
- Extensive set of assembler directives.
- Supported on PC/DOS platform.
- Generates National standard COFF output files.
- Integrated Linker and Librarian.
- Integrated utilities to generate ROM code file outputs.
- DUMPCOFF utility.

This product is integrated as a part of MetaLink tools as a development kit, fully supported by the MetaLink debugger. It may be ordered separately or it is bundled with the MetaLink products at no additional cost.

Order Information

Assembler SDK	
COP8-DEV-IBMA	Assembler SDK on installable 3.5" PC/DOS Floppy Disk Drive format. Periodic upgrades and most recent version is available on National's BBS and Internet.

COP8 C COMPILER

A C Compiler is developed and marketed by Byte Craft Limited. The COP8C compiler is a fully integrated development tool specifically designed to support the compact embedded configuration of the COP8 family of products.

Features are summarized as follows:

- ANSI C with some restrictions and extensions that optimize development for the COP8 embedded application.
- BITS data type extension. Register declaration #pragma with direct bit level definitions.
- C language support for interrupt routines.
- Expert system, rule based code generation and optimization.

Approved List

Manufacturer	North America	Europe	Asia
BP Microsystems	(800) 225-2102 (713) 688-4600 Fax: (713) 688-0920	+49-8152-4183 +49-8856-932616	+852-234-16611 +852-2710-8121
Data I/O	(800) 426-1045 (206) 881-6444 Fax: (206) 882-1043	+44-0734-440011	Call North America
HI-LO	(510) 623-8860	Call Asia	+886-2-764-0215 Fax: +886-2-756-6403
ICE Technology	(800) 624-8949 (919) 430-7915	+44-1226-767404 Fax: 0-1226-370-434	
MetaLink	(800) 638-2423 (602) 926-0797 Fax: (602) 693-0681	+49-80 9156 96-0 Fax: +49-80 9123 86	+852-737-1800
Systems General	(408) 263-6667	+41-1-9450300	+886-2-917-3005 Fax: +886-2-911-1283
Needhams	(916) 924-8037 Fax: (916) 924-8065		

Development Support (Continued)

- Performs consistency checks against the architectural definitions of the target COP8 device.
- Generates program memory code.
- Supports linking of compiled object or COP8 assembled object formats.
- Global optimization of linked code.
- Symbolic debug load format fully source level supported by the MetaLink debugger.

INDUSTRY WIDE OTP/EPROM PROGRAMMING SUPPORT

Programming support, in addition to the MetaLink development tools, is provided by a full range of independent approved vendors to meet the needs from the engineering laboratory to full production.

AVAILABLE LITERATURE

For more information, please see the COP8 Basic Family User's Manual, Literature Number 620895, COP8 Feature Family User's Manual, Literature Number 620897 and National's Family of 8-bit Microcontrollers COP8 Selection Guide, Literature Number 630009.

DIAL-A-HELPER SERVICE

Dial-A-Helper is a service provided by the Microcontroller Applications group. The Dial-A-Helper is an Electronic Information System that may be accessed as a Bulletin Board System (BBS) via data modem, as an FTP site on the Internet via standard FTP client application or as an FTP site on the Internet using a standard Internet browser such as Netscape or Mosaic.

The Dial-A-Helper system provides access to an automated information storage and retrieval system. The system capabilities include a MESSAGE SECTION (electronic mail, when accessed as a BBS) for communications to and from the Microcontroller Applications Group and a FILE SECTION which consists of several file areas where valuable application software and utilities could be found.

DIAL-A-HELPER BBS via a Standard Modem

Modem:	CANADA/U.S.:	(800) NSC-MICRO
		(800) 672-6427
	EUROPE:	(+49) 0-8141-351332
	Baud:	14.4k
	Set-Up:	Length: 8-Bit
		Parity: None
		Stop Bit: 1
	Operation:	24 Hrs., 7 Days

DIAL-A-HELPER via FTP

ftp nscmicro.nsc.com

user: anonymous

password: username@yourhost.site.domain

DIAL-A-HELPER via a WorldWide Web Browser

ftp://nscmicro.nsc.com

National Semiconductor on the WorldWide Web

See us on the WorldWide Web at: http://www.national.com

CUSTOMER RESPONSE CENTER

Complete product information and technical support is available from National's customer response centers.

CANADA/U.S.:	Tel:	(800) 272-9959
	email:	support @tevm2.nsc.com
EUROPE:	email:	europe.support@nsc.com
	Deutsch Tel:	+49 (0) 180-530 85 85
	English Tel:	+49 (0) 180-532 78 32
	Français Tel:	+49 (0) 180-532 93 58
	Italiano Tel:	+49 (0) 180-534 16 80
JAPAN:	Tel:	+81-043-299-2309
S. E. ASIA:	Beijing Tel:	(+86) 10-6865-8601
	Shanghai Tel:	(+86) 21-6415-4092
	Hong Kong Tel:	(+852) 2737-1600
	Korea Tel:	(+82) 2-3771-6909
	Malaysia Tel:	(+60-4) 644-9061
	Singapore Tel:	(+65) 255-2226
	Taiwan Tel:	+886-2-521-3288
AUSTRALIA:	Tel:	(+61) 3-9558-9999
INDIA:	Tel:	(+91) 80-559-9467

Section 4

COP8™ Basic Family Products

Section 4 Contents

COP912C/COP912CH 8-Bit Microcontroller 4-3

COP620C/COP622C/COP640C/COP642C/COP820C/COP822C/COP840C/COP842C/COP920C/COP922C/COP940C/COP942C 8-Bit Microcontroller 4-24

COP820CJ/COP822CJ/COP823CJ 8-Bit Microcontroller with Multi-Input Wake-Up and Brown Out Detector 4-47

COP840CJ/COP842CJ/COP940CJ/COP942CJ 8-Bit Microcontrollers with Multi-Input Wake-Up and Brown Out Detector 4-75

COP680C/COP681C/COP682C/COP880C/COP881C/COP882C/COP980C/COP981C/COP982C Microcontrollers 4-106

National Semiconductor

COP912C/COP912CH 8-Bit Microcontroller

General Description

The COP912C/COP912CH are members of the COP8™ 8-bit MicroController family. They are fully static Microcontrollers, fabricated using double-metal silicon gate microCMOS technology. These low cost MicroControllers are complete microcomputers containing all system timing, interrupt logic, ROM, RAM, and I/O necessary to implement dedicated control functions in a variety of applications. Features include an 8-bit memory mapped architecture, MICROWIRE™ serial I/O, a 16-bit timer/counter with capture register and a multi-sourced interrupt. Each I/O pin has software selectable options to adapt the device to the specific application. The device operates over voltage ranges from 2.3V to 4.0V (COP912C) and from 4.0V to 5.5V (COP912CH). High throughput is achieved with an efficient, regular instruction set operating at a minimum of 2 μs per instruction rate.

Key Features

- Lowest cost COP8 microcontroller
- 16-bit multi-function timer supporting
  - — PWM mode
  - — External event counter mode
  - — Input capture mode
- 768 bytes of ROM
- 64 bytes of RAM

I/O Features

- Memory mapped I/O
- Software selectable I/O options (TRI-STATE® Output, Push-Pull Output, Weak Pull-Up Input, High Impedance Input)
- Schmitt trigger inputs on Port G
- MICROWIRE/PLUS™ Serial I/O
- Packages: 20 DIP/SO with 16 I/O pins

CPU/Instruction Set Features

- Instruction cycle time of 2 μs for COP912CH and 2.5 μs for COP912C
- Three multi-sourced interrupts servicing
  - — External Interrupt with selectable edge
  - — Timer interrupt
  - — Software interrupt
- Versatile and easy to use instruction set
- 8-bit Stack Pointer (SP)—stack in RAM
- Two 8-bit Register Indirect Memory Pointers (B, X)

Fully Static CMOS

- Low current drain (typically < 1 μA)
- Single supply operation: 2.3V to 4.0V or 4.0V to 5.5V
- Temperature range: 0°C to +70°C

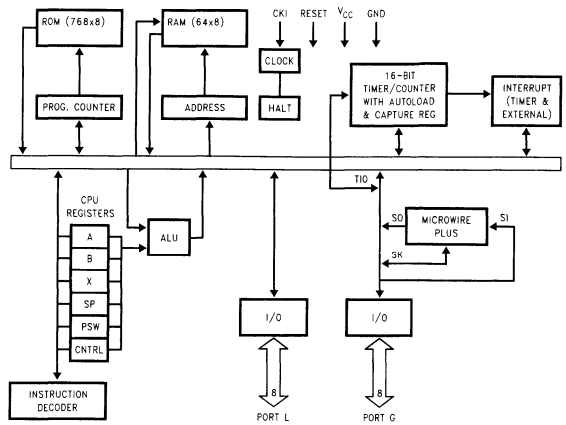
Development Support

- Emulation and OTP devices
- Real time emulation and full program debug offered by MetaLink Development System

Applications

- Electronic keys and switches
- Remote Control
- Timers
- Alarms
- Small industrial control units
- Low cost slave controllers
- Temperature meters
- Small domestic appliances
- Toys and games

Block Diagram



TL/DD/12060–1

4

Absolute Maximum Ratings

If Military/Aerospace specified devices are required, please contact the National Semiconductor Sales Office/Distributors for availability and specifications.

Supply Voltage (V_{CC})	6.0V
Voltage at Any Pin	−0.3V to V_{CC} +0.3V
Total Current into V_{CC} Pin (Source)	80 mA
Total Current out of GND Pin (Sink)	80 mA
Storage Temperature Range	−65°C to +150°C

Note: ***Absolute maximum ratings indicate limits beyond which damage to the device may occur. DC and AC electrical specifications are not ensured when operating the device at absolute maximum ratings.***

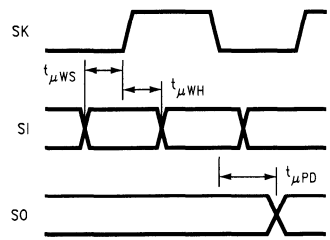
DC Electrical Characteristics COP912C/COP912CH; 0°C ≤ T_A ≤ +70°C unless other specified

Parameter	Conditions	Min	Typ	Max	Units
Operating Voltage					
912C		2.3		4.0	V
912CH		4.0		5.5	V
Power Supply Ripple 1 (Note 1)	Peak to Peak			0.1 V_{CC}	V
Supply Current (Note 2)					
CKI = 4 MHz	V_{CC} = 5.5V, tc = 2.5 μs			6.0	mA
CKI = 4 MHz	V_{CC} = 4.0V, tc = 2.5 μs			2.5	mA
HALT Current	V_{CC} = 5.5V, CKI = 0 MHz		<1	8	μA
INPUT LEVELS (V_{IH}, V_{IL})					
Reset, CKI:					
Logic High		0.9 V_{CC}			V
Logic Low				0.1 V_{CC}	V
All Other Inputs					
Logic High		0.7 V_{CC}			V
Logic Low				0.2 V_{CC}	V
Hi-Z Input Leakage/TRI-STATE Leakage	V_{CC} = 5.5V	−2		+2	μA
Input Pullup Current	V_{CC} = 5.5V			250	μA
G-Port Hysteresis			0.05 V_{CC}	0.35 V_{CC}	V
Output Current Levels					
Source (Push-Pull Mode)	V_{CC} = 4.0V, V_{OH} = 3.8V	0.4			mA
	V_{CC} = 2.3V, V_{OH} = 1.8V	0.2			mA
Sink (Push-Pull Mode)	V_{CC} = 4.0V, V_{OL} = 1.0V	4.0			mA
	V_{CC} = 2.3V, V_{OL} = 0.4V	0.7			mA
Allowable Sink/Source Current Per Pin				3	mA
Input Capacitance (Note 3)				7	pF
Load Capacitance on D2 (Note 3)				1000	pF

Note 1: Rate of voltage change must be less then 0.5 V/ms.

Note 2: Supply current is measured after running 2000 cycles with a square wave CKI input, CKO open, inputs at rails and outputs open.

Note 3: Characterized, not tested.

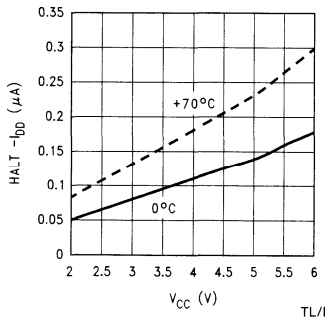


TL/DD/12060–2

FIGURE 1. MICROWIRE/PLUS Timing

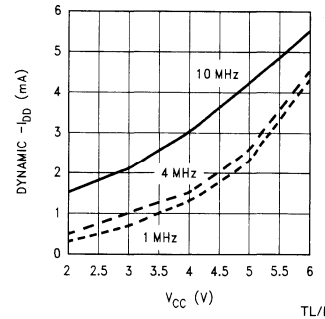
Typical Performance Characteristics

Halt—I_{DD}



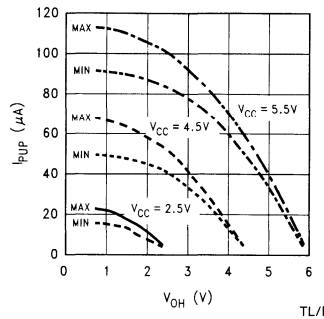
TL/DD/12060–16

Dynamic—I_{DD} (Crystal Clock Option)



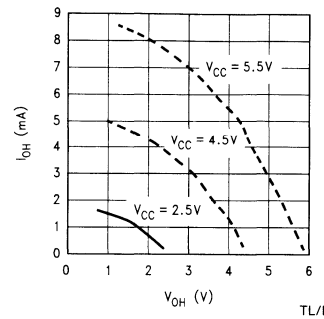
TL/DD/12060–17

Port L/G Weak Pull-Up Source Current



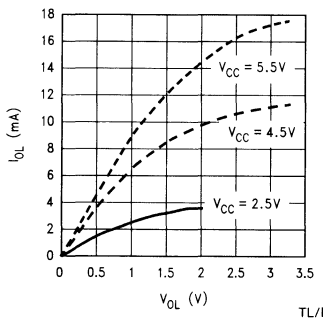
TL/DD/12060–18

Port L/G Push-Pull Source Current



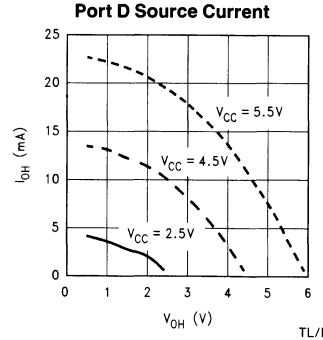
TL/DD/12060–19

Port L/G Push-Pull Sink Current



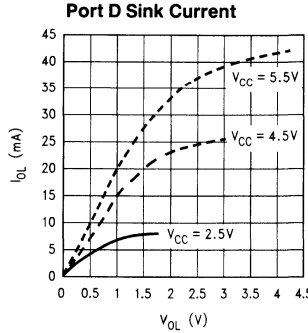
TL/DD/12060–20

Port D Source Current



TL/DD/12060–21

Port D Sink Current



TL/DD/12060–22

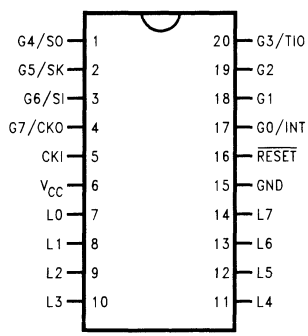
AC Electrical Characteristics COP912C/COP912CH; 0°C ≤ T_A ≤ +70°C unless otherwise specified

Parameter	Conditions	Min	Typ	Max	Units
INSTRUCTION CYCLE TIME (tc)					
Crystal/Resonator	4.0V ≤ V_{CC} ≤ 5.5V	2		DC	μs
	2.3V ≤ V_{CC} < 4.0V	2.5		DC	μs
R/C Oscillator	4.0V ≤ V_{CC} ≤ 5.5V	3		DC	μs
	2.3V ≤ V_{CC} < 4.0V	7.5		DC	μs
Inputs					
t_{Setup}	4.0V ≤ V_{CC} ≤ 5.5V	200			ns
	2.3V ≤ V_{CC} < 4.0V	500			ns
t_{Hold}	4.0V ≤ V_{CC} ≤ 5.5V	60			ns
	2.3V ≤ V_{CC} < 4.0V	150			ns
Output Propagation Delay	R_L = 2.2 kΩ, C_L = 100 pF				
t_{PD1}, t_{PD0}					
SO, SK	4.0V ≤ V_{CC} ≤ 5.5V			0.7	μs
	2.3V ≤ V_{CC} < 4.0V			1.75	μs
All Others	4.0V ≤ V_{CC} ≤ 5.5V			1	μs
	2.3V ≤ V_{CC} < 4.0V			5	μs
Input Pulse Width					
Interrupt Input High Time		1 tc			
Interrupt Input Low Time		1 tc			
Timer Input High Time		1 tc			
Timer Input Low Time		1 tc			
MICROWIRE Setup Time ($t_{\mu WS}$)		20			ns
MICROWIRE Hold Time ($t_{\mu WH}$)		56			ns
MICROWIRE Output Propagation Delay ($t_{\mu PD}$)				220	ns
Reset Pulse Width		1.0			μs

COP912C/COP912CH Pinout

Top View

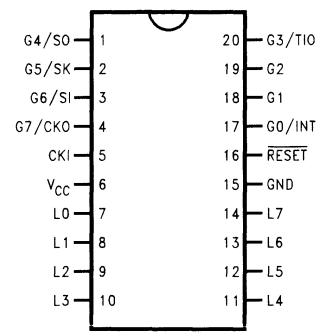
20 DIP



TL/DD/12060–3

Order Number COP912C-XXX/N, COP912CH-XXX/N

20 SO Wide



TL/DD/12060–4

Order Number COP912C-XXX/WM, COP912CH-XXX/WM

FIGURE 2. COP912C/COP912CH Pinout

Pin Description

V_{CC} and **GND** are the power supply pins.

CKI is the clock input. This can come from an external source, a R/C generated oscillator or a crystal (in conjunction with CKO). See Oscillator description.

$\overline{\text{RESET}}$ is the master reset input. See Reset description.

PORT L is an 8-bit I/O port.

There are two registers associated to configure the L port: a data register and a configuration register Therefore, each L I/O bit can be individually configured under software control as shown below:

Port L Config.	Port L Data	PORT L Setup
0	0	Hi-Z Input (TRI-STATE)
0	1	Input with Weak Pull-Up
1	0	Push-Pull Zero Output
1	1	Push-Pull One Output

Three data memory address locations are allocated for this port, one each for data register [00D0], configuration register [00D1] and the input pins [00D2].

PORT G is an 8-bit port with 6 I/O pins (G0–G5) and 2 input pins (G6, G7).

All eight G-pins have Schmitt Triggers on the inputs.

There are two registers associated to configure the G port: a data register and a configuration register. Therefore each G port bit can be individually configured under software control as shown below:

Port G Config.	Port G Data	PORT G Setup
0	0	Hi-Z Input (TRI-STATE)
0	1	Input with Weak Pull-Up
1	0	Push-Pull Zero Output
1	1	Push-Pull One Output

Three data memory address locations are allocated for this port, one for data register [00D4], one for configuration register [00D5] and one for the input pins [00D6]. Since G6 and G7 are Hi-Z input only pins, any attempt by the user to configure them as outputs by writing a one to the configuration register will be disregarded. Reading the G6 and G7 configuration bits will return zeroes. Note that the chip will be placed in the Halt mode by writing a "1" to the G7 data bit.

Six pins of Port G have alternate features:

G0 INTR (an external interrupt)

G3 TIO (timer/counter input/output)

G4 SO (MICROWIRE serial data output)

G5 SK (MICROWIRE clock I/O)

G6 SI (MICROWIRE serial data input)

G7 CKO crystal oscillator output (selected by mask option) or HALT restart input/general purpose input (if clock option is R/C- or external clock)

Pins G1 and G2 currently do not have any alternate functions.

The selection of alternate Port G functions are done through registers PSW [00EF] to enable external interrupt and CNTRL [00EE] to select TIO and MICROWIRE operations.

Functional Description

The internal architecture is shown in the block diagram. Data paths are illustrated in simplified form to depict how the various logic elements communicate with each other in implementing the instruction set of the device.

ALU AND CPU REGISTERS

The ALU can do an 8-bit addition, subtraction, logical or shift operations in one cycle time. There are five CPU registers:

A is the 8-bit Accumulator register

PC is the 15-bit Program Counter register

PU is the upper 7 bits of the program counter (PC)

PL is the lower 8 bits of the program counter (PC)

B is the 8-bit address register and can be auto incremented or decremented

X is the 8-bit alternate address register and can be auto incremented or decremented.

SP is the 8-bit stack pointer which points to the subroutine stack (in RAM).

B, X and SP registers are mapped into the on chip RAM. The B and X registers are used to address the on chip RAM. The SP register is used to address the stack in RAM during subroutine calls and returns. The SP must be preset by software upon initialization.

MEMORY

The memory is separated into two memory spaces: program and data.

PROGRAM MEMORY

Program memory consists of 768 x 8 ROM. These bytes of ROM may be instructions or constant data. The memory is addressed by the 15-bit program counter (PC). There are no "pages" of ROM, the PC counts all 15 bits. ROM can be indirectly read by the LAID instruction for table lookup.

DATA MEMORY

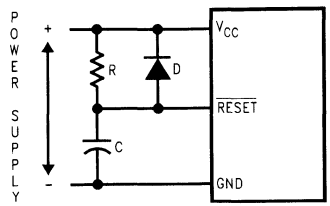
The data memory address space includes on chip RAM, I/O and registers. Data memory is addressed directly by the instruction or indirectly through B, X and SP registers. The device has 64 bytes of RAM. Sixteen bytes of RAM are mapped as "registers", these can be loaded immediately, decremented and tested. Three specific registers: X, B, and SP are mapped into this space, the other registers are available for general usage.

Any bit of data memory can be directly set, reset or tested. I/O and registers (except A and PC) are memory mapped; therefore, I/O bits and register bits can be directly and individually set, reset and tested.

RESET

The RESET input pin when pulled low initializes the microcontroller. Upon initialization, the ports L and G are placed in the TRI-STATE mode. The PC, PSW and CNTRL registers are cleared. The data and configuration registers for ports L and G are cleared. The external RC network shown in *Figure 3* should be used to ensure that the RESET pin is held low until the power supply to the chip stabilizes.

Functional Description (Continued)



TL/DD/12060–5

RC > 5 x POWER SUPPLY RISE TIME

FIGURE 3. Recommended Reset Circuit

OSCILLATOR CIRCUITS

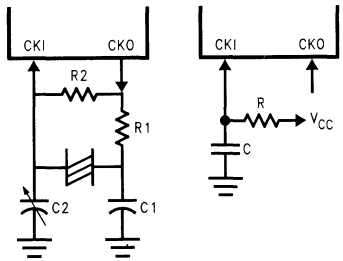
The device can be driven by a clock input which can be between DC and 5 MHz.

CRYSTAL OSCILLATOR

By selecting CKO as a clock output, CKI and CKO can be connected to create a crystal controlled oscillator. Table I shows the component values required for various standard crystal values.

R/C OSCILLATOR

By selecting CKI as a single pin oscillator, CKI can make an R/C oscillator. CKO is available as a general purpose input and/or HALT control. Table II shows variation in the oscillator frequencies as functions of the component (R and C) value.



TL/DD/12060–6

FIGURE 4. Clock Oscillator Configurations

TABLE I. Crystal Oscillator Configuration

R1 (kΩ)	R2 (mΩ)	C1 (pF)	C2 (pF)	CKI Freq. (MHz)
0	1	30	30–36	5
0	1	30	30–36	4
5.6	1	200	100–150	0.455

TABLE II. RC Oscillator Configuration (Part-to-Part Variation, T_A = 25°C)

R (kΩ)	C (pF)	CKI Freq. (MHz)	Intr. Cycle (μs)
3.3	82	2.2 to 2.7	3.7 to 4.6
5.6	100	1.1 to 1.3	7.4 to 9
6.8	100	0.9 to 1.1	8.8 to 10.8

Note: 3k ≤ R ≤ 200 kΩ, 50 pF ≤ C ≤ 200 pF.

HALT MODE

The device is a fully static device. The device enters the HALT mode by writing a one to the G7 bit of the G data register. Once in the HALT mode, the internal circuitry does not receive any clock signal and is therefore frozen in the exact state it was in when halted. In this mode the chip will only draw leakage current.

The device supports two different ways of exiting the HALT mode. The first method is with a low to high transition on the CKO (G7) pin. This method precludes the use of the crystal clock configuration (since CKO is a dedicated output), and so may be used either with an RC clock configuration (or an external clock configuration). The second method of exiting the HALT mode is to pull the RESET low.

Note: To allow clock resynchronization, it is necessary to program two NOP's immediately after the device comes out of the HALT mode. The user must program two NOP's following the "enter HALT mode" (set G7 data bit) instruction.

Functional Description (Continued)

MICROWIRE/PLUS

MICROWIRE/PLUS is a serial synchronous communications interface. The MICROWIRE/PLUS capability enables the device to interface with any of National Semiconductor's MICROWIRE peripherals (i.e., A/D converters, display drivers, EEPROMS etc.) and with other microcontrollers which support the MICROWIRE interface. It consists of an 8-bit serial shift register (SIO) with serial data input (SI), serial data output (SO) and serial shift clock (SK). *Figure 5* shows a block diagram of the MICROWIRE logic.

The shift clock can be derived from either the internal source or from an external source. Operating the MICROWIRE arrangement with the internal clock source is called the Master mode of operation. Similarly, operating the MICROWIRE arrangement with an external shift clock is called the Slave mode of operation.

The CNTRL register is used to configure and control the MICROWIRE mode. To use the MICROWIRE, the MSEL bit in the CNTRL register is set to one. The SK clock rate is selected by the two bits, SL0 and SL1, in the CNTRL register.

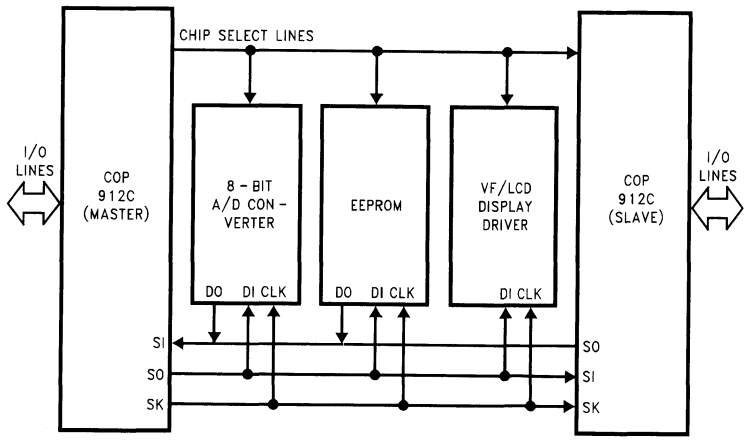
The following table details the different clock rates that may be selected.

SK Divide Clock Rates

SL1	SL0	SK
0	0	2 x tc
0	1	4 x tc
1	x	8 x tc
Where tc is the instruction cycle clock.		

MICROWIRE/PLUS OPERATION

Setting the BUSY bit in the PSW register causes the MICROWIRE/PLUS to start shifting the data. It gets reset when eight data bits have been shifted. The user may reset the BUSY bit by software to allow less than 8 bits to shift. The device may enter the MICROWIRE/PLUS mode either as a Master or as a Slave. *Figure 5* shows how two microcontrollers and several peripherals may be interconnected using the MICROWIRE/PLUS arrangement.



TL/DD/12060–7

FIGURE 5. MICROWIRE/PLUS Application

Functional Description (Continued)

WARNING: The SIO register should only be loaded when the SK clock is low. Loading the SIO register while the SK clock is high will result in undefined data in the SIO register.

Setting the BUSY flag when the input SK clock is high in the MICROWIRE/PLUS slave mode may cause the current SK clock for the SIO shift register to be narrow. For safety, the BUSY flag should only be set when the input SK clock is low.

Table III summarizes the settings required to enter the Master/Slave modes of operations.

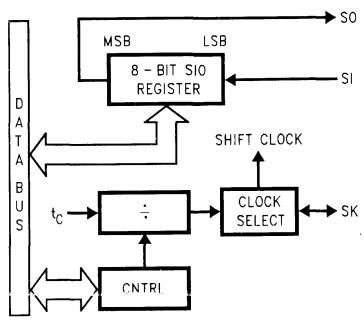
The table assumes that the control flag MSEL is set.

TABLE III. MICROWIRE/PLUS G Port Configuration

G4 (SO) Config. Bit	G5 (SK) Config. Bit	G4 Pin	G5 Pin	G6 Pin	Operation
1	1	SO	Int. SK	SI	MICROWIRE Master
0	1	TRI-STATE	Int. SK	SI	MICROWIRE Master
1	0	SO	Ext. SK	SI	MICROWIRE Slave
0	0	TRI-STATE	Ext. SK	SI	MICROWIRE Slave

MICROWIRE/PLUS MASTER MODE OPERATION

In MICROWIRE/PLUS Master mode operation, the SK shift clock is generated internally. The MSEL bit in the CNTRL register must be set to allow the SK and SO functions onto the G5 and G4 pins. The G5 and G4 pins must also be selected as outputs by setting the appropriate bits in the Port G configuration register. The MICROWIRE Master mode always initiates all data exchanges. The MSEL bit in the CNTRL register is set to enable MICROWIRE/PLUS. G4 and G5 are selected as output.



TL/DD/12060–8

FIGURE 6. MICROWIRE/PLUS Block Diagram

MICROWIRE/PLUS SLAVE MODE

In MICROWIRE/PLUS Slave mode operation, the SK shift clock is generated by an external source. Setting the MSEL bit in the CNTRL register enables the SO and SK functions onto the G port. The SK pin must be selected as an input and the SO pin as an output by resetting and setting their respective bits in the G port configuration register.

The user must set the BUSY flag immediately upon entering the slave mode. This will ensure that all data bits sent by the master will be shifted in properly. After eight clock pulses, the BUSY flag will be cleared and the sequence may be repeated.

Note: In the Slave mode the SIO register does not stop shifting even after the busy flag goes low. Since SK is an external output, the SIO register stops shifting only when SK is turned off by the master.

Note: Setting the BUSY flag when the input SK clock is high in the MICROWIRE/PLUS slave mode may cause the current SK clock for the SIO register to be narrow. When the BUSY flag is set, the MICROWIRE logic becomes active with the internal SIO shift clock enabled. If SK is high in slave mode, this will cause the internal shift clock to go from low in standby mode to high in active mode. This generates a rising edge, and causes one bit to be shifted into the SIO register from the SI input. For safety, the BUSY flag should only be set when the input SK clock is low.

Note: The SIO register must be loaded only when the SK shift clock is low. Loading the SIO register while the SK clock is high will result in undefined data in the SIO register.

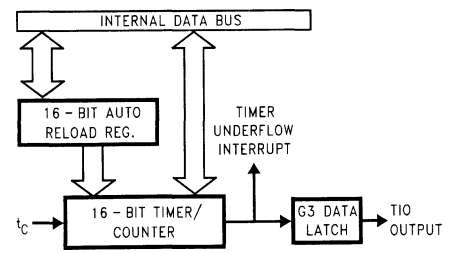
Timer/Counter

The device has an on board 16-bit timer/counter (organized as two 8-bit registers) with an associated 16-bit autoreload/capture register (also organized as two 8-bit registers). Both are read/write registers.

The timer has three modes of operation:

PWM (PULSE WIDTH MODULATION) MODE

The timer counts down at the instruction cycle rate (2 μs max). When the timer count underflows, the value in the autoreload register is copied into the timer. Consequently, the timer is programmable to divide by any value from 1 to 65536. Bit 5 of the timer CNTRL register selects the timer underflow to toggle the G3 output. This allows the user to generate a square wave output or a pulse-width-modulated output. The timer underflow can also be enabled to interrupt the processor. The timer PWM mode is shown in *Figure 7*.



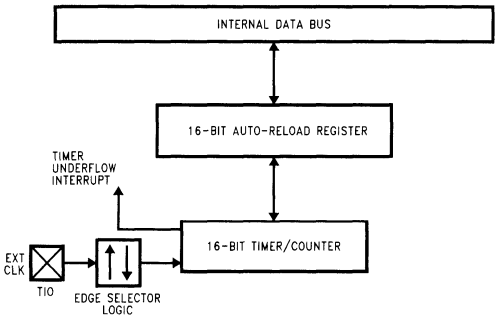
TL/DD/12060–10

FIGURE 7. Timer in PWM Mode

Functional Description (Continued)

EXTERNAL EVENT COUNTER MODE

In this mode, the timer becomes a 16-bit external event counter, clocked from an input signal applied to the G3 input. The maximum frequency for this G3 input clock is 250 kHz (half of the 0.5 MHz instruction cycle clock). When the external event counter underflows, the value in the autoreload register is copied into the timer. This timer underflow may also be used to generate an interrupt. Bit 5 of the CNTRL register is used to select whether the external event counter clocks on positive or negative edges from the G3 input. Consequently, half cycles of an external input signal could be counted. The External Event counter mode is shown in *Figure 8*.

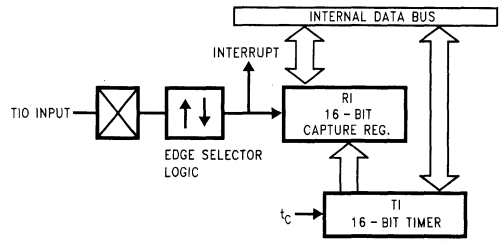


TL/DD/12060-11

FIGURE 8. Timer in External Event Mode

INPUT CAPTURE MODE

In this mode, the timer counts down at the instruction clock rate. When an external edge occurs on pin G3, the value in the timer is copied into the capture register. Consequently, the time of an external edge on the G3 pin is "captured". Bit 5 of the CNTRL register is used to select the polarity of the external edge. This external edge capture can also be programmed to generate an interrupt. The duration of an input signal can be computed by capturing the time of the leading edge, saving this captured value, changing the capture edge, capturing the time of the trailing edge, and then subtracting this trailing edge time from the earlier leading edge time. The Input Capture mode is shown in *Figure 9*.



TL/DD/12060-12

FIGURE 9. Timer in Input Capture Mode

Table IV below details the TIMER modes of operation and their associated interrupts. Bit 4 of CNTRL is used to start and stop the timer/counter. Bits 5, 6 and 7 of the CNTRL register select the timer modes. The ENTI (Enable Timer Interrupt) and TPND (Timer Interrupt Pending) bits in the PSW register are used to control the timer interrupts.

Care must be taken when reading from and writing to the timer and its associated autoreload/capture register. The timer and autoreload/capture register are both 16-bit, but they are read from and written to one byte at a time. It is recommended that the timer be stopped before writing a new value into it. The timer may be read "on the fly" without stopping it if suitable precautions are taken. One method of reading the timer "on the fly" is to read the upper byte of the timer first, and then read the lower byte. If the most significant bit of the lower byte is then tested and found to be high, then the upper byte of the timer should be read again and this new value used.

TABLE IV. Timer Modes and Control Bits

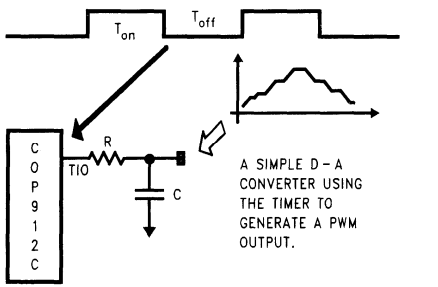
CNTRL Bits			Operation Mode	Timer Interrupt	Timer Counts On
7	6	5			
0	0	0	External Event Counter with Autoreload Register	Timer Underflow	TIO Positive Edge
0	0	1	External Event Counter with Autoreload Register	Timer Underflow	TIO Negative Edge
0	1	0	Not Allowed	Not Allowed	Not Allowed
0	1	1	Not Allowed	Not Allowed	Not Allowed
1	0	0	Timer with Autoreload Register	Timer Underflow	tc
1	0	1	Timer with Autoreload Regiter and Toggle TIO Out	Timer Underflow	tc
1	1	0	Timer with Capture Register	TIO Positive Edge	tc
1	1	1	Timer with Capture Register	TIO Negative Edge	tc

Functional Description (Continued)

TIMER APPLICATION EXAMPLE

The timer has an autoreload register that allows any frequency to be programmed in the timer PWM mode. The timer underflow can be programmed to toggle output bit G3, and may also be programmed to generate a timer interrupt. Consequently, a fully programmable PWM output may be easily generated.

The timer counts down and when it underflows, the value from the autoreload register is copied into the timer. The CNTRL register is programmed to both toggle the G3 output and generate a timer interrupt when the timer underflows. Following each timer interrupt, the user's program alternately loads the values of the "on" time and the "off" time into the timer autoreload register. Consequently, a pulse-width-modulated (PWM) output waveform is generated to a resolution of one instruction cycle time. This PWM application example is shown in *Figure 10*.



TL/DD/12060-13

FIGURE 10. Timer Based PWM Application

Interrupts

There are three interrupt sources:

1. A maskable interrupt on external G0 input positive or negative edge sensitive under software control
2. A maskable interrupt on timer underflow or timer capture
3. A non-maskable software/error interrupt on opcode zero. The GIE (global interrupt enable) bit enables the interrupt function. This is used in conjunction with ENI and ENTI to select one or both of the interrupt sources. This bit is reset when interrupt is acknowledged.

ENI and ENTI bits select external and timer interrupt respectively. Thus the user can select either or both sources to interrupt the microcontroller when GIE is enabled. IEDG selects the external interrupt edge (1 = rising edge, 0 = falling edge). The user can get an interrupt on both rising and falling edges by toggling the state of IEDG bit after each interrupt.

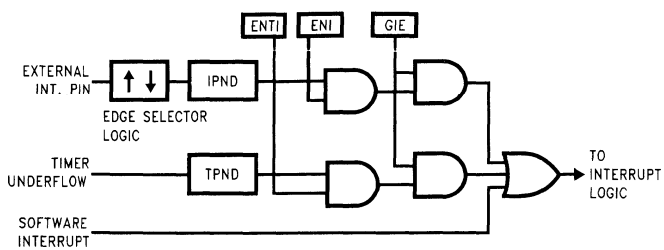
IPND and TPND bits signal which interrupt is pending. After interrupt is acknowledged, the user can check these two bits to determine which interrupt is pending. The user can prioritize the interrupt and clear the pending bit that corresponds to the interrupt being serviced. The user can also enable GIE at this point for nesting interrupts. Two things have to be kept in mind when using the software interrupt. The first is that executing a simple RET instruction will take the program control back to the software interrupt instruction itself. In other words, the program will be stuck in an infinite loop. To avoid the infinite loop, the software interrupt service routine should end with a RETSK instruction or with a JMP instruction. The second thing to keep in mind is that unlike the other interrupt sources, the software interrupt does not reset the GIE bit. This means that the device can be interrupted by other interrupt sources while servicing the software interrupt.

Interrupts push the PC to the stack, reset the GIE bit to disable further interrupts and branch to address 00FF. The RETI instruction will pop the stack to PC and set the GIE bit to enable further interrupts. The user should use the RETI or the RET instruction when returning from a hardware (maskable) interrupt subroutine. The user should use the RETSK instruction when returning from a software interrupt subroutine to avoid an infinite loop situation.

The software interrupt is a special kind of non-maskable interrupt which occurs when the INTR instruction (opcode 00 used to acknowledge interrupts) is fetched from ROM and placed inside the instruction register. This may happen when the PC is pointing beyond the available ROM address space or when the stack is over-popped. When the software interrupt occurs, the user can re-initialize the stack pointer and do a recovery procedure (similar to reset, but not necessarily containing all of the same initialization procedures) before restarting.

Hardware and Software interrupts are treated differently. The software interrupt is not gated by the GIE bit. However, it has the lowest arbitration ranking. Also the fact that all interrupts vector to the same address 00FF Hex means that a software interrupt happening at the same time as a hardware interrupt will be missed.

Note: There is always the possibility of an interrupt occurring during an instruction which is attempting to reset the GIE bit or any other interrupt enable bit. If this occurs when a single cycle instruction is being used to reset the interrupt enable bit, the interrupt enable bit will be reset but an interrupt may still occur. This is because interrupt processing is started at the same time as the interrupt bit is being reset. To avoid this scenario, the user should always use a two, three, or four cycle instruction to reset interrupt enable bits.



TL/DD/12060-14

FIGURE 11. Interrupt Block Diagram

Interrupts (Continued)

DETECTION OF ILLEGAL CONDITIONS

Reading of undefined ROM gets zeroes. The opcode for software interrupt is zero. If the program fetches instructions from undefined ROM, this will force a software interrupt, thus signalling that an illegal condition has occurred.

Note: A software interrupt is acted upon only when a timer or external interrupt is not pending as hardware interrupts have priority over software interrupt. In addition, the Global Interrupt bit is not set when a software interrupt is being serviced thereby opening the door for the hardware interrupts to occur. The subroutine stack grows down for each call and grows up for each return. If the stack pointer is initialized to 2F Hex, then if there are more returns than calls, the stack pointer will point to addresses 30 and 31 (which are undefined RAM). Undefined RAM is read as all 1's, thus, the program will return to address FFFF. This is a undefined ROM location and the instruction fetched will generate a software interrupt signalling an illegal condition. The device can detect the following illegal conditions:

1. Executing from undefined ROM
2. Over "POP"ing the stack by having more returns than calls.

Illegal conditions may occur from coding errors, "brown out" voltage drops, static, supply noise, etc. When the software interrupt occurs, the user can re-initialize the stack pointer and do a recovery procedure before restarting (this recovery program is probably similar to RESET but might not clear the RAM). Examination of the stack can help in identifying the source of the error. For example, upon a software interrupt, if the SP = 30, 31 it implies that the stack was over "POP"ed (with the SP = 2F hex initially). If the SP contains a legal value (less than or equal to the initialized SP value), then the value in the PC gives a clue as to where in the user program an attempt to access an illegal (an address over 300 Hex) was made. The opcode returned in this case is 00 which is a software interrupt.

The detection of illegal conditions is illustrated with an example:

```
0043    CLRA
0044    RC
0045    JMP 04FF
0046    NOP
```

When the device is executing this program, it seemingly "locks-up" having executed a software interrupt. To debug this condition, the user takes a look at the SP and the contents of the stack. The SP has a legal value and the contents of the stack are 04FF. The perceptive user immediately realizes that an illegal ROM location (04FF) was accessed and the opcode returned (00) was a software interrupt. Another way to decode this is to run a trace and follow the sequence of steps that ended in a software interrupt. The damaging jump statement is changed.

Control Registers

CNTRL REGISTER (ADDRESS X'00EE)

The Timer and MICROWIRE control register contains the following bits:

SL1 and SL0 Select the MICROWIRE clock divide-by (00 = 2, 01 = 4, 1x = 8)

IEDG External interrupt edge polarity select

MSEL Selects G5 and G4 as MICROWIRE signals SK and SO respectively

TRUN Used to start and stop the timer/counter (1 = run, 0 = stop)

TC1 Timer Mode Control Bit

TC2 Timer Mode Control Bit

TC3 Timer Mode Control Bit

7							0
TC1	TC2	TC3	TRUN	MSEL	IEDG	SL1	SL0

PSW REGISTER (ADDRESS X'00EF)

The PSW register contains the following select bits:

GIE Global interrupt enable (enables interrupts)

ENI External interrupt enable

BUSY MICROWIRE busy shifting flag

IPND External interrupt pending

ENTI Timer interrupt enable

TPND Timer interrupt pending (timer underflow or capture edge)

C Carry Flip/flop

HC Half carry Flip/flop

7							0
HC	C	TPND	ENTI	IPND	BUSY	ENI	GIE

The Half-Carry bit is also effected by all the instructions that effect the Carry flag. The flag values depend upon the instruction. For example, after executing the ADC instruction the values of the Carry and the Half-Carry flag depend upon the operands involved. However, instructions like SET C and RESET C will set and clear both the carry flags. Table V lists out the instructions that effect the HC and the C flags.

TABLE V. Instructions Effecting HC and C Flags

Instr.	HC Flag	C Flag
ADC	Depends on Operands	Depends on Operands
SUBC	Depends on Operands	Depends on Operands
SETC	Set	Set
RESET C	Set	Set
RRC	Depends on Operands	Depends on Operands

MEMORY MAP

All RAM, ports and registers (except A and PC) are mapped into data memory address space.

Control Registers (Continued)

TABLE VI. Memory Map

Address	Contents
00 to 2F	On-chip RAM Bytes (48 Bytes)
30 to 7F	Unused RAM Address Space (Reads as all ones)
80 to BF	Expansion Space for On-Chip EERAM (Reads Undefined Data)
C0 to CF	Expansion Space for I/O and Registers
D0	Port L Data Register
D1	Port L Configuration Register
D2	Port L Input Pins (read only)
D3	Reserved for Port L
D4	Port G Data Register
D5	Port G Configuration Register
D6	Port G Input Pins (read only)
D7	Reserved
D8 to DB	Reserved
DC to DF	Reserved
E0 to EF	On-Chip Functions and Registers
E0 to E7	Reserved for Future Parts
E8	Reserved
E9	MICROWIRE Shift Register
EA	Timer Lower Byte
EB	Timer Upper Byte
EC	Timer Autoreload Register Lower Byte
ED	Timer Auto reload Register Upper Byte
EE	CNTRL Control Register
EF	PSW Register
F0 to FF	On-Chip RAM Mapped as Registers (16 Bytes)
FC	X Register
FD	SP Register
FE	B Register

Reading other unused memory locations will return undefined data.

Addressing Modes

The device has ten addressing modes, six for operand addressing and four for transfer of control.

OPERAND ADDRESSING MODES

Register Indirect

This is the "normal" addressing mode for the chip. The operand is the data memory addressed by the **B** or **X** pointer.

Register Indirect With Auto Post Increment Or Decrement

This addressing mode is used with the LD and X instructions. The operand is the data memory addressed by the **B** or **X** pointer. This is a register indirect mode that automatically post increments or post decrements the **B** or **X** pointer after executing the instruction.

Direct

The instruction contains an 8-bit address field that directly points to the data memory for the operand.

Immediate

The instruction contains an 8-bit immediate field as the operand.

Short Immediate

This addressing mode issued with the LD B,# instruction, where the immediate # is less than 16. The instruction contains a 4-bit immediate field as the operand.

Indirect

This addressing mode is used with the LAID instruction. The contents of the accumulator are used as a partial address (lower 8 bits of PC) for accessing a data operand from the program memory.

TRANSFER OF CONTROL ADDRESSING MODES

Relative

This mode is used for the JP instruction with the instruction field being added to the program counter to produce the next instruction address. JP has a range from −31 to +32 to allow a one byte relative jump (JP + 1 is implemented by a NOP instruction). There are no "blocks" or "pages" when using JP since all 15 bits of the PC are used.

Absolute

This mode is used with the JMP and JSR instructions with the instruction field of 12 bits replacing the lower 12 bits of the program counter (PC). This allows jumping to any location in the current 4k program memory segment.

Absolute Long

This mode is used with the JMPL and JSRL instructions with the instruction field of 15 bits replacing the entire 15 bits of the program counter (PC). This allows jumping to any location in the entire 32k program memory space.

Indirect

This mode is used with the JID instruction. The contents of the accumulator are used as a partial address (lower 8 bits of PC) for accessing a location in the program memory. The contents of this program memory location serves as a partial address (lower 8 bits of PC) for the jump to the next instruction.

Instruction Set

REGISTER AND SYMBOL DEFINITIONS

Registers

A	8-Bit Accumulator Register
B	8-Bit Address Register
X	8-Bit Address Register
SP	8-Bit Stack Pointer Register
S	8-Bit Data Segment Address Register
PC	15-Bit Program Counter Register
PU	Upper 7 Bits of PC
PL	Lower 8 Bits of PC
C	1-Bit of PSW Register for Carry
HC	1-Bit of PSW Register for Half Carry
GIE	1-Bit of PSW Register for Global Interrupt Enable

Symbols

[B]	Memory Indirectly Addressed by B Register
[X]	Memory Indirectly Addressed by X Register
MD	Direct Addressed Memory
Mem	Direct Addressed Memory, or B
Meml	Direct Addressed Memory, B, or Immediate Data
Imm	8-Bit Immediate Data
Reg	Register Memory: Addresses F0 to FF (Includes B, X, and SP)
Bit	Bit Number (0 to 7)
←	Loaded with
⟷	Exchanged with

Instruction Set (Continued)

TABLE VII. Instruction Set

Instr		Function	Register Operation
ADD	A, Meml	Add	A ← A + Meml
ADC	A, Meml	Add with Carry	A ← A + Meml + C, C ← Carry
SUBC	A, Meml	Subtract with Carry	A ← A − Meml + C, C ← Carry
AND	A, Meml	Logical AND	A ← A and Meml
OR	A, Meml	Logical OR	A ← A or Meml
XOR	A, Meml	Logical Exclusive-OR	A ← A xor Meml
IFEQ	A, Meml	IF Equal	Compare A and Meml, Do Next if A = Meml
IFGT	A, Meml	IF Greater than	Compare A and Meml, Do Next if A > Meml
IFBNE	#	IF B not Equal	Do Next If Lower 4 Bits of B not = Imm
DRSZ	Reg	Decrement Reg, Skip if Zero	Reg ← Reg - 1, Skip if Reg Goes to Zero
SBIT	#, Mem	Set Bit	1 to Mem.Bit (Bit = 0 to 7 Immediate)
RBIT	#, Mem	Reset Bit	0 to Mem.Bit (Bit = 0 to 7 Immediate)
IFBIT	#, Mem	If Bit	If Mem.Bit is True, Do Next Instruction
X	A, Mem	Exchange A with Memory	A ⟷ Mem
LD	A, Meml	Load A with Memory	A ← Meml
LD	Mem, Imm	Load Direct Memory Immed.	Mem ← Imm
LD	Reg, Imm	Load Register Memory Immed.	Reg ← Imm
X	A, [B±]	Exchange A with Memory [B]	A ⟷ [B] (B ← B±1)
X	A, [X±]	Exchange A with Memory [X]	A ⟷ [X] (X ← X±1)
LD	A, [B±]	Load A with Memory [B]	A ← [B] (B ← B±1)
LD	A, [X±]	Load A with Memory [X]	A ← [X] (X ← X±1)
LD	[B±], Imm	Load Memory Immediate	[B] ← Imm (B ← B±1)
CLRA		Clear A	A ← 0
INC		Increment A	A ← A + 1
DEC		Decrement A	A ← A − 1
LAID	A	Load A Indirect from ROM	A ← ROM(PU, A)
DCOR	A	Decimal Correct A	A ← BCD Correction (follows ADC, SUBC)
RRC		Rotate Right Through Carry	C → A7 → . . . → A0 → C
SWAP	A	Swap Nibbles of A	A7 . . . A4 ⟷ A3 . . . A0
SC	A	Set C	C ← 1
RC	A	Reset C	C ← 0
IFC		If C	If C is True, do Next Instruction
IFNC		If Not C	If C is not True, do Next Instruction
JMPL		Jump Absolute Long	PC ← ii (ii = 15 Bits, 0k to 32k)
JMP		Jump Absolute	PC11 . . . PC0 ← i (i = 12 Bits) PC15 . . . PC12 Remain Unchanged
JP		Jump Relative Short	PC ← PC + r (r is −31 to +32, not 1)
JSRL	Addr.	Jump Subroutine Long	[SP] ← PL, [SP−1] ← PU, SP−2, PC ← ii
JSR	Addr.	Jump Subroutine	[SP] ← PL, [SP−1] ← PU, SP−2, PC11..PC0 ← ii
JID	Disp.	Jump Indirect	PL ← ROM(PU, A)
RET	Addr.	Return from Subroutine	SP+2, PL ← [SP], PU ← [SP−1]
RETSK	Addr.	Return and Skip	SP+2, PL ← [SP], PU ← [SP−1], Skip next Instr.
RETI		Return from Interrupt	SP+2, PL ← [SP], PU ← [SP−1], GIE ← 1
INTR		Generate an Interrupt	[SP] ← PL, [SP−1] ← PU, SP−2, PC ← 0FF
NOP		No Operation	PC ← PC+1

Instruction Set (Continued)

- Most instructions are single byte (with immediate addressing mode instructions requiring two bytes).
- Most single byte instructions take one cycle time to execute.
- Skipped instructions require x number of cycles to be skipped, where x equals the number of bytes in the skipped instruction opcode.

The following tables show the number of bytes and cycles for each instruction in the format byte/cycle.

Arithmetic and Logic Instructions (Bytes/Cycles)

Instr	[B]	Direct	Immediate
ADD	1/1	3/4	
ADC	1/1	3/4	2/2
SUBC	1/1	3/4	2/2
AND	1/1	3/4	2/2
OR	1/1	3/4	2/2
XOR	1/1	3/4	2/2
IFEQ	1/1	3/4	2/2
IFNE	1/1	3/4	2/2
IFGT	1/1	3/4	2/2
IFBNE	1/1		2/2
DRSZ	1/1	1/3	
SBIT	1/1	3/4	
RBIT	1/1	3/4	
IFBIT	1/1	3/4	

Instructions Using A and C (Bytes/Cycles)

Instr	Bytes/Cycles
CLRA	1/1
INCA	1/1
DECA	1/1
LAID	1/3
DCOR	1/1
RRCA	1/1
SWAPA	1/1
SC	1/1
RC	1/1
IFC	1/1
IFNC	1/1

Transfer of Control Instructions (Bytes/Cycles)

Instr	Bytes/Cycles
JMPL	3/4
JMP	2/3
JP	1/3
JSRL	3/5
JSR	2/5
JID	1/3
RET	1/5
RETSK	1/5
RETI	1/5
INTR	1/7
NOP	1/1

Memory Transfer Instructions (Bytes/Cycles)

Instr	Register Indirect		Direct	Immed.	Register Indirect Auto Incr and Decr	
	[B]	[X]			[B+, B−]	[X+, X−]
X A,[a]	1/1		2/3		1/2	
LD A,*	1/1		2/3		1/2	
LD B,Imm		1/3		2/2		1/3
LD B,Imm		1/3		1/1[b]		1/3
LD Mem,Imm	2/2		3/3	2/3[c]	2/2	
LD Reg,Imm			2/3			

a. Memory location addressed by B or X directly

b. IF B < 16

c. IF B > 15

4

UPPER NIBBLE BITS 7–4																
F	**E**	**D**	**C**	**B**	**A**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**	
JP-15	JP-31	LD 0F0, #i	DRSZ 0F0	RRCA	RC	ADCA, 3 I	ADCA, (B)	IFBIT 0, (B)	*	LD B, 0F	IFBNE 0	JSR 0000–00FF	JMP 0000–00FF	JP+17	INTR	0
JP-14	JP-30	LD 0F1, #1	DRSZ,0F1	*	SC	SUBCA, #i	SUBC A,(B)	IFBIT 1,(B)	*	LD B, 0E	IFBNE 1	JSR 0100-01FF	JMP 0100–01FF	JP+18	JP+2	1
JP-13	JP-29	LD 0F2, #i	DRSZ 0F2	XA (X+)	XA, (X+)	IFEQA, #i	IFEQ, #i	IFBIT A,(B)	*	LD B, 0D	IFBNE 2	JSR 0200–02FF	JMP 0200–02FF	JP+19	UJP+3	2
JP-12	JP-28	LD 0F3#i	DRSZ 0F3	XA, (X−)	XA, (B−)	IFGT A, #i	IFGT A, (B)	IFBIT 3, (B)	*	LDB, 0C	IFBNE 3	JSR 0300–03FF	JMP 0300–03FF	JP+20	JP+4	3
JP-11	JP-27	LD 0F4, #i	DRSZ 0F4	*	LAID	ADD A, #i	ADD A, (B)	IFBIT 4, (B)	CLRA	LD B, 0B	IFBNE 4	JSR 0400–04FF	JMP 0400–04FF	JP+21	JP+5	4
JP-10	JP-26	LD 0F5, #i	DRSZ 0F5	*	JID	AND A, #i	AND A, (B)	IFBIT 5, (B)	SWAPA	LD B, 0A	IFBNE 5	JSR 0500–05FF	JMP 0500–05FF	JP+22	JP+6	5
JP-9	JP-25	LD 0F6, #i	DRSZ 0F6	XA, (X)	XA, (B)	XOR A, #i	XOR A, (B)	IFBIT 6, (B)	DCORA	LD B, 9	IFBNE 6	JSR 0600–06FF	JMP 0600–06FF	JP+23	JP+7	6
JP-8	JP-24	LD 0F7, #i	DRSZ 0F7	*	*	OR A, #i	OR A, (B)	IFBIT 7, (B)	*	LD B, 8	IFBNE 7	JSR 0700–07FF	JMP 0700–07FF	JP+24	JP+8	7
JP-7	JP-23	LD 0F8, #i	DRSZ 0F8	NOP	*	LD A, #i	IFC	SBIT 0,(B)	RBIT 0, (B)	LD B, 7	IFBNE 8	JSR 0800–08FF	JMP 0800–08FF	JP+25	JP+9	8
JP-6	JP-22	LD 0F9, #i	DRSZ 0F9	*	*	*	IFNC	SBIT 1,(B)	RBIT 1(B)	LD B, 6	IFBNE 9	JSR 0900–09FF	JMP 0900–09FF	JP+26	JP+10	9
JP-5	JP-21	LD 0FA, #i	DRSZ 0FA	LDA, X(+)	LD A, (B+)	LD (B+), #i	INCA	SBIT 2, (B)	RBIT 2, (B)	LD B, 5	IFBNE 0A	JSR 0A00–0AFF	JMP 0A00–0AFF	JP+27	JP+11	A
JP-4	JP-20	LD 0FB, #i	DRSZ 0FB	LDA, X(−)	LD A, (B−)	LD (B−), #i	DECA	SBIT 3, (B)	RBIT 3, (B)	LD B, 4	IFBNE 0B	JSR 0B00–0BFF	JMP 0B00–0BFF	JP+28	JP+12	B
JP-3	JP-19	LD 0FC, #i	DRSZ 0FC	LD Md, #i	JMPL	X A,Md	*	SBIT 4, (B)	RBIT 4, (B)	LD B, 3	IFBNE 0C	JSR 0C00–0CFF	JMP 0C00–0CFF	JP+29	JP+13	C
JP-2	JP-18	LD 0D, #i	DRSZ 0D	DIR	JSRL	LD A, Md	RETSK	SBIT 5, (B)	RBIT 5, (B)	LD B, 2	IFBNE 0D	JSR 0D00–0DFF	JMP 0D00–0DFF	JP+30	JP+14	D
JP-1	JP-17	LD 0FE, #i	DRSZ 0FE	LD A, (X)	LD A, (B)	LD B, #i	RET	SBIT 6, (B)	RBIT 6, (B)	LD B, 1	IFBNE 0E	JSR 0E00–0EFF	JMP 0E00–0EFF	JP+31	JP+15	E
JP-0	JP-16	LD 0FF, #i	DRSZ 0FF	*	*	*	RETI	SBIT 7, (B)	RBIT 7, (B)	LD B, 0	IFBNE 0F	JSR 0F00–0FFF	JMP 0F00–0FFF	JP+32	JP+16	F

LOWER NIBBLE BITS 3–0

Option List

The mask programmable options are listed out below. The options are programmed at the same time as the ROM pattern to provide the user with hardware flexibility to use a variety of oscillator configuration.

OPTION 1: CKI INPUT

= 1 Crystal (CKI/10) CKO for crystal configuration
= 2 NA
= 3 R/C (CKI/10) CKO available as G7 input

OPTION 2: BONDING

= 1 NA
= 2 NA
= 3 20 pin DIP package
= 4 20 pin SO package
= 5 NA

The following option information is to be sent to National along with the EPROM.

Option Data

Option 1 Value__is: CKI Input
Option 2 Value__is: COP Bonding

Development Support

SUMMARY

- iceMASTER™: IM-COP8/400—Full feature in-circuit emulation for all COP8 products. A full set of COP8 Basic and Feature Family device and package specific probes are available.
- COP8 Debug Module: Moderate cost in-circuit emulation and development programming unit.
- COP8 Evaluation and Programming Unit: EPU-COP8780—low cost in-circuit simulation and development programming unit.
- Assembler: COP8-DEV-IBMA. A DOS installable cross development Assembler, Linker, Librarian and Utility Software Development Tool Kit.
- C Compiler: COP8C. A DOS installable cross development Software Tool Kit.
- OTP/EPROM Programmer Support: Covering needs from engineering prototype, pilot production to full production environments.

Development Support (Continued)

IceMASTER (IM) IN-CIRCUIT EMULATION

The iceMASTER IM-COP8/400 is a full feature, PC based, in-circuit emulation tool developed and marketed by MetaLink Corporation to support the whole COP8 family of products. National is a resale vendor for these products.

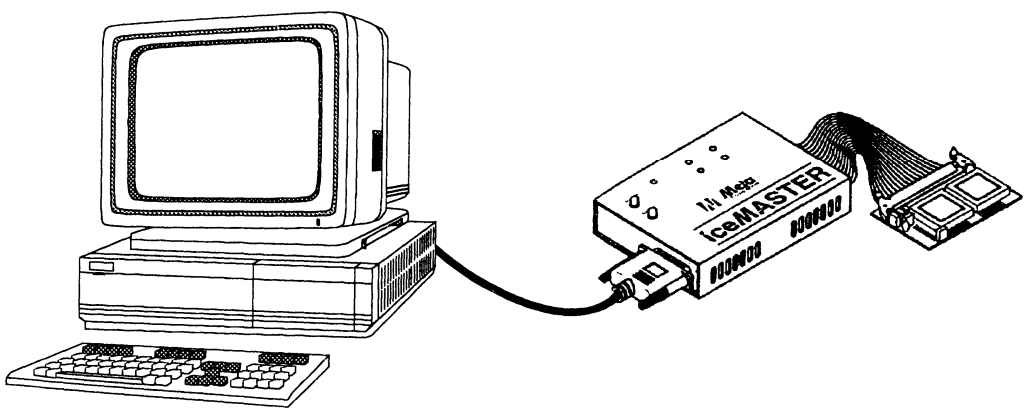
See *Figure 12* for configuration.

The iceMASTER IM-COP8/400 with its device specific COP8 Probe provides a rich feature set for developing, testing and maintaining product:

- Real-time in-circuit emulation; full 2.4V–5.5V operation range, full DC-10 MHz clock. Chip options are programmable or jumper selectable.
- Direct connection to application board by package compatible socket or surface assembly.
- Full 32 kbyte of loadable programming space that overlays (replaces) the on-chip ROM or EPROM. On-chip RAM and I/O blocks are used directly or recreated on the probe as necessary.
- Full 4k frame synchronous trace memory. Address, instruction, and 8 unspecified, circuit connectable trace lines. Display can be HLL source (e.g., C source), assembly or mixed.
- A full 64k hardware configurable break, trace on, trace off control, and pass count increment events.
- Tool set integrated interactive symbolic debugger—supports both assembler (COFF) and C Compiler (.COD) linked object formats.
- Real time performance profiling analysis; selectable bucket definition.
- Watch windows, content updated automatically at each execution break.
- Instruction by instruction memory/register changes displayed on source window when in single step operation.
- Single base unit and debugger software reconfigurable to support the entire COP8 family; only the probe personality needs to change. Debugger software is processor customized, and reconfigured from a master model file.
- Processor specific symbolic display of registers and bit level assignments, configured from master model file.
- Halt/Idle mode notification.
- On-Line HELP customized to specific processor using master model file.
- Includes a copy of COP8-DEV-IBMA assembler and linker SDK.

IM Order Information

Base Unit	
IM-COP8/400-1	iceMASTER Base Unit, 110V Power Supply
IM-COP8/400-2	iceMASTER Base Unit, 220V Power Supply
iceMASTER Probe	
MHW-880C-20DWPC	20 DIP
Adapter for SO Package	
MHW-SOIC20	20 SO



TL/DD/12060–15

FIGURE 12. COP8 iceMASTER Environment

Development Support (Continued)

iceMASTER DEBUG MODULE (DM)

The iceMASTER Debug Module is a PC based, combination in-circuit emulation tool and COP8 based OTP/EPROM programming tool developed and marketed by MetaLink Corporation to support the whole COP8 family of products. National is a resale vendor for these products.

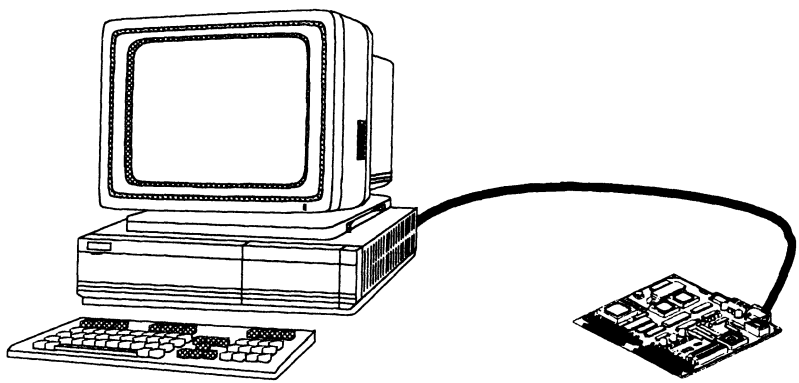
See *Figure 13* for configuration.

The iceMASTER Debug Module is a moderate cost development tool. It has the capability of in-circuit emulation for a specific COP8 microcontroller and in addition serves as a programming tool for COP8 OTP and EPROM product families. Summary of features is as follows:

- Real-time in-circuit emulation; full operating voltage range operation, full DC-10 MHz clock.
- All processor I/O pins can be cabled to an application development board with package compatible cable to socket and surface mount assembly.
- Full 32 kbyte of loadable programming space that overlays (replaces) the on-chip ROM or EPROM. On-chip RAM and I/O blocks are used directly or recreated as necessary.
- 100 frames of synchronous trace memory. The display can be HLL source (C source), assembly or mixed. The most recent history prior to a break is available in the trace memory.
- Configured break points; uses INTR instruction which is modestly intrusive.
- Software—only supported features are selectable.
- Tool set integrated interactive symbolic debugger—supports both assembler (COFF) and C Compiler (.COD) SDK linked object formats.
- Instruction by instruction memory/register changes displayed when in single step operation.
- Debugger software is processor customized, and reconfigured from a master model file.
- Processor specific symbolic display of registers and bit level assignments, configured from master model file.
- Halt/Idle mode notification.
- Programming menu supports full product line of programmable OTP and EPROM COP8 products. Program data is taken directly from the overlay RAM.
- Programming of 44 PLCC and 68 PLCC parts requires external programming adapters.
- Includes wallmount power supply.
- On-board V_{PP} generator from 5V input or connection to external supply supported. Requires V_{PP} level adjustment per the family programming specification (correct level is provided on an on-screen pop-down display).
- On-line HELP customized to specific processor using master model file.
- Includes a copy of COP8-DEV-IBMA assembler and linker SDK.

DM Order Information

Debug Module Unit	
COP8/DM/880C	
Cable Adapters	
DM-COP8/20D	20 DIP
Adapter for SO Package	
MHW-SOIC20	20 SO



TL/DD/12060–23

FIGURE 13. COP8-DM Environment

Development Support (Continued)

COP8 ASSEMBLER/LINKER SOFTWARE DEVELOPMENT TOOL KIT

National Semiconductor offers a relocatable COP8 macro cross assembler, linker, librarian and utility software development tool kit. Features are summarized as follows:

- Basic and Feature Family instruction set by "device" type.
- Nested macro capability.
- Extensive set of assembler directives.
- Supported on PC/DOS platform.
- Generates National standard COFF output files.
- Integrated Linker and Librarian.
- Integrated utilities to generate ROM code file outputs.
- DUMPCOFF utility.

This product is integrated as a part of MetaLink tools as a development kit, fully supported by the MetaLink debugger. It may be ordered separately or it is bundled with the MetaLink products at no additional cost.

Order Information

Assembler SDK:	
COP8-DEV-IBMA	Assembler SDK on installable 3.5" PC®/DOS Floppy Disk Drive format. Periodic upgrades and most recent version is available on National's BBS and Internet.

COP8 C COMPILER

A C Compiler is developed and marketed by Byte Craft Limited. The COP8C compiler is a fully integrated development tool specifically designed to support the compact embedded configuration of the COP8 family of products.

Features are summarized as follows:

- ANSI C with some restrictions and extensions that optimize development for the COP8 embedded application.
- BITS data type extension. Register declaration #pragma with direct bit level definitions.
- C language support for interrupt routines.
- Expert system, rule based code generation and optimization.
- Performs consistency checks against the architectural definitions of the target COP8 device.
- Generates program memory code.
- Supports linking of compiled object or COP8 assembled object formats.
- Global optimization of linked code.
- Symbolic debug load format fully source level supported by the MetaLink debugger.

SINGLE CHIP OTP/EMULATOR SUPPORT

The COP8 family is supported by single chip OTP emulators. For detailed information refer to the emulator specific datasheet and the emulator selection table below:

Approved List

Manufacturer	North America	Europe	Asia
BP Microsystems	(800) 225-2102 (713) 688-4600 Fax: (713) 688-0920	+49-8152-4183 +49-8856-932616	+852-234-16611 +852-2710-8121
Data I/O	(800) 426-1045 (206) 881-6444 Fax: (206) 882-1043	+44-0734-440011	Call North America
HI-LO	(510) 623-8860	Call Asia	+886-2-764-0215 Fax: +886-2-756-6403
ICE Technology	(800) 624-8949 (919) 430-7915	+44-1226-767404 Fax: 0-1226-370-434	
MetaLink	(800) 638-2423 (602) 926-0797 Fax: (602) 693-0681	+49-80 9156 96-0 Fax: +49-80 9123 86	+852-737-1800
Systems General	(408) 263-6667	+41-1-9450300	+886-2-917-3005 Fax: +886-2-911-1283
Needhams	(916) 924-8037 Fax: (916) 924-8065		

Development Support (Continued)

OTP Emulator Ordering Information

Device Number	Clock Option	Package	Emulates
COP8782CN	Programmable	20 N	COP912C, COP912CH
COP8782CWM	Programmable	20 SO	COP912C, COP912CH

INDUSTRY WIDE OTP/EPROM PROGRAMMING SUPPORT

Programming support, in addition to the MetaLink development tools, is provided by a full range of independent approved vendors to meet the needs from the engineering laboratory to full production.

AVAILABLE LITERATURE

For more information, please see the COP8 Basic Family User's Manual, Literature Number 620895, COP8 Feature Family User's Manual, Literature Number 620897 and National's Family of 8-bit Microcontrollers COP8 Selection Guide, Literature Number 630009.

DIAL-A-HELPER SERVICE

Dial-A-Helper is a service provided by the Microcontroller Applications group. The Dial-A-Helper is an Electronic Information System that may be accessed as a Bulletin Board System (BBS) via data modem, as an FTP site on the Internet via standard FTP client application or as an FTP site on the Internet using a standard Internet browser such as Netscape or Mosaic.

The Dial-A-Helper system provides access to an automated information storage and retrieval system. The system capabilities include a MESSAGE SECTION (electronic mail, when accessed as a BBS) for communications to and from the Microcontroller Applications Group and a FILE SECTION which consists of several file areas where valuable application software and utilities could be found.

DIAL-A-HELPER BBS via a Standard Modem

Modem: CANADA/U.S.: (800) NSC-MICRO
(800) 672-6427
EUROPE: (+49) 0-8141-351332
Baud: 14.4k
Set-up: Length: 8-Bit
Parity: None
Stop Bit: 1
Operation: 24 Hours, 7 Days

DIAL-A-HELPER via FTP

ftp nscmicro.nsc.com
user: anonymous
password: username@yourhost.site.domain

DIAL-A-HELPER via a WorldWide Web Browser

ftp://nscmicro.nsc.com

National Semiconductor on the WorldWide Web

See us on the WorldWide Web at: http://www.national.com

CUSTOMER RESPONSE CENTER

Complete product information and technical support is available from National's customer response centers.

CANADA/U.S.:	Tel:	(800) 272-9959
	email:	support @tevm2.nsc.com
EUROPE:	email:	europe.support@nsc.com
	Deutsch Tel:	+49 (0) 180-530 85 85
	English Tel:	+49 (0) 180-532 78 32
	Français Tel:	+49 (0) 180-532 93 58
	Italiano Tel:	+49 (0) 180-534 16 80
JAPAN:	Tel:	+81-043-299-2309
S.E. ASIA:	Beijing Tel:	(+86) 10-6856-8601
	Shanghai Tel:	(+86) 21-6415-4092
	Hong Kong Tel:	(+852) 2737-1600
	Korea Tel:	(+82) 2-3771-6909
	Malaysia Tel:	(+60-4) 644-9061
	Singapore Tel:	(+65) 255-2226
	Taiwan Tel:	+886-2-521-3288
AUSTRALIA:	Tel:	(+61) 3-9558-9999
INDIA:	Tel:	(+91) 80-559-9467

National Semiconductor

COP620C/COP622C/COP640C/COP642C/ COP820C/COP822C/COP840C/COP842C/ COP920C/COP922C/COP940C/COP942C 8-Bit Microcontroller

General Description

The COP820C and COP840C are members of the COP8™ microcontroller family. They are fully static parts, fabricated using double-metal silicon gate microCMOS technology. This low cost microcontroller is a complete microcomputer containing all system timing, interrupt logic, ROM, RAM, and I/O necessary to implement dedicated control functions in a variety of applications. Features include an 8-bit memory mapped architecture, MICROWIRE/PLUS™ serial I/O, a 16-bit timer/counter with capture register and a multi-sourced interrupt. Each I/O pin has software selectable options to adapt the device to the specific application. The part operates over a voltage range of 2.5 to 6.0V. High throughput is achieved with an efficient, regular instruction set operating at a 1 microsecond per instruction rate.

Key Features

- 16-bit multi-function timer supporting
  - — PWM mode
  - — External event counter mode
  - — Input capture mode
- 1024 bytes ROM/64 bytes RAM-COP820C family
- 2048 bytes ROM/128 bytes RAM-COP840C family

I/O Features

- Memory mapped I/O
- Software selectable I/O options (TRI-STATE® Output, Push-Pull Output, Weak Pull-Up Input, High Impedance Input)
- High current outputs
- Schmitt trigger inputs on Port G
- MICROWIRE/PLUS serial I/O
- Packages:
  - — 20 DIP/SO with 16 I/O pins
  - — 28 DIP/SO with 24 I/O pins

CPU/Instruction Set Feature

- 1 μs instruction cycle time
- Three multi-source interrupts servicing
  - — External interrupt with selectable edge
  - — Timer interrupt
  - — Software interrupt
- Versatile and easy to use instruction set
- 8-bit Stack point (SP)—stack in RAM
- Two 8-bit Register Indirect Memory Pointers (B, X)

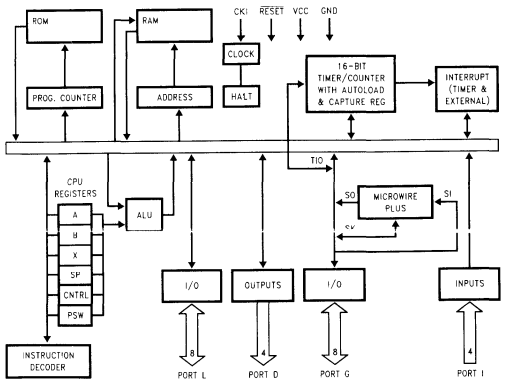
Fully Static CMOS

- Low current drain (typically < 1 μA)
- Single supply operation: 2.5V to 6.0V
- Temperature range: 0°C to +70°C, −40°C to +85°C, −55°C to +125°C

Development Support

- Emulation and OTP devices
- Real time emulation and full program debug offered by MetaLink's Development System

Block Diagram



TL/DD/9103–1

FIGURE 1

Absolute Maximum Ratings

If Military/Aerospace specified devices are required, please contact the National Semiconductor Sales Office/Distributors for availability and specifications.

Supply Voltage (V_{CC})	7V
Voltage at any Pin	−0.3V to V_{CC} + 0.3V
Total Current into V_{CC} Pin (Source)	50 mA
Total Current out of GND Pin (Sink)	60 mA
Storage Temperature Range	−65°C to +140°C

Note: Absolute maximum ratings indicate limits beyond which damage to the device may occur. DC and AC electrical specifications are not ensured when operating the device at absolute maximum ratings.

DC Electrical Characteristics COP92XC, COP94XC; 0°C ≤ T_A ≤ +70°C unless otherwise specified

Parameter	Condition	Min	Typ	Max	Units
Operating Voltage					
COP9XXC		2.3		4.0	V
COP9XXCH		4.0		6.0	V
Power Supply Ripple (Note 1)	Peak to Peak			0.1 V_{CC}	V
Supply Current (Note 2)					
CKI = 10 MHz	V_{CC} = 6V, tc = 1 μs			6.0	mA
CKI = 4 MHz	V_{CC} = 6V, tc = 2.5 μs			4.0	mA
CKI = 4 MHz	V_{CC} = 4V, tc = 2.5 μs			2.0	mA
CKI = 1 MHz	V_{CC} = 4V, tc = 10 μs			1.2	mA
HALT Current	V_{CC} = 6V, CKI = 0 MHz		<0.7	8.0	μA
(Note 3)	V_{CC} = 4V, CKI = 0 MHz		<0.4	5.0	μA
Input Levels					
$\overline{\text{RESET}}$, CKI					
Logic High		0.9 V_{CC}			V
Logic Low				0.1 V_{CC}	V
All Other Inputs					
Logic High		0.7 V_{CC}			V
Logic Low				0.2 V_{CC}	V
Hi-Z Input Leakage	V_{CC} = 6.0V	−1		+1	μA
Input Pullup Current	V_{CC} = 6.0V, V_{IN} = 0V	−40		−250	μA
G Port Input Hysteresis				0.35 V_{CC}	V
Output Current Levels					
D Outputs					
Source	V_{CC} = 4.5V, V_{OH} = 3.8V	−0.4			mA
	V_{CC} = 2.3V, V_{OH} = 1.6V	−0.2			mA
Sink	V_{CC} = 4.5V, V_{OL} = 1.0V	10			mA
	V_{CC} = 2.3V, V_{OL} = 0.4V	2			mA
All Others					
Source (Weak Pull-Up)	V_{CC} = 4.5V, V_{OH} = 3.2V	−10		−110	μA
	V_{CC} = 2.3V, V_{OH} = 1.6V	−2.5		−33	μA
Source (Push-Pull Mode)	V_{CC} = 4.5V, V_{OH} = 3.8V	−0.4			mA
	V_{CC} = 2.3V, V_{OH} = 1.6V	−0.2			
Sink (Push-Pull Mode)	V_{CC} = 4.5V, V_{OL} = 0.4V	1.6			mA
	V_{CC} = 2.3V, V_{OL} = 0.4V	0.7			
TRI-STATE Leakage	V_{CC} = 6.0V	−1.0		+1.0	μA
Allowable Sink/Source Current Per Pin					
D Outputs (Sink)				15	mA
All Others				3	mA
Maximum Input Current (Note 4) Without Latchup (Room Temp)	Room Temp			±100	mA
RAM Retention Voltage, Vr	500 ns Rise and Fall Time (Min)	2.0			V
Input Capacitance				7	pF
Load Capacitance on D2				1000	pF

COP920C/COP922C/COP940C/COP942C

DC Electrical Characteristics (Continued)

Note 1: Rate of voltage change must be less than 0.5V/ms.

Note 2: Supply current is measured after running 2000 cycles with a square wave CKI input, CKO open, inputs at rails and outputs open.

Note 3: The HALT mode will stop CKI from oscillating in the RC and the Crystal configurations. Test conditions: All inputs tied to V_{CC}, L and G0—G5 configured as outputs and set high. The D port set to zero.

Note 4: Except pin G7: +100 mA, −25 mA (COP920C only). Sampled and not 100% tested. Pins G6 and $\overline{\text{RESET}}$ are designed with a high voltage input network for factory testing. These pins allow input voltages greater than V_{CC} and the pins will have sink current to V_{CC} when biased at voltages greater than V_{CC} (the pins do not have source current when biased at a voltage below V_{CC}). The effective resistance to V_{CC} is 750Ω (typical). These two pins will not latch up. The voltage at the pins must be limited to less than 14V.

AC Electrical Characteristics $0°C \le T_A \le +70°C$ unless otherwise specified

Parameter	Condition	Min	Typ	Max	Units
Instruction Cycle Time (tc)					
Ext., Crystal/Resonator	$V_{CC} \ge 4.0V$	1		DC	μs
(Div-by 10)	$2.3V \le V_{CC} \le 4.0V$	2.5		DC	μs
R/C Oscillator Mode	$V_{CC} \ge 4.0V$	3		DC	μs
(Div-by 10)	$2.3V \le V_{CC} \le 4.0V$	7.5		DC	μs
CKI Clock Duty Cycle (Note 5)	fr = Max	40		60	%
Rise Time (Note 5)	fr = 10 MHz Ext Clock			12	ns
Fall Time (Note 5)	fr = 10 MHz Ext Clock			8	ns
Inputs					
t_{SETUP}	$V_{CC} \ge 4.0V$	200			ns
	$2.3V \le V_{CC} \le 4.0V$	500			ns
t_{HOLD}	$V_{CC} \ge 4.0V$	60			ns
	$2.3V \le V_{CC} \le 4.0V$	150			ns
Output Propagation Delay	$C_L = 100$ pF, $R_L = 2.2$ kΩ				
t_{PD1}, t_{PD0}					
SO, SK	$V_{CC} \ge 4.0V$			0.7	μs
	$2.5V \le V_{CC} \le 4.0V$			1.75	μs
All Others	$V_{CC} \ge 4.0V$			1	μs
	$2.5V \le V_{CC} \le 4.0V$			2.5	μs
MICROWIRE™ Setup Time (t_{UWS})		20			ns
MICROWIRE Hold Time (t_{UWH})		56			ns
MICROWIRE Output Propagation Delay (t_{UPD})				220	ns
Input Pulse Width					
Interrupt Input High Time		t_C			
Interrupt Input Low Time		t_C			
Timer Input High Time		t_C			
Timer Input Low Time		t_C			
Reset Pulse Width		1.0			μs

Note 5: Parameter sampled (not 100% tested).

COP820C/COP822C/COP840C/COP842C

Absolute Maximum Ratings

If Military/Aerospace specified devices are required, please contact the National Semiconductor Sales Office/Distributors for availability and specifications.

Supply Voltage (V_{CC})	7V
Voltage at any Pin	−0.3V to V_{CC} + 0.3V
Total Current into V_{CC} Pin (Source)	50 mA
Total Current out of GND Pin (Sink)	60 mA
Storage Temperature Range	−65°C to +140°C

Note: Absolute maximum ratings indicate limits beyond which damage to the device may occur. DC and AC electrical specifications are not ensured when operating the device at absolute maximum ratings.

DC Electrical Characteristics

COP82XC, COP84XC: $-40°C \le T_A \le +85°C$ unless otherwise specified

Parameter	Condition	Min	Typ	Max	Units
Operating Voltage		2.5		6.0	V
Power Supply Ripple (Note 1)	Peak to Peak			0.1 V_{CC}	V
Supply Current (Note 2)					
CKI = 10 MHz	V_{CC} = 6V, tc = 1 μs			6.0	mA
CKI = 4 MHz	V_{CC} = 6V, tc = 2.5 μs			4.0	mA
CKI = 4 MHz	V_{CC} = 4.0V, tc = 2.5 μs			2.0	mA
CKI = 1 MHz	V_{CC} = 4.0V, tc = 10 μs			1.2	mA
HALT Current (Note 3)	V_{CC} = 6V, CKI = 0 MHz		<1	10	μA
Input Levels					
RESET, CKI					
Logic High		0.9 V_{CC}			V
Logic Low				0.1 V_{CC}	V
All Other Inputs					
Logic High		0.7 V_{CC}			V
Logic Low				0.2 V_{CC}	V
Hi-Z Input Leakage	V_{CC} = 6.0V	−2		+2	μA
Input Pullup Current	V_{CC} = 6.0V, V_{IN} = 0V	−40		−250	μA
G Port Input Hysteresis				0.35 V_{CC}	V
Output Current Levels					
D Outputs					
Source	V_{CC} = 4.5V, V_{OH} = 3.8V	−0.4			mA
	V_{CC} = 2.5V, V_{OH} = 1.8V	−0.2			mA
Sink	V_{CC} = 4.5V, V_{OL} = 1.0V	10			mA
	V_{CC} = 2.5V, V_{OL} = 0.4V	2			mA
All Others					
Source (Weak Pull-Up)	V_{CC} = 4.5V, V_{OH} = 3.2V	−10		−110	μA
	V_{CC} = 2.5V, V_{OH} = 1.8V	−2.5		−33	μA
Source (Push-Pull Mode)	V_{CC} = 4.5V, V_{OH} = 3.8V	−0.4			mA
	V_{CC} = 2.5V, V_{OH} = 1.8V	−0.2			
Sink (Push-Pull Mode)	V_{CC} = 4.5V, V_{OL} = 0.4V	1.6			mA
	V_{CC} = 2.5V, V_{OL} = 0.4V	0.7			
TRI-STATE Leakage		−2.0		+2.0	μA
Allowable Sink/Source Current Per Pin					
D Outputs (Sink)				15	mA
All Others				3	mA
Maximum Input Current (Note 4) Without Latchup (Room Temp)	Room Temp			±100	mA
RAM Retention Voltage, Vr	500 ns Rise and Fall Time (Min)	2.0			V
Input Capacitance				7	pF
Load Capacitance on D2				1000	pF

Note 1: Rate of voltage change must be less than 0.5V/ms.

Note 2: Supply current is measured after running 2000 cycles with a square wave CKI input, CKO open, inputs at rails and outputs open.

Note 3: The HALT mode will stop CKI from oscillating in the RC and the Crystal configurations. Test conditions: All inputs tied to V_{CC}, L and G0—G5 configured as outputs and set high. The D port set to zero.

Note 4: Except pin G7: +100 mA, −25 mA (COP820C only). Sampled and not 100% tested. Pins G6 and RESET are designed with a high voltage input network for factory testing. These pins allow input voltages greater than V_{CC} and the pins will have sink current to V_{CC} when biased at voltages greater than V_{CC} (the pins do not have source current when biased at a voltage below V_{CC}). The effective resistance to V_{CC} is 750Ω (typical). These two pins will not latch up. The voltage at the pins must be limited to less than 14V.

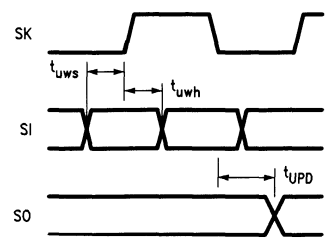
COP820C/COP822C/COP840C/COP842C

AC Electrical Characteristics $-40°C \le T_A \le +85°C$ unless otherwise specified

Parameter	Condition	Min	Typ	Max	Units
Instruction Cycle Time (tc)					
Ext. or Crystal/Resonator	$V_{CC} \ge 4.5V$	1		DC	μs
(Div-by 10)	$2.5V \le V_{CC} < 4.5V$	2.5		DC	μs
R/C Oscillator Mode	$V_{CC} \ge 4.5V$	3		DC	μs
(Div-by 10)	$2.5V \le V_{CC} < 4.5V$	7.5		DC	μs
CKI Clock Duty Cycle (Note 5)	fr = Max	40		60	%
Rise Time (Note 5)	fr = 10 MHz Ext Clock			12	ns
Fall Time (Note 5)	fr = 10 MHz Ext Clock			8	ns
Inputs					
t_{SETUP}	$V_{CC} \ge 4.5V$	200			ns
	$2.5V \le V_{CC} < 4.5V$	500			ns
t_{HOLD}	$V_{CC} \ge 4.5V$	60			ns
	$2.5V \le V_{CC} < 4.5V$	150			ns
Output Propagation Delay	C_L = 100 pF, R_L = 2.2 kΩ				
t_{PD1}, t_{PD0}					
SO, SK	$V_{CC} \ge 4.5V$			0.7	μs
	$2.5V \le V_{CC} < 4.5V$			1.75	μs
All Others	$V_{CC} \ge 4.5V$			1	μs
	$2.5V \le V_{CC} < 4.5V$			2.5	μs
MICROWIRE Setup Time (t_{UWS})		20			ns
MICROWIRE Hold Time (t_{UWH})		56			ns
MICROWIRE Output Propagation Delay (t_{UPD})				220	ns
Input Pulse Width					
Interrupt Input High Time		t_C			
Interrupt Input Low Time		t_C			
Timer Input High Time		t_C			
Timer Input Low Time		t_C			
Reset Pulse Width		1.0			μs

Note 5: Parameter sampled (not 100% tested).

Timing Diagram



TL/DD/9103-19

FIGURE 2. MICROWIRE/PLUS Timing

COP620C/COP622C/COP640C/COP642C

Absolute Maximum Ratings

If Military/Aerospace specified devices are required, please contact the National Semiconductor Sales Office/Distributors for availability and specifications.

Supply Voltage (V_{CC})	6V
Voltage at any Pin	$-0.3V$ to $V_{CC} + 0.3V$
Total Current into V_{CC} Pin (Source)	40 mA
Total Current out of GND Pin (Sink)	48 mA
Storage Temperature Range	$-65°C$ to $+140°C$

Note: *Absolute maximum ratings indicate limits beyond which damage to the device may occur. DC and AC electrical specifications are not ensured when operating the device at absolute maximum ratings.*

DC Electrical Characteristics

COP62XC, COP64XC: $-55°C \le T_A \le +125°C$ unless otherwise specified

Parameter	Condition	Min	Typ	Max	Units
Operating Voltage		4.5		5.5	V
Power Supply Ripple (Note 1)	Peak to Peak			0.1 V_{CC}	V
Supply Current (Note 2)					
CKI = 10 MHz	V_{CC} = 5.5V, tc = 1 μs			6.0	mA
CKI = 4 MHz	V_{CC} = 5.5V, tc = 2.5 μs			4	mA
HALT Current (Note 3)	V_{CC} = 5.5V, CKI = 0 MHz		<10	30	μA
Input Levels					
$\overline{\text{RESET}}$, CKI					
Logic High		0.9 V_{CC}			V
Logic Low				0.1 V_{CC}	V
All Other Inputs					
Logic High		0.7 V_{CC}			V
Logic Low				0.2 V_{CC}	V
Hi-Z Input Leakage	V_{CC} = 5.5V	−5		+5	μA
Input Pullup Current	V_{CC} = 4.5V, V_{IN} = 0V	−35		−300	μA
G Port Input Hysteresis				0.35 V_{CC}	V
Output Current Levels					
D Outputs					
Source	V_{CC} = 4.5V, V_{OH} = 3.8V	−0.35			mA
Sink	V_{CC} = 4.5V, V_{OL} = 1.0V	9			mA
All Others					
Source (Weak Pull-Up)	V_{CC} = 4.5V, V_{OH} = 3.2V	−9		−120	μA
Source (Push-Pull Mode)	V_{CC} = 4.5V, V_{OH} = 3.8V	−0.35			mA
Sink (Push-Pull Mode)	V_{CC} = 4.5V, V_{OL} = 0.4V	1.4			mA
TRI-STATE Leakage		−5.0		+5.0	μA
Allowable Sink/Source Current Per Pin					
D Outputs (Sink)				12	mA
All Others				2.5	mA
Maximum Input Current (Room Temp) Without Latchup (Note 5)	Room Temp			±100	mA
RAM Retention Voltage, Vr	500 ns Rise and Fall Time (Min)	2.5			V
Input Capacitance				7	pF
Load Capacitance on D2				1000	pF

Note 1: Rate of voltage change must be less than 0.5V/ms.

Note 2: Supply current is measured after running 2000 cycles with a square wave CKI input, CKO open, inputs at rails and outputs open.

Note 3: The HALT mode will stop CKI from oscillating in the RC and the Crystal configurations. Test conditions: All inputs tied to V_{CC}, L and G0—G5 configured as outputs and set high. The D port set to zero.

Note 4: Except pin G7: +100 mA, −25 mA (COP620C only). Sampled and not 100% tested. Pins G6 and $\overline{\text{RESET}}$ are designed with a high voltage input network for factory testing. These pins allow input voltages greater than V_{CC} and the pins will have sink current to V_{CC} when biased at voltages greater than V_{CC} (the pins do not have source current when biased at a voltage below V_{CC}). The effective resistance to V_{CC} is 750Ω (typical). These two pins will not latch up. The voltage at the pins must be limited to less than 14V.

COP620C/COP622C/COP640C/COP642C

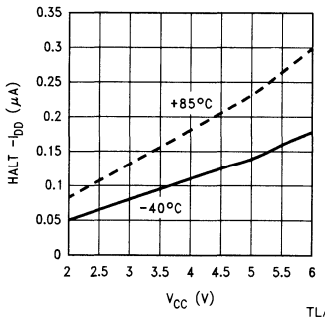
AC Electrical Characteristics $-55°C \leq T_A \leq +125°C$ unless otherwise specified

Parameter	Condition	Min	Typ	Max	Units
Instruction Cycle Time (tc)					
Ext. or Crystal/Resonant (Div-by 10)	$V_{CC} \geq 4.5V$	1		DC	μs
CKI Clock Duty Cycle (Note 5)	fr = Max	40		60	%
Rise Time (Note 5)	fr = 10 MHz Ext Clock			12	ns
Fall Time (Note 5)	fr = 10 MHz Ext Clock			8	ns
Inputs					
t_{SETUP}	$V_{CC} \geq 4.5V$	220			ns
t_{HOLD}	$V_{CC} \geq 4.5V$	66			ns
Output Propagation Delay t_{PD1}, t_{PD0}	$R_L = 2.2k, C_L = 100$ pF				
SO, SK	$V_{CC} \geq 4.5V$			0.8	μs
All Others	$V_{CC} \geq 4.5V$			1.1	μs
MICROWIRE Setup Time (t_{UWS})		20			ns
MICROWIRE Hold Time (t_{UWH})		56			ns
MICROWIRE Output Valid Time (t_{UPD})				220	ns
Input Pulse Width					
Interrupt Input High Time		t_C			
Interrupt Input Low Time		t_C			
Timer Input High Time		t_C			
Timer Input Low Time		t_C			
Reset Pulse Width		1			μs

Note 5: Parameter sampled (not 100% tested).

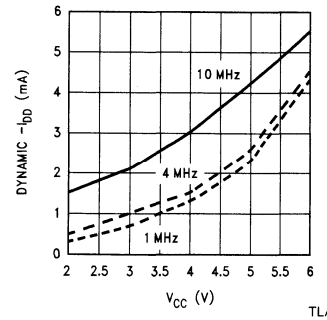
Typical Performance Characteristics ($-40°C \le T_A \le +85°C$)

Halt—I_{DD}



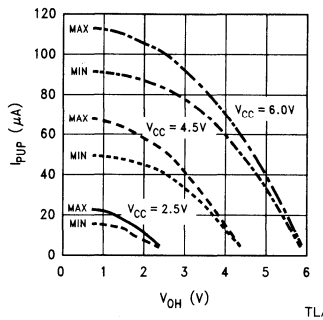
TL/DD/9103-20

Dynamic—I_{DD} (Crystal Clock Option)



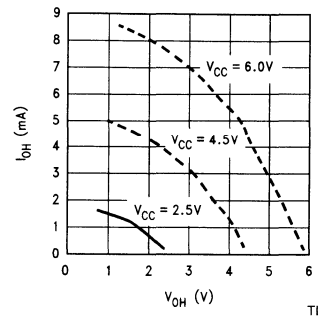
TL/DD/9103-21

Port L/G Weak Pull-Up Source Current



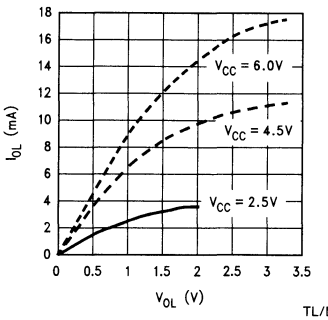
TL/DD/9103-22

Port L/G Push-Pull Source Current



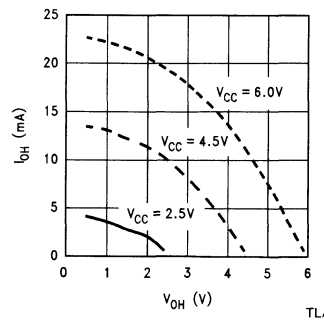
TL/DD/9103-23

Port L/G Push-Pull Sink Current



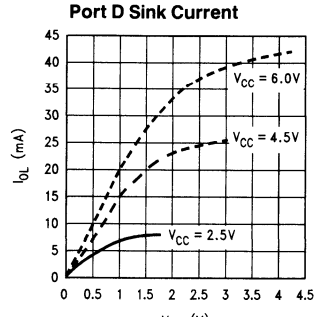
TL/DD/9103-24

Port D Source Current



TL/DD/9103-25

Port D Sink Current



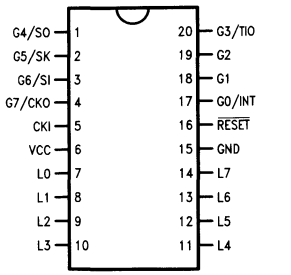
TL/DD/9103-26

4

Connection Diagrams

DUAL-IN-LINE PACKAGE

20 DIP

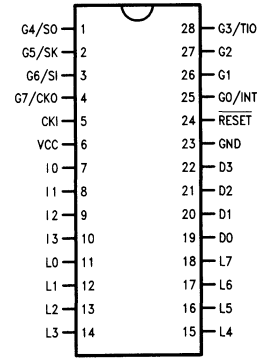


TL/DD/9103-3

Top View

Order Number COP622C-XXX/N, COP642C-XXX/N, COP822C-XXX/N, COP842C-XXX/N, COP922C-XXX/N, COP942C-XXX/N, COP922CH-XXX/N or COP942CH-XXX/N
See NS Package Number N20A

28 DIP

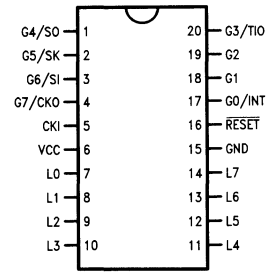


TL/DD/9103-5

Order Number COP620C-XXX/N, COP640C-XXX/N, COP820C-XXX/N, COP840C-XXX/D,COP920C-XXX/N, COP940C-XXX/N, COP920CH-XXX/N or COP940CH-XXX/N
See NS Package Number N28B

SURFACE MOUNT

20 SO Wide

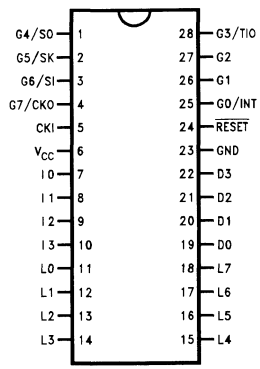


TL/DD/9103-3

Top View

Order Number COP822C-XXX/WM, COP842C-XXX/WM, COP922C-XXX/WM, COP942C-XXX/WM, COP922CH-XXX/WM or COP942CH-XXX/WM
See NS Package Number M20B

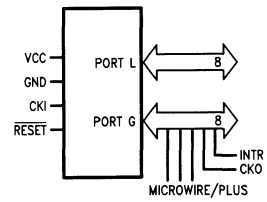
28-Lead SO



TL/DD/9103-18

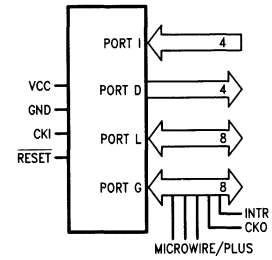
Order Number COP820C-XXX/WM, COP840C-XXX/WM, COP920C-XXX/WM, COP940C-XXX/WM, COP920CH-XXX/WM or COP940CH-XXX/WM
See NS Package Number M28B

20 DIP/SO



TL/DD/9103-6

28 DIP/SO



TL/DD/9103-8

FIGURE 3. Connection Diagrams

Pin Descriptions

V_{CC} and GND are the power supply pins.

CKI is the clock input. This can come from an external source, a R/C generated oscillator or a crystal (in conjunction with CKO). See Oscillator description.

$\overline{\text{RESET}}$ is the master reset input. See Reset description.

PORT I is a four bit Hi-Z input port.

PORT L is an 8-bit I/O port.

There are two registers associated with each L I/O port: a data register and a configuration register. Therefore, each L I/O bit can be individually configured under software control as shown below:

Port L Config.	Port L Data	Port L Setup
0	0	Hi-Z Input (TRI-STATE)
0	1	Input With Weak Pull-Up
1	0	Push-Pull "0" Output
1	1	Push-Pull "1" Output

Three data memory address locations are allocated for these ports, one for data register, one for configuration register and one for the input pins.

PORT G is an 8-bit port with 6 I/O pins (G0–G5) and 2 input pins (G6, G7). All eight G-pins have Schmitt Triggers on the inputs. The G7 pin functions as an input pin under normal operation and as the continue pin to exit the HALT mode. There are two registers with each I/O port: a data register and a configuration register. Therefore, each I/O bit can be individually configured under software control as shown below.

Port G Config.	Port G Data	Port G Setup
0	0	Hi-Z Input (TRI-STATE)
0	1	Input With Weak Pull-Up
1	0	Push-Pull "0" Output
1	1	Push-Pull "1" Output

Three data memory address locations are allocated for these ports, one for data register, one for configuration register and one for the input pins. Since G6 and G7 are input only pins, any attempt by the user to set them up as outputs by writing a one to the configuration register will be disregarded. Reading the G6 and G7 configuration bits will return zeros. Note that the chip will be placed in the HALT mode by setting the G7 data bit.

Six bits of Port G have alternate features:

G0 INTR (an external interrupt)

G3 TIO (timer/counter input/output)

G4 SO (MICROWIRE serial data output)

G5 SK (MICROWIRE clock I/O)

G6 SI (MICROWIRE serial data input)

G7 CKO crystal oscillator output (selected by mask option) or HALT restart input (general purpose input)

Pins G1 and G2 currently do not have any alternate functions.

PORT D is a four bit output port that is set high when $\overline{\text{RESET}}$ goes low. Care must be exercised with the D2 pin operation. At RESET, the external load on this pin must ensure that the output voltage stays above 0.9 V_{CC} to prevent the device from entering special modes. Also, keep the external loading on the D2 pin to less than 1000 pf.

Functional Description

Figure 1 shows the block diagram of the internal architecture. Data paths are illustrated in simplified form to depict how the various logic elements communicate with each other in implementing the instruction set of the device.

ALU AND CPU REGISTERS

The ALU can do an 8-bit addition, subtraction, logical or shift operation in one cycle time.

There are five CPU registers:

A is the 8-bit Accumulator register

PU is the upper 7 bits of the program counter (PC)

PL is the lower 8 bits of the program counter (PC)

B is the 8-bit address register, can be auto incremented or decremented.

X is the 8-bit alternate address register, can be incremented or decremented.

SP is the 8-bit stack pointer, points to subroutine stack (in RAM).

B, X and SP registers are mapped into the on chip RAM. The B and X registers are used to address the on chip RAM. The SP register is used to address the stack in RAM during subroutine calls and returns.

PROGRAM MEMORY

Program memory for the COP820C family consists of 1024 bytes of ROM (2048 bytes of ROM for the COP840C family). These bytes may hold program instructions or constant data. The program memory is addressed by the 15-bit program counter (PC). ROM can be indirectly read by the LAID instruction for table lookup.

DATA MEMORY

The data memory address space includes on chip RAM, I/O and registers. Data memory is addressed directly by the instruction or indirectly by the B, X and SP registers.

The COP820C family has 64 bytes of RAM and the COP840C family has 128 bytes of RAM. Sixteen bytes of RAM are mapped as "registers" that can be loaded immediately, decremented or tested. Three specific registers: B, X and SP are mapped into this space, the other bytes are available for general usage.

The instruction set permits any bit in memory to be set, reset or tested. All I/O and registers (except the A & PC) are memory mapped; therefore, I/O bits and register bits can be directly and individually set, reset and tested.

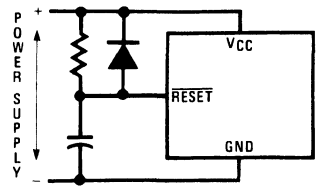
Note: RAM contents are undefined upon power-up.

RESET

The $\overline{\text{RESET}}$ input when pulled low initializes the microcontroller. Initialization will occur whenever the $\overline{\text{RESET}}$ input is pulled low. Upon initialization, the ports L and G are placed in the TRI-STATE mode and the Port D is set high. The PC, PSW and CNTRL registers are cleared. The data and configuration registers for Ports L & G are cleared.

The external RC network shown in Figure 4 should be used to ensure that the $\overline{\text{RESET}}$ pin is held low until the power supply to the chip stabilizes.

Functional Description (Continued)



TL/DD/9103–9

RC ≥ 5X Power Supply Rise Time

FIGURE 4. Recommended Reset Circuit

OSCILLATOR CIRCUITS

Figure 5 shows the three clock oscillator configurations.

A. CRYSTAL OSCILLATOR

The device can be driven by a crystal clock. The crystal network is connected between the pins CKI and CKO.

Table I shows the component values required for various standard crystal values.

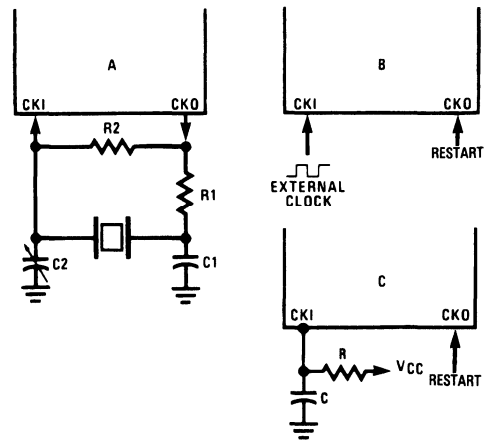
B. EXTERNAL OSCILLATOR

CKI can be driven by an external clock signal. CKO is available as a general purpose input and/or HALT restart control.

C. R/C OSCILLATOR

CKI is configured as a single pin RC controlled Schmitt trigger oscillator. CKO is available as a general purpose input and/or HALT restart control.

Table II shows the variation in the oscillator frequencies as functions of the component (R and C) values.



TL/DD/9103–10

FIGURE 5. Crystal and R-C Connection Diagrams

OSCILLATOR MASK OPTIONS

The device can be driven by clock inputs between DC and 10 MHz.

TABLE I. Crystal Oscillator Configuration, T_A = 25°C

R1 (kΩ)	R2 (MΩ)	C1 (pF)	C2 (pF)	CKI Freq (MHz)	Conditions
0	1	30	30–36	10	V_{CC} = 5V
0	1	30	30–36	4	V_{CC} = 5V
0	1	200	100–150	0.455	V_{CC} = 5V

TABLE II. RC Oscillator Configuration, T_A = 25°C

R (kΩ)	C (pF)	CKI Freq. (MHz)	Instr. Cycle (μs)	Conditions
3.3	82	2.2 to 2.7	3.7 to 4.6	V_{CC} = 5V
5.6	100	1.1 to 1.3	7.4 to 9.0	V_{CC} = 5V
6.8	100	0.9 to 1.1	8.8 to 10.8	V_{CC} = 5V

Note: 3k ≤ R ≤ 200k, 50 pF ≤ C ≤ 200 pF

Functional Description (Continued)

The device has three mask options for configuring the clock input. The CKI and CKO pins are automatically configured upon selecting a particular option.

— Crystal (CKI/10) CKO for crystal configuration

— External (CKI/10) CKO available as G7 input

— R/C (CKI/10) CKO available as G7 input

G7 can be used either as a general purpose input or as a control input to continue from the HALT mode.

HALT MODE

The device supports a power saving mode of operation: HALT. The controller is placed in the HALT mode by setting the G7 data bit, alternatively the user can stop the clock input. In the HALT mode all internal processor activities including the clock oscillator are stopped. The fully static architecture freezes the state of the controller and retains all information until continuing. In the HALT mode, power requirements are minimal as it draws only leakage currents and output current. The applied voltage (V_{CC}) may be decreased down to Vr (minimum RAM retention voltage) without altering the state of the machine.

There are two ways to exit the HALT mode: via the $\overline{\text{RESET}}$ or by the CKO pin. A low on the $\overline{\text{RESET}}$ line reinitializes the microcontroller and starts executing from the address 0000H. A low to high transition on the CKO pin (only if the external or the R/C clock option is selected) causes the microcontroller to continue with no reinitialization from the address following the HALT instruction. This also resets the G7 data bit.

INTERRUPTS

There are three interrupt sources, as shown below.

A maskable interrupt on external G0 input (positive or negative edge sensitive under software control)

A maskable interrupt on timer underflow or timer capture

A non-maskable software/error interrupt on opcode zero

INTERRUPT CONTROL

The GIE (global interrupt enable) bit enables the interrupt function. This is used in conjunction with ENI and ENTI to select one or both of the interrupt sources. This bit is reset when interrupt is acknowledged.

ENI and ENTI bits select external and timer interrupt respectively. Thus the user can select either or both sources to interrupt the microcontroller when GIE is enabled.

IEDG selects the external interrupt edge (0 = rising edge, 1 = falling edge). The user can get an interrupt on both rising and falling edges by toggling the state of IEDG bit after each interrupt.

IPND and TPND bits signal which interrupt is pending. After interrupt is acknowledged, the user can check these two bits to determine which interrupt is pending. This permits the interrupts to be prioritized under software. The pending flags have to be cleared by the user. Setting the GIE bit high inside the interrupt subroutine allows nested interrupts.

The software interrupt does not reset the GIE bit. This means that the controller can be interrupted by other interrupt sources while servicing the software interrupt.

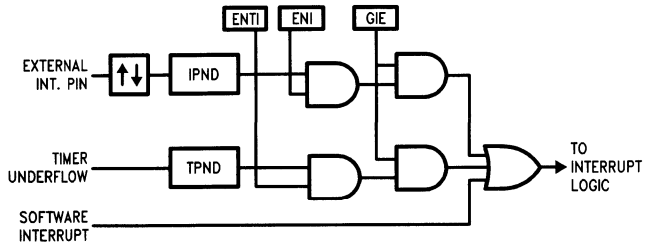
INTERRUPT PROCESSING

The interrupt, once acknowledged, pushes the program counter (PC) onto the stack and the stack pointer (SP) is decremented twice. The Global Interrupt Enable (GIE) bit is reset to disable further interrupts. The microcontroller then vectors to the address 00FFH and resumes execution from that address. This process takes 7 cycles to complete. At the end of the interrupt subroutine, any of the following three instructions return the processor back to the main program: RET, RETSK or RETI. Either one of the three instructions will pop the stack into the program counter (PC). The stack pointer is then incremented twice. The RETI instruction additionally sets the GIE bit to re-enable further interrupts.

Any of the three instructions can be used to return from a hardware interrupt subroutine. The RETSK instruction should be used when returning from a software interrupt subroutine to avoid entering an infinite loop.

Note: There is always the possibility of an interrupt occurring during an instruction which is attempting to reset the GIE bit or any other interrupt enable bit. If this occurs when a single cycle instruction is being used to reset the interrupt enable bit, the interrupt enable bit will be reset but an interrupt may still occur. This is because interrupt processing is started at the same time as the interrupt bit is being reset. To avoid this scenario, the user should always use a two, three, or four cycle instruction to reset interrupt enable bits.

Functional Description (Continued)



TL/DD/9103–11

FIGURE 6. Interrupt Block Diagram

DETECTION OF ILLEGAL CONDITIONS

The device contains a hardware mechanism that allows it to detect illegal conditions which may occur from coding errors, noise and 'brown out' voltage drop situations. Specifically it detects cases of executing out of undefined ROM area and unbalanced stack situations.

Reading an undefined ROM location returns 00 (hexadecimal) as its contents. The opcode for a software interrupt is also '00'. Thus a program accessing undefined ROM will cause a software interrupt.

Reading an undefined RAM location returns an FF (hexadecimal). The subroutine stack grows down for each subroutine call. By initializing the stack pointer to the top of RAM, the first unbalanced return instruction will cause the stack pointer to address undefined RAM. As a result the program will attempt to execute from FFFF (hexadecimal), which is an undefined ROM location and will trigger a software interrupt.

MICROWIRE/PLUS™

MICROWIRE/PLUS is a serial synchronous bidirectional communications interface. The MICROWIRE/PLUS capability enables the device to interface with any of National Semiconductor's MICROWIRE peripherals (i.e. A/D converters, display drivers, EEPROMS, etc.) and with other microcontrollers which support the MICROWIRE/PLUS interface. It consists of an 8-bit serial shift register (SIO) with serial data input (SI), serial data output (SO) and serial shift clock (SK). *Figure 7* shows the block diagram of the MICROWIRE/PLUS interface.

The shift clock can be selected from either an internal source or an external source. Operating the MICROWIRE/PLUS interface with the internal clock source is called the Master mode of operation. Similarly, operating the MICROWIRE/PLUS interface with an external shift clock is called the Slave mode of operation.

The CNTRL register is used to configure and control the MICROWIRE/PLUS mode. To use the MICROWIRE/PLUS, the MSEL bit in the CNTRL register is set to one. The SK clock rate is selected by the two bits, SL0 and SL1, in the CNTRL register. Table III details the different clock rates that may be selected.

TABLE III

SL1	SL0	SK Cycle Time
0	0	$2t_C$
0	1	$4t_C$
1	x	$8t_C$

where,

t_C is the instruction cycle clock.

MICROWIRE/PLUS OPERATION

Setting the BUSY bit in the PSW register causes the MICROWIRE/PLUS arrangement to start shifting the data. It gets reset when eight data bits have been shifted. The user may reset the BUSY bit by software to allow less than 8 bits to shift. The device may enter the MICROWIRE/PLUS mode either as a Master or as a Slave. *Figure 8* shows how two microcontrollers and several peripherals may be interconnected using the MICROWIRE/PLUS arrangement.

Master MICROWIRE/PLUS Operation

In the MICROWIRE/PLUS Master mode of operation the shift clock (SK) is generated internally. The MICROWIRE/PLUS Master always initiates all data exchanges. (See *Figure 8*). The MSEL bit in the CNTRL register must be set to enable the SO and SK functions onto the G Port. The SO and SK pins must also be selected as outputs by setting appropriate bits in the Port G configuration register. Table IV summarizes the bit settings required for Master mode of operation.

SLAVE MICROWIRE/PLUS OPERATION

In the MICROWIRE/PLUS Slave mode of operation the SK clock is generated by an external source. Setting the MSEL bit in the CNTRL register enables the SO and SK functions onto the G Port. The SK pin must be selected as an input and the SO pin is selected as an output pin by appropriately setting up the Port G configuration register. Table IV summarizes the settings required to enter the Slave mode of operation.

The user must set the BUSY flag immediately upon entering the Slave mode. This will ensure that all data bits sent by the Master will be shifted properly. After eight clock pulses the BUSY flag will be cleared and the sequence may be repeated. (See *Figure 8*.)

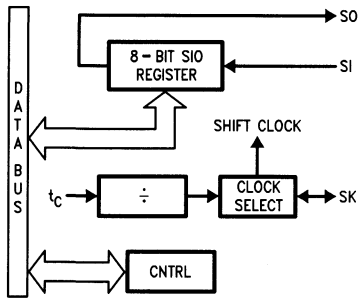
Functional Description (Continued)

TABLE IV

G4 Config. Bit	G5 Config. Bit	G4 Fun.	G5 Fun.	G6 Fun.	Operation
1	1	SO	Int. SK	SI	MICROWIRE Master
0	1	TRI-STATE	Int. SK	SI	MICROWIRE Master
1	0	SO	Ext. SK	SI	MICROWIRE Slave
0	0	TRI-STATE	Ext. SK	SI	MICROWIRE Slave

TIMER/COUNTER

The device has a powerful 16-bit timer with an associated 16-bit register enabling them to perform extensive timer functions. The timer T1 and its register R1 are each organized as two 8-bit read/write registers. Control bits in the register CNTRL allow the timer to be started and stopped under software control. The timer-register pair can be operated in one of three possible modes. Table V details various timer operating modes and their requisite control settings.



TL/DD/9103–12

FIGURE 7. MICROWIRE/PLUS Block Diagram

MODE 1. TIMER WITH AUTO-LOAD REGISTER

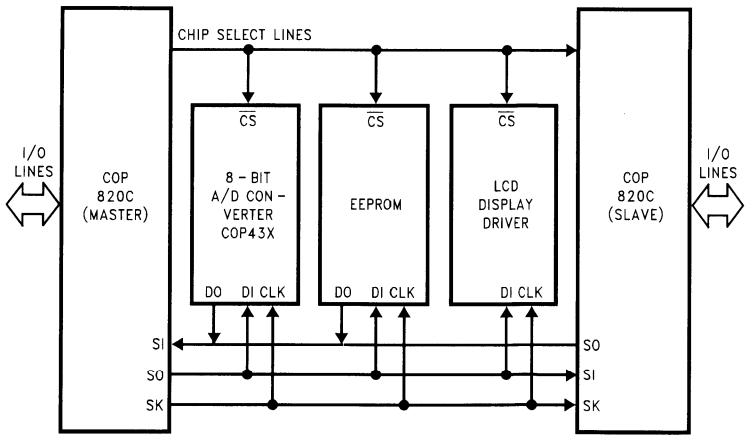
In this mode of operation, the timer T1 counts down at the instruction cycle rate. Upon underflow the value in the register R1 gets automatically reloaded into the timer which continues to count down. The timer underflow can be programmed to interrupt the microcontroller. A bit in the control register CNTRL enables the TIO (G3) pin to toggle upon timer underflows. This allow the generation of square-wave outputs or pulse width modulated outputs under software control. (See *Figure 9*)

MODE 2. EXTERNAL COUNTER

In this mode, the timer T1 becomes a 16-bit external event counter. The counter counts down upon an edge on the TIO pin. Control bits in the register CNTRL program the counter to decrement either on a positive edge or on a negative edge. Upon underflow the contents of the register R1 are automatically copied into the counter. The underflow can also be programmed to generate an interrupt. (See *Figure 9*)

MODE 3. TIMER WITH CAPTURE REGISTER

Timer T1 can be used to precisely measure external frequencies or events in this mode of operation. The timer T1 counts down at the instruction cycle rate. Upon the occurrence of a specified edge on the TIO pin the contents of the timer T1 are copied into the register R1. Bits in the control register CNTRL allow the trigger edge to be specified either as a positive edge or as a negative edge. In this mode the user can elect to be interrupted on the specified trigger edge. (See *Figure 10.*)



TL/DD/9103–13

FIGURE 8. MICROWIRE/PLUS Application

Functional Description (Continued)

TABLE V. Timer Operating Modes

CNTRL Bits 7 6 5	Operation Mode	T Interrupt	Timer Counts On
0 0 0	External Counter W/Auto-Load Reg.	Timer Underflow	TIO Pos. Edge
0 0 1	External Counter W/Auto-Load Reg.	Timer Underflow	TIO Neg. Edge
0 1 0	Not Allowed	Not Allowed	Not Allowed
0 1 1	Not Allowed	Not Allowed	Not Allowed
1 0 0	Timer W/Auto-Load Reg.	Timer Underflow	t_C
1 0 1	Timer W/Auto-Load Reg./Toggle TIO Out	Timer Underflow	t_C
1 1 0	Timer W/Capture Register	TIO Pos. Edge	t_C
1 1 1	Timer W/Capture Register	TIO Neg. Edge	t_C

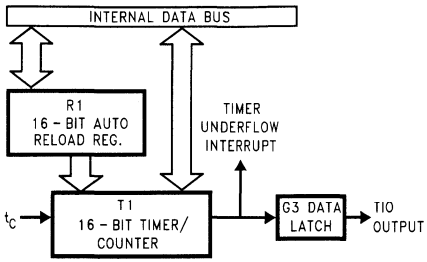


FIGURE 9. Timer/Counter Auto Reload Mode Block Diagram

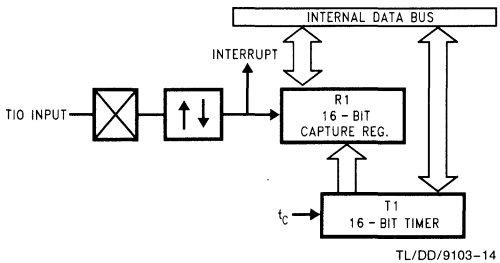


FIGURE 10. Timer Capture Mode Block Diagram

TIMER PWM APPLICATION

Figure 11 shows how a minimal component D/A converter can be built out of the Timer-Register pair in the Auto-Reload mode. The timer is placed in the "Timer with auto reload" mode and the TIO pin is selected as the timer output. At the outset the TIO pin is set high, the timer T1 holds the on time and the register R1 holds the signal off time. Setting TRUN bit starts the timer which counts down at the instruction cycle rate. The underflow toggles the TIO output and copies the off time into the timer, which continues to run. By alternately loading in the on time and the off time at each successive interrupt a PWM frequency can be easily generated.

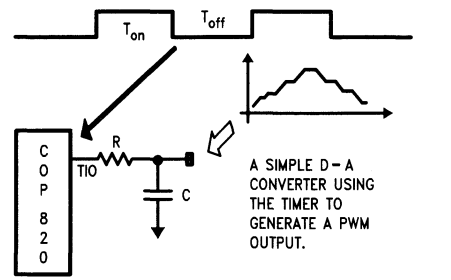


FIGURE 11. Timer Application

Control Registers

CNTRL REGISTER (ADDRESS X'00EE)

The Timer and MICROWIRE/PLUS control register contains the following bits:

SL1 & SL0 Select the MICROWIRE/PLUS clock divide-by

IEDG External interrupt edge polarity select (0 = rising edge, 1 = falling edge)

MSEL Enable MICROWIRE/PLUS functions SO and SK

TRUN Start/Stop the Timer/Counter (1 = run, 0 = stop)

TC3 Timer input edge polarity select (0 = rising edge, 1 = falling edge)

TC2 Selects the capture mode

TC1 Selects the timer mode

TC1	TC2	TC3	TRUN	MSEL	IEDG	SL1	SL0

BIT 7 BIT 0

PSW REGISTER (ADDRESS X'00EF)

The PSW register contains the following select bits:

GIE Global interrupt enable

ENI External interrupt enable

BUSY MICROWIRE/PLUS busy shifting

IPND External interrupt pending

ENTI Timer interrupt enable

TPND Timer interrupt pending

C Carry Flag

HC Half carry Flag

HC	C	TPND	ENTI	IPND	BUSY	ENI	GIE

Bit 7 Bit 0

Addressing Modes

REGISTER INDIRECT

This is the "normal" mode of addressing. The operand is the memory addressed by the B register or X register.

DIRECT

The instruction contains an 8-bit address field that directly points to the data memory for the operand.

IMMEDIATE

The instruction contains an 8-bit immediate field as the operand.

REGISTER INDIRECT (AUTO INCREMENT AND DECREMENT)

This is a register indirect mode that automatically increments or decrements the B or X register after executing the instruction.

RELATIVE

This mode is used for the JP instruction, the instruction field is added to the program counter to get the new program location. JP has a range of from −31 to +32 to allow a one byte relative jump (JP + 1 is implemented by a NOP instruction). There are no 'pages' when using JP, all 15 bits of PC are used.

Memory Map

All RAM, ports and registers (except A and PC) are mapped into data memory address space.

Address	Contents
COP820C Family	
00 to 2F	On Chip RAM Bytes
30 to 7F	Unused RAM Address Space (Reads as all Ones)
COP840C Family	
00 to 6F	On Chip RAM Bytes
70 to 7F	Unused RAM Address Space (Reads as all Ones)
COP820C and COP840C Families	
80 to BF	Expansion Space for on Chip EERAM
C0 to CF	Expansion Space for I/O and Registers
D0 to DF	On Chip I/O and Registers
D0	Port L Data Register
D1	Port L Configuration Register
D2	Port L Input Pins (Read Only)
D3	Reserved for Port L
D4	Port G Data Register
D5	Port G Configuration Register
D6	Port G Input Pins (Read Only)
D7	Port I Input Pins (Read Only)
D8–DB	Reserved for Port C
DC	Port D Data Register
DD–DF	Reserved for Port D
E0 to EF	On Chip Functions and Registers
E0–E7	Reserved for Future Parts
E8	Reserved
E9	MICROWIRE/PLUS Shift Register
EA	Timer Lower Byte
EB	Timer Upper Byte
EC	Timer Autoload Register Lower Byte
ED	Timer Autoload Register Upper Byte
EE	CNTRL Control Register
EF	PSW Register
F0 to FF	On Chip RAM Mapped as Registers
FC	X Register
FD	SP Register
FE	B Register

Reading unused memory locations below 7FH will return all ones. Reading other unused memory locations will return undefined data.

Instruction Set

REGISTER AND SYMBOL DEFINITIONS

Registers

A	8-bit Accumulator register
B	8-bit Address register
X	8-bit Address register
SP	8-bit Stack pointer register
PC	15-bit Program counter register
PU	upper 7 bits of PC
PL	lower 8 bits of PC
C	1-bit of PSW register for carry
HC	Half Carry
GIE	1-bit of PSW register for global interrupt enable

Symbols

[B]	Memory indirectly addressed by B register
[X]	Memory indirectly addressed by X register
Mem	Direct address memory or [B]
MemI	Direct address memory or [B] or Immediate data
Imm	8-bit Immediate data
Reg	Register memory: addresses F0 to FF (Includes B, X and SP)
Bit	Bit number (0 to 7)
←	Loaded with
⟷	Exchanged with

Instruction Set

ADD	add	A ← A + MemI
ADC	add with carry	A ← A + MemI + C, C ← Carry HC ← Half Carry
SUBC	subtract with carry	A ← A + $\overline{\text{MemI}}$ + C, C ← Carry HC ← Half Carry
AND	Logical AND	A ← A and MemI
OR	Logical OR	A ← A or MemI
XOR	Logical Exclusive-OR	A ← A xor MemI
IFEQ	IF equal	Compare A and MemI, Do next if A = MemI
IFGT	IF greater than	Compare A and MemI, Do next if A > MemI
IFBNE	IF B not equal	Do next if lower 4 bits of B ≠ Imm
DRSZ	Decrement Reg. ,skip if zero	Reg ← Reg − 1, skip if Reg goes to 0
SBIT	Set bit	1 to bit, Mem (bit = 0 to 7 immediate)
RBIT	Reset bit	0 to bit, Mem
IFBIT	If bit	If bit, Mem is true, do next instr.
X	Exchange A with memory	A ⟷ Mem
LD A	Load A with memory	A ← MemI
LD mem	Load Direct memory Immed.	Mem ← Imm
LD Reg	Load Register memory Immed.	Reg ← Imm
X	Exchange A with memory [B]	A ⟷ [B] (B ← B ± 1)
X	Exchange A with memory [X]	A ⟷ [X] (X ← X ± 1)
LD A	Load A with memory [B]	A ← [B] (B ← B ± 1)
LD A	Load A with memory [X]	A ← [X] (X ← X ± 1)
LD M	Load Memory Immediate	[B] ← Imm (B ← B ± 1)
CLRA	Clear A	A ← 0
INCA	Increment A	A ← A + 1
DECA	Decrement A	A ← A − 1
LAID	Load A indirect from ROM	A ← ROM(PU,A)
DCORA	DECIMAL CORRECT A	A ← BCD correction (follows ADC, SUBC)
RRCA	ROTATE A RIGHT THRU C	C → A7 → . . . → A0 → C
SWAPA	Swap nibbles of A	A7 . . . A4 ⟷ A3 . . . A0
SC	Set C	C ← 1, HC ← 1
RC	Reset C	C ← 0, HC ← 0
IFC	If C	If C is true, do next instruction
IFNC	If not C	If C is not true, do next instruction
JMPL	Jump absolute long	PC ← ii (ii = 15 bits. 0 to 32k)
JMP	Jump absolute	PC11..0 ← i (i = 12 bits)
JP	Jump relative short	PC ← PC + r (r is −31 to +32, not 1)
JSRL	Jump subroutine long	[SP] ← PL,[SP-1] ← PU,SP-2,PC ← ii
JSR	Jump subroutine	[SP] ← PL,[SP-1] ← PU,SP-2,PC11.. 0 ← i
JID	Jump indirect	PL ← ROM(PU,A)
RET	Return from subroutine	SP + 2,PL ← [SP],PU ← [SP-1]
RETSK	Return and Skip	SP + 2,PL ← [SP],PU ← [SP-1],Skip next instruction
RETI	Return from Interrupt	SP + 2,PL ← [SP],PU ← [SP-1],GIE ← 1
INTR	Generate an interrupt	[SP] ← PL,[SP − 1] ← PU,SP-2,PC ← 0FF
NOP	No operation	PC ← PC + 1

Bits 7–4

F	E	D	C	B	A	9	8	7	6	5	4	3	2	1	0	
JP -15	JP -31	LD 0F0,#i	DRSZ 0F0	RRCA	RC	ADC A, #i	ADC A, [B]	IFBIT 0,[B]	*	LD B, 0F	IFBNE 0	JSR 0000-00FF	JMP 0000-00FF	JP + 17	INTR	0
JP -14	JP -30	LD 0F1,#i	DRSZ 0F1	*	SC	SUBC A, #i	SUBC A,[B]	IFBIT 1,[B]	*	LD B, 0E	IFBNE 1	JSR 0100-01FF	JMP 0100-01FF	JP + 18	JP + 2	1
JP -13	JP -29	LD 0F2,#i	DRSZ 0F2	X A, [X+]	X A, [B+]	IFEQ A, #i	IFEQ A,[B]	IFBIT 2,[B]	*	LD B, 0D	IFBNE 2	JSR 0200-02FF	JMP 0200-02FF	JP + 19	JP + 3	2
JP -12	JP -28	LD 0F3,#i	DRSZ 0F3	X A, [X−]	X A, [B−]	IFGT A, #i	IFGT A,[B]	IFBIT 3,[B]	*	LD B, 0C	IFBNE 3	JSR 0300-03FF	JMP 0300-03FF	JP + 20	JP + 4	3
JP -11	JP -27	LD 0F4,#i	DRSZ 0F4	*	LAID	ADD A, #i	ADD A,[B]	IFBIT 4,[B]	CLRA	LD B, 0B	IFBNE 4	JSR 0400-04FF	JMP 0400-04FF	JP + 21	JP + 5	4
JP -10	JP -26	LD 0F5,#i	DRSZ 0F5	*	JID	AND A, #i	AND A,[B]	IFBIT 5,[B]	SWAPA	LD B, 0A	IFBNE 5	JSR 0500-05FF	JMP 0500-05FF	JP + 22	JP + 6	5
JP -9	JP -25	LD 0F6,#i	DRSZ 0F6	X A, [X]	X A, [B]	XOR A, #i	XOR A,[B]	IFBIT 6,[B]	DCORA	LD B, 9	IFBNE 6	JSR 0600-06FF	JMP 0600-06FF	JP + 23	JP + 7	6
JP -8	JP -24	LD 0F7,#i	DRSZ 0F7	*	*	OR A, #i	OR A,[B]	IFBIT 7,[B]	*	LD B, 8	IFBNE 7	JSR 0700-07FF	JMP 0700-07FF	JP + 24	JP + 8	7
JP -7	JP -23	LD 0F8,#i	DRSZ 0F8	NOP	*	LD A, #i	IFC	SBIT 0,[B]	RBIT 0,[B]	LD B, 7	IFBNE 8	JSR 0800-08FF	JMP 0800-08FF	JP + 25	JP + 9	8
JP -6	JP -22	LD 0F9,#i	DRSZ 0F9	*	*	*	IFNC	SBIT 1,[B]	RBIT 1,[B]	LD B, 6	IFBNE 9	JSR 0900-09FF	JMP 0900-09FF	JP + 26	JP + 10	9
JP -5	JP -21	LD 0FA,#i	DRSZ 0FA	LD A, [X+]	LD A, [B+]	LD [B+],#i	INCA	SBIT 2,[B]	RBIT 2,[B]	LD B, 5	IFBNE 0A	JSR 0A00-0AFF	JMP 0A00-0AFF	JP + 27	JP + 11	A
JP -4	JP -20	LD 0FB,#i	DRSZ 0FB	LD A, [X−]	LD A, [B−]	LD [B−],#i	DECA	SBIT 3,[B]	RBIT 3,[B]	LD B, 4	IFBNE 0B	JSR 0B00-0BFF	JMP 0B00-0BFF	JP + 28	JP + 12	B
JP -3	JP -19	LD 0FC,#i	DRSZ 0FC	LD Md, #i	JMPL	X A,Md	*	SBIT 4,[B]	RBIT 4,[B]	LD B, 3	IFBNE 0C	JSR 0C00-0CFF	JMP 0C00-0CFF	JP + 29	JP +13	C
JP -2	JP -18	LD 0FD,#i	DRSZ 0FD	DIR	JSRL	LD A, Md	RETSK	SBIT 5,[B]	RBIT 5,[B]	LD B, 2	IFBNE 0D	JSR 0D00-0DFF	JMP 0D00-0DFF	JP + 30	JP +14	D
JP -1	JP -17	LD 0FE,#i	DRSZ 0FE	LD A, [X]	LD A, [B]	LD [B], #i	RET	SBIT 6, [B]	RBIT 6, [B]	LD B, 1	IFBNE 0E	JSR 0E00−0EFF	JMP 0E00−0EFF	JP + 31	JP +15	E
JP -0	JP -16	LD 0FF,#1	DRSZ 0FF	*	*	*	RETI	SBIT 7,[B]	RBIT 7,[B]	LD B, 0	IFBNE 0F	JSR 0F00-0FFF	JMP 0F00-0FFF	JP + 32	JP + 16	F

Bits 3–0

where, i is the immediate data Md is a directly addressed memory location * is an unused opcode (see following table)

Instruction Execution Time

Most instructions are single byte (with immediate addressing mode instruction taking two bytes).

Most single instructions take one cycle time to execute.

Skipped instructions require x number of cycles to be skipped, where x equals the number of bytes in the skipped instruction opcode.

See the BYTES and CYCLES per INSTRUCTION table for details.

Bytes and Cycles per Instruction

The following table shows the number of bytes and cycles for each instruction in the format of byte/cycle.

Arithmetic and Logic Instructions

	[B]	Direct	Immed.
ADD	1/1	3/4	2/2
ADC	1/1	3/4	2/2
SUBC	1/1	3/4	2/2
AND	1/1	3/4	2/2
OR	1/1	3/4	2/2
XOR	1/1	3/4	2/2
IFEQ	1/1	3/4	2/2
IFGT	1/1	3/4	2/2
IFBNE	1/1		
DRSZ		1/3	
SBIT	1/1	3/4	
RBIT	1/1	3/4	
IFBIT	1/1	3/4	

The following table shows the instructions assigned to unused opcodes. This table is for information only. The operations performed are subject to change without notice. Do not use these opcodes.

Unused Opcode	Instruction	Unused Opcode	Instruction
60	NOP	A9	NOP
61	NOP	AF	LD A, [B]
62	NOP	B1	C → HC
63	NOP	B4	NOP
67	NOP	B5	NOP
8C	RET	B7	X A, [X]
99	NOP	B9	NOP
9F	LD [B], #i	BF	LD A, [X]
A7	X A, [B]		
A8	NOP		

Memory Transfer Instructions

	Register Indirect [B]	Register Indirect [X]	Direct	Immed.	Register Indirect Auto Incr & Decr [B+, B−]	Register Indirect Auto Incr & Decr [X+, X−]	
X A,*	1/1	1/3	2/3		1/2	1/3	
LD A,*	1/1	1/3	2/3	2/2	1/2	1/3	
LD B,Imm				1/1			(If B < 16)
LD B,Imm				2/3			(If B > 15)
LD Mem,Imm	2/2		3/3		2/2		
LD Reg,Imm				2/3			

* => Memory location addressed by B or X or directly.

Instructions Using A & C

CLRA	1/1
INCA	1/1
DECA	1/1
LAID	1/3
DCORA	1/1
RRCA	1/1
SWAPA	1/1
SC	1/1
RC	1/1
IFC	1/1
IFNC	1/1

Transfer of Control Instructions

JMPL	3/4
JMP	2/3
JP	1/3
JSRL	3/5
JSR	2/5
JID	1/3
RET	1/5
RETSK	1/5
RETI	1/5
INTR	1/7
NOP	1/1

Option List

The mask programmable options are listed out below. The options are programmed at the same time as the ROM pattern to provide the user with hardware flexibility to use a variety of oscillator configuration.

OPTION 1: CKI INPUT

= 1 Crystal (CKI/10) CKO for crystal configuration
= 2 External (CKI/10) CKO available as G7 input
= 3 R/C (CKI/10) CKO available as G7 input

OPTION 2: BONDING

= 1 28-pin DIP package
= 2 N.A.
= 3 20-pin DIP package
= 4 20-SO package
= 5 28-SO package

The following option information is to be sent to National along with the EPROM.

Option Data

Option 1 Value_is: CKI Input
Option 2 Value_is: COP Bonding

Development Support

SUMMARY

- iceMASTER™: IM-COP8/400—Full feature in-circuit emulation for all COP8 products. A full set of COP8 Basic and Feature Family device and package specific probes are available.
- COP8 Debug Module: Moderate cost in-circuit emulation and development programming unit.
- COP8 Evaluation and Programming Unit: EPU-COP888GG—low cost In-circuit simulation and development programming unit.
- Assembler: COP8-DEV-IBMA. A DOS installable cross development Assembler, Linker, Librarian and Utility Software Development Tool Kit.
- C Compiler: COP8C. A DOS installable cross development Software Tool Kit.
- OTP/EPROM Programmer Support: Covering needs from engineering prototype, pilot production to full production environments.

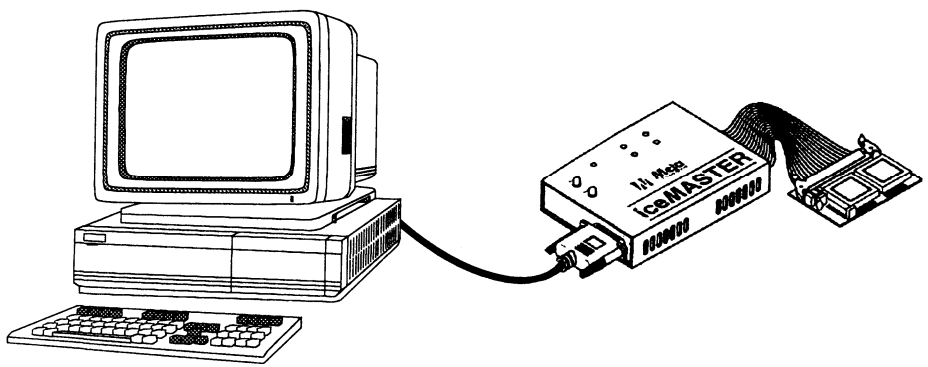
iceMASTER (IM) IN-CIRCUIT EMULATION

The iceMASTER IM-COP8/400 is a full feature, PC based, in-circuit emulation tool developed and marketed by MetaLink Corporation to support the whole COP8 family of products. National is a resale vendor for these products.

See *Figure 19* for configuration.

The iceMASTER IM-COP8/400 with its device specific COP8 Probe provides a rich feature set for developing, testing and maintaining product:

- Real-time in-circuit emulation; full 2.4V–5.5V operation range, full DC-10 MHz clock. Chip options are programmable or jumper selectable.
- Direct connection to application board by package compatible socket or surface mount assembly.
- Full 32K byte of loadable programming space that overlays (replaces) the on-chip ROM or EPROM. On-chip RAM and I/O blocks are used directly or recreated on the probe as necessary.
- Full 4K frame synchronous trace memory. Address, instruction, and 8 unspecified, circuit connectable trace lines. Display can be HLL source (e.g., C source), assembly or mixed.
- A full 64K hardware configurable break, trace on, trace off control, and pass count increment events.
- Tool set integrated interactive symbolic debugger—supports both assembler (COFF) and C Compiler (.COD) linked object formats.
- Real time performance profiling analysis; selectable bucket definition.
- Watch windows, content updated automatically at each execution break.
- Instruction by instruction memory/register changes displayed on source window when in single step operation.
- Single base unit and debugger software reconfigurable to support the entire COP8 family; only the probe personality needs to change. Debugger software is processor customized, and reconfigured from a master model file.
- Processor specific symbolic display of registers and bit level assignments, configured from master model file.
- Halt/Idle mode notification.



TL/DD/9103–27

FIGURE 19. COP8 iceMASTER Environment

Development Support (Continued)

- On-line HELP customized to specific processor using master model file.
- Includes a copy of COP8-DEV-IBMA assembler and linker SDK.

IM Order Information

Base Unit	
IM-COP8/400-1	iceMASTER base unit, 100V power supply
IM-COP8/400-2	iceMASTER base unit, 220V power supply
iceMASTER Probe	
MHW-880C20DWPC	20 DIP
MHW-880C28DWPC	28 DIP
Adapters for SO Packages	
MHW-SOIC 20	20 SO
MHW-SOIC 28	28 SO

iceMASTER DEBUG MODULE (DM)

The iceMASTER Debug Module is a PC based, combination in-circuit emulation tool and COP8based OTP/EPROM programming tool developed and marketed by MetaLink Corporation to support the whole COP8 family of products. National is a resale vendor for these products.

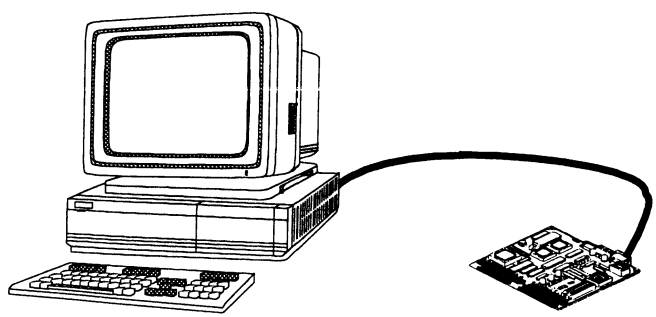
See *Figure 20* for configuration.

The iceMASTER Debug Module is a moderate cost development tool. It has the capability of in-circuit emulation for a specific COP8 microcontroller and in addition serves as a programming tool for COP8 OTP and EPROM product families. Summary of features is as follows:

- Real-time in-circuit emulation; full operating voltage range operation, full DC-10 MHz clock.
- All processor I/O pins can be cabled to an application development board with package compatible cable to socket and surface mount assembly.
- Full 32K byte of loadable programming space that overlays (replaces) the on-chip ROM or EPROM. On-chip RAM and I/O blocks are used directly or recreated as necessary.
- 100 frames of synchronous trace memory. The display can be HLL source (C source), assembly or mixed. The most recent history prior to a break is available in the trace memory.
- Configured break points; uses INTR instruction which is modestly intrusive.
- Software—only supported features are selectable.
- Tool set integrated interactive symbolic debugger—supports both assembler (COFF) and C Compiler (.COD) SDK linked object formats.
- Instruction by instruction memory/register changes displayed when in single step operation.
- Debugger software is processor customized, and reconfigured from a master model file.
- Processor specific symbolic display of registers and bit level assignments, configured from master model file.
- Halt/Idle mode notification.
- Programming menu supports full product line of programmable OTP and EPROM COP8 products. Program data is taken directly from the overlay RAM.
- Programming of 44 PLCC and 68 PLCC parts requires external programming adapters.
- Includes wallmount power supply.
- On-board V_{PP} generator from 5V input or connection to external supply supported. Requires V_{PP} level adjustment per the family programming specification (correct level is provided on an on-screen pop-down display).
- On-line HELP customized to specific processor using master model file.
- Includes a copy of COP8-DEV-IBMA assembler and linker SDK.

DM Order Information

Debug Model Unit	
COP8-DM/880C	
Cable Adapters	
DM-COP8/20D	20 DIP
DM-COP8/28D	28 DIP
Adapter for SO Packages	
MHW-SOIC 20	20 SO
MHW-SOIC 28	28 SO



TL/DD/9103-28

FIGURE 20. COP8-DM Environment

Development Support (Continued)

COP8 ASSEMBLER/LINKER SOFTWARE DEVELOPMENT TOOL KIT

National Semiconductor offers a relocateable COP8 macro cross assembler, linker, librarian and utility software development tool kit. Features are summarized as follows:

- Basic and Feature Family instruction set by "device" type.
- Nested macro capability.
- Extensive set of assembler directives.
- Supported on PC/DOS platform.
- Generates National standard COFF output files.
- Integrated Linker and Librarian.
- Integrated utilities to generate ROM code file outputs.
- DUMPCOFF utility.

This product is integrated as a part of MetaLink tools as a development kit, fully supported by the MetaLink debugger. It may be ordered separately or it is bundled with the MetaLink products at no additional cost.

Order Information

Assembler SDK:	
COP8-DEV-IBMA	Assembler SDK on installable 3.5" PC/DOS Floppy Disk Drive format. Periodic upgrades and most recent version is available on National's BBS and Internet.

COP8 C COMPILER

A C Compiler is developed and marketed by Byte Craft Limited. The COP8C compiler is a fully integrated development tool specifically designed to support the compact embedded configuration of the COP8 family of products.

Features are summarized as follows:

- ANSI C with some restrictions and extensions that optimize development for the COP8 embedded application.
- BITS data type extension. Register declaration #pragma with direct bit level definitions.
- C language support for interrupt routines.
- Expert system, rule based code geration and optimization.
- Performs consistency checks against the architectural definitions of the target COP8 device.
- Generates program memory code.
- Supports linking of compilied object or COP8 assembled object formats.
- Global optimization of linked code.
- Symbolic debug load format fully sourced level supported by the MetaLink debugger.

SINGLE CHIP OTP/EMULATOR SUPPORT

The COP8 family is supported by single chip OTP emulators. For detailed information refer to the emulator specific datasheet and the emulator selection table below:

OTP Emulator Ordering Information

Device Number	Clock Option	Package	Description	Emulates
COP8781CN	Programmable	28 DIP	One Time Programmable (OTP)	COP840C, COP820C
COP8781CWM	Programmable	28 SO	OTP	COP840C, COP820C
COP8782CN	Programmable	20 DIP	OTP	COP842C, COP822C
COP8782CWM	Programmable	20 SO	OTP	COP842C, COP822C

Development Support (Continued)

Approved List

Manufacturer	North America	Europe	Asia
BP Microsystems	(800) 225-2102 (713) 688-4600 Fax: (713) 688-0920	+49-8152-4183 +49-8856-932616	+852-234-16611 +852-2710-8121
Data I/O	(800) 426-1045 (206) 881-6444 Fax: (206) 882-1043	+44-0734-440011	Call North America
HI-LO	(510) 623-8860	Call Asia	+886-02-764-0215 Fax: +886-2-756-6403
ICE Technology	(800) 624-8949 (919) 430-7915	+44-1226-767404 Fax: 0-1226-370-434	
MetaLink	(800) 638-2423 (602) 926-0797 Fax: (602) 693-0681	+49-80 9156 96-0 Fax: +49-80 9123 86	+852-737-1800
Systems General	(408) 263-6667	+41-1-9450300	+886-2-917-3005 Fax: +886-2-911-1283
Needhams	(916) 924-8037 Fax: (916) 924-8065		

INDUSTRY WIDE OTP/EPROM PROGRAMMING SUPPORT

Programming support, in addition to the MetaLink development tools, is provided by a full range of independent approved vendors to meet the needs from the engineering laboratory to full production.

AVAILABLE LITERATURE

For more information, please see the COP8 Basic Family User's Manual, Literature Number 620895, COP8 Feature Family User's Manual, Literature Number 620897 and National's Family of 8-bit Microcontrollers COP8 Selection Guide, Literature Number 630009.

DIAL-A-HELPER SERVICE

Dial-A-Helper is a service provided by the Microcontroller Applications group. The Dial-A-Helper is an Electronic Information System that may be accessed as a Bulletin Board System (BBS) via data modem, as an FTP site on the Internet via standard FTP client application or as an FTP site on the Internet using a standard Internet browser such as Netscape or Mosaic.

The Dial-A-Helper system provides access to an automated information storage and retrieval system . The system capabilities include a MESSAGE SECTION (electronic mail, when accessed as a BBS) for communications to and from the microcontroller Application Group and a FILE SECTION which consists of several file areas where valuable application software and utilities could be found.

DIAL-A-HELPER BBS via a Standard Modem

Modem: CANADA/U.S.: (800) NSC-MICRO
(800) 672-6427
EUROPE: (+49) 0-8141-351332
Baud: 14.4K
Set-Up: Length: 8-Bit
Parity: None
Stop Bit: 1
Operation: 24 Hours, 7 Days

DIAL-A-HELPER via FTP

ftp nscmicro.nsc.com
user: anonymous
password: username@yourhost.site.domain

DIAL-A-HELPER via WorldWide Web Browswer

ftp://nscmicro.nsc.com

National Semiconductor on the WorldWide Web

See us on the WorldWide Web at http://www.national.com

CUSTOMER RESPONSE CENTER

Complete product information and technical support is available from National's customer response centers.

CANADA/U.S.:	Tel:	(800) 272-9959
	email:	support@tevm2.nsc.com
EUROPE:	email:	europe.support@nsc.com
	Deutsch Tel:	+49 (0) 180-530 85 85
	English Tel:	+49 (0) 180-532 78 32
	Français Tel:	+49 (0) 180-532 93 58
	Italiano Tel:	+49 (0) 180-534 16 80
JAPAN:	Tel:	+81-043-299-2309
S.E. ASIA:	Beijing Tel:	(+86) 10-6856-8601
	Shanghai Tel:	(+86) 21-6415-4092
	Hong Kong Tel:	(+852) 2737-1600
	Korea Tel:	(+82) 2-3771-6909
	Malaysia Tel:	(+60-4) 644-9061
	Singapore Tel:	(+65) 255-2226
	Taiwan Tel:	+886-2-521-3288
AUSTRALIA:	Tel:	(+61) 3-9558-9999
INDIA:	Tel:	(+91) 80-559-9467

National Semiconductor

COP820CJ/COP822CJ/COP823CJ 8-Bit Microcontroller with Multi-Input Wake Up and Brown Out Detector

General Description

The COP820CJ is a member of the COP8™ 8-bit Microcontroller family. It is a fully static Microcontroller, fabricated using double-metal silicon gate microCMOS technology. This low cost Microcontroller is a complete microcomputer containing all system timing, interrupt logic, ROM, RAM, and I/O necessary to implement dedicated control functions in a variety of applications. Features include an 8-bit memory mapped architecture, MICROWIRE™ serial I/O, a 16-bit timer/counter with capture register, a multi-sourced interrupt, Comparator, WATCHDOG™ Timer, Modulator/Timer, Brown out protection and Multi-Input Wakeup. Each I/O pin has software selectable options to adapt the device to the specific application. The device operates over a voltage range of 2.5V to 6.0V. High throughput is achieved with an efficient, regular instruction set operating at a 1 μs per instruction rate.

Key Features

- Multi-Input Wake Up (on the 8-bit Port L)
- Brown out detector
- Analog comparator
- Modulator/timer (High speed PWM for IR transmission)
- 16-bit multi-function timer supporting
  - — PWM mode
  - — External event counter mode
  - — Input capture mode
- 1024 bytes of ROM
- 64 bytes of RAM

I/O Features

- Memory mapped I/O
- Software selectable I/O options (TRI-STATE® output, push-pull output, weak pull-up input, high impedance input)
- High current outputs (8 pins)
- Schmitt trigger inputs on Port G
- MICROWIRE/PLUS™ serial I/O
- Packages
  - — 16 SO with 12 I/O pins
  - — 20 DIP/SO with 16 I/O pins
  - — 28 DIP/SO with 24 I/O pins

CPU/Instruction Set Feature

- 1 μs instruction cycle time
- Three multi-source vectored interrupts servicing
  - — External interrupt with selectable edge
  - — Timer interrupt
  - — Software interrupt
- Versatile and easy to use instruction set
- 8-bit Stack Pointer (SP)—stack in RAM
- Two 8-bit register indirect data memory pointers (B, X)

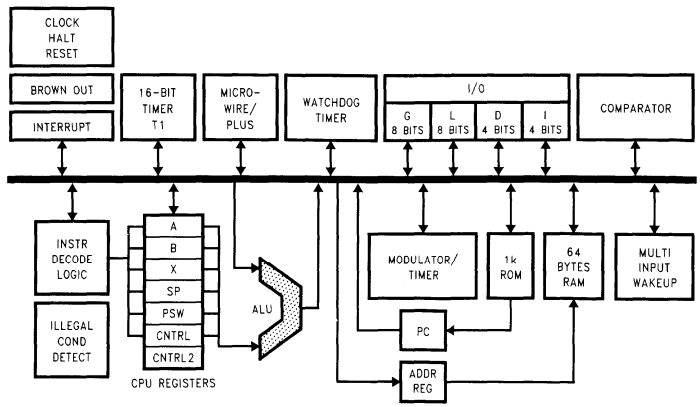
Fully Static CMOS

- Low current drain (typically < 1 μA)
- Single supply operation: 2.5V to 6.0V
- Temperature range: −40°C to +85°C

Development Support

- Emulation and OTP devices
- Real time emulation and full program debug offered by MetaLink Development System

Block Diagram



TL/DD/11208–1

FIGURE 1. Block Diagram

COP820CJ/COP822CJ/COP823CJ

Absolute Maximum Ratings

If Military/Aerospace specified devices are required, please contact the National Semiconductor Sales Office/Distributors for availability and specifications.

Supply Voltage (V_{CC})	7.0V
Voltage at any Pin	−0.3V to V_{CC} + 0.3V
Total Current into V_{CC} pin (Source)	80 mA
Total Current out of GND pin (sink)	80 mA
Storage Temperature Range	−65°C to +150°C

Note: *Absolute maximum ratings indicate limits beyond which damage to the device may occur.*

DC and AC electrical specifications are not ensured when operating the device at absolute maximum ratings.

DC Electrical Characteristics $-40°C \le T_A \le +85°C$ unless otherwise specified

Parameter	Conditions	Min	Typ	Max	Units
Operating Voltage	Brown Out Disabled	2.5		6.0	V
Power Supply Ripple 1 (Note 1)	Peak to Peak			0.1 V_{CC}	V
Supply Current (Note 2)					
CKI = 10 MHz	V_{CC} = 6V, tc = 1 μs			6.0	mA
CKI = 4 MHz	V_{CC} = 6V, tc = 2.5 μs			3.5	mA
CKI = 4 MHz	V_{CC} = 4.0V, tc = 2.5 μs			2.0	mA
CKI = 1 MHz	V_{CC} = 4.0V, tc = 10 μs			1.5	mA
HALT Current with Brown Out Disbled (Note 3)	V_{CC} = 6V, CKI = 0 MHz		<1	10	μA
HALT Current with Brown Out Enabled	V_{CC} = 6V, CKI = 0 MHz		<50	110	μA
Brown Out Trip Level (Brown Out Enabled)		1.8	3.1	4.2	V
INPUT LEVELS (V_{IH}, V_{IL})					
Reset, CKI:					
Logic High		0.8 V_{CC}			V
Logic Low				0.2 V_{CC}	V
All Other Inputs					
Logic High		0.7 V_{CC}			V
Logic Low				0.2 V_{CC}	V
Hi-Z Input Leakage	V_{CC} = 6.0V	−2		+2	μA
Input Pullup Current	V_{CC} = 6.0V, V_{IN} = 0V	−40		−250	μA
L- and G-Port Hysteresis (Note 5)				0.35 V_{CC}	V
Output Current Levels					
D Outputs:					
Source	V_{CC} = 4.5V, V_{OH} = 3.8V	−0.4			mA
	V_{CC} = 2.5V, V_{OH} = 1.8V	−0.2			mA
Sink	V_{CC} = 4.5V, V_{OL} = 1.0V	10			mA
	V_{CC} = 2.5V, V_{OH} = 0.4V	2			mA
L4–L7 Output Sink	V_{CC} = 4.5V, V_{OL} = 2.5V	15			mA
All Others					
Source (Weak Pull-up Mode)	V_{CC} = 4.5V, V_{OH} = 3.2V	−10		−110	μA
	V_{CC} = 2.5V, V_{OH} = 1.8V	−2.5		−33	μA
Source (Push-pull Mode)	V_{CC} = 4.5V, V_{OH} = 3.8V	−0.4			mA
	V_{CC} = 2.5V, V_{OH} = 1.8V	−0.2			mA
Sink (Push-pull Mode)	V_{CC} = 4.5V, V_{OL} = 0.4V	1.6			mA
	V_{CC} = 2.5V, V_{OL} = 0.4V	0.7			mA
TRI-STATE Leakage		−2.0		+2.0	μA
Allowable Sink/Source Current Per Pin					
D Outputs				15	mA
L4–L7 (Sink)				20	mA
All Others				3	mA

DC Electrical Characteristics $-40°C \leq T_A \leq +85°C$ unless otherwise specified (Continued)

Parameter	Conditions	Min	Typ	Max	Units
Maximum Input Current without Latchup (Note 4)	Room Temperature			±100	mA
RAM Retention Voltage, V_r	500 ns Rise and Fall Time (Min)	2.0			V
Input Capacitance				7	pF
Load Capacitance on D2				1000	pF

Note 1: Rate of voltage change must be less than 10 V/mS.

Note 2: Supply current is measured after running 2000 cycles with a square wave CKI input, CKO open, inputs at rails and outputs open.

Note 3: The HALT mode will stop CKI from oscillating in the RC and crystal configurations. HALT test conditions: L, and G0..G5 ports configured as outputs and set high. The D port set to zero. All inputs tied to V_{CC}. The comparator and the Brown Out circuits are disabled.

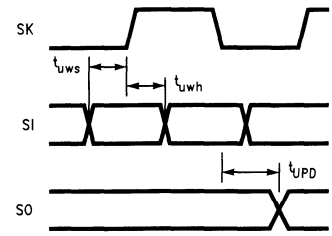
Note 4: Pins G6 and $\overline{\text{RESET}}$ are designed with a high voltage input network. These pins allow input voltages greater than V_{CC} and the pins will have sink current to V_{CC} when biased at voltages greater than V_{CC} (the pins do not have source current when biased at a voltage below V_{CC}). The effective resistance to V_{CC} is 750Ω (typical). These two pins will not latch up. The voltage at the pins must be limited to less than 14V.

AC Electrical Characteristics $-40°C \leq T_A \leq +85°C$ unless otherwise specified

Parameter	Conditions	Min	Typ	Max	Units
Instruction Cycle Time (tc)					
Crystal/Resonator	$4.5V \leq V_{CC} \leq 6.0V$	1		DC	μs
	$2.5V \leq V_{CC} \leq 4.5V$	2.5		DC	μs
R/C Oscillator	$4.5V \leq V_{CC} \leq 6.0V$	3		DC	μs
	$2.5V \leq V_{CC} \leq 4.5V$	7.5		DC	μs
V_{CC} Rise Time when Using Brown Out	V_{CC} = 0V to 6V	50			μs
Frequency at Brown Out Reset				4	MHz
CKI Frequency For Modular Output				4	MHz
CKI Clock Duty Cycle (Note 5)	fr = Max	40		60	%
Rise Time (Note 5)	fr = 10 MHz ext. Clock			12	ns
Fall Time (Note 5)	fr = 10 MHz ext. Clock			8	ns
Inputs					
t_{Setup}	$4.5V \leq V_{CC} \leq 6.0V$	200			ns
	$2.5V \leq V_{CC} \leq 4.5V$	500			ns
t_{Hold}	$4.5V \leq V_{CC} \leq 6.0V$	60			ns
	$2.5V \leq V_{CC} \leq 4.5V$	150			ns
Output Propagation Delay	R_L = 2.2k, CL = 100 pF				
t_{PD1}, t_{PD0}					
SO, SK	$4.5V \leq V_{CC} \leq 6.0V$			0.7	μs
	$2.5V \leq V_{CC} \leq 4.5V$			1.75	μs
All Others	$4.5V \leq V_{CC} \leq 6.0V$			1	μs
	$2.5V \leq V_{CC} \leq 4.5V$			5	μs
Input Pulse Width					
Interrupt Input High Time		1			tc
Interrupt Input Low Time		1			tc
Timer Input High Time		1			tc
Timer Input Low Time		1			tc
MICROWIRE Setup Time ($t_{\mu WS}$)		20			ns
MICROWIRE Hold Time ($t_{\mu WH}$)		56			ns
MICROWIRE Output Propagation Delay ($t_{\mu PD}$)				220	ns
Reset Pulse Width		1.0			μs

Note 5: Parameter characterized but not production tested.

AC Electrical Characteristics (Continued)



TL/DD/11208–2

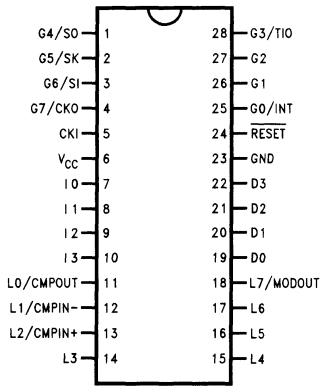
FIGURE 2. MICROWIRE/PLUS Timing

Comparator DC and AC Characteristics $4V \le V_{CC} \le 6V$, $-40°C \le T_A \le +85°C$ (Note 1)

Parameters	Conditions	Min	Type	Max	Units
Input Offset Voltage	$0.4V < V_{IN} < V_{CC} - 1.5V$		±10	±25	mV
Input Common Mode Voltage Range		0.4		$V_{CC} - 1.5$	V
Voltage Gain			300k		V/V
DC Supply Current (when enabled)	$V_{CC} = 6.0V$			250	μA
Response Time	TBD mV Step, TBD mV Overdrive, 100 pF Load			1	μs

Note 1: For comparator output current characteristics see L-Port specs.

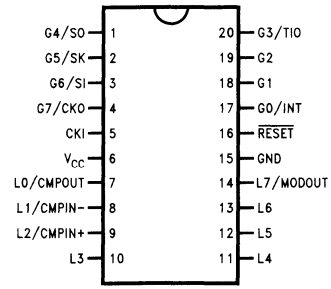
Connection Diagrams



TL/DD/11208–3

Top View

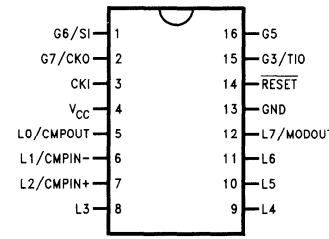
Order Number COPCJ820-XXX/N or COPCJ820-XXX/WM



TL/DD/11208–4

Top View

Order Number COPCJ822-XXX/N or COPCJ822-XXX/WM



TL/DD/11208–5

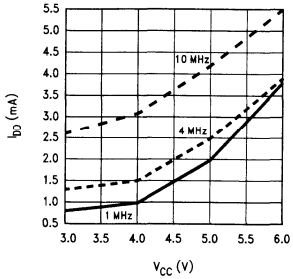
Top View

Order Number COPCJ823-XXX/WM

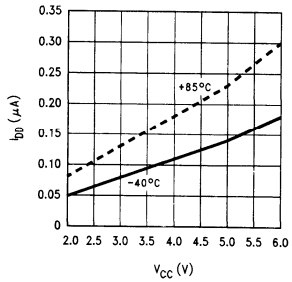
FIGURE 3. Connection Diagrams

Typical Performance Characteristics

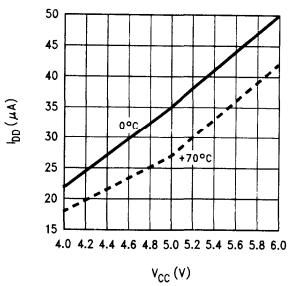
Dynamic—I_{DD} vs V_{CC} (Crystal Clock Option)



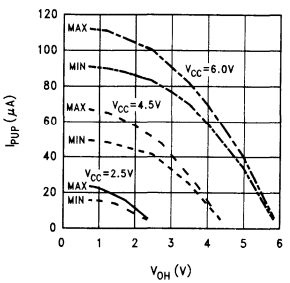
Halt—I_{DD} vs V_{CC} (Brown Out Disabled)



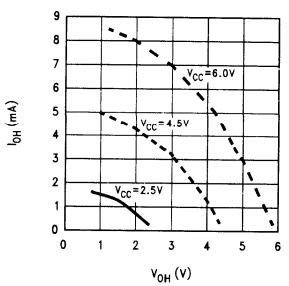
Halt—I_{DD} vs V_{CC} (Brown Out Enabled)



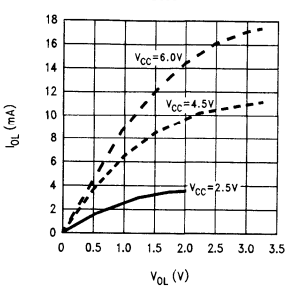
Ports L/G Weak Pull-Up Source Current



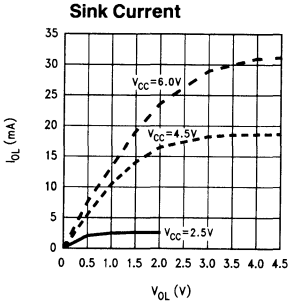
Ports L/G Push-Pull Source Current



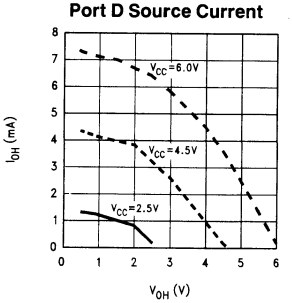
Ports L/G Push-Pull Sink Current



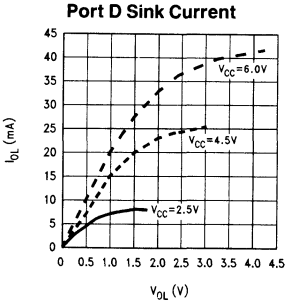
Ports L4–L7 Sink Current



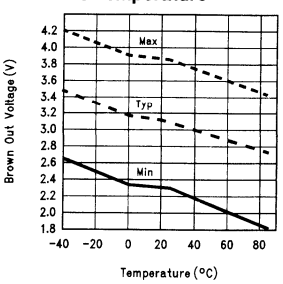
Port D Source Current



Port D Sink Current



Brown Out Voltage vs Temperature



TL/DD/11208–28

COP820CJ Pin Assignment

Port Pin	Typ	ALT Funct.	16 Pin	20 Pin	28 Pin
L0	I/O	MIWU/CMPOUT	5	7	11
L1	I/O	MIWU/CMPIN−	6	8	12
L2	I/O	MIWU/CMPIN+	7	9	13
L3	I/O	MIWU	8	10	14
L4	I/O	MIWU	9	11	15
L5	I/O	MIWU	10	12	16
L6	I/O	MIWU	11	13	17
L7	I/O	MIWU/MODOUT	12	14	18
G0	I/O	INTR		17	25
G1	I/O			18	26
G2	I/O			19	27
G3	I/O	TIO	15	20	28
G4	I/O	SO		1	1
G5	I/O	SK	16	2	2
G6	I	SI	1	3	3
G7	I	CKO	2	4	4
I0	I				7
I1	I				8
I2	I				9
I3	I				10
D0	O				19
D1	O				20
D2	O				21
D3	O				22
V_{CC}			4	6	6
GND			13	15	23
CKI			3	5	5
$\overline{\text{RESET}}$			14	16	24

Pin Description

V_{CC} and **GND** are the power supply pins.

CKI is the clock input. This can come from an external source, a R/C generated oscillator or a crystal (in conjunction with CKO). See Oscillator description.

$\overline{\text{RESET}}$ is the master reset input. See Reset description.

PORT I is a 4-bit Hi-Z input port.

PORT L is an 8-bit I/O port.

There are two registers associated with the L port: a data register and a configuration register. Therefore, each L I/O bit can be individually configured under software control as shown below:

Port L Config.	Port L Data	Port L Setup
0	0	Hi-Z Input (TRI-STATE)
0	1	Input with Weak Pull-up
1	0	Push-pull Zero Output
1	1	Push-pull One Output

Three data memory address locations are allocated for this port, one each for data register [00D0], configuration register [00D1] and the input pins [00D2].

Port L has the following alternate features:

L0 MIWU or CMPOUT
L1 MIWU or CMPIN−
L2 MIWU or CMPIN+
L3 MIWU
L4 MIWU (high sink current capability)
L5 MIWU (high sink current capability)
L6 MIWU (high sink current capability)
L7 MIWU or MODOUT (high sink current capability)

The selection of alternate Port L functions is done through registers WKEN [00C9] to enable MIWU and CNTRL2 [00CC] to enable comparator and modulator.

All eight L-pins have Schmitt Triggers on their inputs.

PORT G is an 8-bit port with 6 I/O pins (G0–G5) and 2 input pins (G6, G7).

All eight G-pins have Schmitt Triggers on the inputs.

There are two registers associated with the G port: a data register and a configuration register. Therefore each G port bit can be individually configured under software control as shown below:

Port G Config.	Port G Data	Port G Setup
0	0	Hi-Z Input (TRI-STATE)
0	1	Input with Weak Pull-up
1	0	Push-pull Zero Output
1	1	Push-pull One Output

Three data memory address locations are allocated for this port, one for data register [00D3], one for configuration register [00D5] and one for the input pins [00D6]. Since G6 and G7 are Hi-Z input only pins, any attempt by the user to configure them as outputs by writing a one to the configuration register will be disregarded. Reading the G6 and G7 configuration bits will return zeros. Note that the device will be placed in the Halt mode by writing a "1" to the G7 data bit.

Six pins of Port G have alternate features:

G0 INTR (an external interrupt)
G3 TIO (timer/counter input/output)
G4 SO (MICROWIRE serial data output)
G5 SK (MICROWIRE clock I/O)
G6 SI (MICROWIRE serial data input)
G7 CKO crystal oscillator output (selected by mask option) or HALT restart input/general purpose input (if clock option is R/C or external clock)

Pin Description (Continued)

Pins G1 and G2 currently do not have any alternate functions.

The selection of alternate Port G functions are done through registers PSW [00EF] to enable external interrupt and CNTRL1 [00EE] to select TIO and MICROWIRE operations.

PORT D is a four bit output port that is preset when RESET goes low. One data memory address location is allocated for the data register [00DC].

Note: Care must be exercised with the D2 pin operation. At RESET, the external loads on this pin must ensure that the output voltages stay above 0.8 V_{CC} to prevent the chip from entering special modes. Also keep the external loading on D2 to less than 1000 pF.

Functional Description

The internal architecture is shown in the block diagram. Data paths are illustrated in simplified form to depict how the various logic elements communicate with each other in implementing the instruction set of the device.

ALU and CPU Registers

The ALU can do an 8-bit addition, subtraction, logical or shift operations in one cycle time. There are five CPU registers:

A is the 8-bit Accumulator register

PC is the 15-bit Program Counter register

PU is the upper 7 bits of the program counter (PC)

PL is the lower 8 bits of the program counter (PC)

B is the 8-bit address register and can be auto incremented or decremented.

X is the 8-bit alternate address register and can be auto incremented or decremented.

SP is the 8-bit stack pointer which points to the subroutine stack (in RAM).

B, X and SP registers are mapped into the on chip RAM. The B and X registers are used to address the on chip RAM. The SP register is used to address the stack in RAM during subroutine calls and returns. The SP must be preset by software upon initialization.

Memory

The memory is separated into two memory spaces: program and data.

PROGRAM MEMORY

Program memory consists of 1024 x 8 ROM. These bytes of ROM may be instructions or constant data. The memory is addressed by the 15-bit program counter (PC). ROM can be indirectly read by the LAID instruction for table lookup.

DATA MEMORY

The data memory address space includes on chip RAM, I/O and registers. Data memory is addressed directly by the instruction or indirectly through B, X and SP registers. The device has 64 bytes of RAM. Sixteen bytes of RAM are mapped as "registers", these can be loaded immediately, decremented and tested. Three specific registers: X, B, and SP are mapped into this space, the other registers are available for general usage.

Any bit of data memory can be directly set, reset or tested. All I/O and registers (except A and PC) are memory mapped; therefore, I/O bits and register bits can be directly and individually set, reset and tested, except the write once only bit (WDREN, WATCHDOG Reset Enable), and the unused and read only bits in CNTRL2 and WDREG registers.

Note: RAM contents are undefined upon power-up.

Reset

EXTERNAL RESET

The RESET input pin when pulled low initializes the microcontroller. The user must insure that the RESET pin is held low until V_{CC} is within the specified voltage range and the clock is stabilized. An R/C circuit with a delay 5x greater than the power supply rise time is recommended *(Figure 4)*. The device immediately goes into reset state when the RESET input goes low. When the RESET pin goes high the device comes out of reset state synchronously. The device will be running within two instruction cycles of the RESET pin going high. The following actions occur upon reset:

Port L	TRI-STATE
Port G	TRI-STATE
Port D	HIGH
PC	CLEARED
RAM Contents	RANDOM with Power-On-Reset UNAFFECTED with external Reset (power already applied)
B, X, SP	Same as RAM
PSW, CNTRL1, CNTRL2 and WDREG Reg.	CLEARED
Multi-Input Wakeup Reg. WKEDG, WKEN WKPND	 CLEARED UNKNOWN
Data and Configuration Registers for L & G	CLEARED
WATCHDOG Timer	Prescaler/Counter each loaded with FF

The device comes out of the HALT mode when the RESET pin is pulled low. In this case, the user has to ensure that the RESET signal is low long enough to allow the oscillator to restart. An internal 256 t_C delay is normally used in conjunction with the two pin crystal oscillator. When the device comes out of the HALT mode through Multi-Input Wakeup, this delay allows the oscillator to stabilize.

The following additional actions occur after the device comes out of the HALT mode through the RESET pin.

If a two pin crystal/resonator oscillator is being used:

RAM Contents	UNCHANGED
Timer T1 and A Contents	UNKNOWN
WATCHDOG Timer Prescaler/Counter	ALTERED

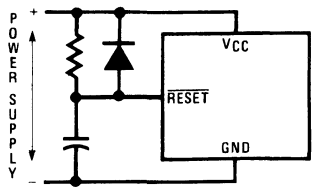
Functional Description (Continued)

If the external or RC Clock option is being used:

RAM Contents	UNCHANGED
Timer T1 and A Contents	UNCHANGED
WATCHDOG Timer Prescaler/Counter	ALTERED

The external RESET takes priority over the Brown Out Reset.

Note: If the RESET pin is pulled low while Brown Out occurs (Brown Out circuit has detected Brown Out condition), the external reset will not occur until the Brown Out condition is removed. External reset has priority only if V_{CC} is greater than the Brown Out voltage.



RC > 5 × Power Supply Rise Time TL/DD/11208–6

FIGURE 4. Recommended Reset Circuit

WATCHDOG RESET

With WATCHDOG enabled, the WATCHDOG logic resets the device if the user program does not service the WATCHDOG timer within the selected service window. The WATCHDOG reset does not disable the WATCHDOG. Upon WATCHDOG reset, the WATCHDOG Prescaler/Counter are each initialized with FF Hex.

The following actions occur upon WATCHDOG reset that are different from external reset.

WDREN WATCHDOG Reset Enable bit UNCHANGED
WDUDF WATCHDOG Underflow bit UNCHANGED

Additional initialization actions that occur as a result of WATCHDOG reset are as follows:

Port L	TRI-STATE
Port G	TRI-STATE
Port D	HIGH
PC	CLEARED
Ram Contents	UNCHANGED
B, X, SP	UNCHANGED
PSW, CNTRL1 and CNTRL2 (except WDUDF Bit) Registers	CLEARED
Multi-Input Wakeup Registers WKEDG, WKEN WKPND	CLEARED UNKNOWN
Data and Configuration Registers for L & G	CLEARED
WATCHDOG Timer	Prescalar/Counter each loaded with FF

BROWN OUT RESET

The on-board Brown Out protection circuit resets the device when the operating voltage (V_{CC}) is lower than the Brown Out voltage. The device is held in reset when V_{CC} stays below the Brown Out Voltage. The device will remain in RESET as long as V_{CC} is below the Brown Out voltage. The Device will resume execution if V_{CC} rises above the Brown Out Voltage. If a two pin crystal/resonator clock option is selected, the Brown Out reset will trigger a 256tc delay. This delay allows the oscillator to stabilize before the device exits the reset state. The delay is not used if the clock option is either R/C or external clock. The contents of data registers and RAM are unknown following a Brown Out reset. The external reset takes priority over Brown Out Reset and will deactivate the 256 tc cycles delay if in progress. The Brown Out reset takes priority over the WATCHDOG reset.

The following actions occur as a result of Brown Out reset:

Port L	TRI-STATE
Port G	TRI-STATE
Port D	HIGH
PC	CLEARED
RAM Contents	RANDOM
B, X, SP	UNKNOWN
PSW, CNTRL1, CNTRL2 and WDREG Registers	CLEARED
Multi-Input Wakeup Registers WKEDG, WKEN WKPND	CLEARED UNKNOWN
Data and Configuration Registers for L & G	CLEARED
WATCHDOG Timer	Prescalar/Counter each loaded with FF
Timer T1 and Accumulator	Unknown data after coming out of the HALT (through Brown Out Reset) with any Clock option

Note: The development system will detect the BROWN OUT RESET externally and will force the $\overline{\text{RESET}}$ pin low. The Development System does not emulate the 256tc delay.

Brown Out Detection

An on-board detection circuit monitors the operating voltage (V_{CC}) and compares it with the minimum operating voltage specified. The Brown Out circuit is designed to reset the device if the operating voltage is below the Brown Out voltage (between 1.8V to 4.2V at −40°C to +85°C). The Minimum operating voltage for the device is 2.5V with Brown Out disabled, but with BROWN OUT enabled the device is guaranteed to operate properly down to minimum Brown Out voltage (Max frequency 4 MHz), For temperature range of 0°C to 70°C the Brown Out voltage is expected to be between 1.9V to 3.9V. The circuit can be enabled or disabled by Brown Out mask option. If the device is intended to operate at lower V_{CC} (lower than Brown Out voltage VBO max), the Brown Out circuit should be disabled by the mask option.

The Brown Out circuit may be used as a power-up reset provided the power supply rise time is slower than 50 μs (0V to 6.0V).

Note: Brown Out Circuit is active in HALT mode (with the Brown Out mask option selected).

Functional Description (Continued)

Oscillator Circuits

EXTERNAL OSCILLATOR

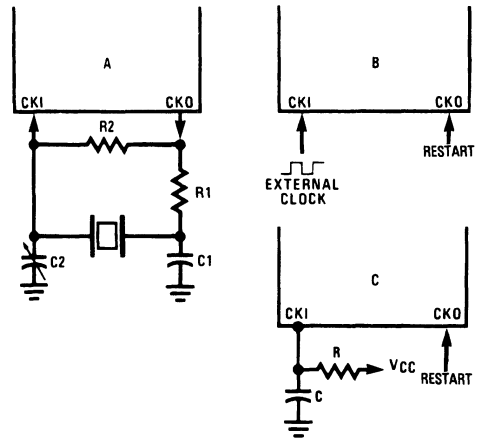
CKI can be driven by an external clock signal provided it meets the specified duty cycle, rise and fall times, and input levels. CKO is available as a general purpose input G7 and/or Halt control.

CRYSTAL OSCILLATOR

By selecting CKO as a clock output, CKI and CKO can be connected to create a crystal controlled oscillator. Table I shows the component values required for various standard crystal values.

R/C OSCILLATOR

By selecting CKI as a single pin oscillator, CKI can make a R/C oscillator. CKO is available as a general purpose input and/or HALT control. Table II shows variation in the oscillator frequencies as functions of the component (R and C) values.



TL/DD/11208–7

FIGURE 5. Clock Oscillator Configurations

TABLE I. Crystal Oscillator Configuration

R1 (kΩ)	R2 (MΩ)	C1 (pF)	C2 (pF)	CKI Freq. (MHz)	Conditions
0	1	30	30–36	10	V_{CC} = 5V
0	1	30	30–36	4	V_{CC} = 5V
5.6	1	100	100–156	0.455	V_{CC} = 5V

TABLE II. RC Oscillator Configuration (Part-To-Part Variation)

R (kΩ)	C (pF)	CK1 Freq. (MHz)	Instr. Cycle (μs)	Conditions
3.3	82	2.2 to 2.7	3.7 to 4.6	V_{CC} = 5V
5.6	100	1.1 to 1.3	7.4 to 9.0	V_{CC} = 5V
6.8	100	0.9 to 1.1	8.8 to 10.8	V_{CC} = 5V

Functional Description (Continued)

Halt Mode

The device is a fully static device. The device enters the HALT mode by writing a one to the G7 bit of the G data register. Once in the HALT mode, the internal circuitry does not receive any clock signal and is therefore frozen in the exact state it was in when halted. In this mode the chip will only draw leakage current (output current and DC current due to the Brown Out circuit if Brown Out is enabled).

The device supports four different methods of exiting the HALT mode. The first method is with a low to high transition on the CKO (G7) pin. This method precludes the use of the crystal clock configuration (since CKO is a dedicated output). It may be used either with an RC clock configuration or an external clock configuration. The second method of exiting the HALT mode is with the multi-Input Wakeup feature on the L port. The third method of exiting the HALT mode is by pulling the RESET input low. The fourth method is with the operating voltage going below Brown Out voltage (if Brown Out is enabled by mask option).

If the two pin crystal/resonator oscillator is being used and Multi-Input Wakeup or Brown Out causes the device to exit the HALT mode, the WAKEUP signal does not allow the chip to start running immediately since crystal oscillators have a delayed start up time to reach full amplitude and freuqency stability. The WATCHDOG timer (consisting of an 8-bit prescaler followed by an 8-bit counter) is used to generate a fixed delay of 256tc to ensure that the oscillator has indeed stabilized before allowing instruction execution. In this case, upon detecting a valid WAKEUP signal only the oscillator circuitry is enabled. The WATCHDOG Counter and Prescaler are each loaded with a value of FF Hex. The WATCHDOG prescaler is clocked with the tc instruction cycle. (The tc clock is derived by dividing the oscillator clock down by a factor of 10). The Schmitt trigger following the CKI inverter on the chip ensures that the WATCHDOG timer is clocked only when the oscillator has a sufficiently large amplitude to meet the Schmitt trigger specs. This Schmitt trigger is not part of the oscillator closed loop. The start-up timeout from the WATCHDOG timer enables the clock signals to be routed to the rest of the chip. The delay is not activated when the device comes out of HALT mode through RESET pin. Also, if the clock option is either RC or External clock, the delay is not used, but the WATCHDOG Prescaler/-Counter contents are changed. The Development System will not emulate the 256tc delay.

The $\overline{\text{RESET}}$ pin or Brown Out will cause the device to reset and start executing from address X'0000. A low to high transition on the G7 pin (if single pin oscillator is used) or Multi-Input Wakeup will cause the device to start executing from the address following the HALT instruction.

When $\overline{\text{RESET}}$ pin is used to exit the device from the HALT mode and the two pin crystal/resonator (CKI/CKO) clock option is selected, the contents of the Accumulator and the Timer T1 are undetermined following the reset. All other information except the WATCHDOG Prescaler/Counter contents is retained until continuing. If the device comes out of the HALT mode through Brown Out reset, the contents of data registers and RAM are unknown following the reset. All information except the WATCHDOG Prescaler/Counter contents is retained if the device exits the HALT mode through G7 pin or Multi-Input Wakeup.

G7 is the HALT-restart pin, but it can still be used as an input. If the device is not halted, G7 can be used as a general purpose input.

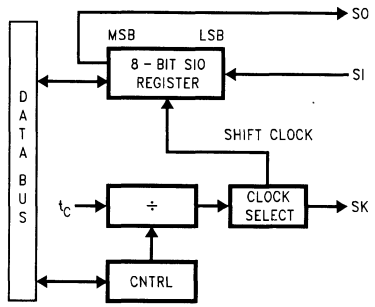
If the Brown Out Enable mask option is selected, the Brown Out circuit remains active during the HALT mode causing additional current to be drawn.

Note: To allow clock resynchronization, it is necessary to program two NOP's immediately after the device comes out of the HALT mode. The user must program two NOP's following the "enter HALT mode" (set G7 data bit) instruction.

Functional Description (Continued)

MICROWIRE/PLUS

MICROWIRE/PLUS is a serial synchronous bidirectional communications interface. The MICROWIRE/PLUS capability enables the device to interface with any of National Semiconductor's MICROWIRE peripherals (i.e. A/D converters, display drivers, EEPROMS, etc.) and with other microcontrollers which support the MICROWIRE/PLUS interface. It consists of an 8-bit serial shift register (SIO) with serial data input (SI), serial data output (SO) and serial shift clock (SK). *Figure 6* shows the block diagram of the MICROWIRE/PLUS interface.



TL/DD/11208–8

FIGURE 6. MICROWIRE/PLUS Block Diagram

The shift clock can be selected from either an internal source or an external source. Operating the MICROWIRE/PLUS interface with the internal clock source is called the Master mode of operation. Operating the MICROWIRE/PLUS interface with an external shift clock is called the Slave mode of operation.

The CNTRL register is used to configure and control the MICROWIRE/PLUS mode. To use the MICROWIRE/PLUS, the MSEL bit in the CNTRL register is set to one. The SK clock rate is selected by the two bits, SL0 and SL1, in the CNTRL register. Table III details the different clock rates that may be selected.

TABLE III

SL1	SL0	SK Cycle Time
0	0	$2t_C$
0	1	$4t_C$
1	x	$8t_C$

where,

t_C is the instruction cycle time.

MICROWIRE/PLUS OPERATION

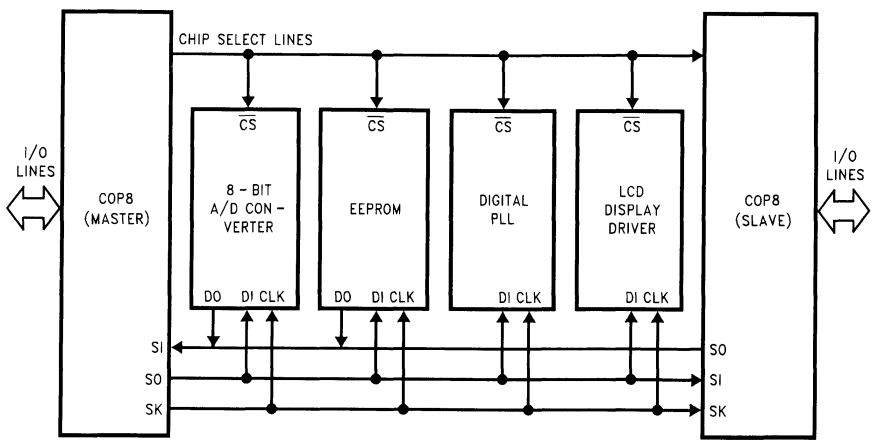
Setting the BUSY bit in the PSW register causes the MICROWIRE/PLUS arrangement to start shifting the data. It gets reset when eight data bits have been shifted. The user may reset the BUSY bit by software to allow less than 8 bits to shift. The device may enter the MICROWIRE/PLUS mode either as a Master or as a Slave. *Figure 7* shows how two device microcontrollers and several peripherals may be interconnected using the MICROWIRE/PLUS arrangement.

Master MICROWIRE/PLUS Operation

In the MICROWIRE/PLUS Master mode of operation the shift clock (SK) is generated internally by the device. The MICROWIRE/PLUS Master always initiates all data exchanges *(Figure 7)*. The MSEL bit in the CNTRL register must be set to enable the SO and SK functions on the G Port. The SO and SK pins must also be selected as outputs by setting appropriate bits in the Port G configuration register. Table IV summarizes the bit settings required for Master mode of operation.

SLAVE MICROWIRE/PLUS OPERATION

In the MICROWIRE/PLUS Slave mode of operation the SK clock is generated by an external source. Setting the MSEL bit in the CNTRL register enables the SO and SK functions on the G Port. The SK pin must be selected as an input and the SO pin selected as an output pin by appropriately setting up the Port G configuration register. Table IV summarizes the settings required to enter the Slave mode of operation.



TL/DD/11208–23

FIGURE 7. MICROWIRE/PLUS Application

Functional Description (Continued)

The user must set the BUSY flag immediately upon entering the Slave mode. This will ensure that all data bits sent by the Master will be shifted properly. After eight clock pulses the BUSY flag will be cleared and the sequence may be repeated.

TABLE IV

G4 Config. Bit	G5 Config. Bit	G4 Fun.	G5 Fun.	G6 Fun.	Operation
1	1	SO	Int. SK	SI	MICROWIRE Master
0	1	TRI-STATE	Int. SK	SI	MICROWIRE Master
1	0	SO	Ext. SK	SI	MICROWIRE Slave
0	0	TRI-STATE	Ext. SK	SI	MICROWIRE Slave

Timer/Counter

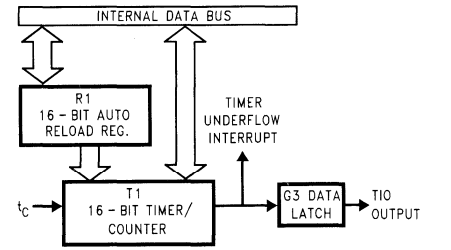
The device has a powerful 16-bit timer with an associated 16-bit register enabling it to perform extensive timer functions. The timer T1 and its register R1 are each organized as two 8-bit read/write registers. Control bits in the register CNTRL allow the timer to be started and stopped under software control. The timer-register pair can be operated in one of three possible modes. Table V details various timer operating modes and their requisite control settings.

MODE 1. TIMER WITH AUTO-LOAD REGISTER

In this mode of operation, the timer T1 counts down at the instruction cycle rate. Upon underflow the value in the register R1 gets automatically reloaded into the timer which continues to count down. The timer underflow can be programmed to interrupt the microcontroller. A bit in the control register CNTRL enables the TIO (G3) pin to toggle upon timer underflows. This allows the generation of square-wave outputs or pulse width modulated outputs under software control *(Figure 8)*.

MODE 2. EXTERNAL COUNTER

In this mode, the timer T1 becomes a 16-bit external event counter. The counter counts down upon an edge on the TIO pin. Control bits in the register CNTRL program the counter to decrement either on a positive edge or on a negative edge. Upon underflow the contents of the register R1 are automatically copied into the counter. The underflow can also be programmed to generate an interrupt *(Figure 9)*.

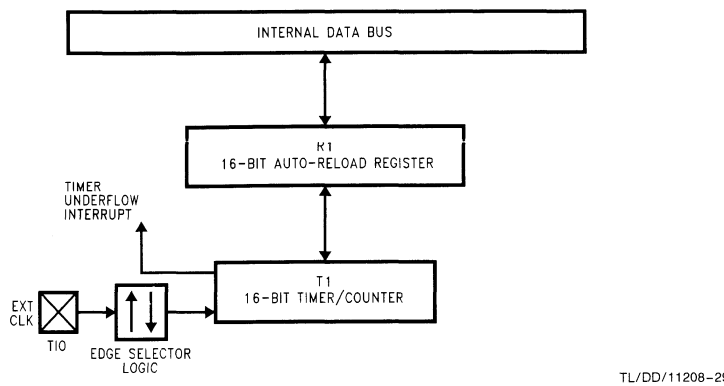


TL/DD/11208-24

FIGURE 8. Timer/Counter Auto Reload Mode Block Diagram

TABLE V. Timer Operating Modes

CNTRL Bits 7 6 5	Operation Mode	T Interrupt	Timer Counts On
0 0 0	External Counter w/Auto-Load Reg.	Timer Underflow	TIO Pos. Edge
0 0 1	External Counter w/Auto-Load Reg.	Timer Underflow	TIO Neg. Edge
0 1 0	Not Allowed	Not Allowed	Not Allowed
0 1 1	Not Allowed	Not Allowed	Not Allowed
1 0 0	Timer w/Auto-Load Reg.	Timer Underflow	t_C
1 0 1	Timer w/Auto-Load Reg./Toggle TIO Out	Timer Underflow	t_C
1 1 0	Timer w/Capture Register	TIO Pos. Edge	t_C
1 1 1	Timer w/Capture Register	TIO Neg. Edge	t_C



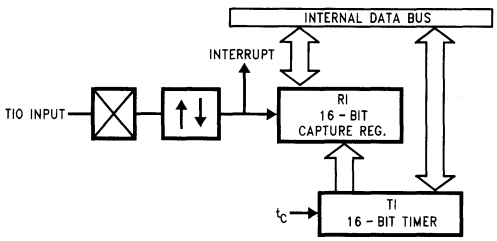
TL/DD/11208-29

FIGURE 9. Timer in External Event Counter Mode

Timer/Counter (Continued)

MODE 3. TIMER WITH CAPTURE REGISTER

Timer T1 can be used to precisely measure external frequencies or events in this mode of operation. The timer T1 counts down at the instruction cycle rate. Upon the occurrence of a specified edge on the TIO pin the contents of the timer T1 are copied into the register R1. Bits in the control register CNTRL allow the trigger edge to be specified either as a positive edge or as a negative edge. In this mode the user can elect to be interrupted on the specified trigger edge *(Figure 10)*.

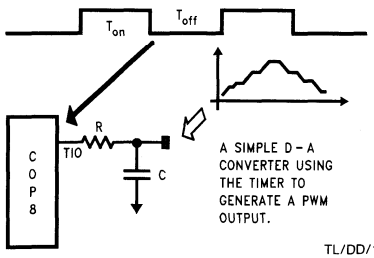


TL/DD/11208–25

FIGURE 10. Timer Capture Mode Block Diagram

TIMER PWM APPLICATION

Figure 11 shows how a minimal component D/A converter can be built out of the Timer-Register pair in the Auto-Reload mode. The timer is placed in the "Timer with auto reload" mode and the TIO pin is selected as the timer output. At the outset the TIO pin is set high, the timer T1 holds the on time and the register R1 holds the signal off time. Setting TRUN bit starts the timer which counts down at the instruction cycle rate. The underflow toggles the TIO output and copies the off time into the timer, which continues to run. By alternately loading in the on time and the off time at each successive interrupt a PWM frequency can be easily generated.



TL/DD/11208–26

FIGURE 11. Timer Application

Watchdog

The device has an on-board 8-bit WATCHDOG timer. The timer contains an 8-bit READ/WRITE down counter clocked by an 8-bit prescaler. Under software control the timer can be dedicated for the WATCHDOG or used as a general purpose counter. *Figure 12* shows the WATCHDOG timer block diagram.

MODE 1: WATCHDOG TIMER

The WATCHDOG is designed to detect user programs getting stuck in infinite loops resulting in loss of program control or "runaway" programs. The WATCHDOG can be enabled or disabled (only once) after the device is reset as a result of brown out reset or external reset. On power-up the WATCHDOG is disabled. The WATCHDOG is enabled by writing a "1" to WDREN bit (resides in WDREG register). Once enabled, the user program should write periodically into the 8-bit counter before the counter underflows. The 8-bit counter (WDCNT) is memory mapped at address 0CE Hex. The counter is loaded with n-1 to get n counts. The counter underflow resets the device, but does not disable the WATCHDOG. Loading the 8-bit counter initializes the prescaler with FF Hex and starts the prescaler/counter. Prescaler and counter are stopped upon counter underflow. Prescaler and counter are each loaded with FF Hex when the device goes into the HALT mode. The prescaler is used for crystal/resonator start-up when the device exits the HALT mode through Multi-Input Wakeup. In this case, the prescaler/counter contents are changed.

MODE 2: TIMER

In this mode, the prescaler/counter is used as a timer by keeping the WDREN (WATCHDOG reset enable) bit at 0. The counter underflow sets the WDUDF (underflow) bit and the underflow does not reset the device. Loading the 8-bit counter (load n-1 for n counts) sets the WDTEN bit (WATCHDOG Timer Enable) to "1", loads the prescaler with FF, and starts the timer. The counter underflow stops the timer. The WDTEN bit serves as a start bit for the WATCHDOG timer. This bit is set when the 8-bit counter is loaded by the user program. The load could be as a result of WATCHDOG service (WATCHDOG timer dedicated for WATCHDOG function) or write to the counter (WATCHDOG timer used as a general purpose counter). The bit is cleared upon Brown Out reset, WATCHDOG reset or external reset. The bit is not memory mapped and is transparent to the user program.

TABLE VI. WATCHDOG Control/Status

Parameter	HALT Mode	WD Reset	EXT/BOR Reset (Note 1)	Counter Load
8-Bit Prescaler	FF	FF	FF	FF
8-Bit WD Counter	FF	FF	FF	User Value
WDREN Bit	Unchanged	Unchanged	0	No Effect
WDUDF Bit	0	Unchanged	0	0
WDTEN Signal	Unchanged	0	0	1

Note 1: BOR is Brown Out Reset.

Functional Description (Continued)

CONTROL/STATUS BITS

WDUDF: WATCHDOG Timer Underflow Bit

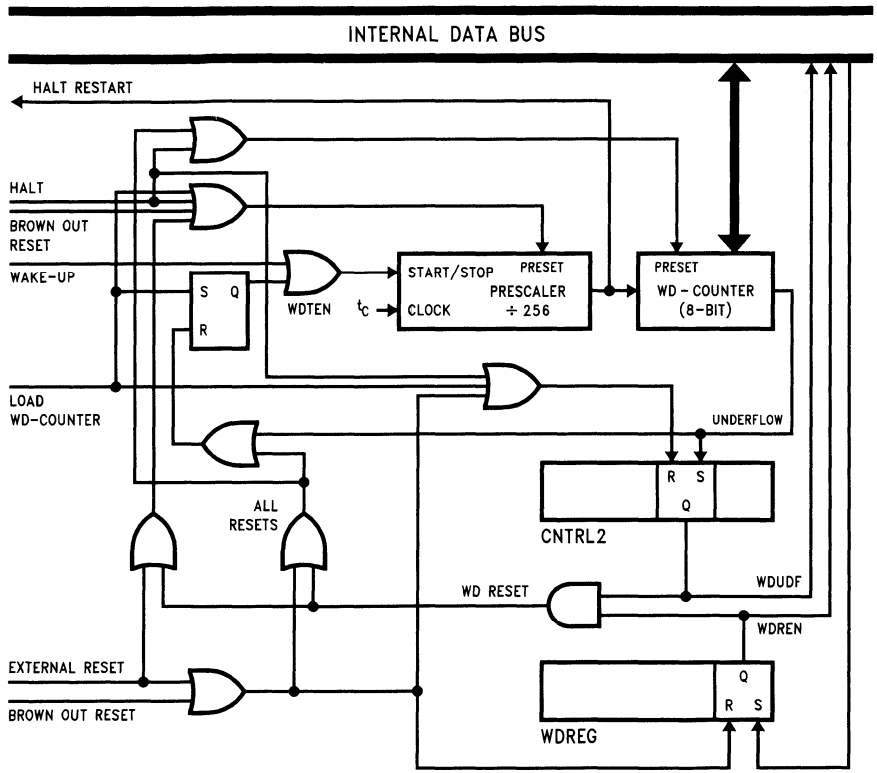
This bit resides in the CNTRL2 Register. The bit is set when the WATCHDOG timer underflows. The underflow resets the device if the WATCHDOG reset enable bit is set (WDREN = 1). Otherwise, WDUDF can be used as the timer underflow flag. The bit is cleared upon Brown-Out reset, external reset, load to the 8-bit counter, or going into the HALT mode. It is a read only bit.

WDREN: WD Reset Enable

WDREN bit resides in a separate register (bit 0 of WDREG). This bit enables the WATCHDOG timer to generate a reset. The bit is cleared upon Brown Out reset, or external reset. The bit under software control can be written to only once (once written to, the hardware does not allow the bit to be changed during program execution).

WDREN = 1 WATCHDOG reset is enabled.
WDREN = 0 WATCHDOG reset is disabled.

Table VI shows the impact of Brown Out Reset, WATCHDOG Reset, and External Reset on the Control/Status bits.



TL/DD/11208–15

FIGURE 12. WATCHDOG Timer Block Diagram

Modulator/Timer

The Modulator/Timer contains an 8-bit counter and an 8-bit autoreload register (MODRL address 0CF Hex). The Modulator/Timer has two modes of operation, selected by the control bit MC3. The Modulator/Timer Control bits MC1, MC2 and MC3 reside in CNTRL2 Register.

MODE 1: MODULATOR

The Modulator is used to generate high frequency pulses on the modulator output pin (L7). The L7 pin should be configured as an output. The number of pulses is determined by the 8-bit down counter. Under software control the modulator input clock can be either CKI or tC. The tC clock is derived by dividing down the oscillator clock by a factor of 10. Three control bits (MC1, MC2, and MC3) are used for the Modulator/Timer output control. When MC2 = 1 and MC3 = 1, CKI is used as the modulator input clock. When MC2 = 0, and MC3 = 1, tC is used as the modulator input clock. The user loads the counter with the desired number of counts (256 max) and sets MC1 to start the counter. The modulator autoreload register is loaded with n-1 to get n pulses. CKI or tc pulses are routed to the modulator output (L7) until the counter underflows *(Figure 13).* Upon underflow the hardware resets MC1 and stops the counter. The L7 pin goes low and stays low until the counter is restarted by the user program. The user program has the responsibility to timeout the low time. Unless the number of counts is changed, the user program does not have to load the counter each time the counter is started. The counter can simply be started by setting the MC1 bit. Setting MC1 by software will load the counter with the value of the autoreload register. The software can reset MC1 to stop the counter.

MODE 2: PWM TIMER

The counter can also be used as a PWM Timer. In this mode, an 8-bit register is used to serve as an autoreload register (MODRL).

a. 50% Duty Cycle:

When MC1 is 1 and MC2, MC3 are 0, a 50% duty cycle free running signal is generated on the L7 output pin *(Figure 14).* The L7 pin must be configured as an output pin. In this mode the 8-bit counter is clocked by tC. Setting the MC1 control bit by software loads the counter with the value of the autoreload register and starts the counter. The counter underflow toggles the (L7) output pin. The 50% duty cycle signal will be continuously generated until MC1 is reset by the user program.

b. Variable Duty Cycle:

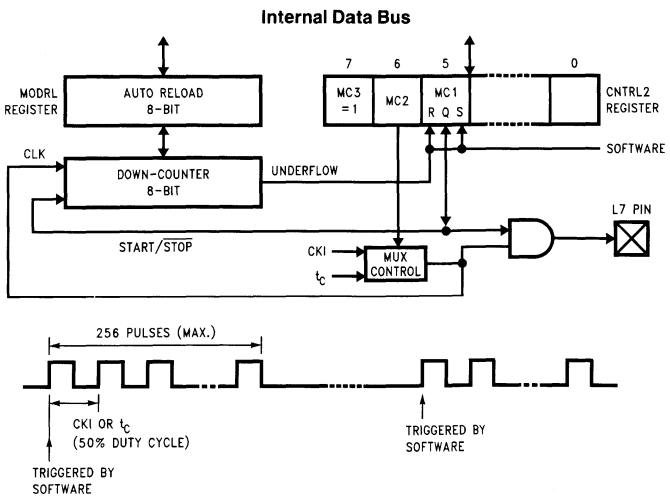
When MC3 = 0 and MC2 = 1, a variable duty cycle PWM signal is generated on the L7 output pin. The counter is clocked by tC. In this mode the 16-bit timer T1 along with the 8-bit down counter are used to generate a variable duty cycle PWM signal. The timer T1 underflow sets MC1 which starts the down counter and it also sets L7 high (L7 should be configured as an output).When the counter underflows the MC1 control bit is reset and the L7 output will go low until the next timer T1 underflow. Therefore, the width of the output pulse is controlled by the 8-bit counter and the pulse duration is controlled by the 16-bit timer T1 *(Figure 15).* Timer T1 must be configured in "PWM Mode/Toggle TIO Out" (CNTRL1 Bits 7,6,5 = 101).

Table VII shows the different operation modes for the Modulator/Timer.

TABLE VII. Modulator/Timer Modes

Control Bits in CNTRL2(00CC)			Operation Mode L7 Function
MC3	MC2	MC1	
0	0	0	Normal I/O
0	0	1	50% Duty Cycle Mode (Clocked by tc)
0	1	X	Variable Duty Cycle Mode (Clocked by tc) Using Timer 1 Underflow
1	0	X	Modulator Mode (Clocked by tc)
1	1	X	Modulator Mode (Clocked by CKI)

Note: MC1, MC2 and MC3 control bits are cleared upon reset.



TL/DD/11208–16

FIGURE 13. Mode 1: Modulator Block Diagram/Output Waveform

Modulator/Timer (Continued)

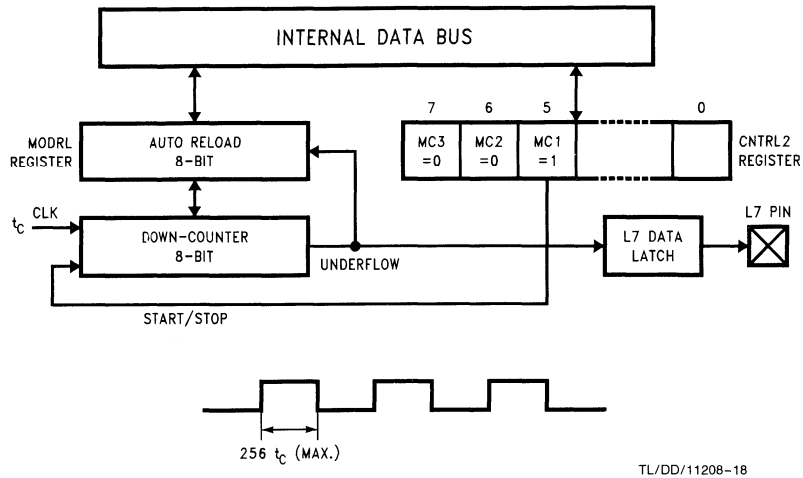


FIGURE 14. Mode 2a: 50% Duty Cycle Output

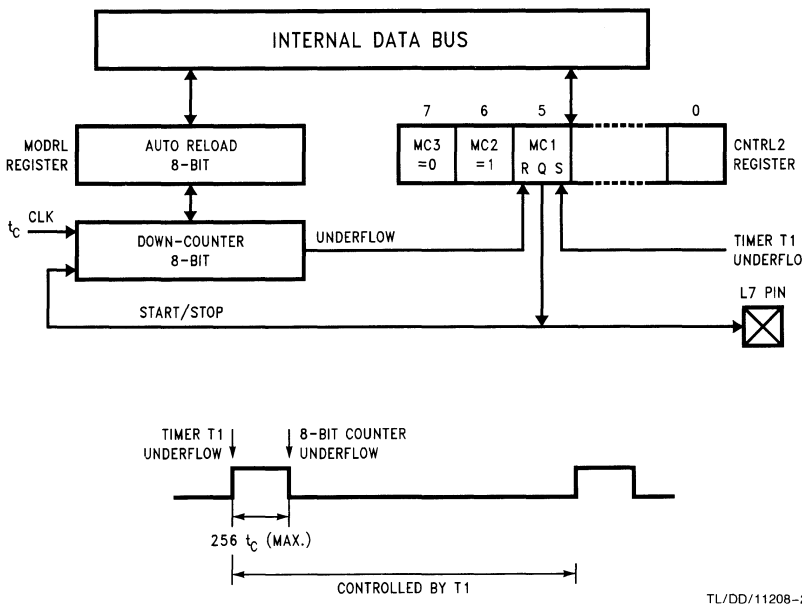


FIGURE 15. Mode 2b: Variable Duty Cycle Output

Comparator

The device has one differential comparator. Ports L0–L2 are used for the comparator. The output of the comparator is brought out to a pin. Port L has the following assignments:

L0 Comparator output
L1 Comparator negative input
L2 Comparator positive input

THE COMPARATOR STATUS/CONTROL BITS

These bits reside in the CNTRL2 Register (Address 0CC)

CMPEN Enables comparator ("1" = enable)

CMPRD Reads comparator output internally (CMPEN = 1, CMPOE = X)

CMPOE Enables comparator output to pin L0 ("1" = enable), CMPEN bit must be set to enable this function. If CMPEN = 0, L0 will be 0.

The Comparator Select/Control bits are cleared on RESET (the comparator is disabled). To save power the program should also disable the comparator before the device enters the HALT mode.

The user program must set up L0, L1 and L2 ports correctly for comparator Inputs/Output: L1 and L2 need to be configured as inputs and L0 as output.

Multi-Input Wake Up

The Multi-Input Wakeup feature is used to return (wakeup) the device from the HALT mode. *Figure 16* shows the Multi-Input Wakeup logic.

This feature utilizes the L Port. The user selects which particular L port bit or combination of L Port bits will cause the device to exit the HALT mode. Three 8-bit memory mapped registers, Reg:WKEN, Reg:WKEDG, and Reg:WKPND are used in conjunction with the L port to implement the Multi-Input Wakeup feature.

All three registers Reg:WKEN, Reg:WKPND, and Reg:WKEDG are read/write registers, and are cleared at reset, except WKPND. WKPND is unknown on reset.

The user can select whether the trigger condition on the selected L Port pin is going to be either a positive edge (low to high transition) or a negative edge (high to low transition). This selection is made via the Reg:WKEDG, which is an 8-bit control register with a bit assigned to each L Port pin. Setting the control bit will select the trigger condition to be a negative edge on that particular L Port pin. Resetting the bit selects the trigger condition to be a positive edge. Changing an edge select entails several steps in order to avoid a pseudo Wakeup condition as a result of the edge change. First, the associated WKEN bit should be reset, followed by the edge select change in WKEDG. Next, the associated WKPND bit should be cleared, followed by the associated WKEN bit being re-enabled.

An example may serve to clarify this procedure. Suppose we wish to change the edge select from positive (low going high) to negative (high going low) for L port bit 5, where bit 5 has previously been enabled for an input. The program would be as follows:

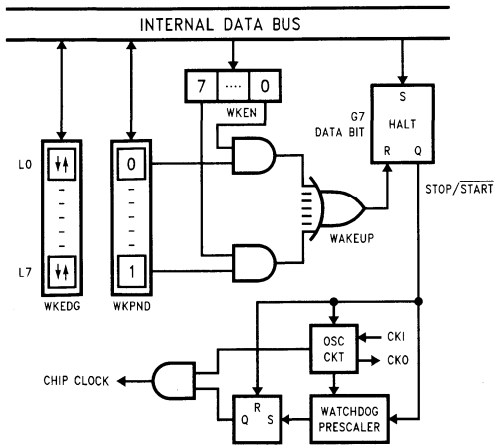
```
RBIT  5,WKEN
SBIT  5,WKEDG
RBIT  5,WKPND
SBIT  5,WKEN
```

If the L port bits have been used as outputs and then changed to inputs with Multi-Input Wakeup, a safety procedure should also be followed to avoid inherited pseudo wakeup conditions. After the selected L port bits have been changed from output to input but before the associated WKEN bits are enabled, the associated edge select bits in WKEDG should be set or reset for the desired edge selects, followed by the associated WKPND bits being cleared. This same procedure should be used following RESET, since the L port inputs are left floating as a result of RESET.

The occurrence of the selected trigger condition for Multi-Input Wakeup is latched into a pending register called Reg:WKPND. The respective bits of the WKPND register will be set on the occurrence of the selected trigger edge on the corresponding Port L pin. The user has the responsibility of clearing these pending flags. Since the Reg:WKPND is a pending register for the occurrence of selected wakeup conditions, the device will not enter the HALT mode if any Wakeup bit is both enabled and pending. Setting the G7 data bit under this condition will not allow the device to enter the HALT mode. Consequently, the user has the responsibility of clearing the pending flags before attempting to enter the HALT mode.

If a crystal oscillator is being used, the Wakeup signal will not start the chip running immediately since crystal oscillators have a finite start up time. The WATCHDOG timer prescaler generates a fixed delay to ensure that the oscillator has indeed stabilized before allowing the device to execute instructions. In this case, upon detecting a valid Wakeup signal only the oscillator circuitry and the WATCHDOG timer are enabled. The WATCHDOG timer prescaler is loaded with a value of FF Hex (256 counts) and is clocked from the tc instruction cycle clock. The tc clock is derived by dividing down the oscillator clock by a factor of 10. A Schmitt trigger following the CKI on chip inverter ensures that the WATCHDOG timer is clocked only when the oscillator has a sufficiently large amplitude to meet the Schmitt trigger specs. This Schmitt trigger is not part of the oscillator closed loop. The startup timeout from the WATCHDOG timer enables the clock signals to be routed to the rest of the chip.

Multi-Input Wakeup (Continued)



TL/DD/11208–21

FIGURE 16. Multi-Input Wakeup Logic

INTERRUPTS

The device has a sophisticated interrupt structure to allow easy interface to the real world. There are three possible interrupt sources, as shown below.

A maskable interrupt on external G0 input (positive or negative edge sensitive under software control)

A maskable interrupt on timer carry or timer capture

A non-maskable software/error interrupt on opcode zero

INTERRUPT CONTROL

The GIE (global interrupt enable) bit enables the interrupt function. This is used in conjunction with ENI and ENTI to select one or both of the interrupt sources. This bit is reset when interrupt is acknowledged.

ENI and ENTI bits select external and timer interrupts respectively. Thus the user can select either or both sources to interrupt the microcontroller when GIE is enabled.

IEDG selects the external interrupt edge (0 = rising edge, 1 = falling edge). The user can get an interrupt on both rising and falling edges by toggling the state of IEDG bit after each interrupt.

IPND and TPND bits signal which interrupt is pending. After an interrupt is acknowledged, the user can check these two bits to determine which interrupt is pending. This permits the interrupts to be prioritized under software. The pending flags have to be cleared by the user. Setting the GIE bit high inside the interrupt subroutine allows nested interrupts.

The software interrupt does not reset the GIE bit. This means that the controller can be interrupted by other interrupt sources while servicing the software interrupt.

INTERRUPT PROCESSING

The interrupt, once acknowledged, pushes the program counter (PC) onto the stack and the stack pointer (SP) is decremented twice. The Global Interrupt Enable (GIE) bit is reset to disable further interrupts. The microcontroller then vectors to the address 00FFH and resumes execution from that address. This process takes 7 cycles to complete. At the end of the interrupt subroutine, any of the following three instructions return the processor back to the main program: RET, RETSK or RETI. Either one of the three instructions will pop the stack into the program counter (PC). The stack pointer is then incremented twice. The RETI instruction additionally sets the GIE bit to re-enable further interrupts.

Any of the three instructions can be used to return from a hardware interrupt subroutine. The RETSK instruction should be used when returning from a software interrupt subroutine to avoid entering an infinite loop.

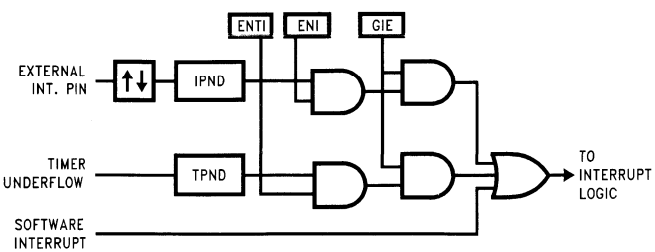
Note: There is always the possibility of an interrupt occurring during an instruction which is attempting to reset the GIE bit or any other interrupt enable bit. If this occurs when a single cycle instruction is being used to reset the interrupt enable bit, the interrupt enable bit will be reset but an interrupt may still occur. This is because interrupt processing is started at the same time as the interrupt bit is being reset. To avoid this scenario, the user should always use a two, three, or four cycle instruction to reset interrupt enable bits.

DETECTION OF ILLEGAL CONDITIONS

The device incorporates a hardware mechanism that allows it to detect illegal conditions which may occur from coding errors, noise, and "brown out" voltage drop situations. Specifically, it detects cases of executing out of undefined ROM area and unbalanced tack situations.

Reading an undefined ROM location returns 00 (hexadecimal) as its contents. The opcode for a software interrupt is also "00". Thus a program accessing undefined ROM will cause a software interrupt.

Reading an undefined RAM location returns an FF (hexadecimal). The subroutine stack on the device grows down for each subroutine call. By initializing the stack pointer to the top of RAM, the first unbalanced return instruction will cause the stack pointer to address undefined RAM. As a result the program will attempt to execute from FFFF (hexadecimal), which is an undefined ROM location and will trigger a software interrupt.



TL/DD/11208–27

FIGURE 17. Interrupt Block Diagram

Control Registers

CNTRL1 REGISTER (ADDRESS 00EE)

The Timer and MICROWIRE control register contains the following bits:

SL1 and SL0 Select the MICROWIRE clock divide-by (00 = 2, 01 = 4, 1x = 8)

IEDG External interrupt edge polarity select

MSEL Selects G5 and G4 as MICROWIRE signals SK and SO respectively

TRUN Used to start and stop the timer/counter (1 = run, 0 = stop)

TC1 Timer T1 Mode Control Bit

TC2 Timer T1 Mode Control Bit

TC3 Timer T1 Mode Control Bit

Bit 7							Bit 0
TC1	TC2	TC3	TRUN	MSEL	IEDG	SL1	SL0

PSW REGISTER (ADDRESS 00EF)

The PSW register contains the following select bits:

GIE Global interrupt enable (enables interrupts)

ENI External interrupt enable

BUSY MICROWIRE busy shifting flag

PND External interrupt pending

ENTI Timer T1 interrupt enable

TPND Timer T1 interrupt pending (timer Underflow or capture edge)

C Carry Flip/Flop

HC Half-Carry Flip/Flop

Bit 7							Bit 0
HC	C	TPND	ENTI	IPND	BUSY	ENI	GIE

The Half-Carry bit is also effected by all the instructions that effect the Carry flag. The flag values depend upon the instruction. For example, after executing the ADC instruction the values of the Carry and the Half-Carry flag depend upon the operands involved. However, instructions like SET C and RESET C will set and clear both the carry flags. Table XIII lists the instructions that effect the HC and the C flags.

TABLE XIII. Instructions Effecting HC and C Flags

Instr.	HC Flag	C Flag
ADC	Depends on Operands	Depends on Operands
SUBC	Depends on Operands	Depends on Operands
SET C	Set	Set
RESET C	Set	Set
RRC	Depends on Operands	Depends on Operands

CNTRL2 REGISTER (ADDRESS 00CC)

Bit 7							Bit 0
MC3	MC2	MC1	CMPEN	CMPRD	CMPOE	WDUDF	unused
R/W	R/W	R/W	R/W	R/O	R/W	R/O	

MC3 Modulator/Timer Control Bit

MC2 Modulator/Timer Control Bit

MC1 Modulator/Timer Control Bit

CMPEN Comparator Enable Bit

CMPRD Comparator Read Bit

CMPOE Comparator Output Enable Bit

WDUDF WATCHDOG Timer Underflow Bit (Read Only)

WDREG REGISTER (ADDRESS 00CD)

WDREN WATCHDOG Reset Enable Bit (Write Once Only)

Bit 7							Bit 0
UNUSED							WDREN

Memory Map

All RAM, ports and registers (except A and PC) are mapped into data memory address space.

TABLE IX. Memory Map

Address	Contents
00 to 2F	On-chip RAM bytes (48 bytes)
30 to 7F	Unused RAM Address Space (Reads as All Ones)
80 to BF	Expansion Space for On-Chip EERAM (Reads Undefined Data)
C0 to C7	Reserved
C8	MIWU Edge Select Register (Reg:WKEDG)
C9	MIWU Enable Register (Reg:WKEN)
CA	MIWU Pending Register (Reg:WKPND)
CB	Reserved
CC	Control2 Register (CNTRL2)
CD	WATCHDOG Register (WDREG)
CE	WATCHDOG Counter (WDCNT)
CF	Modulator Reload (MODRL)
D0	Port L Data Register
D1	Port L Configuration Register
D2	Port L Input Pins (Read Only)
D3	Reserved for Port L
D4	Port G Data Register
D5	Port G Configuration Register
D6	Port G Input Pins (Read Only)
D7	Port I Input Pins (Read Only)
D8 to DB	Reserved for Port C
DC	Port D Data Register
DD to DF	Reserved for Port D
E0 to EF	On-Chip Functions and Registers
E0 to E7	Reserved for Future Parts
E8	Reserved
E9	MICROWIRE Shift Register
EA	Timer Lower Byte
EB	Timer Upper Byte
EC	Timer1 Autoreload Register Lower Byte
ED	Timer1 Autoreload Register Upper Byte
EE	CNTRL1 Control Register
EF	PSW Register
F0 to FF	On-Chip RAM Mapped as Registers
FC	X Register
FD	SP Register
FE	B Register

Reading other unused memory locations will return undefined data.

Addressing Modes

There are ten addressing modes, six for operand addressing and four for transfer of control.

OPERAND ADDRESSING MODES

REGISTER INDIRECT

This is the "normal" addressing mode for the chip. The operand is the data memory addressed by the **B** or **X** pointer.

REGISTER INDIRECT WITH AUTO POST INCREMENT OR DECREMENT

This addressing mode is used with the LD and X instructions. The operand is the data memory addressed by the **B** or **X** pointer. This is a register indirect mode that automatically post increments or post decrements the **B** or **X** pointer after executing the instruction.

DIRECT

The instruction contains an 8-bit address field that directly points to the data memory for the operand.

IMMEDIATE

The instruction contains an 8-bit immediate field as the operand.

SHORT IMMEDIATE

This addressing mode issued with the LD B,# instruction, where the immediate # is less than 16. The instruction contains a 4-bit immediate field as the operand.

INDIRECT

This addressing mode is used with the LAID instruction. The contents of the accumulator are used as a partial address (lower 8 bits of PC) for accessing a data operand from the program memory.

TRANSFER OF CONTROL ADDRESSING MODES

RELATIVE

This mode is used for the JP instruction with the instruction field being added to the program counter to produce the next instruction address. JP has a range from −31 to +32 to allow a one byte relative jump (JP + 1 is implemented by a NOP instruction). There are no "blocks" or "pages" when using JP since all 15 bits of the PC are used.

ABSOLUTE

This mode is used with the JMP and JSR instructions with the instruction field of 12 bits replacing the lower 12 bits of the program counter (PC). This allows jumping to any location in the current 4k program memory segment.

ABSOLUTE LONG

This mode is used with the JMPL and JSRL instructions with the instruction field of 15 bits replacing the entire 15 bits of the program counter (PC). This allows jumping to any location in the entire 32k program memory space.

INDIRECT

This mode is used with the JID instruction. The contents of the accumulator are used as a partial address (lower 8 bits of PC) for accessing a location in the program memory. The contents of this program memory location serves as a partial address (lower 8 bits of PC) for the jump to the next instruction.

Instruction Set

REGISTER AND SYMBOL DEFINITIONS

Registers

A	8-bit Accumulator register
B	8-bit Address register
X	8-bit Address register
SP	8-bit Stack pointer register
PC	15-bit Program counter register
PU	upper 7 bits of PC
PL	lower 8 bits of PC
C	1-bit of PSW register for carry
HC	Half Carry
GIE	1-bit of PSW register for global interrupt enable

Symbols

[B]	Memory indirectly addressed by B register
[X]	Memory indirectly addressed by X register
Mem	Direct address memory or [B]
Meml	Direct address memory or [B] or Immediate data
Imm	8-bit Immediate data
Reg	Register memory: addresses F0 to FF (Includes B, X and SP)
Bit	Bit number (0 to 7)
←	Loaded with
↔	Exchanged with

Instruction Set

ADD	add	A ← A + Meml
ADC	add with carry	A ← A + Meml + C, C ← Carry HC ← Half Carry
SUBC	subtract with carry	A ← A + $\overline{\text{Meml}}$ + C, C ← Carry HC ← Half Carry
AND	Logical AND	A ← A and Meml
OR	Logical OR	A ← A or Meml
XOR	Logical Exclusive-OR	A ← A xor Meml
IFEQ	IF equal	Compare A and Meml, Do next if A = Meml
IFGT	IF greater than	Compare A and Meml, Do next if A > Meml
IFBNE	IF B not equal	Do next if lower 4 bits of B ≠ Imm
DRSZ	Decrement Reg. ,skip if zero	Reg ← Reg − 1, skip if Reg goes to 0
SBIT	Set bit	1 to bit, Mem (bit = 0 to 7 immediate)
RBIT	Reset bit	0 to bit, Mem
IFBIT	If bit	If bit, Mem is true, do next instr.
X	Exchange A with memory	A ↔ Mem
LD A	Load A with memory	A ← Meml
LD mem	Load Direct memory Immed.	Mem ← Imm
LD Reg	Load Register memory Immed.	Reg ← Imm
X	Exchange A with memory [B]	A ↔ [B] (B ← B ± 1)
X	Exchange A with memory [X]	A ↔ [X] (X ← X ± 1)
LD A	Load A with memory [B]	A ← [B] (B ← B ± 1)
LD A	Load A with memory [X]	A ← [X] (X ← X ± 1)
LD M	Load Memory Immediate	[B] ← Imm (B ← B ± 1)
CLRA	Clear A	A ← 0
INCA	Increment A	A ← A + 1
DECA	Decrement A	A ← A − 1
LAID	Load A indirect from ROM	A ← ROM(PU,A)
DCORA	DECIMAL CORRECT A	A ← BCD correction (follows ADC, SUBC)
RRCA	ROTATE A RIGHT THRU C	C → A7 → . . . → A0 → C
SWAPA	Swap nibbles of A	A7 . . . A4 ↔ A3 . . . A0
SC	Set C	C ← 1, HC ← 1
RC	Reset C	C ← 0, HC ← 0
IFC	If C	If C is true, do next instruction
IFNC	If not C	If C is not true, do next instruction
JMPL	Jump absolute long	PC ← ii (ii = 15 bits, 0 to 32k)
JMP	Jump absolute	PC11..0 ← i (i = 12 bits)
JP	Jump relative short	PC ← PC + r (r is −31 to +32, not 1)
JSRL	Jump subroutine long	[SP] ← PL,[SP-1] ← PU,SP-2,PC ← ii
JSR	Jump subroutine	[SP] ← PL,[SP-1] ← PU,SP-2,PC11..0 ← i
JID	Jump indirect	PL ← ROM(PU,A)
RET	Return from subroutine	SP+2,PL ← [SP],PU ← [SP-1]
RETSK	Return and Skip	SP+2,PL ← [SP],PU ← [SP-1],Skip next instruction
RETI	Return from Interrupt	SP+2,PL ← [SP],PU ← [SP-1],GIE ← 1
INTR	Generate an interrupt	[SP] ← PL,[SP−1] ← PU,SP-2,PC ← 0FF
NOP	No operation	PC ← PC + 1

Bits 7–4

F	E	D	C	B	A	9	8	7	6	5	4	3	2	1	0	Bits 3–0
JP -15	JP -31	LD 0F0,#i	DRSZ 0F0	RRCA	RC	ADC A, #i	ADC A, [B]	IFBIT 0,[B]	*	LD B, 0F	IFBNE 0	JSR 0000–00FF	JMP 0000–00FF	JP + 17	INTR	0
JP -14	JP -30	LD 0F1,#i	DRSZ 0F1	*	SC	SUBC A, #i	SUBC A,[B]	IFBIT 1,[B]	*	LD B, 0E	IFBNE 1	JSR 0100–01FF	JMP 0100–01FF	JP + 18	JP + 2	1
JP -13	JP -29	LD 0F2,#i	DRSZ 0F2	X A, [X+]	X A, [B+]	IFEQ A, #i	IFEQ A,[B]	IFBIT 2,[B]	*	LD B, 0D	IFBNE 2	JSR 0200–02FF	JMP 0200–02FF	JP + 19	JP + 3	2
JP -12	JP -28	LD 0F3,#i	DRSZ 0F3	X A, [X−]	X A, [B−]	IFGT A, #i	IFGT A,[B]	IFBIT 3,[B]	*	LD B, 0C	IFBNE 3	JSR 0300–03FF	JMP 0300–03FF	JP + 20	JP + 4	3
JP -11	JP -27	LD 0F4,#i	DRSZ 0F4	*	LAID	ADD A, #i	ADD A,[B]	IFBIT 4,[B]	CLRA	LD B, 0B	IFBNE 4	JSR 0400–04FF	JMP 0400–04FF	JP + 21	JP + 5	4
JP -10	JP -26	LD 0F5,#i	DRSZ 0F5	*	JID	AND A, #i	AND A,[B]	IFBIT 5,[B]	SWAPA	LD B, 0A	IFBNE 5	JSR 0500–05FF	JMP 0500–05FF	JP + 22	JP + 6	5
JP -9	JP -25	LD 0F6,#i	DRSZ 0F6	X A, [X]	X A, [B]	XOR A, #i	XOR A,[B]	IFBIT 6,[B]	DCORA	LD B, 9	IFBNE 6	JSR 0600–06FF	JMP 0600–06FF	JP + 23	JP + 7	6
JP -8	JP -24	LD 0F7,#i	DRSZ 0F7	*	*	OR A, #i	OR A,[B]	IFBIT 7,[B]	*	LD B, 8	IFBNE 7	JSR 0700–07FF	JMP 0700–07FF	JP + 24	JP + 8	7
JP -7	JP -23	LD 0F8,#i	DRSZ 0F8	NOP	*	LD A, #i	IFC	SBIT 0,[B]	RBIT 0,[B]	LD B, 7	IFBNE 8	JSR 0800–08FF	JMP 0800–08FF	JP + 25	JP + 9	8
JP -6	JP -22	LD 0F9,#i	DRSZ 0F9	*	*	*	IFNC	SBIT 1,[B]	RBIT 1,[B]	LD B, 6	IFBNE 9	JSR 0900–09FF	JMP 0900–09FF	JP + 26	JP + 10	9
JP -5	JP -21	LD 0FA,#i	DRSZ 0FA	LD A, [X+]	LD A, [B+]	LD [B+],#i	INCA	SBIT 2,[B]	RBIT 2,[B]	LD B, 5	IFBNE 0A	JSR 0A00–0AFF	JMP 0A00–0AFF	JP + 27	JP + 11	A
JP -4	JP -20	LD 0FB,#i	DRSZ 0FB	LD A, [X−]	LD A, [B−]	LD [B−],#i	DECA	SBIT 3,[B]	RBIT 3,[B]	LD B, 4	IFBNE 0B	JSR 0B00–0BFF	JMP 0B00–0BFF	JP + 28	JP + 12	B
JP -3	JP -19	LD 0FC,#i	DRSZ 0FC	LD Md, #i	JMPL	X A,Md	*	SBIT 4,[B]	RBIT 4,[B]	LD B, 3	IFBNE 0C	JSR 0C00–0CFF	JMP 0C00–0CFF	JP + 29	JP +13	C
JP -2	JP -18	LD 0FD,#i	DRSZ 0FD	DIR	JSRL	LD A, Md	RETSK	SBIT 5,[B]	RBIT 5,[B]	LD B, 2	IFBNE 0D	JSR 0D00–0DFF	JMP 0D00–0DFF	JP + 30	JP +14	D
JP -1	JP -17	LD 0FE,#i	DRSZ 0FE	LD A, [X]	LD A, [B]	LD [B], #i	RET	SBIT 6, [B]	RBIT 6, [B]	LD B, 1	IFBNE 0E	JSR 0E00–0EFF	JMP 0E00–0EFF	JP + 31	JP +15	E
JP -0	JP -16	LD 0FF,#1	DRSZ 0FF	*	*	*	RETI	SBIT 7,[B]	RBIT 7,[B]	LD B, 0	IFBNE 0F	JSR 0F00–0FFF	JMP 0F00–0FFF	JP + 32	JP + 16	F

where, i is the immediate data Md is a directly addressed memory location * is an unused opcode (see following table)

Instruction Execution Time

Most instructions are single byte (with immediate addressing mode instruction taking two bytes).

Most single instructions take one cycle time to execute.

Skipped instructions require x number of cycles to be skipped, where x equals the number of bytes in the skipped instruction opcode.

See the BYTES and CYCLES per INSTRUCTION table for details.

Bytes and Cycles per Instruction

The following table shows the number of bytes and cycles for each instruction in the format of byte/cycle.

Arithmetic Instructions (Bytes/Cycles)

	[B]	Direct	Immed.
ADD	1/1	3/4	2/2
ADC	1/1	3/4	2/2
SUBC	1/1	3/4	2/2
AND	1/1	3/4	2/2
OR	1/1	3/4	2/2
XOR	1/1	3/4	2/2
IFEQ	1/1	3/4	2/2
IFGT	1/1	3/4	2/2
IFBNE	1/1		
DRSZ		1/3	
SBIT	1/1	3/4	
RBIT	1/1	3/4	
IFBIT	1/1	3/4	

Memory Transfer Instructions (Bytes/Cycles)

	Register Indirect		Direct	Immed.	Register Indirect Auto Incr & Decr		
	[B]	[X]			[B+, B−]	[X+, X−]	
X A,*	1/1	1/3	2/3		1/2	1/3	
LD A,*	1/1	1/3	2/3	2/2	1/2	1/3	
LD B,Imm				1/1			(If B < 16)
LD B,Imm				2/3			(If B > 15)
LD Mem,Imm			3/3		2/2		
LD Reg,Imm				2/3			

* => Memory location addressed by B or X or directly.

Instructions Using A & C

Instructions	Bytes/Cycles
CLRA	1/1
INCA	1/1
DECA	1/1
LAID	1/3
DCORA	1/1
RRCA	1/1
SWAPA	1/1
SC	1/1
RC	1/1
IFC	1/1
IFNC	1/1

Transfer of Control Instructions

Instructions	Bytes/Cycles
JMPL	3/4
JMP	2/3
JP	1/3
JSRL	3/5
JSR	2/5
JID	1/3
RET	1/5
RETSK	1/5
RETI	1/5
INTR	1/7
NOP	1/1

Bytes and Cycles per Instruction (Continued)

The following table shows the instructions assigned to unused opcodes. This table is for information only. The operations performed are subject to change without notice. Do not use these opcodes.

Unused Opcode	Instruction	Unused Opcode	Instruction
60	NOP	A9	NOP
61	NOP	AF	LD A, [B]
62	NOP	B1	C → HC
63	NOP	B4	NOP
67	NOP	B5	NOP
8C	RET	B7	X A, [X]
99	NOP	B9	NOP
9F	LD [B], #i	BF	LD A, [X]
A7	X A, [B]		
A8	NOP		

Option List

The mask programmable options are listed below. The options are programmed at the same time as the ROM pattern to provide the user with hardware flexibility to a variety of oscillation and packaging configuration.

OPTION 1: CKI INPUT

= 1 Crystal (CKI/IO) CKO for crystal configuration
= 2 External (CKI/IO) CKO available as G7 input
= 3 R/C (CKI/IO) CKO available as G7 input

OPTION 2: BROWN OUT

= 1 Enable Brown Out Detection
= 2 Disable Brown Out Detection

OPTION 3: BONDING

= 1 28-pin DIP
= 2 20-pin DIP/SO
= 3 16-pin SO
= 4 28-pin SO

Development Support

SUMMARY

- iceMASTER™: IM-COP8/400—Full feature in-circuit emulation for all COP8 products. A full set of COP8 Basic and Feature Family device and package specific probes are available.
- COP8 Debug Module: Moderate cost in-circuit emulation and development programming unit.
- COP8 Evaluation and Programming Unit: EPU-COP8780—low cost in-circuit simulation and development programming unit.
- Assembler: COP8-DEV-IBMA. A DOS installable cross development Assembler, Linker, Librarian and Utility Software Development Tool Kit.
- C Compiler: COP8C. A DOS installable cross development Software Tool Kit.
- OTP/EPROM Programmer Support: Covering needs from engineering prototype, pilot production to full production environments.

Development Support (Continued)

IceMASTER (IM) IN-CIRCUIT EMULATION

The iceMASTER IM-COP8/400 is a full feature, PC based, in-circuit emulation tool developed and marketed by MetaLink Corporation to support the whole COP8 family of products. National is a resale vendor for these products.

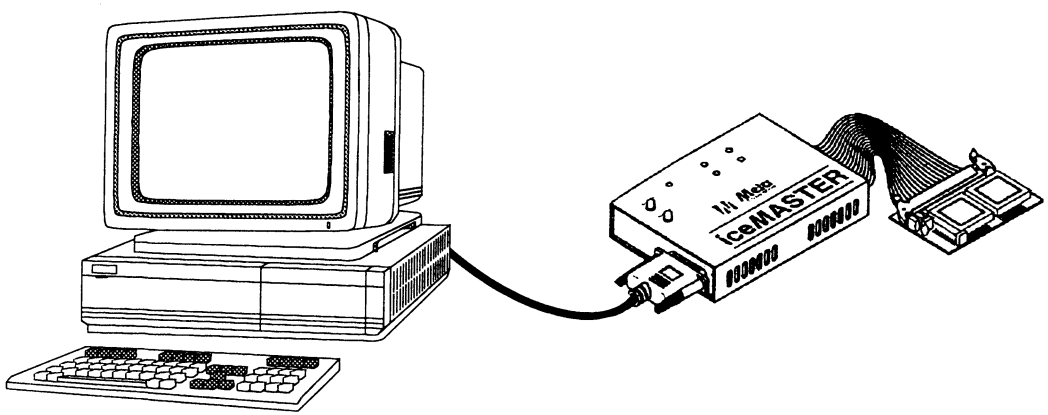
See *Figure 18* for configuration.

The iceMASTER IM-COP8/400 with its device specific COP8 Probe provides a rich feature set for developing, testing and maintaining product:

- Real-time in-circuit emulation; full 2.4V–5.5V operation range, full DC-10 MHz clock. Chip options are programmable or jumper selectable.
- Direct connection to application board by package compatible socket or surface assembly.
- Full 32 kbytes of loadable programming space that overlays (replaces) the on-chip ROM or EPROM. On-chip RAM and I/O blocks are used directly or recreated on the probe as necessary.
- Full 4k frame synchronous trace memory. Address, instruction, and 8 unspecified, circuit connectable trace lines. Display can be HLL source (e.g., C source), assembly or mixed.
- A full 64k hardware configurable break, trace on, trace off control, and pass count increment events.
- Tool set integrated interactive symbolic debugger—supports both assembler (COFF) and C Compiler (.COD) linked object formats.
- Real time performance profiling analysis; selectable bucket definition.
- Watch windows, content updated automatically at each execution break.
- Instruction by instruction memory/register changes displayed on source window when in single step operation.
- Single base unit and debugger software reconfigurable to support the entire COP8 family; only the probe personality needs to change. Debugger software is processor customized, and reconfigured from a master model file.
- Processor specific symbolic display of registers and bit level assignments, configured from master model file.
- Halt/Idle mode notification.
- On-Line HELP customized to specific processor using master model file.
- Includes a copy of COP8-DEV-IBMA assembler and linker SDK.

IM Order Information

Base Unit	
IM-COP8/400-1	iceMASTER Base Unit, 110V Power Supply
IM-COP8/400-2	iceMASTER Base Unit, 220V Power Supply
iceMASTER Probe	
MHW-840CJ20DWPC MHW-840CJ28DWPC	20 DIP 28 DIP
Adapters for SO Packages	
MHW-SOIC16	16 SO
MHW-SOIC20	20 SO
MHW-SOIC28	28 SO



TL/DD/11208–30

FIGURE 18. COP8 iceMASTER Environment

Development Support (Continued)

iceMASTER DEBUG MODULE (DM)

The iceMASTER Debug Module is a PC based, combination in-circuit emulation tool and COP8 based OTP/EPROM programming tool developed and marketed by MetaLink Corporation to support the whole COP8 family of products. National is a resale vendor for these products.

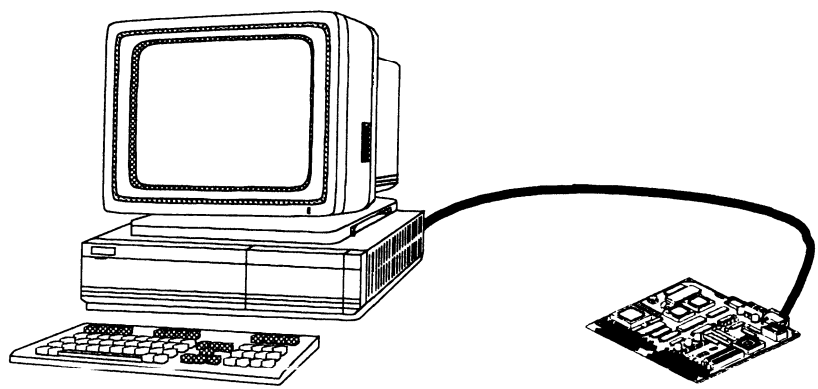
See *Figure 19* for configuration.

The iceMASTER Debug Module is a moderate cost development tool. It has the capability of in-circuit emulation for a specific COP8 microcontroller and in addition serves as a programming tool for COP8 OTP and EPROM product families. Summary of features is as follows:

- Real-time in-circuit emulation; full operating voltage range operation, full DC-10 MHz clock.
- All processor I/O pins can be cabled to an application development board with package compatible cable to socket and surface mount assembly.
- Full 32 kbytes of loadable programming space that overlays (replaces) the on-chip ROM or EPROM. On-chip RAM and I/O blocks are used directly or recreated as necessary.
- 100 frames of synchronous trace memory. The display can be HLL source (C source), assembly or mixed. The most recent history prior to a break is available in the trace memory.
- Configured break points; uses INTR instruction which is modestly intrusive.
- Software-only supported features are selectable.
- Tool set integrated interactive symbolic debugger—supports both assembler (COFF) and C Compiler (.COD) SDK linked object formats.
- Instruction by instruction memory/register changes displayed when in single step operation.
- Debugger software is processor customized, and reconfigured from a master model file.
- Processor specific symbolic display of registers and bit level assignments, configured from master model file.
- Halt/Idle mode notification.
- Programming menu supports full product line of programmable OTP and EPROM COP8 products. Program data is taken directly from the overlay RAM.
- Programming of 44 PLCC and 68 PLCC parts requires external programming adapters.
- Includes wallmount power supply.
- On-board V_{PP} generator from 5V input or connection to external supply supported. Requires V_{PP} level adjustment per the family programming specification (correct level is provided on an on-screen pop-down display).
- On-line HELP customized to specific processor using master model file.
- Includes a copy of COP8-DEV-IBMA assembler and linker SDK.

DM Order Information

Debug Module Unit	
COP8-DM/840CJ	
Cable Adapters	
DM-COP8/20D	20 DIP
DM-COP8/28D	28 DIP
Adapters for SO Package	
MHW-SOIC16	16 SO
MHW-SOIC20	20 SO
MHW-SOIC28	28 SO



TL/DD/11208–31

FIGURE 19. COP8-DM Environment

Development Support (Continued)

COP8 ASSEMBLER/LINKER SOFTWARE DEVELOPMENT TOOL KIT

National Semiconductor offers a relocatable COP8 macro cross assembler, linker, librarian and utility software development tool kit. Features are summarized as follows:

- Basic and Feature Family instruction set by "device" type.
- Nested macro capability.
- Extensive set of assembler directives.
- Supported on PC/DOS platform.
- Generates National standard COFF output files.
- Integrated Linker and Librarian.
- Integrated utilities to generate ROM code file outputs.
- DUMPCOFF utility.

This product is integrated as a part of MetaLink tools as a development kit, fully supported by the MetaLink debugger. It may be ordered separately or it is bundled with the MetaLink products at no additional cost.

Order Information

Assembler SDK	
COP8-DEV-IBMA	Assembler SDK on installable 3.5" PC/DOS Floppy Disk Drive format. Periodic upgrades and most recent version is available on National's BBS and Internet.

COP8 C COMPILER

A C Compiler is developed and marketed by Byte Craft Limited. The COP8C compiler is a fully integrated development tool specifically designed to support the compact embedded configuration of the COP8 family of products.

Features are summarized as follows:

- ANSI C with some restrictions and extensions that optimize development for the COP8 embedded application.
- BITS data type extension. Register declaration #pragma with direct bit level definitions.
- C language support for interrupt routines.
- Expert system, rule based code generation and optimization.
- Performs consistency checks against the architectural definitions of the target COP8 device.
- Generates program memory code.
- Supports linking of compiled object or COP8 assembled object formats.
- Global optimization of linked code.
- Symbolic debug load format fully source level supported by the MetaLink debugger.

Approved List

Manufacturer	North America	Europe	Asia
BP Microsystems	(800) 225-2102 (713) 688-4600 Fax: (713) 688-0920	+49-8152-4183 +49-8856-932616	+852-234-16611 +852-2710-8121
Data I/O	(800) 426-1045 (206) 881-6444 Fax: (206) 882-1043	+44-0734-440011	Call North America
HI-LO	(510) 623-8860	Call Asia	+886-2-764-0215 Fax: +886-2-756-6403
ICE Technology	(800) 624-8949 (919) 430-7915	+44-1226-767404 Fax: 0-1226-370-434	
MetaLink	(800) 638-2423 (602) 926-0797 Fax: (602) 693-0681	+49-80 9156 96-0 Fax: +49-80 9123 86	+852-737-1800
Systems General	(408) 263-6667	+41-1-9450300	+886-2-917-3005 Fax: +886-2-911-1283
Needhams	(916) 924-8037 Fax: (916) 924-8065		

Development Support (Continued)

SINGLE CHIP OTP/EMULATOR SUPPORT

The COP8 family is supported by single chip OTP emulators. For detailed information refer to the emulator specific datasheet and the emulator selection table below:

OTP Emulator Ordering Information

Device Number	Clock Option	Package	Emulates
COP87L22CJN-1N	Crystal	20 DIP	COP822CJ
COP87L22CJN-2N	External	20 DIP	COP822CJ
COP87L22CJN-3N	R/C	20 DIP	COP822CJ
COP87L22CJM-1N	Crystal	20 SO	COP822CJ
COP87L22CJM-2N	External	20 SO	COP822CJ
COP87L22CJM-3N	R/C	20 SO	COP822CJ
COP87L20CJN-1N	Crystal	28 DIP	COP820CJ
COP87L20CJN-2N	External	28 DIP	COP820CJ
COP87L20CJN-3N	R/C	28 DIP	COP820CJ
COP87L20CJM-1N	Crystal	28 SO	COP820CJ
COP87L20CJM-2N	External	28 SO	COP820CJ
COP87L20CJM-3N	R/C	28 SO	COP820CJ

INDUSTRY WIDE OTP/EPROM PROGRAMMING SUPPORT

Programming support, in addition to the MetaLink development tools, is provided by a full range of independent approved vendors to meet the needs from the engineering laboratory to full production.

AVAILABLE LITERATURE

For more information, please see the COP8 Basic Family User's Manual, Literature Number 620895, COP8 Feature Family User's Manual, Literature Number 620897 and National's Family of 8-bit Microcontrollers COP8 Selection Guide, Literature Number 630009.

DIAL-A-HELPER SERVICE

Dial-A-Helper is a service provided by the Microcontroller Applications group. The Dial-A-Helper is an Electronic Information System that may be accessed as a Bulletin Board System (BBS) via data modem, as an FTP site on the Internet via standard FTP client application or as an FTP site on the Internet using a standard Internet browser such as Netscape or Mosaic.

The Dial-A-Helper system provides access to an automated information storage and retrieval system. The system capabilities include a MESSAGE SECTION (electronic mail, when accessed as a BBS) for communications to and from the Microcontroller Applications Group and a FILE SECTION which consists of several file areas where valuable application software and utilities could be found.

DIAL-A-HELPER BBS via a Standard Modem

Modem: CANADA/U.S.: (800) NSC-MICRO
(800) 672-6427
EUROPE: (+49) 0-8141-351332
Baud: 14.4k
Set-up: Length: 8-Bit
Parity: None
Stop Bit: 1
Operation: 24 Hours, 7 Days

DIAL-A-HELPER via FTP

ftp nscmicro.nsc.com
user: anonymous
password: username@yourhost.site.domain

DIAL-A-HELPER via a WorldWide Web Browser

ftp://nscmicro.nsc.com

National Semiconductor on the WorldWide Web

See us on the WorldWide Web at: http://www.national.com

CUSTOMER RESPONSE CENTER

Complete product information and technical support is available from National's customer response centers.

CANADA/U.S.:	Tel:	(800) 272-9959
	email:	support @tevm2.nsc.com
EUROPE:	email:	europe.support@nsc.com
	Deutsch Tel:	+49 (0) 180-530 85 85
	English Tel:	+49 (0) 180-532 78 32
	Français Tel:	+49 (0) 180-532 93 58
	Italiano Tel:	+49 (0) 180-534 16 80
JAPAN:	Tel:	+81-043-299-2309
S.E. ASIA:	Beijing Tel:	(+86) 10-6856-8601
	Shanghai Tel:	(+86) 21-6415-4092
	Hong Kong Tel:	(+852) 2737-1600
	Korea Tel:	(+82) 2-3771-6909
	Malaysia Tel:	(+60-4) 644-9061
	Singapore Tel:	(+65) 255-2226
	Taiwan Tel:	+886-2-521-3288
AUSTRALIA:	Tel:	(+61) 3-9558-9999
INDIA:	Tel:	(+91) 80-559-9467

National Semiconductor

COP840CJ/COP842CJ/COP940CJ/COP942CJ 8-Bit Microcontrollers with Multi-Input Wake-Up and Brown Out Detector

General Description

The COP840CJ is a member of the COP8™ feature family 8-bit microcontroller. It is a fully static Microcontroller, fabricated using double-metal silicon gate microCMOS technology. This low cost microcontroller is a complete microcomputer containing all system timing, interrupt logic, ROM, RAM, and I/O necessary to implement dedicated control functions in a variety of applications. Features include an 8-bit memory mapped architecture, MICROWIRE™ serial I/O, a 16-bit timer/counter with capture register, a multi-sourced interrupt, Comparator, WATCHDOG™ Timer, Modulator/Timer, Brown out detection and Multi Input Wake-up. Each I/O pin has software selectable options to adapt the device to the specific application. The device operates over a voltage range of 2.5V–6.0V. High throughput is achieved with an efficient, regular instruction set operating at a 1 ms per instruction rate. Low radiated emissions are achieved by gradual turn on output drivers and internal I_{CC} filters on the chip logic and crystal oscillator.

Key Features

- Multi-Input wake-up (on the 8-bit Port L)
- Brown out detection
- Analog comparator
- Modulator/Timer (high speed PWM timer for IR transmission)
- 16-bit multi-function timer supporting
  - — PWM mode
  - — External event counter mode
  - — Input capture mode
- Quiet design (low radiated emissions)
- Integrated capacitor for the R/C oscillator
- 2048 bytes of ROM
- 128 bytes of RAM

I/O Features

- Memory mapped I/O
- Software selectable I/O options (TRI-STATE®, push-pull, weak pull-up input, high impedance input)
- High current outputs (8 pins)
- Schmitt trigger inputs on Port G
- MICROWIRE/PLUS™ serial I/O
- Packages: 20 DIP/SO with 16 I/O pins
  28 DIP/SO with 24 I/O pins

CPU/Instruction Set Features

- 1 μs instruction cycle time
- Three multi-source interrupts servicing
  - — External interrupt with selectable edge
  - — Timer interrupt
  - — Software interrupt
- Versatile and easy to use instruction set
- 8-bit Stack Pointer (SP)—stack in RAM
- Two 8-bit register indirect data memory pointers (B and X)

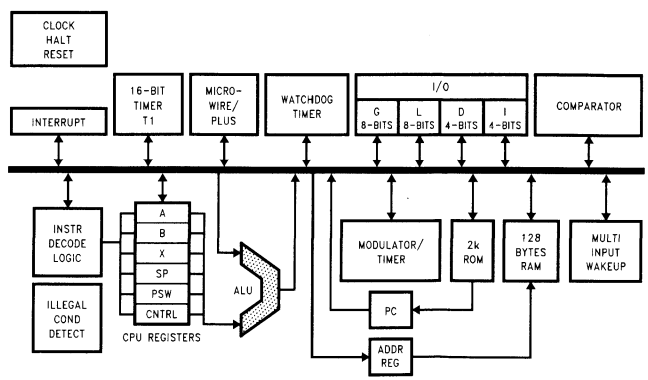
Fully Static CMOS

- Low current drain (typically < 1 μA)
- Single supply operation: 2.5V to 6.0V
- Temperature ranges: 0°C to +70°C, −40°C to +85°C
  −55°C to +125°C

Development Support

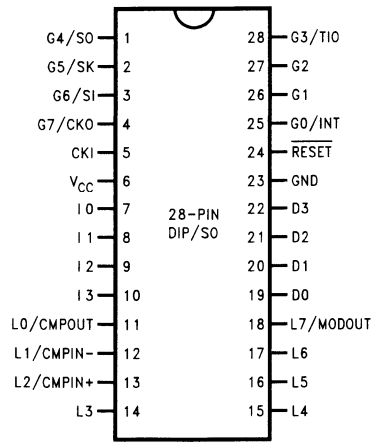
- Emulation and OTP devices
- Real time emulation and full program debug offered by MetaLink's development system

Block Diagram



TL/DD/12851–1

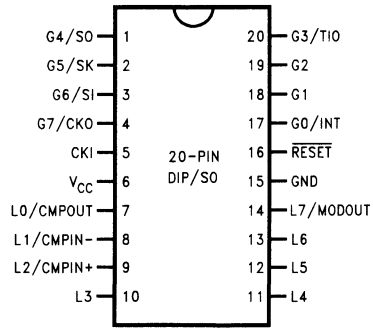
Connection Diagrams



TL/DD/12851-2

Top View

Order Number COP840CJ-XXX/N or COP840CJ-XXX/M, COP940CJ-XXX/N or COP940CJ-XXX/M
See NS Package Number N28B or M28B



TL/DD/12851-3

Top View

Order Number COP842CJ-XXX/N or COP842CJ-XXX/M COP942CJ-XXX/N or COP942CJ-XXX/M
See NS Package Number N20A or M20B

FIGURE 1. Connection Diagrams

Pin Assignment

Port PIN	Type	ALT Funct.	20-Pin	28-Pin
L0	I/O	MIWU/ CMPOUT	7	11
L1	I/O	MIWU/ CMPIN−	8	12
L2	I/O	MIWU/ CMPIN+	9	13
L3	I/O	MIWU	10	14
L4	I/O	MIWU	11	15
L5	I/O	MIWU	12	16
L6	I/O	MIWU	13	17
L7	I/O	MIWU/ MODOUT	14	18
G0	I/O	INTR	17	25
G1	I/O		18	26
G2	I/O		19	27
G3	I/O	TIO	20	28
G4	I/O	SO	1	1
G5	I/O	SK	2	2
G6	I	SI	3	3
G7	I	CKO	4	4
I0	I			7
I1	I			8
I2	I			9
I3	I			10
D0	O			19
D1	O			20
D2	O			21
D3	O			22
V_{CC}			6	6
GND			15	23
CKI			5	5
$\overline{\text{RESET}}$			16	24

Absolute Maximum Ratings

If Military/Aerospace specified devices are required, please contact the National Semiconductor Sales Office/Distributors for availability and specifications.

Supply Voltage (V_{CC})	7.0V
Voltage at Any Pin	−0.3V to V_{CC} +0.3V
Total Current into V_{CC} Pin (source)	80 mA
Total Current out of GND Pin (sink)	80 mA
Storage Temperature Range	−65°C to +150°C

Note: *Absolute maximum ratings indicate limits beyond which damage to the device may occur. DC and AC electrical specifications are not ensured when operating the device at absolute maximum ratings.*

DC Electrical Characteristics COP94x 0°C ≤ T_A ≤ +70°C unless otherwise specified

Parameter	Condition	Min	Typ	Max	Units
Operating Voltage					
COP94xCJ	Brown Out Disabled	2.5		4.5	V
COP94xCJH		4.5		6.0	V
Power Supply Ripple (Note 1)	Peak-to-Peak			0.1 V_{CC}	V
Supply Current (Notes 2, 5)					
CKI = 10 MHz, R = 2.2k	V_{CC} = 6V, t_C = 1 μs			8.0	mA
CKI = 4 MHz, R = 4.7k	V_{CC} = 6V, t_C = 2.5 μs			6.0	mA
CKI = 4 MHz, R = 4.7k	V_{CC} = 4.5V, t_C = 2.5 μs			2.5	mA
CKI = 1 MHz, R = 20k	V_{CC} = 4.5V, t_C = 10 μs			1.5	mA
HALT Current with Brown Out Disabled (Note 3)	V_{CC} = 6V, CKI = 0 MHz		< 2.2	8	μA
HALT Current with Brown Out Enabled	V_{CC} = 6V, CKI = 0 MHz		< 50	100	μA
Brown Out Trip Level (Brown Out Enabled)		1.9	3.1	3.9	V
INPUT LEVELS (V_{IH}, V_{IL})					
Reset, CKI:					
Logic High		0.8 V_{CC}			V
Logic Low				0.2 V_{CC}	V
All Other Inputs					
Logic High		0.7 V_{CC}			V
Logic Low				0.2 V_{CC}	V
Hi-Z Input Leakage	V_{CC} = 6.0V	−2		+2	μA
Input Pullup Current	V_{CC} = 6.0V	−40		−250	μA
L- and G-Port Hysteresis			0.05 V_{CC}	0.35 V_{CC}	V
Output Current Levels					
D Outputs:					
Source	V_{CC} = 4.5V, V_{OH} = 3.8V	−0.4			mA
	V_{CC} = 2.5V, V_{OH} = 1.8V	−0.2			mA
Sink	V_{CC} = 4.5V, V_{OL} = 1.0V	10			mA
	V_{CC} = 2.5V, V_{OH} = 0.4V	2			mA
L4–L7 Output Sink	V_{CC} = 4.5V, V_{OL} = 2.5V	15			mA
All others					
Source (Weak Pull-up Mode)	V_{CC} = 4.5V, V_{OH} = 3.2V	−10		−110	μA
	V_{CC} = 2.5V, V_{OH} = 1.8V	−2.5		−33	μA
Sink (Push-Pull Mode)	V_{CC} = 4.5V, V_{OL} = 0.4V	1.6			mA
	V_{CC} = 2.5V, V_{OL} = 0.4V	0.7			mA
TRI-STATE Leakage		−2.0		+2.0	μA

DC Electrical Characteristics COP94x $0°C \leq T_A \leq +70°C$ unless otherwise specified (Continued)

Parameter	Condition	Min	Typ	Max	Units
Allowable Sink/Source Current per Pin					
D Outputs				15	mA
L4–L7 (sink)				20	mA
All Others				3	mA
Maximum Input Current without Latchup (Note 4)				±100	mA
RAM Retention Voltage, Vr	500 ns Rise and Fall Time (min)	2.0			V
Input Capacitance				7	pF
Load Capacitance on D2				1000	pF

Note 1: Rate of voltage change must be < 10V/ms.

Note 2: Supply current is measured after running 2000 cycles with a square wave CKI input, CKO open, inputs at rails and outputs open.

Note 3: The HALT mode will stop CKI from oscillating in the RC and crystal configurations. HALT test conditions: L, and G0..G5 ports configured as outputs and set high. The D port set to zero. All inputs tied to V_{CC}. The comparator and the Brown Out circuits are disabled.

Note 4: Pins G6 and $\overline{\text{RESET}}$ are designed with a high voltage input network. These pins allow input voltages greater than V_{CC} and the pins will have sink current to V_{CC} when biased at voltages greater than V_{CC} (the pins do not have source current when biased at a voltage below V_{CC}). The effective resistance to V_{CC} is 750Ω (typical). These two pins will not latch up. The voltage at the pins must be limited to < 14V.

Note 5: The Resistor values are for R/C only.

AC Electrical Characteristics COP94x $0°C \leq T_A \leq +70°C$ unless otherwise specified

Parameter	Condition	Min	Typ	Max	Units
Instruction Cycle Time (t_C)					
Crystal/Resonator	$4.5V \leq V_{CC} < 6.0V$	1		DC	μs
	$2.5V \leq V_{CC} < 4.5V$	2.5		DC	μs
R/C Oscillator	$4.5V \leq V_{CC} < 6.0V$	2		DC	μs
	$2.5V \leq V_{CC} < 4.5V$	5		DC	μs
V_{CC} Rise Time when Using Brown Out		50			μs
Frequency at Brown Out Reset				4	MHz
CKI Frequency for Modulator Output				4	MHz
CKI Clock Duty Cycle (Note 1)	fr = Max	40		60	%
Rise Time (Note 1)	fr = 10 MHz ext. clock			12	ns
Fall Time (Note 1)	fr = 10 MHz ext. clock			8	ns
Inputs					
t_{Setup}	$4.5V \leq V_{CC} < 6.0$	200			ns
	$2.5V \leq V_{CC} < 4.5$	500			ns
tHold	$4.5V \leq V_{CC} < 6.0$	60			ns
	$2.5V \leq V_{CC} < 4.5$	150			ns
Output Propagation Delay	RL = 2.2k, CL = 100 pF				
t_{PD1}, t_{PD0}					
SO, SK	$4.5V \leq V_{CC} < 6.0$			0.7	μs
	$2.5V \leq V_{CC} < 4.5$			1.75	μs
All Others	$4.5V \leq V_{CC} < 6.0$			1	μs
	$2.5V \leq V_{CC} < 4.5$			5	μs
Input Pulse Width		1			t_C
Interrupt Input High Time		1			t_C
Interrupt Input Low Time		1			t_C
Timer Input High Time		1			t_C
Timer Input Low Time		1			t_C
MICROWIRE Setup Time ($t_{\mu WS}$)		20			ns
MICROWIRE Hold Time ($t_{\mu WH}$)		56			ns
MICROWIRE Output Propagation Delay ($t_{\mu PD}$)				220	ns
Reset Pulse Width		1.0			μs

Note 1: Parameter sampled but not 100% tested.

Absolute Maximum Ratings

If Military/Aerospace specified devices are required, please contact the National Semiconductor Sales Office/Distributors for availability and specifications.

Supply Voltage (V_{CC})	7.0V
Voltage at any Pin	−0.3V to V_{CC} +0.3V
Total current into V_{CC} pin (source)	80 mA
Total current out of GND pin (sink)	80 mA
Storage Temperature Range	−65°C to +150°C

Note: *Absolute maximum ratings indicate limits beyond which damage to the device may occur. DC and AC electrical specifications are not ensured when operating the device at absolute maximum ratings.*

DC Electrical Characteristics COP84x $-40°C \le T_A \le +85°C$ unless otherwise specified

Parameter	Condition	Min	Typ	Max	Units
Operating Voltage	Brown Out disabled	2.5		6.0	V
Power Supply Ripple (Note 1)	Peak-to-Peak			0.1 V_{CC}	V
Supply Current (Notes 2, 5)					
CKI = 10 MHz, R = 2.2k	V_{CC} = 6V, t_C = 1 μs			8.0	mA
CKI = 4 MHz, R = 4.7k	V_{CC} = 6V, t_C = 2.5 μs			6.0	mA
CKI = 4 MHz, R = 4.7k	V_{CC} = 4.5V, t_C = 2.5 μs			2.5	mA
CKI = 1 MHz, R = 20k	V_{CC} = 4.5V, t_C = 10 μs			1.5	mA
HALT Current with Brown Out Disabled (Note 3)	V_{CC} = 6V, CKI = 0 MHz		< 2.5	10	μA
HALT Current with Brown Out Enabled	V_{CC} = 6V, CKI = 0 MHz		< 50	100	μA
Brown Out Trip Level (Brown Out Enabled)		1.8	3.1	4.2	V
INPUT LEVELS (V_{IH}, V_{IL})					
Reset, CKI:					
Logic High		0.8 V_{CC}			V
Logic Low				0.2 V_{CC}	V
All Other Inputs					
Logic High		0.7 V_{CC}			V
Logic Low				0.2 V_{CC}	V
Hi-Z Input Leakage	V_{CC} = 6.0V	−2		+2	μA
Input Pullup Current	V_{CC} = 6.0V	−40		−250	μA
L- and G-Port Hysteresis			0.05 V_{CC}	0.35 V_{CC}	V
Output Current Levels					
D Outputs:					
Source	V_{CC} = 4.5V, V_{OH} = 3.8V	−0.4			mA
	V_{CC} = 2.5V, V_{OH} = 1.8V	−0.2			mA
Sink	V_{CC} = 4.5V, V_{OL} = 1.0V	10			mA
	V_{CC} = 2.5V, V_{OH} = 0.4V	2			mA
L4–L7 Output Sink	V_{CC} = 4.5V, V_{OL} = 2.5V	15			mA
All Others					
Source (Weak Pull-Up Mode)	V_{CC} = 4.5V, V_{OH} = 3.2V	−10		−110	μA
	V_{CC} = 2.5V, V_{OH} = 1.8V	−2.5		−33	μA
Sink (Push-Pull Mode)	V_{CC} = 4.5V, V_{OL} = 0.4V	1.6			mA
	V_{CC} = 2.5V, V_{OL} = 0.4V	0.7			mA
TRI-STATE Leakage		−2.0		+2.0	μA

DC Electrical Characteristics COP84x $-40°C \leq T_A \leq +85°C$ unless otherwise specified (Continued)

Parameter	Condition	Min	Typ	Max	Units
Allowable Sink/Source Current per Pin					
D Outputs				15	mA
L4–L7 (sink)				20	mA
All Others				3	mA
Maximum Input Current without Latchup (Note 4)				±100	mA
RAM Retention Voltage, Vr	500 ns Rise and Fall Time (min)	2.0			V
Input Capacitance				7	pF
Load Capacitance on D2				1000	pF

Note 1: Rate of voltage change must be < 10V/ms.

Note 2: Supply current is measured after running 2000 cycles with a square wave CKI input, CKO open, inputs at rails and outputs open.

Note 3: The HALT mode will stop CKI from oscillating in the RC and crystal configurations. HALT test conditions: L, and G0..G5 ports configured as outputs and set high. The D port set to zero. All inputs tied to V_{CC}. The comparator and the Brown Out circuits are disabled.

Note 4: Pins G6 and $\overline{\text{RESET}}$ are designed with a high voltage input network. These pins allow input voltages greater than V_{CC} and the pins will have sink current to V_{CC} when biased at voltages greater than V_{CC} (the pins do not have source current when biased at a voltage below V_{CC}). The effective resistance to V_{CC} is 750Ω (typical). These two pins will not latch up. The voltage at the pins must be limited to < 14V.

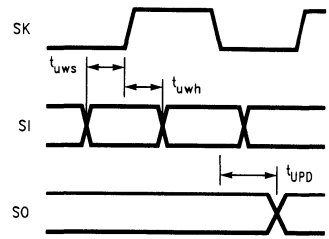
Note 5: The Resistor values are for R/C only.

AC Electrical Characteristics COP84x $-40°C \leq T_A \leq +85°C$ unless otherwise specified.

Parameter	Condition	Min	Typ	Max	Units
Instruction Cycle Time (t_C)					
Crystal/Resonator	$4.5V \leq V_{CC} < 6.0V$	1		DC	μs
	$2.5V \leq V_{CC} < 4.5V$	2.5		DC	μs
R/C Oscillator	$4.5V < V_{CC} < 6.0V$	2		DC	μs
	$2.5V \leq V_{CC} < 4.5V$	5		DC	μs
V_{CC} Rise Time when Using Brown Out		50			μs
Frequency at Brown Out Reset				4	MHz
CKI Frequency for Modulator Output				4	MHz
CKI Clock Duty Cycle (Note 1)	fr = Max	40		60	%
Rise Time (Note 1)	fr = 10 MHz ext. clock			12	ns
Fall Time (Note 1)	fr = 10 MHz ext. clock			8	ns
Inputs					
t_{Setup}	$4.5V \leq V_{CC} < 6.0V$	200			ns
	$2.5V \leq V_{CC} < 4.5V$	500			ns
t_{Hold}	$4.5V \leq V_{CC} < 6.0V$	60			ns
	$2.5V \leq V_{CC} < 4.5V$	150			ns
Output Propagation Delay	$R_L = 2.2k, C_L = 100$ pF				
t_{PD1}, t_{PD0}					
SO, SK	$4.5V \leq V_{CC} < 6.0V$			0.7	μs
	$2.5V \leq V_{CC} < 4.5V$			1.75	μs
All Others	$4.5V \leq V_{CC} < 6.0V$			1	μs
	$2.5V \leq V_{CC} < 4.5V$			5	μs
Input Pulse Width		1			t_C
Interrupt Input High Time		1			t_C
Interrupt Input Low Time		1			t_C
Timer Input High Time		1			t_C
Timer input low time		1			t_C
MICROWIRE Setup Time ($t_{\mu WS}$)		20			ns
MICROWIRE Hold Time ($t_{\mu WH}$)		56			ns
MICROWIRE Output Propagation Delay ($t_{\mu PD}$)				220	ns
Reset Pulse Width		1.0			μs

Note 1: Parameter sampled but not 100% tested.

AC Electrical Characteristics COP84X (Continued)

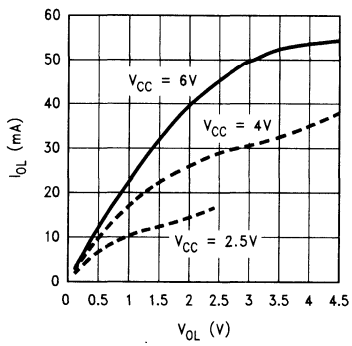


TL/DD/12851-4

FIGURE 1. MICROWIRE/PLUS Timing

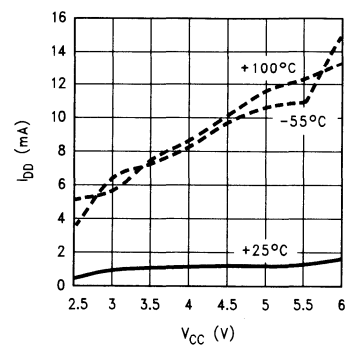
Typical Performance Characteristics ($-40°C \leq T_A \leq +85°C$)

Port D Sink Current



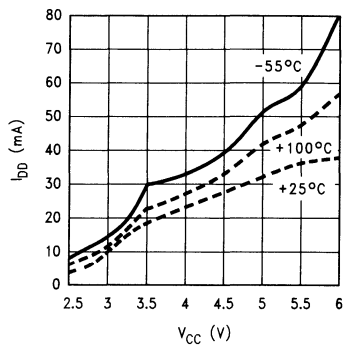
TL/DD/12851-5

Halt Current with Brown Out Disabled



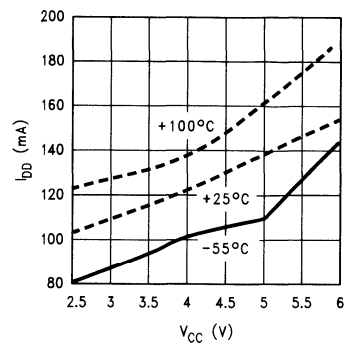
TL/DD/12851-6

Halt Current with Brown Out Enabled



TL/DD/12851-7

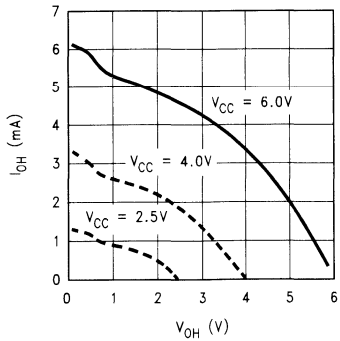
Halt Current with Comparator Enabled



TL/DD/12851-8

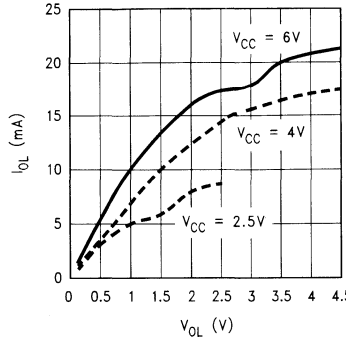
Typical Performance Characteristics ($-40°C \leq T_A \leq +85°C$) (Continued)

Ports L/G Push-Pull Source Current



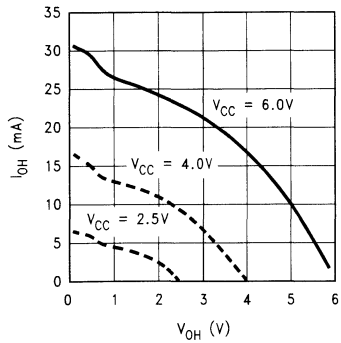
TL/DD/12851-9

Ports L/G Push-Pull Sink Current



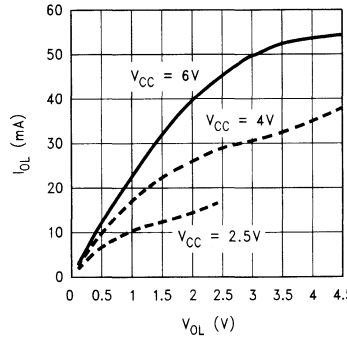
TL/DD/12851-10

Port D Source Current



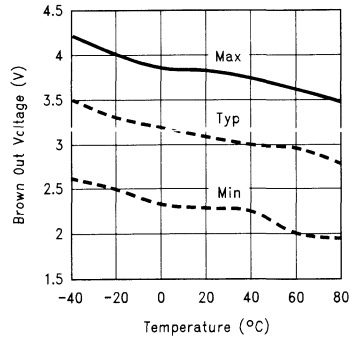
TL/DD/12851-11

Port D Sink Current



TL/DD/12851-12

Brown Out Voltage vs Temperature



TL/DD/12851-13

Pin Descriptions

V_{CC} and **GND** are the power supply pins.

CKI is the clock input. This can come from an external source, a R/C generated oscillator or a crystal (in conjunction with CKO). See Oscillator description.

$\overline{\text{RESET}}$ is the master reset input. See Reset description.

PORT I is a 4-bit Hi-Z input port.

PORT L is an 8-bit I/O port.

There are two registers associated with the L port: a data register and a configuration register. Therefore, each L I/O bit can be individually configured under software control as shown below:

Port L Config.	PORT L Data	Port L Setup
0	0	Hi-Z input (TRI-STATE)
0	1	Input with weak pull-up
1	0	Push-pull zero output
1	1	Push-pull one output

Three data memory address locations are allocated for this port, one each for data register [00D0], configuration register [00D1] and the input pins [00D2].

Port L has the following alternate features:

L0 MIWU or CMPOUT

L1 MIWU or CMPIN−

L2 MIWU or CMPIN+

L3 MIWU

L4 MIWU (high sink current capability)

L5 MIWU (high sink current capability)

L6 MIWU (high sink current capability)

L7 MIWU or MODOUT (high sink current capability)

The selection of alternate Port L functions is done through registers WKEN [00C9] to enable MIWU, and CNTRL2 [00CC] to enable comparator and modulator.

All eight L-pins have Schmitt Triggers on their inputs.

PORT G is an 8-bit port with 6 I/O pins (G0–G5) and 2 input pins (G6, G7).

All eight G-pins have Schmitt Triggers on the inputs.

There are two registers associated with the G port: a data register and a configuration register. Therefore each G port bit can be individually configured under software control as shown below:

Port G Config.	Port G Data	PORT G Setup
0	0	Hi-Z input (TRI-STATE)
0	1	Input with weak pull-up
1	0	Push-pull zero output
1	1	Push-pull one output

Three data memory address locations are allocated for this port, one for data register [00D4], one for configuration register [00D5] and one for the input pins [00D6]. Since G6 and G7 are Hi-Z input only pins, any attempt by the user to configure them as outputs by writing a one to the configuration register will be disregarded. Reading the G6 and G7 configuration bits will return zeros. Note that the device will be placed in the Halt mode by writing a "1" to the G7 data bit.

Six pins of Port G have alternate features:

G0 INTR (an external interrupt)

G3 TIO (timer/counter input/output)

G4 SO (MICROWIRE serial data output)

G5 SK (MICROWIRE clock I/O)

G6 SI (MICROWIRE serial data input)

G7 CKO crystal oscillator output (selected by mask option) or HALT restart input/general purpose input (if clock option is R/C- or external clock)

Pins G1 and G2 currently do not have any alternate functions.

The selection of alternate Port G functions is done through registers PSW [00EF] to enable external interrupt, and CNTRL1 [00EE] to select TIO and MICROWIRE operations.

PORT D is a four bit output port that is preset high when RESET goes low. One data memory address location is allocated for the data register [00DC].

Note: Care must be exercised with the D2 pin operation. At RESET, the external loads on this pin must ensure that the output voltages stay above 0.8 V_{CC} to prevent the chip from entering special modes. Also keep the external loading on D2 to < 1000 pF.

Functional Description

The internal architecture is shown in the block diagram. Data paths are illustrated in simplified form to depict how the various logic elements communicate with each other in implementing the instruction set of the device.

ALU AND CPU REGISTERS

The ALU can do an 8-bit addition, subtraction, logical or shift operations in one cycle time.

There are five CPU registers:

A is the 8-bit Accumulator register

PC is the 15-bit Program Counter register. PU is the upper 7 bits of the program counter (PC). PL is the lower 8 bits of the program counter (PC).

B is the 8-bit address register and can be auto incremented or decremented.

X is the 8-bit alternate address register and can be auto incremented or decremented.

SP is the 8-bit stack pointer which points to the subroutine stack (in RAM).

B, X and SP registers are mapped into the on-chip RAM. The B and X registers are used to address the on-chip RAM. The SP register is used to address the stack in RAM during subroutine calls and returns. The SP must be initialized by software before any subroutine call or interrupts occur.

Functional Description (Continued)

MEMORY

The memory is separated into two memory spaces: program and data.

PROGRAM MEMORY

Program memory consists of 2048 x 8 ROM. These bytes of ROM may hold instructions or constant data. The memory is addressed by the 15-bit program counter (PC). ROM can be indirectly read by the LAID instruction for table lookup.

DATA MEMORY

The data memory address space includes on-chip RAM, I/O and registers. Data memory is addressed directly by instructions or indirectly through the B, X and SP registers. The device has 128 bytes of RAM. Sixteen bytes of RAM are mapped as "registers", these can be loaded immediately, decremented and tested. Three specific registers: X, B, and SP are mapped into this space, the other registers are available for general usage.

The instruction set permits any bit in memory to be directly set, reset or tested. All I/O and registers (except A and PC) are memory mapped; therefore, I/O bits and register bits can be directly and individually set, reset and tested; except the write once only bit (WDREN, WATCHDOG Reset Enable), and the unused and read only bits in the CNTRL2 and WDREG registers.

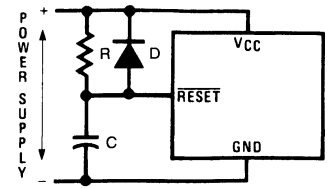
Note: RAM contents are undefined upon power-up.

Reset

EXTERNAL RESET

The RESET input pin when pulled low initializes the microcontroller. The user must insure that the RESET pin is held low until V_{CC} is within the specified voltage range and the clock is stabilized. An R/C circuit with a delay 5x greater than the power supply rise time is recommended *(Figure 2)*. The device immediately goes into reset state when the RESET input goes low. When the RESET pin goes high the device comes out of reset state synchronously. The device will be running within two instruction cycles of the RESET pin going high. The following actions occur upon reset:

Register	Initialization
Port L	TRI-STATE
Port G	TRI-STATE
Port D	HIGH
PC	CLEARED
RAM Contents	RANDOM with Power On Reset UNAFFECTED with external Reset (power already applied)
B, X, SP	Same as RAM
PSW, CNTRL1, CNTRL2 and WDREG Reg.	CLEARED
Multi-Input Wake-up Reg. (WKEDG, WKEN) (WKPND)	CLEARED UNKNOWN
Data and Configuration Registers for L and G	CLEARED
WATCHDOG Timer	Prescaler/Counter each loaded with FF



TL/DD/12851–14

RC > 5 x Power Supply Rise Time

FIGURE 2. Recommended Reset Circuit

The device comes out of the HALT mode when the RESET pin is pulled low. In this case, the user has to ensure that the RESET signal is low long enough to allow the oscillator to restart. An internal 256 t_C delay is normally used in conjunction with the two pin crystal oscillator. When the device comes out of the HALT mode through Multi-Input Wake-up, this delay allows the oscillator to stabilize.

The following additional actions occur after the device comes out of the HALT mode via the RESET pin.

If a two pin crystal/resonator oscillator is being used:

RAM Contents:	UNAFFECTED
Timer T1 and A Contents:	UNKNOWN
WATCHDOG Timer Prescaler/Counter:	CHANGED

If the external or RC clock option is being used:

RAM Contents:	UNCHANGED
Timer T1 and A Contents:	UNCHANGED
WATCHDOG Timer Prescaler/Counter:	CHANGED

The external RESET takes priority over the Brown Out Reset.

Note: If the RESET pin is pulled low while Brown Out occurs (Brown Out circuit has detected Brown Out condition), the external reset will not occur until the Brown Out condition is removed. External reset has priority only if V_{CC} is greater than the Brown Out Voltage.

WATCHDOG RESET

With WATCHDOG enabled, the WATCHDOG logic resets the device if the user program does not service the WATCHDOG timer within the selected service window. The WATCHDOG reset does not disable the WATCHDOG. Upon WATCHDOG reset, the WATCHDOG Prescaler/Counter are each initialized with FF Hex.

The following actions occur upon WATCHDOG reset that are different from external reset.

WDREN WATCHDOG Reset Enable bit UNCHANGED

WDUDF WATCHDOG Underflow bit UNCHANGED

Reset (Continued)

Additional initialization actions that occur as a result of WATCHDOG reset are as follows:

Port L:	TRI-STATE
Port G:	TRI-STATE
Port D:	HIGH
PC:	CLEARED
RAM Contents:	RANDOM
B, X, SP:	UNAFFECTED
PSW, CNTRL1 and CNTRL2 (except WDUDF Bit) Registers:	CLEARED
Multi-Input Wake-up Registers (WKEDG,WKEN): (WKPND):	CLEARED UNKNOWN
Data and Configuration Registers for L and G:	CLEARED
WATCHDOG Timer:	Prescaler/Counter each loaded with FF

BROWN OUT RESET

The on-board Brown Out detection circuit resets the device when the operating voltage (V_{CC}) is lower than the Brown Out voltage. The device is held in reset when V_{CC} stays below the Brown Out voltage. The device will remain in RESET as long as V_{CC} is below the Brown Out Voltage. The device will resume execution if V_{CC} rises above the Brown Out Voltage. If a two pin crystal/resonator clock option is selected, the Brown Out reset will trigger a 256 tc delay. This delay allows the oscillator to stabilize before the device exits the reset state. The delay is not used if the clock option is either R/C or external clock. The contents of data registers and RAM are unknown following a Brown Out reset. The external reset takes priority over Brown Out Reset and will deactivate the 256 tc cycles delay if in progress. The Brown Out reset takes priority over the WATCHDOG reset.

The following actions occur as a result of Brown Out reset:

Port L:	TRI-STATE
Port G:	TRI-STATE
Port D	HIGH
PC:	CLEARED
RAM Contents:	RANDOM
B, X, SP	UNKNOWN
PSW, CNTRL1, CNTRL2 and WDREG Registers:	CLEARED
Multi-Input Wake-up Registers (WKEDG, WKEN):	CLEARED
(WKPND):	UNKNOWN
Data and Configuration Registers for L and G:	CLEARED
WATCHDOG Timer:	Prescaler/Counter each loaded with FF
Timer T1 and Accumulator:	Unknown data after coming out of the HALT (through Brown Out Reset) with any Clock option

Note: The Development system will detect the BROWN OUT RESET externally and will force the $\overline{\text{RESET}}$ pin low. The Development System does not emulate the 256 tc delay.

Brown Out Detection

An on-board detection circuit monitors the operating voltage (V_{CC}) and compares it with the minimum operating voltage specified. The Brown Out circuit is designed to reset the device if the operating voltage is below the Brown Out voltage (between 1.8V–4.2V at −40C to +85C). The Minimum operating voltage for the device is 2.5V with Brown Out disabled, but with Brown Out enabled the device is guaranteed to operate properly down to minimum Brown Out voltage (Max frequency 4 MHz). For temperature range of 0°C–70°C the Brown Out voltage is expected to be between 1.9V and 3.9V. The circuit can be enabled or disabled by Brown Out mask option. If the device is intended to operate at lower V_{CC} (lower than Brown Out voltage VBO max), the Brown Out circuit should be disabled by the mask option.

The Brown Out circuit may be used as a power-up reset provided the power supply rise time is slower than 50 μs (0V–6.0V). Brown Out should not be used in frequencies over 4 MHz.

Note: Brown Out Circuit is active in HALT mode (with the Brown Out mask option selected).

Oscillator Circuits

EXTERNAL OSCILLATOR

By selecting the external oscillator option, the CKI pin can be driven by an external clock signal provided it meets the specified duty cycle, rise and fall times, and input levels. The G7/CKO is available as a general purpose input G7 and/or Halt control.

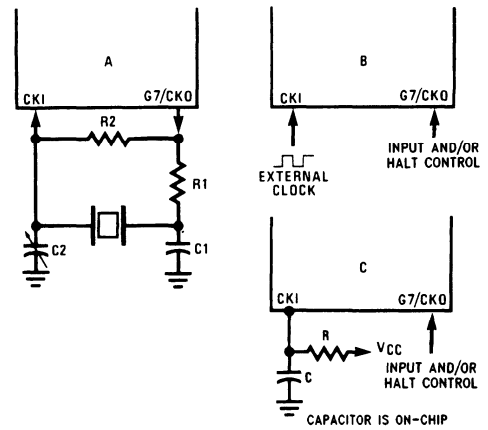
CRYSTAL OSCILLATOR

By selecting the crystal oscillator option, the G7/CKO pin is connected as a clock output, CKI and G7/CKO can be connected to make a crystal controlled oscillator. Table I shows the clock frequency for different component values. See *Figure 3* for the connections.

R/C OSCILLATOR

By selecting the R/C oscillator option, connecting a resistor from the CKI pin to V_{CC} makes a R/C oscillator. The capacitor is on-chip. The G7/CKO pin is available as a general purpose input G7 and/or Halt control. Adding an external capacitor will jeopardize the clock frequency tolerance and increase EMI emissions.

Table II shows the clock frequency for the different resistor values. The capacitor is on-chip. See *Figure 3* for the connections.



TL/DD/12851–15

FIGURE 3. Clock Oscillator Configurations

TABLE I. Crystal Oscillator Configuration (T_A = 25°C)

R1 (kΩ)	R2 (MΩ)	C1 (pF)	C2 (pF)	CKI Freq. (MHz)	Conditions
0	1	30	30–36	10	V_{CC} = 5V
0	1	30	30–36	4	V_{CC} = 5V
5.6	1	200	100–156	0.455	V_{CC} = 5V

TABLE II. R/C Oscillator Configuration (Part- To- Part Variation)

R (kΩ) (Note 1)	CKI Freq. (MHz)	Temp.	V_{CC}
2.2	7.0 ±15%	−40°C – +85°C	4.5V–5.5V
4.7	4.2 ±10%	−40°C – +85°C	4.5V–5.5V
20.0	1.1 ±10%	−40°C – +85°C	4.5V–5.5V

Note 1: The resistance level is calculated with a total of 5.3 pF capacitance added from the printed circuit board. It is important to take this into account when figuring the clock frequency.

Oscillator Circuits (Continued)

HALT Mode

The device is a fully static device. The device enters the HALT mode by writing a one to the G7 bit of the G data register. Once in the HALT mode, the internal circuitry does not receive any clock signal and is therefore frozen in the exact state it was in when halted. In this mode the chip will only draw leakage current (output current and DC current due to the Brown Out circuit if Brown Out is enabled).

The device supports four different methods of exiting the HALT mode. The first method is with a low to high transition on the CKO (G7) pin. This method precludes the use of the crystal clock configuration (since CKO is a dedicated output). It may be used either with an RC clock configuration or an external clock configuration. The second method of exiting the HALT mode is with the Multi-Input Wake-up feature on the L port. The third method of exiting the HALT mode is by pulling the RESET input low. The fourth method is with the operating voltage going below Brown Out voltage (if Brown Out is enabled by mask option).

If the two pin crystal/resonator oscillator is being used and Multi-Input Wake-up or Brown Out causes the device to exit the HALT mode, the WAKE-UP signal does not allow the chip to start running immediately since crystal oscillators have a delayed start up time to reach full amplitude and frequency stability. The WATCHDOG timer (consisting of an 8-bit prescaler followed by an 8-bit counter) is used to generate a fixed delay of 256 t_C to ensure that the oscillator has indeed stabilized before allowing instruction execution. In this case, upon detecting a valid WAKE-UP signal only the oscillator circuitry is enabled. The WATCHDOG Counter and Prescaler are each loaded with a value of FF Hex. The WATCHDOG prescaler is clocked with the t_C instruction cycle (The t_C clock is derived by dividing the oscillator clock down by a factor of 10). The schmitt trigger following the CKI inverter on the chip ensures that the WATCHDOG timer is clocked only when the oscillator has a sufficiently large amplitude to meet the Schmitt trigger specs. This Schmitt trigger is not part of the oscillator closed loop. The start-up time-out from the WATCHDOG timer enables the clock signals to be routed to the rest of the chip. The delay is not activated when the device comes out of HALT mode through RESET pin. Also, if the clock option is either RC or External clock, the delay is not used, but the WATCHDOG Prescaler/Counter contents are changed. The Development System will not emulate the 256 t_C delay.

The RESET pin or Brown Out will cause the device to reset and start executing from address X'0000. A low to high transition on the G7 pin (if single pin oscillator is used) or Multi-Input Wake-up will cause the device to start executing from the address following the HALT instruction.

When RESET/pin is used to exit the device from the HALT mode and the two pin crystal/resonator (CKI/CKO) clock option is selected the contents of the Accumulator and the Timer T1 are undetermined following the reset. All other information except the WATCHDOG Prescaler/Counter contents is retained until continuing. If the device comes out of the HALT mode through Brown Out reset, the contents of data registers and RAM are unknown following the reset. All information except the WATCHDOG Prescaler/Counter contents is retained if the device exits the HALT mode through G7 pin or Multi-Input Wake-up.

G7 is the HALT-restart pin, but it can still be used as an input. If the device is not halted, G7 can be used as a general purpose input.

If the Brown Out Enable mask option is selected, the Brown Out circuit remains active during the HALT mode causing additional current to be drawn.

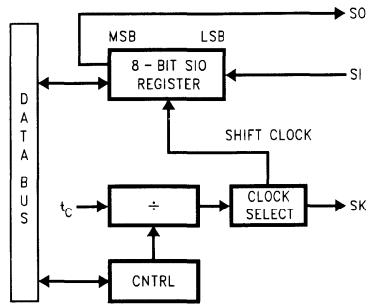
Note: To allow clock resynchronization, it is necessary to program two NOP's immediately after the device comes out of the HALT mode. The user must program two NOP's following the "enter HALT mode" (set G7 data bit) instruction.

MICROWIRE/PLUS

MICROWIRE/PLUS is a serial synchronous bidirectional communications interface. The MICROWIRE/PLUS capability enables the device to interface with any of National Semiconductor's MICROWIRE peripherals (i.e., A/D converters, display drivers, EEPROMS, etc.) and with other microcontrollers which support the MICROWIRE/PLUS interface. It consists of an 8-bit serial shift register (SIO) with serial data input (SI), serial data output (SO) and serial shift clock (SK). *Figure 4* shows the block diagram of the MICROWIRE/PLUS interface.

The shift clock can be selected from either an internal source or an external source. Operating the MICROWIRE/PLUS interface with the internal clock source is called the Master mode of operation. Operating the MICROWIRE/PLUS interface with an external shift clock is called the Slave mode of operation.

MICROWIRE/PLUS (Continued)



TL/DD/12851–16

FIGURE 4. MICROWIRE/PLUS Block Diagram

The CNTRL register is used to configure and control the MICROWIRE/PLUS mode. To use the MICROWIRE/PLUS, the MSEL bit in the CNTRL register is set to one. The SK clock rate is selected by the two bits, S0 and S1, in the CNTRL register. Table III details the different clock rates that may be selected.

TABLE III

S1	S0	SK Cycle Time
0	0	$2t_C$
0	1	$4t_C$
1	x	$8t_C$

where,

t_C is the instruction cycle time.

MICROWIRE/PLUS OPERATION

Setting the BUSY bit in the PSW register causes the MICROWIRE/PLUS arrangement to start shifting the data. It gets reset when eight data bits have been shifted. The user may reset the BUSY bit by software to allow < 8 bits to shift. The device may enter the MICROWIRE/PLUS mode either as a Master or as a Slave. *Figure 5* shows how two devices and several peripherals may be interconnected using the MICROWIRE/PLUS arrangement.

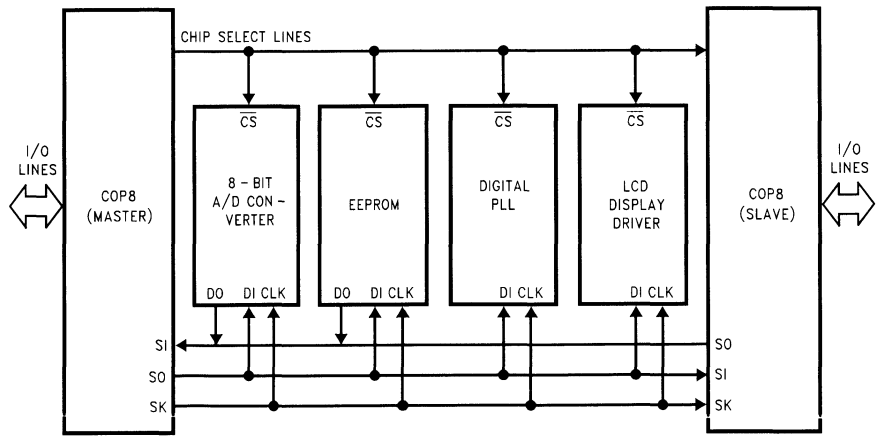
MASTER MICROWIRE/PLUS OPERATION

In the MICROWIRE/PLUS Master mode of operation the shift clock (SK) is generated internally by the device. The MICROWIRE/PLUS Master always initiates all data exchanges. (See *Figure 5*). The MSEL bit in the CNTRL register must be set to enable the SO and SK functions on the G Port. The SO and SK pins must also be selected as outputs by setting appropriate bits in the Port G configuration register. Table IV summarizes the bit settings required for Master mode of operation.

SLAVE MICROWIRE/PLUS OPERATION

In the MICROWIRE/PLUS Slave mode of operation the SK clock is generated by an external source. Setting the MSEL bit in the CNTRL register enables the SO and SK functions on the G Port. The SK pin must be selected as an input and the SO pin selected as an output pin by appropriately setting up the Port G configuration register. Table IV summarizes the settings required to enter the Slave mode of operation.

The user must set the BUSY flag immediately upon entering the Slave mode. This will ensure that all data bits sent by the Master will be shifted properly. After eight clock pulses the BUSY flag will be cleared and the sequence may be repeated. (See *Figure 5*)



TL/DD/12851–17

FIGURE 5. MICROWIRE/PLUS Application

MICROWIRE/PLUS (Continued)

TABLE IV

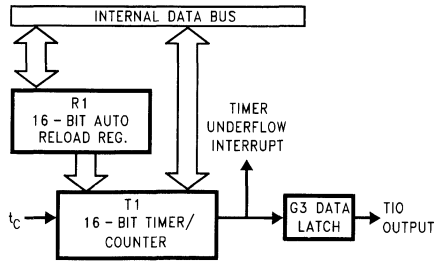
G4 Config. Bit	G5 Config. Bit	G4 Fun.	G5 Fun.	G6 Fun.	Operation
1	1	SO	Intl.SK	SI	MICROWIRE Master
0	1	TRI-STATE	Int. SK	SI	MICROWIRE Master
1	0	SO	Ext. SK	SI	MICROWIRE Slave
0	0	TRI-STATE	Ext. SK	SI	MICROWIRE Slave

TIMER/COUNTER

The device has a powerful 16-bit timer with an associated 16-bit register enabling it to perform extensive timer functions. The timer T1 and its register R1 are each organized as two 8-bit read/write registers. Control bits in the register CNTRL allow the timer to be started and stopped under software control. The timer-register pair can be operated in one of three possible modes. Table V details various timer operating modes and their requisite control settings.

MODE 1. TIMER WITH AUTO-LOAD REGISTER

In this mode of operation, the timer T1 counts down at the instruction cycle rate. Upon underflow the value in the register R1 gets automatically reloaded into the timer which continues to count down. The timer underflow can be programmed to interrupt the microcontroller. A bit in the control register CNTRL enables the TIO (G3) pin to toggle upon timer underflows. This allows the generation of square-wave outputs or pulse width modulated outputs under software control. (See *Figure 6*.)



TL/DD/12851–18

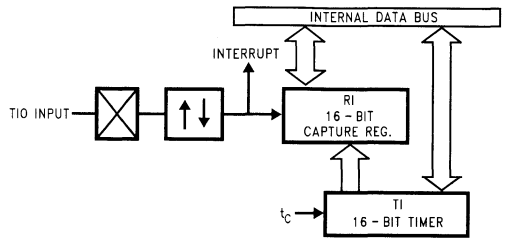
FIGURE 6. Timer/Counter Auto Reload Mode Block Diagram

MODE 2. EXTERNAL COUNTER

In this mode, the timer T1 becomes a 16-bit external event counter. The counter counts down upon an edge on the TIO pin. Control bits in the register CNTRL program the counter to decrement either on a positive edge or on a negative edge. Upon underflow the contents of the register R1 are automatically copied into the counter. The underflow can also be programmed to generate an interrupt. (See *Figure 6*).

MODE 3 TIMER WITH CAPTURE REGISTER

Timer T1 can be used to precisely measure external frequencies or events in this mode of operation. The timer T1 counts down at the instruction cycle rate. Upon the occurrence of a specified edge on the TIO pin the contents of the timer T1 are copied into the register R1. Bits in the control register CNTRL allow the trigger edge to be specified either as a positive edge or as a negative edge. In this mode the user can elect to be interrupted on the specified trigger edge. (See *Figure 7*).

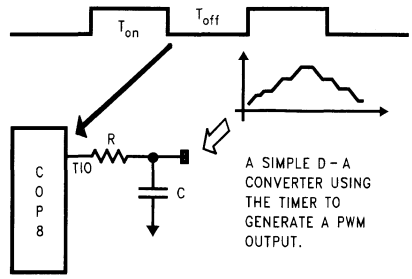


TL/DD/12851–19

FIGURE 7. Timer Capture Mode Block Diagram

TIMER PWM APPLICATION

Figure 8 shows how a minimal component D/A converter can be built out of the Timer-Register pair in the Auto-Reload mode. The timer is place in the "Timer with auto reload" mode and the TIO pin is selected as timer output. At the outset the TIO pin is set high, the timer T1 holds the on time and the register R1 holds the signal off time. Setting TRUN bit starts the timer which counts down at the instruction cycle rate. The underflow toggles the TIO output and copies the off time into the timer, which continues to run. By alternately loading in the on time and the off time at each successive interrupt a PWM frequency can be easily generated.



TL/DD/12851–20

FIGURE 8. Timer Application

MICROWIRE/PLUS (Continued)

TABLE V. Timer Operating Modes

CNTRL Bits 7 6 5	Operating Mode	T Interrupt	Timer Counter On
0 0 0	External Counter W/Auto-Load Reg.	Timer Underflow	TIO Pos. Edge
0 0 1	External Counter W/Auto-Load Reg.	Timer Underflow	TIO Pos. Edge
0 1 0	Not Allowed	Not Allowed	Not Allowed
0 1 1	Not Allowed	Not Allowed	Not Allowed
1 0 0	Timer W/Auto-Load Reg.	Timer Underflow	t_C
1 0 1	Timer W/Auto-Load Reg./Toggle TIO Out	Timer Underflow	t_C
1 1 0	Timer W/Capture Register	TIO Pos. Edge	t_C
1 1 1	Timer W/Capture Register	TIO Neg. Edge	t_C

WATCHDOG

The device has an on board 8-bit WATCHDOG timer. The timer contains an 8-bit READ/WRITE down counter clocked by an 8-bit prescaler. Under software control the timer can be dedicated for the WATCHDOG or used as a general purpose counter. *Figure 9* shows the WATCHDOG timer block diagram.

MODE 1: WATCHDOG TIMER

The WATCHDOG is designed to detect user programs getting stuck in infinite loops resulting in loss of program control or "runaway" programs. The WATCHDOG can be enabled or disabled (only once) after the device is reset as a result of Brown Out reset or external reset. On power-up the WATCHDOG is disabled. The WATCHDOG is enabled by writing a 1 to WDREN bit (resides in WDREG register). Once enabled, the user program should write periodically into the 8-bit counter before the counter underflows. The 8-bit counter (WDCNT) is memory mapped at address 0CE Hex. The counter is loaded with n-1 to get n counts. The counter underflow resets the device, but does not disable the WATCHDOG. Loading the 8-bit counter initializes the prescaler with FF Hex and starts the prescaler/counter. Prescaler and counter are stopped upon counter underflow. Prescaler and counter are each loaded with FF Hex when the device goes into the HALT mode. The prescaler is used for crystal/resonator start-up when the device exits the HALT mode through Multi-Input Wake-up. In this case, the prescaler/counter contents are changed.

MODE 2: TIMER

In this mode, the prescaler/counter is used as a timer by keeping the WDREN (WATCHDOG reset enable) bit at 0. The counter underflow sets the WDUDF (underflow) bit and the underflow does not reset the device. Loading the 8-bit counter (load n-1 for n counts) sets the WDTEN bit (WATCHDOG Timer Enable) to "1", loads the prescaler with FF, and starts the timer. The counter underflow stops the timer. The WDTEN bit serves as a start bit for the WATCHDOG timer. This bit is set when the 8-bit counter is loaded by the user program. The load could be as a result of WATCHDOG service (WATCHDOG timer dedicated for WATCHDOG function) or write to the counter (WATCHDOG timer used as a general purpose counter). The bit is cleared upon Brown Out reset, WATCHDOG reset or external reset. The bit is not memory mapped and is transparent to the user program.

Control/Status Bits

WDUDF: WATCHDOG Timer Underflow Bit

This bit resides in the CNTRL2 Register. The bit is set when the WATCHDOG timer underflows. The underflow resets the device if the WATCHDOG reset enable bit is set (WDREN = 1). Otherwise, WDUDF can be used as the timer underflow flag. The bit is cleared upon Brown-Out reset, external reset, load to the 8-bit counter, or going into the HALT mode. It is a read only bit.

WDREN: WD Reset Enable

WDREN bit resides in a separate register (bit 0 of WDREG). This bit enables the WATCHDOG timer to generate a reset. The bit is cleared upon Brown Out reset, or external reset. The bit under software control can be written to only once (once written to, the hardware does not allow the bit to be changed during program execution).

WDREN = 1 WATCHDOG reset is enabled
WDREN = 0 WATCHDOG reset is disabled.

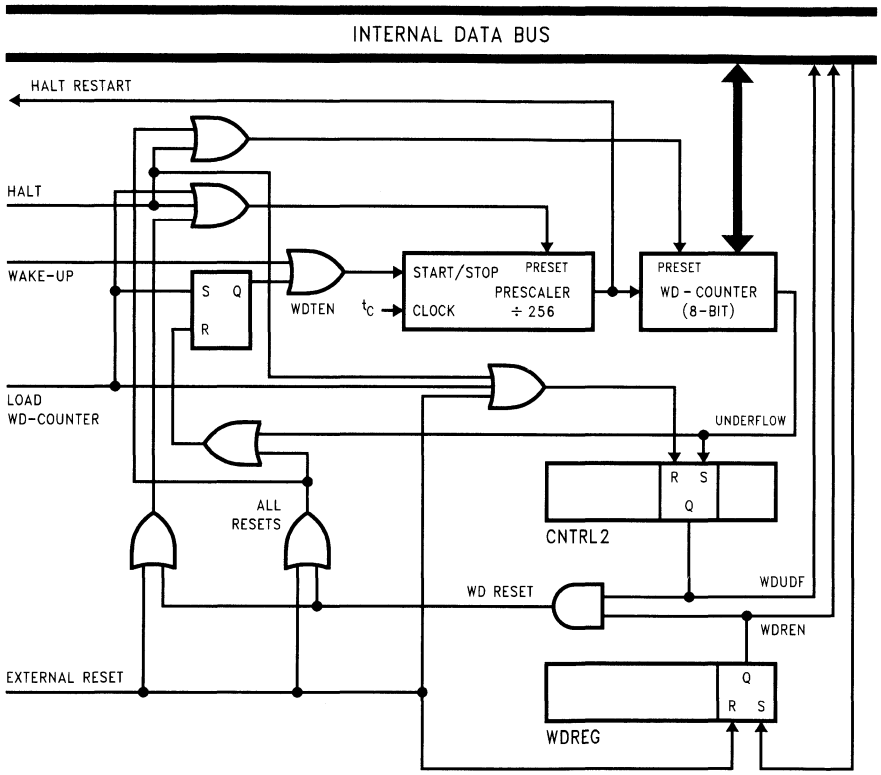
Table VI shows the impact of Brown Out Reset, WATCHDOG Reset, and External Reset on the Control/Status bits.

WATCHDOG (Continued)

TABLE VI. WATCHDOG Control/Status

Parameter	HALT MODE	WD Reset	EXT/BOR (Note 1) Reset	Load Counter
8-bit Prescaler	FF	FF	FF	FF
8-bit WD counter	FF	FF	FF	User Value
WDREN bit	Unchanged	Unchanged	0	No effect
WDUDF bit	0	Unchanged	0	0
WDTEN Signal	Unchanged	0	0	1

Note 1: BOR is Brown Out Reset



TL/DD/12851-21

FIGURE 9. WATCHDOG Timer Block Diagram

Modulator/Timer

The MODULATOR/TIMER contains an 8-bit counter and an 8-bit autoreload register (MODRL address 0CF Hex). The Modulator/Timer has two modes of operation, selected by the control bit MC3. The Modulator/Timer Control bits MC1, MC2 and MC3 reside in CNTRL2 Register.

MODE 1: MODULATOR

The Modulator is used to generate high frequency pulses on the modulator output pin (L7). The L7 pin should be configured as an output. The number of pulses is determined by the 8-bit down counter. Under software control the modulator input clock can be either CKI or t_C. The t_C clock is derived by dividing down the oscillator clock by a factor of 10. Three control bits (MC1, MC2, and MC3) are used for the Modulator/Timer output control. When MC2 = 1 and MC3 = 1, CKI is used as the modulator input clock. When MC2 = 0, and MC3 = 1, t_C is used as the modulator input clock. The user loads the counter with the desired number of counts (256 max) and sets MC1 to start the counter. The modulator autoreload register is loaded with n-1 to get n pulses. CKI or t_C pulses are routed to the modulator output (L7) until the counter underflows *(Figure 10)*. Upon underflow the hardware resets MC1 and stops the counter. The L7 pin goes low and stays low until the counter is restarted by the user program. The user program has the responsibility to time-out the low time. Unless the number of counts is changed, the user program does not have to load the counter each time the counter is started. The counter can simply be started by setting the MC1 bit. Setting MC1 by software will load the counter with the value of the autoreload register. The software can reset MC1 to stop the counter.

MODE 2: PWM TIMER

The counter can also be used as a PWM Timer. In this mode, an 8-bit register is used to serve as an autoreload register (MODRL).

a. 50% Duty Cycle:

When MC1 is 1 and MC2, MC3 are 0, a 50% duty cycle free running signal is generated on the L7 output pin *(Figure 11)*. The L7 pin must be configured as an output pin. In this mode the 8-bit counter is clocked by t_C. Setting the MC1 control bit by software loads the counter with the value of the autoreload register and starts the counter. The counter underflow toggles the (L7) output pin. The 50% duty cycle signal will be continuously generated until MC1 is reset by the user program.

b. Variable Duty Cycle:

When MC3 = 0 and MC2 = 1, a variable duty cycle PWM signal is generated on the L7 output pin. The counter is clocked by t_C. In this mode the 16-bit timer T1 along with the 8-bit down counter are used to generate a variable duty cycle PWM signal. The timer T1 underflow sets MC1 which starts the down counter and it also sets L7 high (L7 should be configured as an output). When the counter underflows the MC1 control bit is reset and the L7 output will go low until the next timer T1 underflow. Therefore, the width of the output pulse is controlled by the 8-bit counter and the pulse duration is controlled by the 16-bit timer T1 *(Figure 12)*. Timer T1 must be configured in "PWM Mode/Toggle TIO Out" (CNTRL1 Bits 7,6,5 = 101).

Table VII shows the different operation modes for the Modulator/Timer.

TABLE VII. Modulator/Timer Modes

Control bits in CNTRL2(00CC)			OPERATION MODE L7 Function
MC3	MC2	MC1	
0	0	0	Normal I/O
0	0	1	50% duty cycle mode (clocked by t_C)
0	1	X	Variable duty cycle mode (clocked by t_C) using Timer 1 underflow
1	0	X	Modulator mode (clocked by t_C)
1	1	X	Modulator mode (clocked by CKI)

Note: MC1, MC2 and MC3 control bits are cleared upon reset.

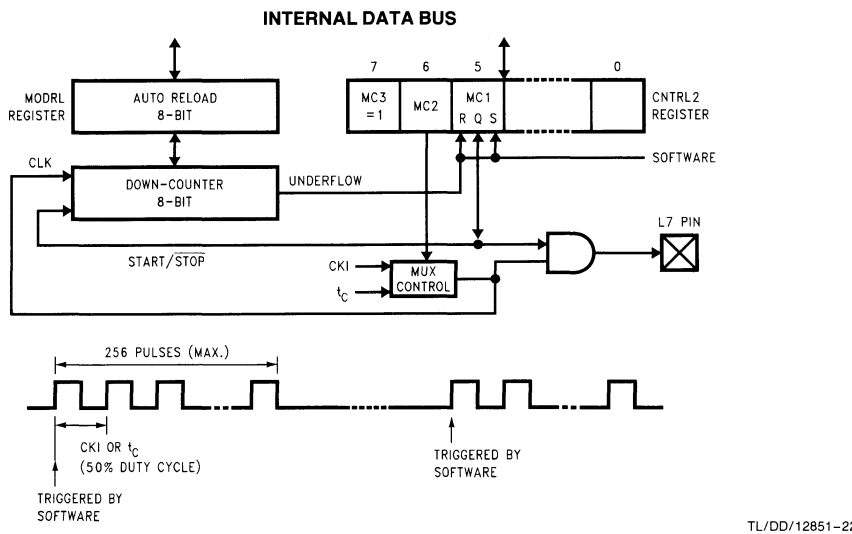


FIGURE 10. Mode 1: Modulator Block Diagram/Output Waveform

Modulator/Timer (Continued)

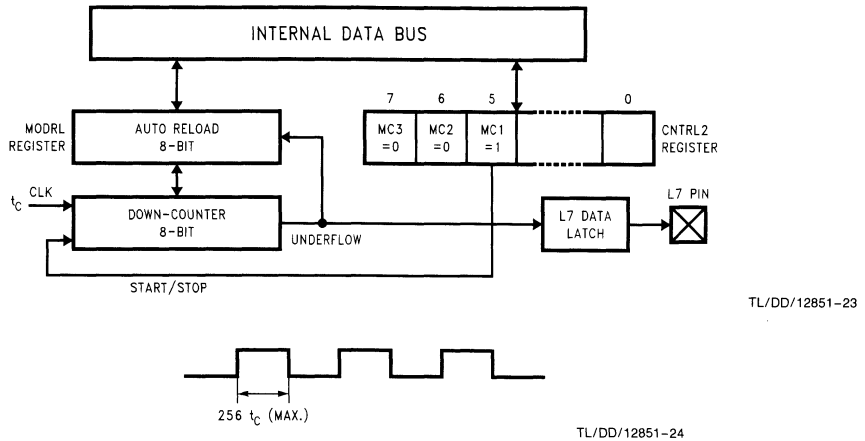


FIGURE 11. Mode 2a: 50% Duty Cycle Output

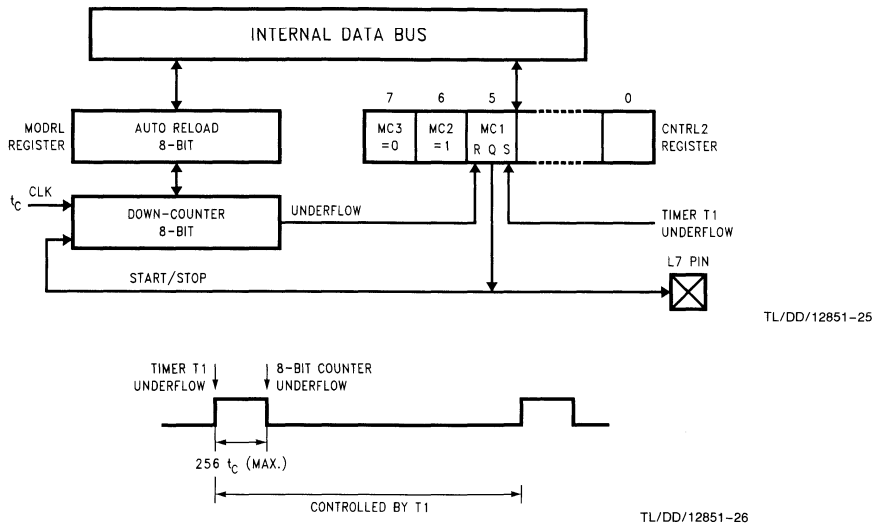


FIGURE 12. Mode 2b: Variable Duty Cycle Output

Comparator

The device has one differential comparator. Ports L0–L2 are used for the comparator. The output of the comparator is brought out to a pin. Port L has the following assignments:

L0 Comparator output
L1 Comparator negative input
L2 Comparator positive input

THE COMPARATOR STATUS/CONTROL BITS

These bits reside in the CNTRL2 Register (Address 0CC)

CMPEN Enables comparator ("1" = enable)

CMPRD Reads comparator output internally (CMPEN = 1, CMPOE = X)

CMPOE Enables comparator output to pin L0 ("1"= enable), CMPEN bit must be set to enable this function. If CMPEN = 0, L0 will be 0.

The Comparator Select/Control bits are cleared on RESET (the comparator is disabled). To save power the program should also disable the comparator before the device enters the HALT mode.

The user program must set up L0, L1, and L2 ports correctly for comparator Inputs/Output. L1 and L2 need to be configured as inputs and L0 as output. Table VIII shows the DC and AC characteristics for the comparator.

Comparator (Continued)

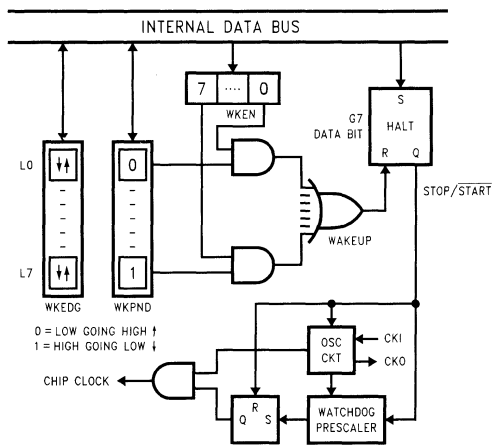
TABLE VIII. DC and AC Characteristics (Note 1) 4V ≤ V_{CC} ≤ 6V, −40°C ≤ T_A ≤ +85°C

Parameters	Conditions	Min	Typ	Max	Units
Input Offset Voltage	0.4V < V_{IN} < V_{CC} −1.5V		±10	±25	mV
Input Common Mode Voltage Range		0.4		V_{CC} −1.5	V
Voltage Gain			300k		V/V
DC Supply Current (when enabled)	V_{CC} = 6.0V			250	μA
Response Time	100 mV Overdrive	60	100	140	ns
	500 mV Overdrive	80	125	165	
	1000 mV Overdrive	135	215	300	

Note 1: For comparator output current characteristics see L-Port specs.

Multi-Input Wake-Up

The Multi-Input Wake-Up feature is used to return (wake-up) the device from the HALT mode. *Figure 13* shows the Multi-Input Wake-Up logic.



TL/DD/12851-27

FIGURE 13. Multi-Input Wake-Up Logic

This feature utilizes the L Port. The user selects which particular L port bit or combination of L Port bits will cause the device to exit the HALT mode. Three 8-bit memory mapped registers, Reg:WKEN, Reg:WKEDG, and Reg:WKPND are used in conjunction with the L port to implement the Multi-Input Wake-Up feature.

All three registers Reg:WKEN, Reg:WKPND, and Reg:WKEDG are read/write registers, and are cleared at reset, except WKPND. WKPND is unknown on reset.

The user can select whether the trigger condition on the selected L Port pin is going to be either a positive edge (low to high transition) or a negative edge (high to low transition). This selection is made via the Reg:WKEDG, which is an 8-bit control register with a bit assigned to each L Port pin. Setting the control bit will select the trigger condition to be a negative edge on that particular L Port pin. Resetting the bit selects the trigger condition to be a positive edge. Changing an edge select entails several steps in order to avoid a pseudo wake-up condition as a result of the edge change. First, the associated WKEN bit should be reset, followed by the edge select change in WKEDG. Next, the associated WKPND bit should be cleared, followed by the associated WKEN bit being re-enabled.

An example may serve to clarify this procedure. Suppose we wish to change the edge select from positive (low going high) to negative (high going low) for L port bit 5, where bit 5 has previously been enabled for an input. The program would be as follows:

```
RBIT 5,WKEN
SBIT 5,WKEDG
RBIT 5,WKPND
SBIT 5,WKEN
```

If the L port bits have been used as outputs and then changed to inputs with Multi-Input Wake-Up, a safety procedure should also be followed to avoid inherited pseudo wake-up conditions. After the selected L port bits have been changed from output to input but before the associated WKEN bits are enabled, the associated edge select bits in WKEDG should be set or reset for the desired edge selects, followed by the associated WKPND bits being cleared. This same procedure should be used following RESET, since the L port inputs are left floating as a result of RESET.

The occurrence of the selected trigger condition for Multi-Input Wake-Up is latched into a pending register called Reg:WKPND. The respective bits of the WKPND register will be set on the occurrence of the selected trigger edge on the corresponding Port L pin. The user has the responsibility of clearing these pending flags. Since the Reg:WKPND is a pending register for the occurrence of selected wake-up conditions, the device will not enter the HALT mode if any wake-up bit is both enabled and pending. Setting the G7 data bit under this condition will not allow the device to enter the HALT mode. Consequently, the user has the responsibility of clearing the pending flags before attempting to enter the HALT mode.

If a crystal oscillator is being used, the wake-up signal will not start the chip running immediately since crystal oscillators have a finite start up time. The WATCHDOG timer prescaler generates a fixed delay to ensure that the oscillator has indeed stabilized before allowing the device to execute instructions. In this case, upon detecting a valid wake-up signal only the oscillator circuitry and the WATCHDOG timer are enabled. The WATCHDOG timer prescaler is loaded with a value of FF Hex (256 counts) and is clocked from the t_C instruction cycle clock. The t_C clock is derived dividing down the oscillator clock by a factor of 10. A Schmitt trigger following the CKI on-chip inverter ensures that the WATCHDOG timer is clocked only when the oscillator has a sufficiently large amplitude to meet the Schmitt trigger specs.

Multi-Input Wake-Up (Continued)

This Schmitt trigger is not part of the oscillator closed loop. The start-up time-out from the WATCHDOG timer enables the clock signals to be routed to the rest of the chip.

Interrupts

The device has a sophisticated interrupt structure to allow easy interface to the real world. There are three possible interrupt sources, as shown below.

A maskable interrupt on external GO input (positive or negative edge sensitive under software control)

A maskable interrupt on timer carry or timer capture

A non-maskable software/error interrupt on opcode zero

INTERRUPT CONTROL

The GIE (global interrupt enable) bit enables the interrupt function. This is used in conjunction with ENI and ENTI to select one or both of the interrupt sources. This bit is reset when interrupt is acknowledged.

ENI and ENTI bits select external and timer interrupts respectively. Thus be user can select either or both source to interrupt the microcontroller when GIE is enabled.

IEDG selects the external interrupt edge (0—rising edge, 1 = falling edge). The user can get an interrupt on both rising and falling edges by toggling the state of IEDG bit after each interrupt.

IPND and TPND bits signal which interrupt is pending. After an interrupt is acknowledged, the user can check these two bits to determine which interrupt is pending. This permits the interrupts to be prioritized under software. The pending flags have to be cleared by the user. Setting the GIE bit high inside the interrupt subroutine allows nested interrupts. The software interrupt does not reset the GIE bit. This means that the controller can be interrupted by other interrupt sources while servicing the software interrupt.

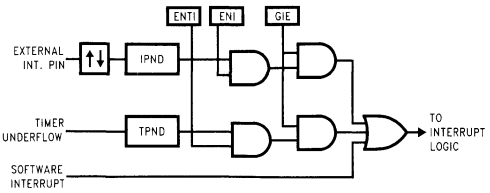
INTERRUPT PROCESSING

The interrupt, once acknowledged, pushes the program countermen counter (PC) onto the stack and the stack pointer (SP) is decremental twice. The Global Interrupt Enable (GIE) bit is reset to disable further interrupts. The microcontroller then vectors to the address 00FFH and resumes execution from that address. This process takes 7 cycles to complete. At end of the interrupt subroutine, any of the following three instructions return the processor back to the main program: RET, RETSK or RETI. Either one of the three instructions will pop the stack into the program counter (PC).

The stack pointer is then incremented twice. The RETI instruction additionally sets the GIE bit to re-enable further interrupts.

Any of the three instructions can be used to return from a hardware interrupt subroutine. The RETSK instruction should be used when returning from a software interrupt subroutine to avoid entering an infinite loop.

Note: There is always the possibility of an interrupt occurring during an instruction which is attempting to reset the GIE bit or any other interrupt enable bit. If this occurs when a single cycle instruction is being used to reset the interrupt enable bit, the interrupt enable bit will be reset but an interrupt may still occur. This is because interrupt processing is started at the same time as the interrupt bit is being reset. To avoid this scenario, the user should always use a two, three, or four cycle instruction to reset interrupt enable bits.



TL/DD/12851–28

FIGURE 14. Interrupt Block Diagram

DETECTION OF ILLEGAL CONDITIONS

The device incorporates a hardware mechanism that allows it to detect illegal conditions which may occur from coding errors, noise and "brown out" voltage drop situations. Specifically, it detects cases of executing out of undefined ROM area and unbalanced stack situations.

Reading an undefined ROM location returns 00 (hexadecimal) as its contents. The opcode for a software interrupt is also "00". Thus a program accessing undefined ROM will cause a software interrupt. Reading undefined RAM location returns an FF (hexadecimal). The subroutine stack on the device grows down for each subroutine call. By initializing the stack pointer to the top of RAM, the first unbalanced return instruction will cause the stack pointer to address undefined RAM. As a result the program will attempt to execute from FFFF (hexadecimal), which is an undefined ROM location and will trigger a software interrupt.

Control Registers

CNTRL1 REGISTER (ADDRESS 00EE)

The Timer and MICROWIRE control register contains the following bits:

SL1 and SL0	Select the MICROWIRE clock divide-by (00 = 2, 01 = 4, 1x = 8)
IEDG	External interrupt edge polarity select
MSEL	Selects G5 and G4 as MICROWIRE signals SK and SO respectively
TRUN	Used to start and stop the timer/counter (1 = run, 0 = stop)
TC1	Timer T1 Mode Control Bit
TC2	Timer T1 Mode Control Bit
TC3	Timer T1 Mode Control Bit

TC1	TC2	TC3	TRUN	MSEL	IEDG	SL1	SL0
Bit 7							Bit 0

Control Registers (Continued)

PSW REGISTER (ADDRESS 00EF)

The PSW register contains the following select bits:

GIE	Global interrupt enable (enables interrupts)
ENI	External interrupt enable
BUSY	MICROWIRE busy shifting flag
PND	External interrupt pending
ENTI	Timer T1 interrupt enable
TPND	Timer T1 interrupt pending (timer Underflow or capture edge)
C	Carry Flip/flop
HC	Half carry Flip/flop

HC	C	TPND	ENTI	IPND	BUSY	ENI	GIE

Bit 7 Bit 0

The Half-Carry bit is also effected by all the instructions that effect the Carry flag. The flag values depend upon the instruction. For example, after executing the ADC instruction the values of the Carry and the Half-Carry flag depend upon the operands involved. However, instructions like SET C and RESET C will set and clear both the carry flags. Table IX lists the instructions that effect the HC and the C flags.

TABLE IX. Instructions Effecting HC and C Flags

Instr.	HC Flag	C Flag
ADC	Depends on operands	Depends on operands
SUBC	Depends on operands	Depends on operands
SET C	Set	Set
RESET C	Set	Set
RRC	Depends on operands	Depends on operands

CNTRL2 REGISTER (ADDRESS 00CC)

MC3	MC2	MC1	CMPEN	CMPRD	CMPOE	WDUDF	unused
R/W	R/W	R/W	R/W	R/O	R/W	R/O	

Bit 7 Bit 0

MC3	Modulator/Timer Control Bit
MC2	Modulator/Timer Control Bit
MC1	Modulator/Timer Control Bit
CMPEN	Comparator Enable Bit
CMPRD	Comparator Read Bit
CMPOE	Comparator Output Enable Bit
WDUDF	WATCHDOG Timer Underflow Bit (Read Only)

WDREN REGISTER (ADDRESS 00CD)

WDRENWATCHDOG Reset Enable Bit (Write Once Only)

UNUSED	WDREN

Bit 7 Bit 0

Memory Map

All RAM, ports and registers (except A and PC) are mapped into data memory address space.

TABLE X. Memory Map

ADDRESS	CONTENTS
00–6F	On-Chip RAM bytes (112 bytes)
70–7F	Unused RAM address (Reads as all ones)
80–BF	Unused RAM address (Reads Undefined Data)
C0–C7	Reserved
C8	MIWU Edge Select Register (Reg:WKEDG)
C9	MIWU Enable Register (Reg:WKEN)
CA	MIWU Pending Register (Reg:WKPND)
CB	Reserved
CC	Control2 Register (CNTRL2)
CD	WATCHDOG Register (WDREG)
CE	WATCHDOG Counter (WDCNT)
CF	Modulator Reload (MODRL)
D0	Port L Data Register
D1	Port L Configuration Register
D2	Port L input Pins (read only)
D3	Reserved for Port L
D4	Port G Data Register
D5	Port G Configuration Register
D6	Port G Input pins (read only)
D7	Port I Input pins (read only)
D8–DB	Reserved for Port C
DC	Port D Data Register
DD–DF	Reserved for Port D
E0–EF	On-Chip Functions and Registers
E0–E7	Reserved for Future Parts
E8	Reserved
E9	MICROWIRE Shift Register
EA	Timer Lower Byte
EB	Timer Upper Byte
EC	Timer1 Autoreload Register Lower Byte
ED	Timer1 Autoreload Register Upper Byte
EE	CNTRL1 Control Register
EF	PSW Register
F0–FF	On-Chip RAM mapped as Registers
FC	X Register
FD	SP Register
FE	B Register

Reading other unused memory locations will return undefined data.

Addressing Modes

There are ten addressing modes, six for operand addressing and four for transfer of control.

OPERAND ADDRESSING MODES

REGISTER INDIRECT

This is the "normal" addressing mode for the chip. The operand is the data memory addressed by the B or X pointer.

REGISTER INDIRECT WITH AUTO POST INCREMENT OR DECREMENT

This addressing mode is used with the LD and X instructions. The operand is the data memory addressed by the **B** or **X** pointer. This is a register indirect mode that automatically post increments or post decrements the **B** or **X** pointer after executing the instruction.

DIRECT

The instruction contains an 8-bit address field that directly points to the data memory for the operand.

IMMEDIATE

The instruction contains an 8-bit immediate field as the operand.

SHORT IMMEDIATE

This addressing mode issued with the LD B,# instruction, where the immediate # is < 16. The instruction contains a 4-bit immediate field as the operand.

INDIRECT

This addressing mode is used with the LAID instruction. The contents of the accumulator are used as a partial address (lower 8 bits of PC) for accessing a data operand from the program memory.

TRANSFER OF CONTROL ADDRESSING MODES

RELATIVE

This mode is used for the JP instruction with the instruction field being added to the program counter to produce the next instruction address. JP has a range from -31 to $+32$ to allow a one byte relative jump (JP $+$ 1 is implemented by a NOP instruction). There are no "blocks" or "pages" when using JP since all 15 bits of the PC are used.

ABSOLUTE

This mode is used with the JMP and JSR instructions with the instruction field of 12 bits replacing the lower 12 bits of the program counter (PC). This allows jumping to any location in the current 4k program memory segment.

ABSOLUTE LONG

This mode is used with the JMPL and JSRL instructions with the instruction field of 15 bits replacing the entire 15 bits of the program counter (PC). This allows jumping to any location in the entire 32k program memory space.

INDIRECT

This mode is used with the JID instruction. The contents of the accumulator are used as a partial address (lower 8 bits of PC) for accessing a location in the program memory. The contents of this program memory location serves as a partial address (lower 8 bits of PC) for the jump to the next instruction.

Instruction Set

REGISTER AND SYMBOL DEFINITIONS

Registers

A	8-bit Accumulator register
B	8-bit Address register
X	8-bit Address register
SP	8-bit Stack pointer register
PC	15-bit Program counter register
PU	Upper 7 bits of PC
PL	Lower 8 bits of PC
C	1-bit of PSW register for carry
HC	1-bit of PSW register for half carry
GIE	1-bit of PSW register for global interrupt enable

Symbols

[B]	Memory indirectly addressed by B register
[X]	Memory indirectly addressed by X register
MD	Direct addressed memory
Mem	Direct addressed memory, or [B]
Meml	Direct addressed memory, [B], or Immediate data
Imm	8-bit Immediate data
Reg	Register memory: addresses F0 to FF (Includes B, X, and SP)
Bit	Bit number (0 to 7)
←	Loaded with
⟷	Exchanged with

Instr		Function	Register Operation
ADD	A, Meml	Add	A ← A + Meml
ADC	A, Meml	Add with carry	A ← A + Meml + C, C ← Carry
SUBC	A, Meml	Subtract with carry	A ← A + $\overline{\text{Meml}}$ + C, C ← Carry
AND	A, Meml	Logical AND	A ← A and Meml
OR	A, Meml	Logical OR	A ← A or Meml
XOR	A, Meml	Logical Exclusive-OR	A ← A xor Meml
IFEQ	A, Meml	IF equal	Compare A and Meml, Do next if A = Meml
IFGT	A, Meml	IF greater than	Compare A and Meml, Do next if A > Meml
IFBNE	#	IF B not equal	Do next if lower 4 bits of B not = Imm
DRSZ	Reg	Decrement Reg. ,skip if zero	Reg ← Reg − 1, skip if Reg goes to 0
SBIT	#, Mem	Set bit	1 to Mem.bit (bit = 0 to 7 immediate)
RBIT	#, Mem	Reset bit	0 to Mem.bit (bit = 0 to 7 immediate)
IFBIT	#, Mem	If bit	If Mem.bit is true, do next instruction
X	A, Mem	Exchange A with memory	A ⟷ Mem
LD	A, Meml	Load A with memory	A ← Meml
LD	Mem, Imm	Load Direct memory Immed.	Mem ← Imm
LD	Reg, Imm	Load Register memory Immed.	Reg ← Imm
X	A, [B±]	Exchange A with memory [B]	A ⟷ [B] (B ← B±1)
X	A, [X±]	Exchange A with memory [X]	A ⟷ [X] (X ← X±1)
LD	A, [B±]	Load A with memory [B]	A ← [B] (B ← B±1)
LD	A, [X±]	Load A with memory [X]	A ← [X] (X ← X±1)
LD	[B±], Imm	Load memory immediate	[B] ← Imm (B ← B±1)
CLRA		Clear A	A ← 0
INC		Increment A	A ← A + 1
DEC		Decrement A	A ← A − 1
LAID	A	Load A indirect from ROM	A ← ROM(PU,A)
DCOR	A	Decimal Correct A	A ← BCD correction (follows ADC, SUBC)
RRC		Rotate right through carry	C → A7 → . . . → A0 → C
SWAP	A	Swap nibbles of A	A7 . . . A4 ⟷ A3 . . . A0
SC	A	Set C	C ← 1
RC		Reset C	C ← 0
IFC		If C	If C is true, do next instruction
IFNC		If Not C	If C is not true, do next instruction
JMPL		Jump absolute long	PC ← ii (ii = 15 bits, 0 to 32k)
JMP		Jump absolute	PC11...PC0 ← i (i = 12 bits) PC15...PC12 remain unchanged
JP	Addr.	Jump relative short	PC ← PC + r (r is −31 to +32, not 1)
JSRL	Addr.	Jump subroutine long	[SP] ← PL, [SP−1] ← PU, SP−2, PC ← ii
JSR		Jump subroutine	[SP] ← PL, [SP−1] ← PU, SP−2, PC11.. PC0 ← i
JID	Disp.	Jump indirect	PL ← ROM(PU, A)
RET	Addr.	Return from subroutine	SP+2, PL ← [SP], PU ← [SP−1]
RETSK	Addr.	Return and skip	SP+2, PL ← [SP], PU ← [SP−1], Skip next instruction
RETI		Return from interrupt	SP+2, PL ← [SP], PU ← [SP−1], GIE ← 1
INTR		Generate an interrupt	[SP] ← PL, [SP−1] ← PU, SP−2, PC ← 0FF
NOP		No operation	PC ← PC + 1

Opcode Table

Upper Nibble Bits 7–4																Lower Nibble Bits 3–0
F	E	D	C	B	A	9	8	7	6	5	4	3	2	1	0	
JP−15	JP−31	LD 0F0, #i	DRSZ 0F0	RRCA	RC	ADC A, #i	ADC A,[B]	IFBIT 0,[B]	*	LD B, 0F	IFBNE 0	JSR 0000−00FF	JMP 0000−00FF	JP+17	INTR	0
JP−14	JP−30	LD 0F1, #i	DRSZ 0F1	*	SC	SUBC A, #i	SUBC A,[B]	IFBIT 1,[B]	*	LD B, 0E	IFBNE 1	JSR 0100−01FF	JMP 0100−01FF	JP+18	JP+2	1
JP−13	JP−29	LD 0F2, #i	DRSZ 0F2	X A,[X+]	X A,[B+]	IFEQ A, #i	IFEQ A,[B]	IFBIT A,[B]	*	LD B, 0D	IFBNE 2	JSR 0200−02FF	JMP 0200−02FF	JP+19	UJP+3	2
JP−12	JP−28	LD 0F3, #i	DRSZ 0F3	X A,[X−]	X A,[B−]	IFGT A, #i	IFGT A,[B]	IFBIT 3,[B]	*	LD B, 0C	IFBNE 3	JSR 0300−03FF	JMP 0300−03FF	JP+20	JP+4	3
JP−11	JP−27	LD 0F4, #i	DRSZ 0F4	*	LAID	ADD A, #i	ADD A,[B]	IFBIT 4,[B]	CLRA	LD B, 0B	IFBNE 4	JSR 0400−04FF	JMP 0400−04FF	JP+21	JP+5	4
JP−10	JP−26	LD 0F5, #i	DRSZ 0F5	*	JID	AND A, #i	AND A,[B]	IFBIT 5,[B]	SWAPA	LD B, 0A	IFBNE 5	JSR 0500−05FF	JMP 0500−05FF	JP+22	JP+6	5
JP−9	JP−25	LD 0F6, #i	DRSZ 0F6	X A,[X]	X A,[B]	XOR A, #i	XOR A,[B]	IFBIT 6,[B]	DCORA	LD B, 9	IFBNE 6	JSR 0600−06FF	JMP 0600−06FF	JP+23	JP+7	6
JP−8	JP−24	LD 0F7, #i	DRSZ 0F7	*	*	OR A, #i	OR A,[B]	IFBIT 7,[B]	*	LD B, 8	IFBNE 7	JSR 0700−07FF	JMP 0700−07FF	JP+24	JP+8	7
JP−7	JP−23	LD 0F8, #i	DRSZ 0F8	NOP	*	LD A, #i	IFC	SBIT 0,[B]	RBIT 0,[B]	LD B, 7	IFBNE 8	JSR 0800−08FF	JMP 0800−08FF	JP+25	JP+9	8
JP−6	JP−22	LD 0F9, #i	DRSZ 0F9	*	*	*	IFNC	SBIT 1,[B]	RBIT 1,[B]	LD B, 6	IFBNE 9	JSR 0900−09FF	JMP 0900−09FF	JP+26	JP+10	9
JP−5	JP−21	LD 0FA, #i	DRSZ 0FA	LD A,[X+]	LD A,[B+]	LD [B+], #i	INCA	SBIT 2,[B]	RBIT 2,[B]	LD B, 5	IFBNE 0A	JSR 0A00−0AFF	JMP 0A00−0AFF	JP+27	JP+11	A
JP−4	JP−20	LD 0FB, #i	DRSZ 0FB	LD A,[X−]	LD A,[B−]	LD [B−], #i	DECA	SBIT 3,[B]	RBIT 3,[B]	LD B, 4	IFBNE 0B	JSR 0B00−0BFF	JMP 0B00−0BFF	JP+28	JP+12	B
JP−3	JP−19	LD 0FC, #i	DRSZ 0FC	LD Md, #i	JMPL	X A,Md	*	SBIT 4,[B]	RBIT 4,[B]	LD B, 3	IFBNE 0C	JSR 0C00−0CFF	JMP 0C00−0CFF	JP+29	JP+13	C
JP−2	JP−18	LD 0FD, #i	DRSZ 0FD	DIR	JSRL	LD A,Md	RETSK	SBIT 5,[B]	RBIT 5,[B]	LD B, 2	IFBNE 0D	JSR 0D00−0DFF	JMP 0D00−0DFF	JP+30	JP+14	D
JP−1	JP−17	LD 0FE, #i	DRSZ 0FE	LD A,[X]	LD A,[B]	LD [B], #i	RET	SBIT 6,[B]	RBIT 6,[B]	LD B, 1	IFBNE 0E	JSR 0E00−0EFF	JMP 0E00−0EFF	JP+31	JP+15	E
JP−0	JP−16	LD 0FF, #i	DRSZ 0FF	*	*	*	RETI	SBIT 7,[B]	RBIT 7,[B]	LD B, 0	IFBNE 0F	JSR 0F00−0FFF	JMP 0F00−0FFF	JP+32	JP+16	F

Where,

i is the immediate data

Md is a directly addressed memory location

* is an unused opcode

Note: The opcode 60 Hex is also the opcode for IFBIT #i,A

Instruction Execution Time

- Most instructions are single byte (with immediate addressing mode instructions requiring two bytes).
- Most single byte instructions take one cycle time to execute.
- Skipped instructions require x number of cycles to be skipped, where x equals the number of bytes in the skipped instruction opcode.

The following tables shows the number of bytes and cycles for each instruction in the format byte/cycle.

Arithmetic and Logic Instructions (Bytes/Cycles)

Instr	[B]	Direct	Immediate
ADD	1/1	3/4	
ADC	1/1	3/4	2/2
SUBC	1/1	3/4	2/2
AND	1/1	3/4	2/2
OR	1/1	3/4	2/2
XOR	1/1	3/4	2/2
IFEQ	1/1	3/4	2/2
IFNE	1/1	3/4	2/2
IFGT	1/1	3/4	2/2
IFBNE	1/1		2/2
DRSZ	1/1	1/3	
SBIT	1/1	3/4	
RBIT	1/1	3/4	
IFBIT	1/1	3/4	

Instuctions Using A and C (Bytes/Cycles)

Instr	Bytes/Cycles
CLRA	1/1
INCA	1/1
DECA	1/1
LAID	1/3
DCOR	1/1
RRCA	1/1
SWAPA	1/1
SC	1/1
RC	1/1
IFC	1/1
IFNC	1/1

Transfer of Control Instructions (Bytes/Cycles)

Instr	Bytes/Cycles
JMPL	3/4
JMP	2/3
JP	1/3
JSRL	3/5
JSR	2/5
JID	1/3
RET	1/5
RETSK	1/5
RETI	1/5
INTR	1/7
NOP	1/1

Memory Transfer Instructions (Bytes/Cycles)

Instr	Register Indirect		Direct	Immed.	Register Indirect Auto Incr and Decr	
	[B]	[X]			[B+, B−]	[X+, X−]
X A, (Note a)	1/1		2/3		1/2	
LD A, (Note a)	1/1		2/3		1/2	
LD B,Imm		1/3		2/2		1/3
LD B,Imm		1/3		1/1 (Note b)		1/3
LD Mem,Imm			3/3	2/3 (Note c)	2/2	
LD Reg,Imm	2/2		2/3			

Note a: Memory location addressed by B or X or directly

Note b: IF B < 16

Note c: IF B > 15

Mask Option

The mask programmable options are listed below. The options are programmed at the same time as the ROM pattern to provide the user with hardware flexibility to use a variety of oscillator configurations and the Brown Out feature.

The following option information is to be sent to National along with the EPROM. Contact the sales office for more details.

OPTION 1: CKI INPUT

= 1 Crystal (CKI/lO) G7/CKO for crystal configuration
= 2 External (CKI/lO) G7 available as input
= 3 R/C (CKI/lO) G7 available as input

OPTION 2: "Brown Out"

= 1 Enable Brown Out Detection
= 2 Disable Brown Out Detection

OPTION 3: BONDING

= 1 28-Pin DIP/SO Package
= 2 20-Pin DIP/SO Package

How to Order

To order a complete development package, select the section for the microcontroller to be developed and order the parts listed. Contact the sales office for more details.

Development Support

SUMMARY

- iceMASTER™: IM-COP8/400—Full feature in-circuit emulation for all COP8 products. A full set of COP8 Basic and Feature Family device and package specific probes are available.
- COP8 Debug Module: Moderate cost in-circuit emulation and development programming unit.
- COP8 Evaluation and Programming Unit: EPU-COP880C—low cost In-circuit simulation and development programming unit.
- Assembler: COP8-DEV-IBMA. A DOS installable cross development Assembler, Linker, Librarian and Utility Software Development Tool Kit.
- C Compiler: COP8C. A DOS installable cross development Software Tool Kit.
- OTP/EPROM Programmer Support: Covering needs from engineering prototype, pilot production to full production environments.

Development Support (Continued)

IceMASTER (IM) IN-CIRCUIT EMULATION

The iceMASTER IM-COP8/400 is a full feature, PC based, in-circuit emulation tool developed and marketed by MetaLink Corporation to support the whole COP8 family of products. National is a resale vendor for these products.

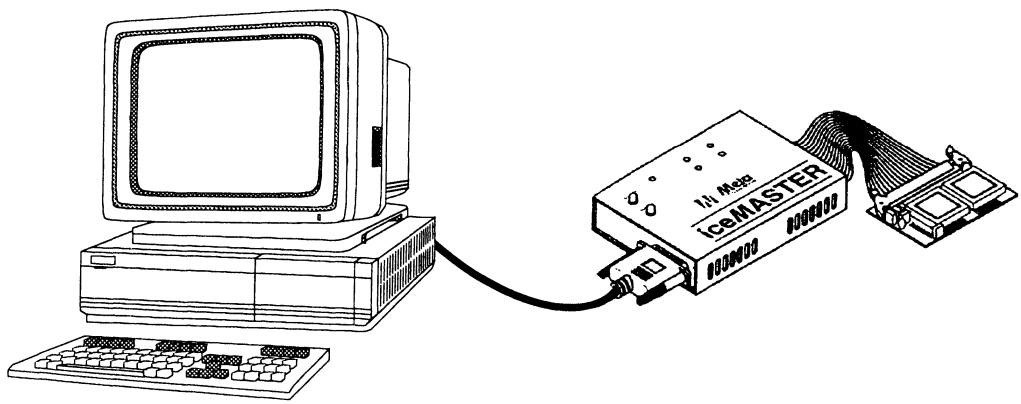
See *Figure 15* for configuration.

The iceMASTER IM-COP8/400 with its device specific COP8 Probe provides a rich feature set for developing, testing and maintaining product:

- Real-time in-circuit emulation; full 2.4VDC–5.5VDC operation range, full DC-10 MHz clock. Chip options are programmable or jumper selectable.
- Direct connection to application board by package compatible socket or surface mount assembly.
- Full 32 kbyte of loadable programming space that overlays (replaces) the on-chip ROM or EPROM. On-chip RAM and I/O blocks are used directly or recreated on the probe as necessary.
- Full 4k frame synchronous trace memory. Address, instruction, and 8 unspecified, circuit connectable trace lines. Display can be HLL source (e.g., C source), assembly or mixed.
- A full 64k hardware configurable break, trace on, trace off control, and pass count increment events.
- Tool set integrated interactive symbolic debugger—supports both assembler (COFF) and C Compiler (.COD) linked object formats.
- Real time performance profiling analysis; selectable bucket definition.
- Watch windows, content updated automatically at each execution break.
- Instruction by instruction memory/register changes displayed on source window when in single step operation.
- Single base unit and debugger software reconfigurable to support the entire COP8 family; only the probe personality needs to change. Debugger software is processor customized, and reconfigured from a master model file.
- Processor specific symbolic display of registers and bit level assignments, configured from master model file.
- Halt/Idle mode notification.
- On-line HELP customized to specific processor using master model file.
- Includes a copy of COP8-DEV-IBMA assembler and linker SDK.

IM Order Information

Base Unit	
IM-COP8/400-1	iceMASTER Base Unit, 110V Power Supply
IM-COP8/400-2	iceMASTER Base Unit, 220V Power Supply
iceMASTER Probe	
MHW-840CJ28DWPC MHW-840CJ20DWPC	28 DIP 20 DIP
MHW-SOIC28 MHW-SOIC20	28 SOIC Adapter Kit 20 SOIC Adapter Kit



TL/DD/12851–29

FIGURE 15. COP8 iceMASTER Environment

Development Support (Continued)

IceMASTER DEBUG MODULE (DM)

The iceMASTER Debug Module is a PC based, combination in-circuit emulation tool and COP8 based OTP/EPROM programming tool developed and marketed by MetaLink Corporation to support the whole COP8 family of products. National is a resale vendor for these products.

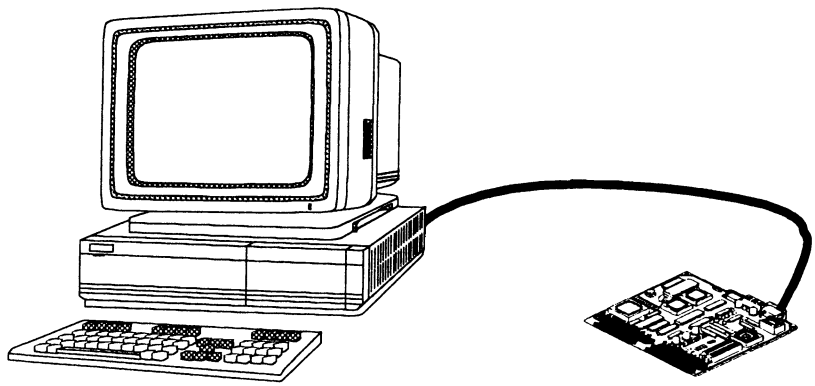
See *Figure 16* for configuration.

The iceMASTER Debug Module is a moderate cost development tool. It has the capability of in-circuit emulation for a specific COP8 microcontroller and in addition serves as a programming tool for COP8 OTP and EPROM product families. Summary of features is as follows:

- Real-time in-circuit emulation; full operating voltage range operation, full DC-10 MHz clock.
- All processor I/O pins can be cabled to an application development board with package compatible cable to socket and surface mount assembly.
- Full 32 kbytes of loadable programming space that overlays (replaces) the on-chip ROM or EPROM. On-chip RAM and I/O blocks are used directly or recreated as necessary.
- 100 frames of synchronous trace memory. The display can be HLL source (C source), assembly or mixed. The most recent history prior to a break is available in the trace memory.
- Configured break points; uses INTR instruction which is modestly intrusive.
- Software—only supported features are selectable.
- Tool set integrated interactive symbolic debugger—supports both assembler (COFF) and C Compiler (.COD) SDK linked object formats.
- Instruction by instruction memory/register changes displayed when in single step operation.
- Debugger software is processor customized, and reconfigured from a master model file.
- Processor specific symbolic display of registers and bit level assignments, configured from master model file.
- Halt/Idle mode notification.
- Programming menu supports full product line of programmable OTP and EPROM COP8 products. Program data is taken directly from the overlay RAM.
- Programming of 44 PLCC and 68 PLCC parts requires external programming adapters.
- Includes wall mount power supply.
- On-board V_{PP} generator from 5V input or connection to external supply supported. Requires V_{PP} level adjustment per the family programming specification (correct level is provided on an on-screen pop-down display).
- On-line HELP customized to specific processor using master model file.
- Includes a copy of COP8-DEV-IBMA assembler and linker SDK.

DM Order Information

Debug Module Unit	
COP8-DM/840CJ	
Cable Adapters	
DM-COP8/28D	28 DIP cable
DM-COP8/28D-SO	28 DIP to 28 SOIC adapter
DM-COP8/20D	20 DIP cable
DM-COP8/20D-SO	20 DIP to 20 SOIC adapter



TL/DD/12851-30

FIGURE 16. COP8-DM Environment

Development Support (Continued)

COP8 ASSEMBLER/LINKER SOFTWARE DEVELOPMENT TOOL KIT

National Semiconductor offers a relocateable COP8 macro cross assembler, linker, librarian and utility software development tool kit. Features are summarized as follows:

- Basic and Feature Family instruction set by "device" type.
- Nested macro capability.
- Extensive set of assembler directives.
- Supported on PC/DOS platform.
- Generates National standard COFF output files.
- Integrated Linker and Librarian.
- Integrated utilities to generate ROM code file outputs.
- DUMPCOFF utility.

This product is integrated as a part of MetaLink tools as a development kit, fully supported by the MetaLink debugger. It may be ordered separately or it is bundled with the MetaLink products at no additional cost.

Order Information

Assembler SDK	
COP8-DEV-IBMA	Assembler SDK on installable 3.5" PC/DOS Floppy Disk Drive format. Periodic upgrades and most recent version is available on National's BBS and Internet

COP8 C COMPILER

A C Compiler is developed and marketed by Byte Craft Limited. The COP8C compiler is a fully integrated development tool specifically designed to support the compact embedded configuration of the COP8 family of products.

Features are summarized as follows:

- ANSI C with some restrictions and extensions that optimize development for the COP8 embedded application.
- BITS data type extension. Register declaration #pragma with direct bit level definitions.
- C language support for interrupt routines.
- Expert system, rule based code generation and optimization.
- Performs consistency checks against the architectural definitions of the target COP8 device.
- Generates program memory code.
- Supports linking of compiled object or COP8 assembled object formats.
- Global optimization of linked code.
- Symbolic debug load format fully source level supported by the MetaLink debugger.

INDUSTRY WIDE OTP/EPROM PROGRAMMING SUPPORT

Programming support, in addition to the MetaLink development tools, is provided by a full range of independent approved vendors to meet the needs from the engineering laboratory to full production.

Approved List

Manufacturer	North America	Europe	Asia
BP Microsystems	(800) 225-2102 (713) 688-4600 Fax: (713) 688-0920	+49-8152-4183 +49-8856-932616	+852-234-16611 +852-2710-8121
Data I/O	(800) 426-1045 (206) 881-6444 Fax: (206) 882-1043	+44-0734-440011	Call North America
HI-LO	(510) 623-8860	Call Asia	+886-2-764-0215 Fax: +886-2-756-6403
ICE Technology	(800) 624-8949 (919) 430-7915	+44-1226-767404 Fax: 0-1226-370-434	
MetaLink	(800) 638-2423 (602) 926-0797 Fax: (602) 693-0681	+49-80 9156 96-0 Fax: +49-80 9123 86	+852-737-1800
Systems General	(408) 263-6667	+41-1-9450300	+886-2-917-3005 Fax: +886-2-911-1283
Needhams	(916) 924-8037 Fax: (916) 924-8065		

Development Support (Continued)

AVAILABLE LITERATURE

For more information, please see the COP8 Basic Family User's Manual, Literature Number 620895, COP8 Feature Family User's Manual, Literature Number 620897 and National's Family of 8-bit Microcontrollers COP8 Selection Guide, Literature Number 630009.

DIAL-A-HELPER SERVICE

Dial-A-Helper is a service provided by the Microcontroller Applications group. The Dial-A-Helper is an Electronic Information System that may be accessed as a Bulletin Board System (BBS) via data modem, as an FTP site on the Internet via standard FTP client application or as an FTP site on the Internet using a standard Internet browser such as Netscape or Mosaic.

The Dial-A-Helper system provides access to an automated information storage and retrieval system. The system capabilities include a MESSAGE SECTION (electronic mail, when accessed as a BBS) for communications to and from the Microcontroller Applications Group and a FILE SECTION which consists of several file areas where valuable application software and utilities could be found.

DIAL-A-HELPER BBS via a Standard Modem

Modem: CANADA/U.S.: (800) NSC-MICRO
(800) 672-6427
EUROPE: (+49) 0-8141-351332
Baud: 14.4k
Set-up: Length: 8-Bit
Parity: None
Stop Bit: 1
Operation: 24 Hrs., 7 Days

DIAL-A-HELPER via FTP

ftp nscmicro.nsc.com
user: anonymous
password: username@yourhost.site.domain

DIAL-A-HELPER via a WorldWide Web Browser

ftp://nscmicro.nsc.com

National Semiconductor on the WorldWide Web

See us on the WorldWide Web at: http://www.national.com

CUSTOMER RESPONSE CENTER

Complete product information and technical support is available from National's customer response centers.

CANADA/U.S.:	Tel:	(800) 272-9959
	email:	support @tevm2.nsc.com
EUROPE:	email:	europe.support@nsc.com
	Deutsch Tel:	+49 (0) 180-530 85 85
	English Tel:	+49 (0) 180-532 78 32
	Français Tel:	+49 (0) 180-532 93 58
	Italiano Tel:	+49 (0) 180-534 16 80
JAPAN:	Tel:	+81-043-299-2309
S.E. ASIA:	Beijing Tel:	(+86) 10-6856-8601
	Shanghai Tel:	(+86) 21-6415-4092
	Hong Kong Tel:	(+852) 2737-1600
	Korea Tel:	(+82) 2-3771-6909
	Malaysia Tel:	(+60-4) 644-9061
	Singapore Tel:	(+65) 255-2226
	Taiwan Tel:	+886-2-521-3288
AUSTRALIA:	Tel:	(+61) 3-9558-9999
INDIA:	Tel:	(+91) 80-559-9467

National Semiconductor

COP680C/COP681C/COP682C/COP880C/COP881C/ COP882C/COP980C/COP981C/COP982C Microcontrollers

General Description

The COP680C/COP681C/COP682C/COP880C/COP881C /COP882C/COP980C/COP981C and COP982C are members of the COPS™ microcontroller family. They are fully static parts, fabricated using double-metal silicon gate microCMOS technology. This low cost microcontroller is a complete microcomputer containing all system timing, interrupt logic, ROM, RAM, and I/O necessary to implement dedicated control functions in a variety of applications. Features include an 8-bit memory mapped architecture, MICROWIRE/PLUS™ serial I/O, a 16-bit timer/counter with capture register and a multi-sourced interrupt. Each I/O pin has software selectable options to adapt the device to the specific application. The part operates over a voltage range of 2.5 to 6.0V. High throughput is achieved with an efficient, regular instruction set operating at a 1 microsecond per instruction rate.

Key Features

- 16-bit multi-function timer supporting
  - — PWM mode
  - — External event counter mode
  - — Input capture mode
- 4 kbytes of ROM
- 128 bytes of RAM

I/O Features

- Memory mapped I/O
- Software selectable I/O options (TRI-STATE®, Push-Pull, Weak Pull-Up Input, High Impedance Input)
- High current outputs (8 pins)
- Schmitt trigger inputs on Port G
- MICROWIRE PLUS serial I/O
- Packages:
  - — 20 DIP/SO with 16 I/O pins
  - — 28 DIP/SO with 24 I/O pins
  - — 40 DIP, 36 I/O pins
  - — 44 PLCC, 36 I/O pins

CPU/Instruction Set Features

- 1 μs instruction cycle time
- Three multi-source interrupts servicing
  - — External interrupt with selectable edge
  - — Timer interrupt
  - — Software interrupt
- Versatile and easy to use instruction set
- 8-bit Stack Pointer (SP)—stack in RAM
- Two 8-bit Register Indirect Data Memory Pointers (B and X)

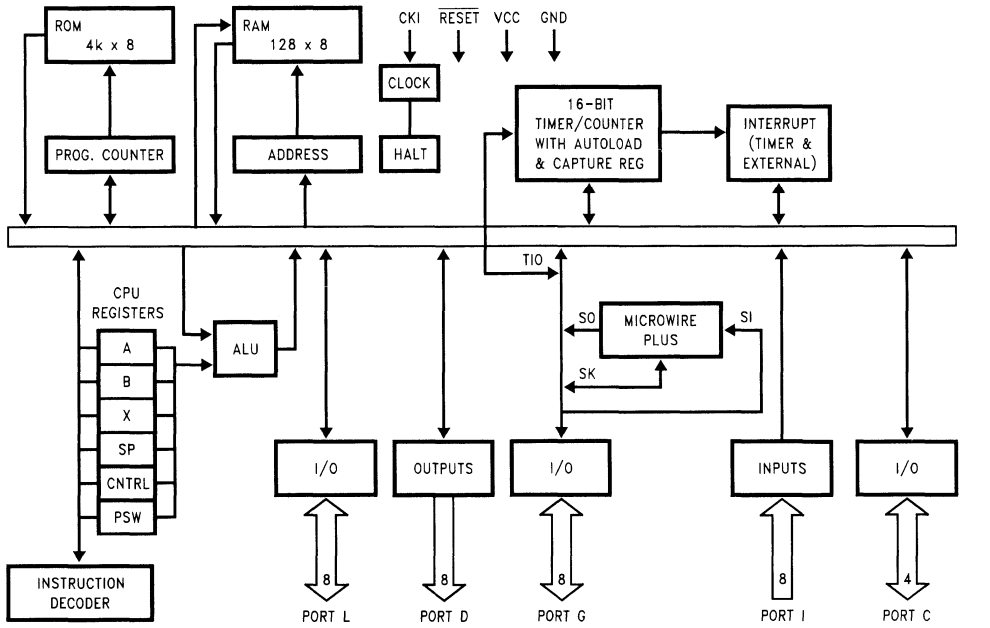
Fully Static CMOS

- Low current drain (typically < 1 μA)
- Single supply operation: 2.5V to 6.0V
- Temperature ranges: 0°C to 70°C, −40°C to +85°C, −55°C to +125°C.

Development Support

- Emulation and OTP devices
- Real time emulation and full program debug offered by MetaLink's development system

Block Diagram

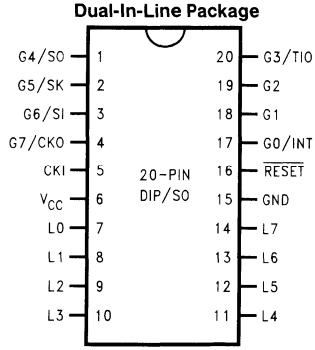


TL/DD/10802–1

FIGURE 1

Connection Diagrams

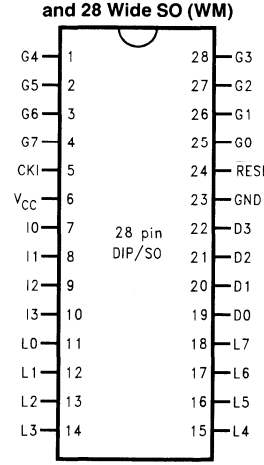
Dual-In-Line Package



TL/DD/10802-23

Top View
Order Number COP882C-XXX/N, COP982C-XXX/N, COP882C-XXX/WM, COP982C-XXX/WM, COP982C-XXX/N or COP982CH-XXX/WM

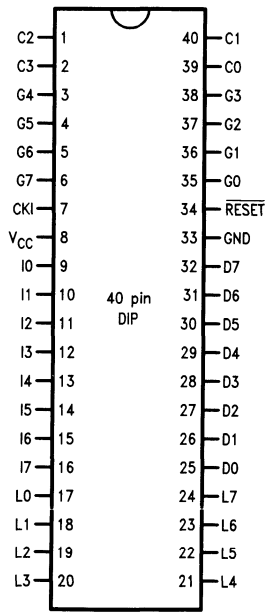
Dual-In-Line Package (N) and 28 Wide SO (WM)



TL/DD/10802-5

Top View
Order Number COP881C-XXX/N, COP981C-XXX/N, COP881C-XXX/WM, COP981C-XXX/WM, COP981CH-XXX/N or COP981CH-XXX/WM

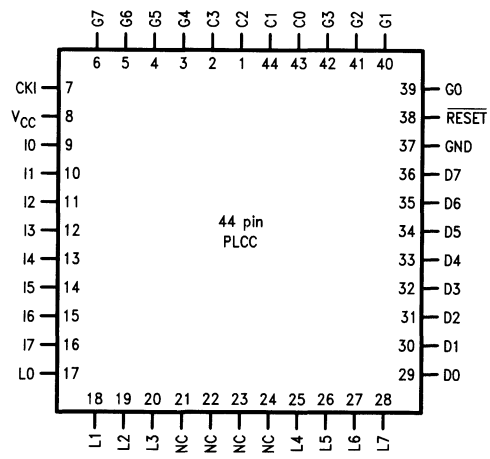
Dual-In-Line Package



TL/DD/10802-4

Top View
Order Number COP680C-XXX/N, COP880C-XXX/N, COP980C-XXX/N or COP980CH-XXX/N

Plastic Chip Carrier



TL/DD/10802-3

Top View
Order Number COP680C-XXX/V, COP880C-XXX/V, COP980C-XXX/V or COP980CH-XXX/V

FIGURE 3. Connection Diagrams

COP980C/COP981C/COP982C

Absolute Maximum Ratings

If Military/Aerospace specified devices are required, please contact the National Semiconductor Sales Office/Distributors for availability and specifications.

Supply Voltage (V_{CC})	7V
Voltage at any Pin	−0.3V to V_{CC} + 0.3V
Total Current into V_{CC} Pin (Source)	50 mA
Total Current out of GND Pin (Sink)	60 mA
Storage Temperature Range	−65°C to +140°C

Note: Absolute maximum ratings indicate limits beyond which damage to the device may occur. DC and AC electrical specifications are not ensured when operating the device at absolute maximum ratings.

DC Electrical Characteristics COP98xC; 0°C ≤ T_A ≤ +70°C unless otherwise specified

Parameter	Condition	Min	Typ	Max	Units
Operating Voltage					
98XC		2.3		4.0	V
98XCH		4.0		6.0	V
Power Supply Ripple (Note 1)	Peak to Peak			0.1 V_{CC}	V
Supply Current					
CKI = 10 MHz	V_{CC} = 6V, tc = 1 μs			6.0	mA
CKI = 4 MHz	V_{CC} = 6V, tc = 2.5 μs			4.4	mA
CKI = 4 MHz	V_{CC} = 4.0V, tc = 2.5 μs			2.2	mA
CKI = 1 MHz	V_{CC} = 4.0V, tc = 10 μs			1.4	mA
(Note 2)					
HALT Current	V_{CC} = 6V, CKI = 0 MHz		<0.7	8	μA
(Note 3)	V_{CC} = 4.0V, CKI = 0 MHz		<0.4	5	μA
Input Levels					
RESET, CKI					
Logic High		0.9 V_{CC}			V
Logic Low				0.1 V_{CC}	V
All Other Inputs					
Logic High		0.7 V_{CC}			V
Logic Low				0.2 V_{CC}	V
Hi-Z Input Leakage	V_{CC} = 6.0V	−1.0		+1.0	μA
Input Pullup Current	V_{CC} = 6.0V, V_{IN} = 0V	−40		−250	μA
G Port Input Hysteresis				0.35 V_{CC}	V
Output Current Levels					
D Outputs					
Source	V_{CC} = 4.5V, V_{OH} = 3.8V	−0.4			mA
	V_{CC} = 2.3V, V_{OH} = 1.6V	−0.2			mA
Sink	V_{CC} = 4.5V, V_{OL} = 1.0V	10			mA
	V_{CC} = 2.3V, V_{OL} = 0.4V	2			mA
All Others					
Source (Weak Pull-Up)	V_{CC} = 4.5V, V_{OH} = 3.2V	−10		−110	μA
	V_{CC} = 2.3V, V_{OH} = 1.6V	−2.5		−33	μA
Source (Push-Pull Mode)	V_{CC} = 4.5V, V_{OH} = 3.8V	−0.4			mA
	V_{CC} = 2.3V, V_{OH} = 1.6V	−0.2			
Sink (Push-Pull Mode)	V_{CC} = 4.5V, V_{OL} = 0.4V	1.6			mA
	V_{CC} = 2.3V, V_{OL} = 0.4V	0.7			
TRI-STATE Leakage	V_{CC} = 6.0V	−1.0		+1.0	μA
Allowable Sink/Source Current Per Pin					
D Outputs (Sink)				15	mA
All Others				3	mA
Maximum Input Current (Note 4) Without Latchup (Room Temp)	Room Temp			±100	mA
RAM Retention Voltage, Vr (Note 5)	500 ns Rise and Fall Time (Min)	2.0			V
Input Capacitance				7	pF
Load Capacitance on D2				1000	pF

DC Electrical Characteristics (Continued)

Note 1: Rate of voltage change must be less than 0.5V/ms.

Note 2: Supply current is measured after running 2000 cycles with a square wave CKI input, CKO open, inputs at rails and outputs open.

Note 3: The HALT mode will stop CKI from oscillating in the RC and the Crystal configurations. Test conditions: All inputs tied to V_{CC}, L, C and G ports TRI-STATE and tied to ground, all outputs low and tied to ground.

Note 4: Pins G6 and $\overline{\text{RESET}}$ are designed with a high voltage input network for factory testing. These pins allow input voltages greater than V_{CC} and the pins will have sink current to V_{CC} when biased at voltages greater than V_{CC} (the pins do not have source current when biased at a voltage below V_{CC}). The effective resistance to V_{CC} is 750Ω (typ). These two pins will not latch up. The voltage at the pins must be limited to less than 14V.

Note 5: To maintain RAM integrity, the voltage must not be dropped or raised instantaneously.

AC Electrical Characteristics $0°C \le T_A \le +70°C$ unless otherwise specified

Parameter	Condition	Min	Typ	Max	Units
Instruction Cycle Time (tc)					
Crystal/Resonator or External	$V_{CC} \ge 4.0V$	1		DC	μs
(Div-by 10)	$2.3V \le V_{CC} \le 4.0V$	2.5		DC	μs
R/C Oscillator Mode	$V_{CC} \ge 4.0V$	3		DC	μs
(Div-by 10)	$2.3V \le V_{CC} \le 4.0V$	7.5		DC	μs
CKI Clock Duty Cycle (Note 6)	fr = Max	40		60	%
Rise Time (Note 6)	fr = 10 MHz Ext Clock			12	ns
Fall Time (Note 6)	fr = 10 MHz Ext Clock			8	ns
Inputs					
t_{SETUP}	$V_{CC} \ge 4.0V$	200			ns
	$2.3V \le V_{CC} \le 4.0V$	500			ns
t_{HOLD}	$V_{CC} \ge 4.0V$	60			ns
	$2.3V \le V_{CC} \le 4.0V$	150			ns
Output Propagation Delay	$C_L = 100$ pF, $R_L = 2.2$ kΩ				
t_{PD1}, t_{PD0}					
SO, SK	$V_{CC} \ge 4.0V$			0.7	μs
	$2.3V \le V_{CC} \le 4.0V$			1.75	μs
All Others	$V_{CC} \ge 4.0V$			1	μs
	$2.3V \le V_{CC} \le 4.0V$			2.5	μs
MICROWIRE™ Setup Time (t_{UWS})		20			ns
MICROWIRE Hold Time (t_{UWH})		56			ns
MICROWIRE Output Propagation Delay (t_{UPD})				220	ns
Input Pulse Width					
Interrupt Input High Time		t_C			
Interrupt Input Low Time		t_C			
Timer Input High Time		t_C			
Timer Input Low Time		t_C			
Reset Pulse Width		1.0			μs

Note 6: Parameter characterized but not production tested.

Absolute Maximum Ratings

If Military/Aerospace specified devices are required, please contact the National Semiconductor Sales Office/Distributors for availability and specifications.

Supply Voltage (V_{CC})	7V
Voltage at any Pin	−0.3V to V_{CC} + 0.3V
Total Current into V_{CC} Pin (Source)	50 mA
Total Current out of GND Pin (Sink)	60 mA
Storage Temperature Range	−65°C to +140°C

Note: Absolute maximum ratings indicate limits beyond which damage to the device may occur. DC and AC electrical specifications are not ensured when operating the device at absolute maximum ratings.

DC Electrical Characteristics

COP88xC; −40°C ≤ T_A ≤ +85°C unless otherwise specified

Parameter	Condition	Min	Typ	Max	Units
Operating Voltage		2.5		6.0	V
Power Supply Ripple (Note 1)	Peak to Peak			0.1 V_{CC}	V
Supply Current					
CKI = 10 MHz	V_{CC} = 6V, tc = 1 μs			6.0	mA
CKI = 4 MHz	V_{CC} = 6V, tc = 2.5 μs			4.4	mA
CKI = 4 MHz	V_{CC} = 4.0V, tc = 2.5 μs			2.2	mA
CKI = 1 MHz	V_{CC} = 4.0V, tc = 10 μs			1.4	mA
(Note 2)					
HALT Current	V_{CC} = 6V, CKI = 0 MHz		<1	10	μA
(Note 3)	V_{CC} = 3.5V, CKI = 0 MHz		<0.5	6	μA
Input Levels					
$\overline{RESET}$, CKI					
Logic High		0.9 V_{CC}			V
Logic Low				0.1 V_{CC}	V
All Other Inputs					
Logic High		0.7 V_{CC}			V
Logic Low				0.2 V_{CC}	V
Hi-Z Input Leakage	V_{CC} = 6.0V	−2		+2	μA
Input Pullup Current	V_{CC} = 6.0V, V_{IN} = 0V	−40		−250	μA
G Port Input Hysteresis				0.35 V_{CC}	V
Output Current Levels					
D Outputs					
Source	V_{CC} = 4.5V, V_{OH} = 3.8V	−0.4			mA
	V_{CC} = 2.5V, V_{OH} = 1.8V	−0.2			mA
Sink	V_{CC} = 4.5V, V_{OL} = 1.0V	10			mA
	V_{CC} = 2.5V, V_{OL} = 0.4V	2			mA
All Others					
Source (Weak Pull-Up)	V_{CC} = 4.5V, V_{OH} = 3.2V	−10		−110	μA
	V_{CC} = 2.5V, V_{OH} = 1.8V	−2.5		−33	μA
Source (Push-Pull Mode)	V_{CC} = 4.5V, V_{OH} = 3.8V	−0.4			mA
	V_{CC} = 2.5V, V_{OH} = 1.8V	−0.2			
Sink (Push-Pull Mode)	V_{CC} = 4.5V, V_{OL} = 0.4V	1.6			mA
	V_{CC} = 2.5V, V_{OL} = 0.4V	0.7			
TRI-STATE Leakage	V_{CC} = 6.0V	−2.0		+2.0	μA
Allowable Sink/Source Current Per Pin					
D Outputs (Sink)				15	mA
All Others				3	mA
Maximum Input Current (Note 4) Without Latchup (Room Temp)	Room Temp			±100	mA
RAM Retention Voltage, Vr (Note 5)	500 ns Rise and Fall Time (Min)	2.0			V
Input Capacitance				7	pF
Load Capacitance on D2				1000	pF

COP880C/COP881C/COP882C

DC Electrical Characteristics (Continued)

Note 1: Rate of voltage change must be less than 0.5V/ms.

Note 2: Supply current is measured after running 2000 cycles with a square wave CKI input, CKO open, inputs at rails and outputs open.

Note 3: The HALT mode will stop CKI from oscillating in the RC and the Crystal configurations. Test conditions: All inputs tied to V_{CC}, L, C and G ports TRI-STATE and tied to ground, all outputs low and tied to ground.

Note 4: Pins G6 and $\overline{\text{RESET}}$ are designed with a high voltage input network for factory testing. These pins allow input voltages greater than V_{CC} and the pins will have sink current to V_{CC} when biased at voltages greater than V_{CC} (the pins do not have source current when biased at a voltage below V_{CC}). The effective resistance to V_{CC} is 750Ω (typ). These two pins will not latch up. The voltage at the pins must be limited to less than 14V.

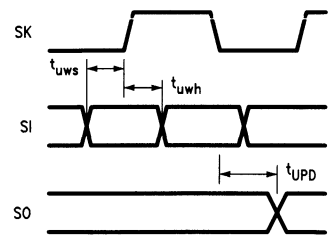
Note 5: To maintain RAM integrity, the voltage must not be dropped or raised instantaneously.

AC Electrical Characteristics $-40°C \le T_A \le +85°C$ unless otherwise specified

Parameter	Condition	Min	Typ	Max	Units
Instruction Cycle Time (tc)					
Crystal/Resonator or External	$V_{CC} \ge 4.5V$	1		DC	μs
(Div-by 10)	$2.5V \le V_{CC} < 4.5V$	2.5		DC	μs
R/C Oscillator Mode	$V_{CC} \ge 4.5V$	3		DC	μs
(Div-by 10)	$2.5V \le V_{CC} < 4.5V$	7.5		DC	μs
CKI Clock Duty Cycle (Note 6)	fr = Max	40		60	%
Rise Time (Note 6)	fr = 10 MHz Ext Clock			12	ns
Fall Time (Note 6)	fr = 10 MHz Ext Clock			8	ns
Inputs					
t_{SETUP}	$V_{CC} \ge 4.5V$	200			ns
	$2.5V \le V_{CC} < 4.5V$	500			ns
t_{HOLD}	$V_{CC} \ge 4.5V$	60			ns
	$2.5V \le V_{CC} < 4.5V$	150			ns
Output Propagation Delay	$C_L = 100$ pF, $R_L = 2.2$ kΩ				
t_{PD1}, t_{PD0}					
SO, SK	$V_{CC} \ge 4.5V$			0.7	μs
	$2.5V \le V_{CC} < 4.5V$			1.75	μs
All Others	$V_{CC} \ge 4.5V$			1	μs
	$2.5V \le V_{CC} < 4.5V$			2.5	μs
MICROWIRE™ Setup Time (t_{UWS})		20			ns
MICROWIRE Hold Time (t_{UWH})		56			ns
MICROWIRE Output					
Propagation Delay (t_{UPD})				220	ns
Input Pulse Width					
Interrupt Input High Time		t_C			
Interrupt Input Low Time		t_C			
Timer Input High Time		t_C			
Timer Input Low Time		t_C			
Reset Pulse Width		1.0			μs

Note 6: Parameter characterized but not production tested.

Timing Diagram



TL/DD/10802-2

FIGURE 2. MICROWIRE/PLUS Timing

COP680C/COP681C/COP682C

Absolute Maximum Ratings

If Military/Aerospace specified devices are required, please contact the National Semiconductor Sales Office/Distributors for availability and specifications.

Supply Voltage (V_{CC})	6V
Voltage at Any Pin	−0.3V to V_{CC} + 0.3V
Total Current into V_{CC} Pin (Source)	40 mA
Total Current Out of GND Pin (Sink)	48 mA
Storage Temperature Range	−65°C to +140°C

Note: *Absolute maximum ratings indicate limits beyond which damage to the device may occur. DC and AC electrical specifications are not ensured when operating the device at absolute maximum ratings.*

DC Electrical Characteristics

COP68xC: $-55°C \le T_A \le +125°C$ unless otherwise specified

Parameter	Condition	Min	Typ	Max	Units
Operating Voltage		4.5		5.5	V
Power Supply Ripple (Note 1)	Peak to Peak			0.1 V_{CC}	V
Supply Current (Note 2)					
CKI = 10 MHz	V_{CC} = 5.5V, tc = 1 μs			8.0	mA
CKI = 4 MHz	V_{CC} = 5.5V, tc = 2.5 μs			4.4	mA
HALT Current (Note 3)	V_{CC} = 5.5V, CKI = 0 MHz		<10	30	μA
Input Levels					
$\overline{RESET}$, CKI					
Logic High		0.9 V_{CC}			V
Logic Low				0.1 V_{CC}	V
All Other Inputs					
Logic High		0.7 V_{CC}			V
Logic Low				0.2 V_{CC}	V
Hi-Z Input Leakage	V_{CC} = 5.5V	−5		+5	μA
Input Pullup Current	V_{CC} = 5.5V, V_{IN} = 0V	−35		−300	μA
G Port Input Hysteresis				0.35 V_{CC}	V
Output Current Levels					
D Outputs					
Source	V_{CC} = 4.5V, V_{OH} = 3.8V	−0.35			mA
Sink	V_{CC} = 4.5V, V_{OL} = 1.0V	9			mA
All Others					
Source (Weak Pull-Up)	V_{CC} = 4.5V, V_{OH} = 3.2V	−9		−120	μA
Source (Push-Pull Mode)	V_{CC} = 4.5V, V_{OH} = 3.2V	−0.35			mA
Sink (Push-Pull Mode)	V_{CC} = 4.5V, V_{OL} = 0.4V	1.4			mA
TRI-STATE Leakage	V_{CC} = 5.5V	−5.0		+5.0	μA
Allowable Sink/Source Current per Pin					
D Outputs (Sink)				12	mA
All Others				2.5	mA
Maximum Input Current (Room Temp) without Latchup (Note 4)	Room Temp			±100	mA
RAM Retention Voltage, Vr (Note 5)	500 ns Rise and Fall Time (Min)	2.5			V
Input Capacitance				7	pF
Load Capacitance on D2				1000	pF

Note 1: Rate of voltage change must be less than 0.5V/ms.

Note 2: Supply current is measured after running 2000 cycles with a square wave CKI input, CKO open, inputs at rails and outputs open.

Note 3: The HALT mode will stop CKI from oscillating in the RC and the Crystal configurations. Test conditions: All inputs tied to V_{CC}, L and G ports TRI-STATE and tied to ground, all outputs low and tied to ground.

Note 4: Pins G6 and $\overline{RESET}$ are designed with a high voltage input network for factory testing. These pins allow input voltages greater than V_{CC} and the pins will have sink current to V_{CC} when biased at voltages greater than V_{CC} (the pins do not have source current when biased at a voltage below V_{CC}). The effective resistance to V_{CC} is 750Ω (typical). These two pins will not latch up. The voltage at the pins must be limited to less than 14V.

Note 5: To maintain RAM integrity, the voltage must not be dropped or raised instantaneously.

COP680C/COP681C/COP682C

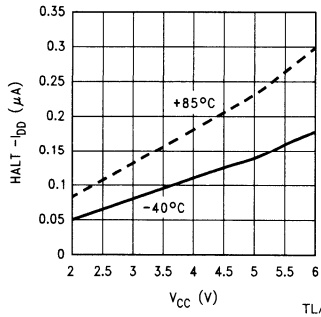
AC Electrical Characteristics $-55°C \leq T_A \leq +125°C$ unless otherwise specified

Parameter	Condition	Min	Typ	Max	Units
Instruction Cycle Time (tc)					
Ext. or Crystal/Resonant (Div-by 10)	$V_{CC} \geq 4.5V$	1		DC	μs
CKI Clock Duty Cycle (Note 6)	fr = Max	40		60	%
Rise Time (Note 6)	fr = 10 MHz Ext Clock			12	ns
Fall Time (Note 6)	fr = 10 MHz Ext Clock			8	ns
MICROWIRE Setup Time (t_{UWS})		20			ns
MICROWIRE Hold Time (t_{UWH})		56			ns
MICROWIRE Output Valid Time (t_{UPD})				220	ns
Input Pulse Width					
Interrupt Input High Time		t_C			
Interrupt Input Low Time		t_C			
Timer Input High Time		t_C			
Timer Input Low Time		t_C			
Reset Pulse Width		1			μs

Note 6: Parameter characterized but not production tested.

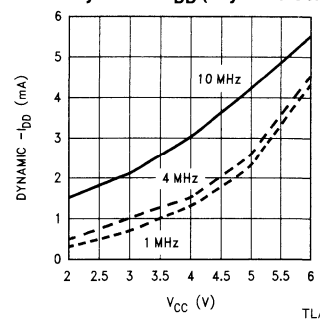
Typical Performance Characteristics ($-40°C \leq T_A \leq +85°C$)

Hall—I_{DD}



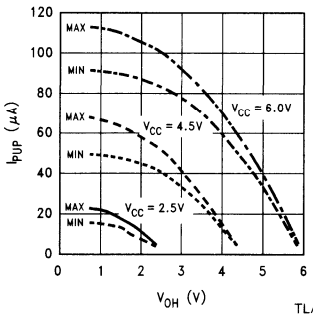
TL/DD/10802-16

Dynamic—I_{DD} (Crystal Clock Option)



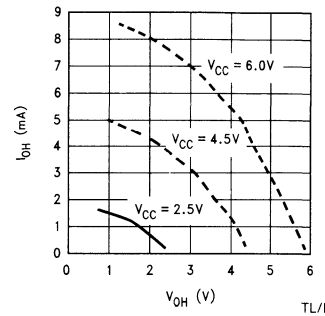
TL/DD/10802-17

Port L/C/G Weak Pull-Up Source Current



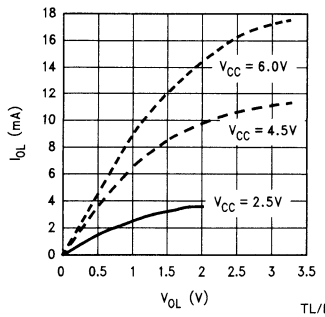
TL/DD/10802-18

Port L/C/G Push-Pull Source Current



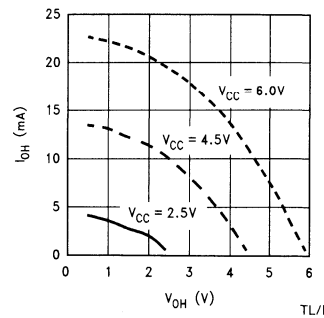
TL/DD/10802-19

Port L/C/G Push-Pull Sink Current



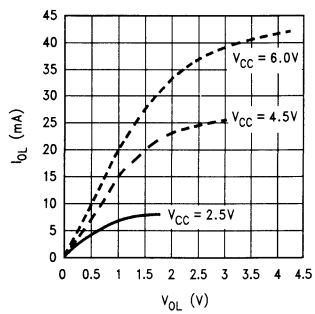
TL/DD/10802-20

Port D Source Current



TL/DD/10802-21

Port D Sink Current



TL/DD/10802-22

Pin Descriptions

V_{CC} and GND are the power supply pins.

CKI is the clock input. This can come from an external source, a R/C generated oscillator or a crystal (in conjunction with CKO). See Oscillator description.

$\overline{\text{RESET}}$ is the master reset input. See Reset description.

PORT I is an 8-bit Hi-Z input port. The 28-pin device does not have a full complement of Port I pins. The unavailable pins are not terminated i.e., they are floating. A read operation for these unterminated pins will return unpredictable values. The user must ensure that the software takes this into account by either masking or restricting the accesses to bit operations. The unterminated Port I pins will draw power only when addressed.

PORT L is an 8-bit I/O port.

PORT C is a 4-bit I/O port.

Three memory locations are allocated for the L, G and C ports, one each for data register, configuration register and the input pins. Reading bits 4–7 of the C-Configuration register, data register, and input pins returns undefined data.

There are two registers associated with the L and C ports: a data register and a configuration register. Therefore, each L and C I/O bit can be individually configured under software control as shown below:

Config.	Data	Ports L and C Setup
0	0	Hi-Z Input (TRI-STATE Output)
0	1	Input with Pull-Up (Weak One Output)
1	0	Push-Pull Zero Output
1	1	Push-Pull One Output

On the 28-pin part, it is recommended that all bits of Port C be configured as outputs.

PORT G is an 8-bit port with 6 I/O pins (G0–G5) and 2 input pins (G6, G7). All eight G-pins have Schmitt Triggers on the inputs.

There are two registers associated with the G port: a data register and a configuration register. Therefore, each G port bit can be individually configured under software control as shown below:

Config.	Data	Port G Setup
0	0	Hi-Z Input (TRI-STATE Output)
0	1	Input with Pull-Up (Weak One Output)
1	0	Push-Pull Zero Output
1	1	Push-Pull One Output

Since G6 and G7 are input only pins, any attempt by the user to configure them as outputs by writing a one to the configuration register will be disregarded. Reading the G6 and G7 configuration bits will return zeros. The device will be placed in the HALT mode by writing to the G7 bit in the G-port data register.

Six pins of Port G have alternate features:

G0 INTR (an external interrupt)

G3 TIO (timer/counter input/output)

G4 SO (MICROWIRE serial data output)

G5 SK (MICROWIRE clock I/O)

G6 SI (MICROWIRE serial data input)

G7 CKO crystal oscillator output (selected by mask option) or HALT restart input (general purpose input)

Pins G1 and G2 currently do not have any alternate functions.

PORT D is an 8-bit output port that is preset high when RESET goes low. Care must be exercised with the D2 pin operation. At RESET, the external loads on this pin must ensure that the output voltages stay above 0.9 V_{CC} to prevent the chip from entering special modes. Also, keep the external loading on D2 to less than 1000 pF.

Functional Description

Figure 1 shows the block diagram of the internal architecture. Data paths are illustrated in simplified form to depict how the various logic elements communicate with each other in implementing the instruction set of the device.

ALU AND CPU REGISTERS

The ALU can do an 8-bit addition, subtraction, logical or shift operation in one cycle time.

There are five CPU registers:

A is the 8-bit Accumulator register

PU is the upper 7 bits of the program counter (PC)

PL is the lower 8 bits of the program counter (PC)

B is the 8-bit address register, can be auto incremented or decremented.

X is the 8-bit alternate address register, can be incremented or decremented.

SP is the 8-bit stack pointer, points to subroutine stack (in RAM).

B, X and SP registers are mapped into the on chip RAM. The B and X registers are used to address the on chip RAM. The SP register is used to address the stack in RAM during subroutine calls and returns.

PROGRAM MEMORY

Program memory consists of 4096 bytes of ROM. These bytes may hold program instructions or constant data. The program memory is addressed by the 15-bit program counter (PC). ROM can be indirectly read by the LAID instruction for table lookup.

DATA MEMORY

The data memory address space includes on chip RAM, I/O and registers. Data memory is addressed directly by the instruction or indirectly by the B, X and SP registers.

The device has 128 bytes of RAM. Sixteen bytes of RAM are mapped as "registers" that can be loaded immediately, decremented or tested. Three specific registers: B, X and SP are mapped into this space, the other bytes are available for general usage.

The instruction set permits any bit in memory to be set, reset or tested. All I/O and registers (except the A & PC) are memory mapped; therefore, I/O bits and register bits can be directly and individually set, reset and tested. A is not memory mapped, but bit operations can be still performed on it.

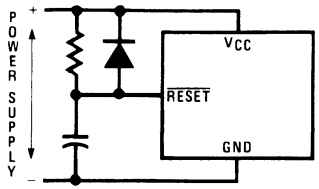
Note: RAM contents are undefined upon power-up.

RESET

The $\overline{\text{RESET}}$ input when pulled low initializes the microcontroller. Initialization will occur whenever the $\overline{\text{RESET}}$ input is pulled low. Upon initialization, the ports L, G and C are placed in the TRI-STATE mode and the Port D is set high. The PC, PSW and CNTRL registers are cleared. The data and configuration registers for Ports L, G and C are cleared.

The external RC network shown in Figure 4 should be used to ensure that the $\overline{\text{RESET}}$ pin is held low until the power supply to the chip stabilizes.

Functional Description (Continued)



TL/DD/10802–6

RC ≥ 5X Power Supply Rise Time

FIGURE 4. Recommended Reset Circuit

OSCILLATOR CIRCUITS

Figure 5 shows the three clock oscillator configurations.

A. CRYSTAL OSCILLATOR

The device can be driven by a crystal clock. The crystal network is connected between the pins CKI and CKO.

Table I shows the component values required for various standard crystal values.

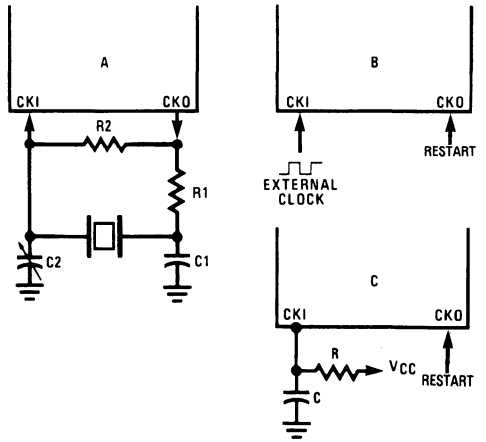
B. EXTERNAL OSCILLATOR

CKI can be driven by an external clock signal. CKO is available as a general purpose input and/or HALT restart control.

C. R/C OSCILLATOR

CKI is configured as a single pin RC controlled Schmitt trigger oscillator. CKO is available as a general purpose input and/or HALT restart control.

Table II shows the variation in the oscillator frequencies as functions of the component (R and C) values.



TL/DD/10802–7

FIGURE 5. Crystal and R-C Connection Diagrams

OSCILLATOR MASK OPTIONS

The device can be driven by clock inputs between DC and 10 MHz.

TABLE I. Crystal Oscillator Configuration, $T_A = 25°C$

R1 (kΩ)	R2 (MΩ)	C1 (pF)	C2 (pF)	CKI Freq (MHz)	Conditions
0	1	30	30–36	10	$V_{CC} = 5V$
0	1	30	30–36	4	$V_{CC} = 2.5V$
5.6	1	200	100–150	0.455	$V_{CC} = 5V$

TABLE II. RC Oscillator Configuration, $T_A = 25°C$

R (kΩ)	C (pF)	CKI Freq. (MHz)	Instr. Cycle (μs)	Conditions
3.3	82	2.2 to 2.7	3.7 to 4.6	$V_{CC} = 5V$
5.6	100	1.1 to 1.3	7.4 to 9.0	$V_{CC} = 5V$
6.8	100	0.9 to 1.1	8.8 to 10.8	$V_{CC} = 5V$

Note: (R/C Oscillator Configuration): 3k ≤ R ≤ 200k, 50 pF ≤ C ≤ 200 pF.

Functional Description (Continued)

The device has three mask options for configuring the clock input. The CKI and CKO pins are automatically configured upon selecting a particular option.

— Crystal (CKI/10); CKO for crystal configuration

— External (CKI/10); CKO available as G7 input

— R/C (CKI/10); CKO available as G7 input

G7 can be used either as a general purpose input or as a control input to continue from the HALT mode.

HALT MODE

The device supports a power saving mode of operation: HALT. The controller is placed in the HALT mode by setting the G7 data bit, alternatively the user can stop the clock input. In the HALT mode all internal processor activities including the clock oscillator are stopped. The fully static architecture freezes the state of the controller and retains all information until continuing. In the HALT mode, power requirements are minimal as it draws only leakage currents and output current. The applied voltage (V_{CC}) may be decreased down to Vr (minimum RAM retention voltage) without altering the state of the machine.

There are two ways to exit the HALT mode: via the $\overline{\text{RESET}}$ or by the CKO pin. A low on the $\overline{\text{RESET}}$ line reinitializes the microcontroller and starts executing from the address 0000H. A low to high transition on the CKO pin (only if the external or R/C clock option selected) causes the microcontroller to continue with no reinitialization from the address following the HALT instruction. This also resets the G7 data bit.

INTERRUPTS

There are three interrupt sources, as shown below.

A maskable interrupt on external G0 input (positive or negative edge sensitive under software control)

A maskable interrupt on timer underflow or timer capture

A non-maskable software/error interrupt on opcode zero

INTERRUPT CONTROL

The GIE (global interrupt enable) bit enables the interrupt function. This is used in conjunction with ENI and ENTI to select one or both of the interrupt sources. This bit is reset when interrupt is acknowledged.

ENI and ENTI bits select external and timer interrupt respectively. Thus the user can select either or both sources to interrupt the microcontroller when GIE is enabled.

IEDG selects the external interrupt edge (0 = rising edge, 1 = falling edge). The user can get an interrupt on both rising and falling edges by toggling the state of IEDG bit after each interrupt.

IPND and TPND bits signal which interrupt is pending. After interrupt is acknowledged, the user can check these two bits to determine which interrupt is pending. This permits the interrupts to be prioritized under software. The pending flags have to be cleared by the user. Setting the GIE bit high inside the interrupt subroutine allows nested interrupts.

The software interrupt does not reset the GIE bit. This means that the controller can be interrupted by other interrupt sources while servicing the software interrupt.

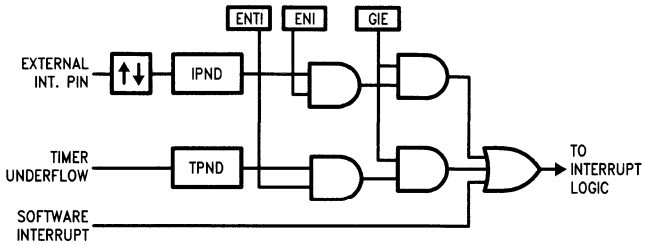
INTERRUPT PROCESSING

The interrupt, once acknowledged, pushes the program counter (PC) onto the stack and the stack pointer (SP) is decremented twice. The Global Interrupt Enable (GIE) bit is reset to disable further interrupts. The microcontroller then vectors to the address 00FFH and resumes execution from that address. This process takes 7 cycles to complete. At the end of the interrupt subroutine, any of the following three instructions return the processor back to the main program: RET, RETSK or RETI. Either one of the three instructions will pop the stack into the program counter (PC). The stack pointer is then incremented twice. The RETI instruction additionally sets the GIE bit to re-enable further interrupts.

Any of the three instructions can be used to return from a hardware interrupt subroutine. The RETSK instruction should be used when returning from a software interrupt subroutine to avoid entering an infinite loop.

Note: There is always the possibility of an interrupt occurring during an instruction which is attempting to reset the GIE bit or any other interrupt enable bit. If this occurs when a single cycle instruction is being used to reset the interrupt enable bit, the interrupt enable bit will be reset but an interrupt may still occur. This is because interrupt processing is started at the same time as the interrupt bit is being reset. To avoid this scenario, the user should always use a two, three or four cycle instruction to reset interrupt enable bits.

Functional Description (Continued)



TL/DD/10802–8

FIGURE 6. Interrupt Block Diagram

DETECTION OF ILLEGAL CONDITIONS

The device contains a hardware mechanism that allows it to detect illegal conditions which may occur from coding errors, noise and 'brown out' voltage drop situations. Specifically it detects cases of executing out of undefined ROM area and unbalanced stack situations.

Reading an undefined ROM location returns 00 (hexadecimal) as its contents. The opcode for a software interrupt is also '00'. Thus a program accessing undefined ROM will cause a software interrupt.

Reading an undefined RAM location returns an FF (hexadecimal). The subroutine stack grows down for each subroutine call. By initializing the stack pointer to the top of RAM, the first unbalanced return instruction will cause the stack pointer to address undefined RAM. As a result the program will attempt to execute from FFFF (hexadecimal), which is an undefined ROM location and will trigger a software interrupt.

MICROWIRE/PLUS™

MICROWIRE/PLUS is a serial synchronous bidirectional communications interface. The MICROWIRE/PLUS capability enables the device to interface with any of National Semiconductor's MICROWIRE peripherals (i.e. A/D converters, display drivers, EEPROMS, etc.) and with other microcontrollers which support the MICROWIRE/PLUS interface. It consists of an 8-bit serial shift register (SIO) with serial data input (SI), serial data output (SO) and serial shift clock (SK). *Figure 7* shows the block diagram of the MICROWIRE/PLUS interface.

The shift clock can be selected from either an internal source or an external source. Operating the MICROWIRE/PLUS interface with the internal clock source is called the Master mode of operation. Similarly, operating the MICROWIRE/PLUS interface with an external shift clock is called the Slave mode of operation.

The CNTRL register is used to configure and control the MICROWIRE/PLUS mode. To use the MICROWIRE/PLUS, the MSEL bit in the CNTRL register is set to one. The SK clock rate is selected by the two bits, SL0 and SL1, in the CNTRL register. Table III details the different clock rates that may be selected.

TABLE III

SL1	SL0	SK Cycle Time
0	0	$2t_C$
0	1	$4t_C$
1	x	$8t_C$

where,

t_C is the instruction cycle clock.

MICROWIRE/PLUS OPERATION

Setting the BUSY bit in the PSW register causes the MICROWIRE/PLUS arrangement to start shifting the data. It gets reset when eight data bits have been shifted. The user may reset the BUSY bit by software to allow less than 8 bits to shift. The devoce may enter the MICROWIRE/PLUS mode either as a Master or as a Slave. *Figure 8* shows how two COP880C microcontrollers and several peripherals may be interconnected using the MICROWIRE/PLUS arrangement.

Master MICROWIRE/PLUS Operation

In the MICROWIRE/PLUS Master mode of operation the shift clock (SK) is generated internally. The MICROWIRE/PLUS Master always initiates all data exchanges. (See *Figure 8*). The MSEL bit in the CNTRL register must be set to enable the SO and SK functions onto the G Port. The SO and SK pins must also be selected as outputs by setting appropriate bits in the Port G configuration register. Table IV summarizes the bit settings required for Master mode of operation.

SLAVE MICROWIRE/PLUS OPERATION

In the MICROWIRE/PLUS Slave mode of operation the SK clock is generated by an external source. Setting the MSEL bit in the CNTRL register enables the SO and SK functions onto the G Port. The SK pin must be selected as an input and the SO pin is selected as an output pin by appropriately setting up the Port G configuration register. Table IV summarizes the settings required to enter the Slave mode of operation.

The user must set the BUSY flag immediately upon entering the Slave mode. This will ensure that all data bits sent by the Master will be shifted properly. After eight clock pulses the BUSY flag will be cleared and the sequence may be repeated. (See *Figure 8*.)

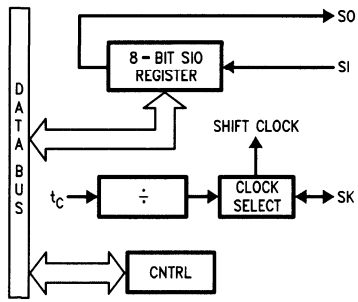
Functional Description (Continued)

TABLE IV

G4 Config. Bit	G5 Config. Bit	G4 Fun.	G5 Fun.	G6 Fun.	Operation
1	1	SO	Int. SK	SI	MICROWIRE Master
0	1	TRI-STATE	Int. SK	SI	MICROWIRE Master
1	0	SO	Ext. SK	SI	MICROWIRE Slave
0	0	TRI-STATE	Ext. SK	SI	MICROWIRE Slave

TIMER/COUNTER

The device has a powerful 16-bit timer with an associated 16-bit register enabling them to perform extensive timer functions. The timer T1 and its register R1 are each organized as two 8-bit read/write registers. Control bits in the register CNTRL allow the timer to be started and stopped under software control. The timer-register pair can be operated in one of three possible modes. Table V details various timer operating modes and their requisite control settings.



TL/DD/10802-9

FIGURE 7. MICROWIRE/PLUS Block Diagram

MODE 1. TIMER WITH AUTO-LOAD REGISTER

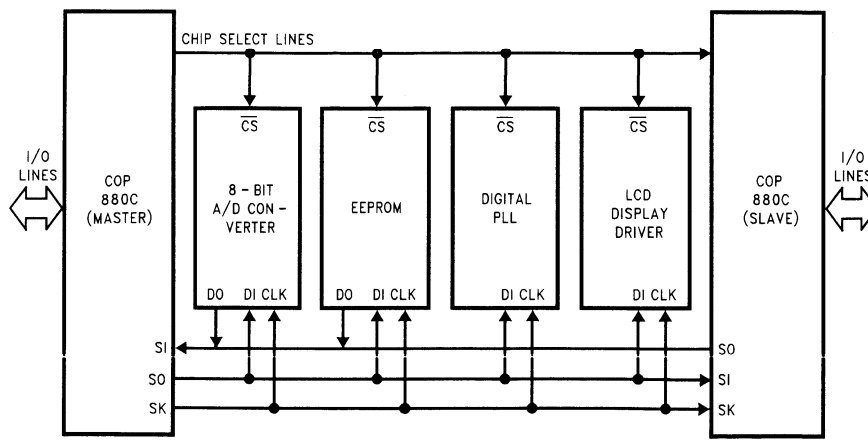
In this mode of operation, the timer T1 counts down at the instruction cycle rate. Upon underflow the value in the register R1 gets automatically reloaded into the timer which continues to count down. The timer underflow can be programmed to interrupt the microcontroller. A bit in the control register CNTRL enables the TIO (G3) pin to toggle upon timer underflows. This allow the generation of square-wave outputs or pulse width modulated outputs under software control. (See *Figure 9.*)

MODE 2. EXTERNAL COUNTER

In this mode, the timer T1 becomes a 16-bit external event counter. The counter counts down upon an edge on the TIO pin. Control bits in the register CNTRL program the counter to decrement either on a positive edge or on a negative edge. Upon underflow the contents of the register R1 are automatically copied into the counter. The underflow can also be programmed to generate an interrupt. (See *Figure 9*)

MODE 3. TIMER WITH CAPTURE REGISTER

Timer T1 can be used to precisely measure external frequencies or events in this mode of operation. The timer T1 counts down at the instruction cycle rate. Upon the occurrence of a specified edge on the TIO pin the contents of the timer T1 are copied into the register R1. Bits in the control register CNTRL allow the trigger edge to be specified either as a positive edge or as a negative edge. In this mode the user can elect to be interrupted on the specified trigger edge. (See *Figure 10.*)



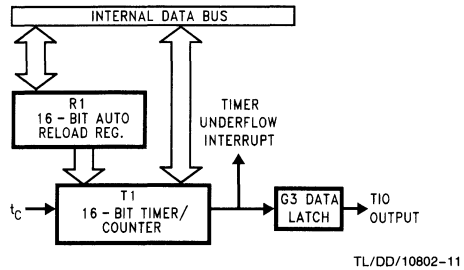
TL/DD/10802-10

FIGURE 8. MICROWIRE/PLUS Application

Functional Description (Continued)

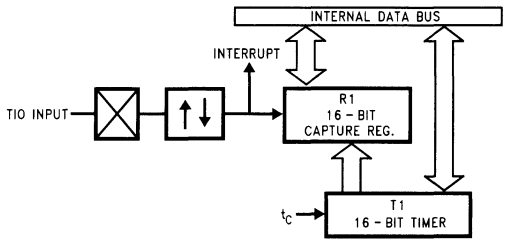
TABLE V. Timer Operating Modes

CNTRL Bits 7 6 5	Operation Mode	T Interrupt	Timer Counts On
0 0 0	External Counter W/Auto-Load Reg.	Timer Underflow	TIO Pos. Edge
0 0 1	External Counter W/Auto-Load Reg.	Timer Underflow	TIO Neg. Edge
0 1 0	Not Allowed	Not Allowed	Not Allowed
0 1 1	Not Allowed	Not Allowed	Not Allowed
1 0 0	Timer W/Auto-Load Reg.	Timer Underflow	t_C
1 0 1	Timer W/Auto-Load Reg./Toggle TIO Out	Timer Underflow	t_C
1 1 0	Timer W/Capture Register	TIO Pos. Edge	t_C
1 1 1	Timer W/Capture Register	TIO Neg. Edge	t_C



TL/DD/10802-11

FIGURE 9. Timer/Counter Auto Reload Mode Block Diagram

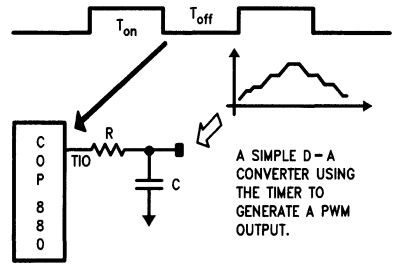


TL/DD/10802-12

FIGURE 10. Timer Capture Mode Block Diagram

TIMER PWM APPLICATION

Figure 11 shows how a minimal component D/A converter can be built out of the Timer-Register pair in the Auto-Reload mode. The timer is placed in the "Timer with auto reload" mode and the TIO pin is selected as the timer output. At the outset the TIO pin is set high, the timer T1 holds the on time and the register R1 holds the signal off time. Setting TRUN bit starts the timer which counts down at the instruction cycle rate. The underflow toggles the TIO output and copies the off time into the timer, which continues to run. By alternately loading in the on time and the off time at each successive interrupt a PWM frequency can be easily generated.



TL/DD/10802-13

FIGURE 11. Timer Application

Control Registers

CNTRL REGISTER (ADDRESS X'00EE)

The Timer and MICROWIRE/PLUS control register contains the following bits:

SL1 & SL0 Select the MICROWIRE/PLUS clock divide-by

IEDG External interrupt edge polarity select (0 = rising edge, 1 = falling edge)

MSEL Enable MICROWIRE/PLUS functions SO and SK

TRUN Start/Stop the Timer/Counter (1 = run, 0 = stop)

TC3 Timer input edge polarity select (0 = rising edge, 1 = falling edge)

TC2 Selects the capture mode

TC1 Selects the timer mode

TC1	TC2	TC3	TRUN	MSEL	IEDG	SL1	SL0

BIT 7 BIT 0

PSW REGISTER (ADDRESS X'00EF)

The PSW register contains the following select bits:

GIE Global interrupt enable

ENI External interrupt enable

BUSY MICROWIRE/PLUS busy shifting

IPND External interrupt pending

ENTI Timer interrupt enable

TPND Timer interrupt pending

C Carry Flag

HC Half carry Flag

HC	C	TPND	ENTI	IPND	BUSY	ENI	GIE

Bit 7 Bit 0

Addressing Modes

REGISTER INDIRECT

This is the "normal" mode of addressing. The operand is the memory addressed by the B register or X register.

DIRECT

The instruction contains an 8-bit address field that directly points to the data memory for the operand.

IMMEDIATE

The instruction contains an 8-bit immediate field as the operand.

REGISTER INDIRECT (AUTO INCREMENT AND DECREMENT)

This is a register indirect mode that automatically increments or decrements the B or X register after executing the instruction.

RELATIVE

This mode is used for the JP instruction, the instruction field is added to the program counter to get the new program location. JP has a range of from −31 to +32 to allow a one byte relative jump (JP + 1 is implemented by a NOP instruction). There are no 'pages' when using JP, all 15 bits of PC are used.

Memory Map

All RAM, ports and registers (except A and PC) are mapped into data memory address space.

Address	Contents
00 to 6F	On Chip RAM Bytes
70 to 7F	Unused RAM Address Space (Reads as all Ones)
80 to BF	Expansion Space for future use
C0 to CF	Expansion Space for I/O and Registers
D0 to DF	On Chip I/O and Registers
D0	Port L Data Register
D1	Port L Configuration Register
D2	Port L Input Pins (Read Only)
D3	Reserved for Port L
D4	Port G Data Register
D5	Port G Configuration Register
D6	Port G Input Pins (Read Only)
D7	Port I Input Pins (Read Only)
D8	Port C Data Register
D9	Port C Configuration Register
DA	Port C Input Pins (Read Only)
DB	Reserved for Port C
DC	Port D Data Register
DD–DF	Reserved for Port D
E0 to EF	On Chip Functions and Registers
E0–E7	Reserved for Future Parts
E8	Reserved
E9	MICROWIRE/PLUS Shift Register
EA	Timer Lower Byte
EB	Timer Upper Byte
EC	Timer Autoload Register Lower Byte
ED	Timer Autoload Register Upper Byte
EE	CNTRL Control Register
EF	PSW Register
F0 to FF	On Chip RAM Mapped as Registers
FC	X Register
FD	SP Register
FE	B Register

Reading unused memory locations below 7FH will return all ones. Reading other unused memory locations will return undefined data.

Instruction Set

REGISTER AND SYMBOL DEFINITIONS

Registers

A	8-bit Accumulator register
B	8-bit Address register
X	8-bit Address register
SP	8-bit Stack pointer register
PC	15-bit Program counter register
PU	upper 7 bits of PC
PL	lower 8 bits of PC
C	1-bit of PSW register for carry
HC	Half Carry
GIE	1-bit of PSW register for global interrupt enable

Symbols

[B]	Memory indirectly addressed by B register
[X]	Memory indirectly addressed by X register
Mem	Direct address memory or [B]
Meml	Direct address memory or [B] or Immediate data
Imm	8-bit Immediate data
Reg	Register memory: addresses F0 to FF (Includes B, X and SP)
Bit	Bit number (0 to 7)
←	Loaded with
⟷	Exchanged with

Instruction Set

ADD	add	A ← A + Meml
ADC	add with carry	A ← A + Meml + C, C ← Carry HC ← Half Carry
SUBC	subtract with carry	A ← A + $\overline{\text{Meml}}$ + C, C ← Carry HC ← Half Carry
AND	Logical AND	A ← A and Meml
OR	Logical OR	A ← A or Meml
XOR	Logical Exclusive-OR	A ← A xor Meml
IFEQ	IF equal	Compare A and Meml, Do next if A = Meml
IFGT	IF greater than	Compare A and Meml, Do next if A > Meml
IFBNE	IF B not equal	Do next if lower 4 bits of B ≠ Imm
DRSZ	Decrement Reg. ,skip if zero	Reg ← Reg − 1, skip if Reg goes to 0
SBIT	Set bit	1 to bit, Mem (bit = 0 to 7 immediate)
RBIT	Reset bit	0 to bit, Mem
IFBIT	If bit	If bit, Mem is true, do next instr.
X	Exchange A with memory	A ⟷ Mem
LD A	Load A with memory	A ← Meml
LD mem	Load Direct memory Immed.	Mem ← Imm
LD Reg	Load Register memory Immed.	Reg ← Imm
X	Exchange A with memory [B]	A ⟷ [B] (B ← B±1)
X	Exchange A with memory [X]	A ⟷ [X] (X ← X±1)
LD A	Load A with memory [B]	A ← [B] (B ← B±1)
LD A	Load A with memory [X]	A ← [X] (X ← X±1)
LD M	Load Memory Immediate	[B] ← Imm (B ← B±1)
CLRA	Clear A	A ← 0
INCA	Increment A	A ← A + 1
DECA	Decrement A	A ← A − 1
LAID	Load A indirect from ROM	A ← ROM(PU,A)
DCORA	DECIMAL CORRECT A	A ← BCD correction (follows ADC, SUBC)
RRCA	ROTATE A RIGHT THRU C	C → A7 → . . . → A0 → C
SWAPA	Swap nibbles of A	A7 . . . A4 ⟷ A3 . . . A0
SC	Set C	C ← 1, HC ← 1
RC	Reset C	C ← 0, HC ← 0
IFC	If C	If C is true, do next instruction
IFNC	If not C	If C is not true, do next instruction
JMPL	Jump absolute long	PC ← ii (ii = 15 bits, 0 to 32k)
JMP	Jump absolute	PC11..0 ← i (i = 12 bits)
JP	Jump relative short	PC ← PC + r (r is −31 to +32, not 1)
JSRL	Jump subroutine long	[SP] ← PL,[SP-1] ← PU,SP-2,PC ← ii
JSR	Jump subroutine	[SP] ← PL,[SP-1] ← PU,SP-2,PC11..0 ← i
JID	Jump indirect	PL ← ROM(PU,A)
RET	Return from subroutine	SP+2,PL ← [SP],PU ← [SP-1]
RETSK	Return and Skip	SP+2,PL ← [SP],PU ← [SP-1],Skip next instruction
RETI	Return from Interrupt	SP+2,PL ← [SP],PU ← [SP-1],GIE ← 1
INTR	Generate an interrupt	[SP] ← PL,[SP−1] ← PU,SP-2,PC ← 0FF
NOP	No operation	PC ← PC + 1

OPCODE LIST

Bits 7–4

F	E	D	C	B	A	9	8	7	6	5	4	3	2	1	0	Bits 3–0
JP -15	JP -31	LD 0F0, #i	DRSZ 0F0	RRCA	RC	ADC A, #i	ADC A, [B]	IFBIT 0,[B]	*	LD B, 0F	IFBNE 0	JSR 0000-00FF	JMP 0000-00FF	JP + 17	INTR	0
JP -14	JP -30	LD 0F1, #i	DRSZ 0F1	*	SC	SUBC A, #i	SUBC A,[B]	IFBIT 1,[B]	*	LD B, 0E	IFBNE 1	JSR 0100-01FF	JMP 0100-01FF	JP + 18	JP + 2	1
JP -13	JP -29	LD 0F2, #i	DRSZ 0F2	X A, [X+]	X A, [B+]	IFEQ A, #i	IFEQ A,[B]	IFBIT 2,[B]	*	LD B, 0D	IFBNE 2	JSR 0200-02FF	JMP 0200-02FF	JP + 19	JP + 3	2
JP -12	JP -28	LD 0F3, #i	DRSZ 0F3	X A, [X−]	X A, [B−]	IFGT A, #i	IFGT A,[B]	IFBIT 3,[B]	*	LD B, 0C	IFBNE 3	JSR 0300-03FF	JMP 0300-03FF	JP + 20	JP + 4	3
JP -11	JP -27	LD 0F4, #i	DRSZ 0F4	*	LAID	ADD A, #i	ADD A,[B]	IFBIT 4,[B]	CLRA	LD B, 0B	IFBNE 4	JSR 0400-04FF	JMP 0400-04FF	JP + 21	JP + 5	4
JP -10	JP -26	LD 0F5, #i	DRSZ 0F5	*	JID	AND A, #i	AND A,[B]	IFBIT 5,[B]	SWAPA	LD B, 0A	IFBNE 5	JSR 0500-05FF	JMP 0500-05FF	JP + 22	JP + 6	5
JP -9	JP -25	LD 0F6, #i	DRSZ 0F6	X A, [X]	X A, [B]	XOR A, #i	XOR A,[B]	IFBIT 6,[B]	DCORA	LD B, 9	IFBNE 6	JSR 0600-06FF	JMP 0600-06FF	JP + 23	JP + 7	6
JP -8	JP -24	LD 0F7, #i	DRSZ 0F7	*	*	OR A, #i	OR A,[B]	IFBIT 7,[B]	*	LD B, 8	IFBNE 7	JSR 0700-07FF	JMP 0700-07FF	JP + 24	JP + 8	7
JP -7	JP -23	LD 0F8, #i	DRSZ 0F8	NOP	*	LD A, #i	IFC	SBIT 0,[B]	RBIT 0,[B]	LD B, 7	IFBNE 8	JSR 0800-08FF	JMP 0800-08FF	JP + 25	JP + 9	8
JP -6	JP -22	LD 0F9, #i	DRSZ 0F9	*	*	*	IFNC	SBIT 1,[B]	RBIT 1,[B]	LD B, 6	IFBNE 9	JSR 0900-09FF	JMP 0900-09FF	JP + 26	JP + 10	9
JP -5	JP -21	LD 0FA, #i	DRSZ 0FA	LD A, [X+]	LD A, [B+]	LD [B+], #i	INCA	SBIT 2,[B]	RBIT 2,[B]	LD B, 5	IFBNE 0A	JSR 0A00-0AFF	JMP 0A00-0AFF	JP + 27	JP + 11	A
JP -4	JP -20	LD 0FB, #i	DRSZ 0FB	LD A, [X−]	LD A, [B−]	LD [B−], #i	DECA	SBIT 3,[B]	RBIT 3,[B]	LD B, 4	IFBNE 0B	JSR 0B00-0BFF	JMP 0B00-0BFF	JP + 28	JP + 12	B
JP -3	JP -19	LD 0FC, #i	DRSZ 0FC	LD Md, #i	JMPL	X A,Md	*	SBIT 4,[B]	RBIT 4,[B]	LD B, 3	IFBNE 0C	JSR 0C00-0CFF	JMP 0C00-0CFF	JP + 29	JP +13	C
JP -2	JP -18	LD 0FD, #i	DRSZ 0FD	DIR	JSRL	LD A, Md	RETSK	SBIT 5,[B]	RBIT 5,[B]	LD B, 2	IFBNE 0D	JSR 0D00-0DFF	JMP 0D00-0DFF	JP + 30	JP +14	D
JP -1	JP -17	LD 0FE, #i	DRSZ 0FE	LD A, [X]	LD A, [B]	LD [B], #i	RET	SBIT 6, [B]	RBIT 6, [B]	LD B, 1	IFBNE 0E	JSR 0E00−0EFF	JMP 0E00−0EFF	JP + 31	JP +15	E
JP -0	JP -16	LD 0FF, #1	DRSZ 0FF	*	*	*	RETI	SBIT 7,[B]	RBIT 7,[B]	LD B, 0	IFBNE 0F	JSR 0F00-0FFF	JMP 0F00-0FFF	JP + 32	JP + 16	F

where, i is the immediate data Md is a directly addressed memory location * is an unused opcode (see following table)

Instruction Execution Time

Most instructions are single byte (with immediate addressing mode instruction taking two bytes).

Most single instructions take one cycle time to execute.

Skipped instructions require x number of cycles to be skipped, where x equals the number of bytes in the skipped instruction opcode.

See the BYTES and CYCLES per INSTRUCTION table for details.

BYTES and CYCLES per INSTRUCTION

The following table shows the number of bytes and cycles for each instruction in the format of byte/cycle.

Arithmetic and Logic Instructions

	[B]	Direct	Immed.
ADD	1/1	3/4	2/2
ADC	1/1	3/4	2/2
SUBC	1/1	3/4	2/2
AND	1/1	3/4	2/2
OR	1/1	3/4	2/2
XOR	1/1	3/4	2/2
IFEQ	1/1	3/4	2/2
IFGT	1/1	3/4	2/2
IFBNE	1/1		
DRSZ		1/3	
SBIT	1/1	3/4	
RBIT	1/1	3/4	
IFBIT	1/1	3/4	

Memory Transfer Instructions

	Register Indirect [B]	Register Indirect [X]	Direct	Immed.	Register Indirect Auto Incr & Decr [B+, B−]	Register Indirect Auto Incr & Decr [X+, X−]	
X A,*	1/1	1/3	2/3		1/2	1/3	
LD A,*	1/1	1/3	2/3	2/2	1/2	1/3	
LD B,Imm				1/1			(If B < 16)
LD B,Imm				2/3			(If B > 15)
LD Mem,Imm	2/2		3/3		2/2		
LD Reg,Imm				2/3			

* = > Memory location addressed by B or X or directly.

Instructions Using A & C		Transfer of Control Instructions	
CLRA	1/1	JMPL	3/4
INCA	1/1	JMP	2/3
DECA	1/1	JP	1/3
LAID	1/3	JSRL	3/5
DCORA	1/1	JSR	2/5
RRCA	1/1	JID	1/3
SWAPA	1/1	RET	1/5
SC	1/1	RETSK	1/5
RC	1/1	RETI	1/5
IFC	1/1	INTR	1/7
IFNC	1/1	NOP	1/1

BYTES and CYCLES per INSTRUCTION (Continued)

The following table shows the instructions assigned to unused opcodes. This table is for information only. The operations performed are subject to change without notice. Do not use these opcodes.

Unused Opcode	Instruction	Unused Opcode	Instruction
60	NOP	A9	NOP
61	NOP	AF	LD A, [B]
62	NOP	B1	C → HC
63	NOP	B4	NOP
67	NOP	B5	NOP
8C	RET	B7	X A, [X]
99	NOP	B9	NOP
9F	LD [B], #i	BF	LD A, [X]
A7	X A, [B]		
A8	NOP		

Option List

The mask programmable options are listed out below. The options are programmed at the same time as the ROM pattern to provide the user with hardware flexibility to use a variety of oscillator configuration.

OPTION 1: CKI INPUT

= 1 Crystal (CKI/10) CKO for crystal configuration

= 2 External (CKI/10) CKO available as G7 input

= 3 R/C (CKI/10) CKO available as G7 input

OPTION 2: BONDING

= 1 44-Pin PLCC

= 2 40-Pin DIP

= 3 28-Pin SO

= 4 28-Pin DIP

The following option information is to be sent to National along with the EPROM.

Option Data

Option 1 Value_is: CKI Input

Option 2 Value_is: COP Bonding

Development Support

SUMMARY

- iceMASTER™: IM-COP8/400—Full feature in-circuit emulation for all COP8 products. A full set of COP8 Basic and Feature Family device and package specific probes are available.
- COP8 Debug Module: Moderate cost in-circuit emulation and development programming unit.
- COP8 Evaluation and Programming Unit: EPU-COP880C—low cost In-circuit simulation and development programming unit.
- Assembler: COP8-DEV-IBMA. A DOS installable cross development Assembler, Linker, Librarian and Utility Software Development Tool Kit.
- C Compiler: COP8C. A DOS installable cross development Software Tool Kit.
- OTP/EPROM Programmer Support: Covering needs from engineering prototype, pilot production to full production environments.

Development Support (Continued)

iceMASTER (IM) IN-CIRCUIT EMULATION

The iceMASTER IM-COP8/400 is a full feature, PC based, in-circuit emulation tool developed and marketed by Meta-Link Corporation to support the whole COP8 family of products. National is a resale vendor for these products.

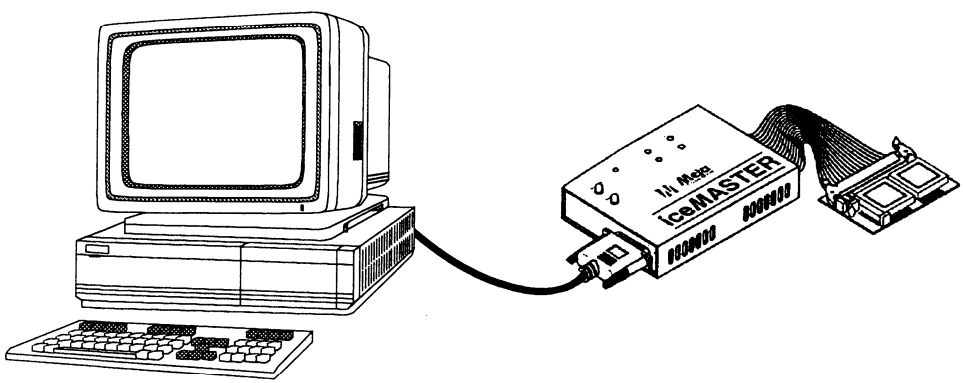
See *Figure 12* for configuration.

The iceMASTER IM-COP8/400 with its device specific COP8 Probe provides a rich feature set for developing, testing and maintaining product:

- Real-time in-circuit emulation; full 2.4V–5.5V operation range, full DC-10 MHz clock. Chip options are programmable or jumper selectable.
- Direct connection to application board by package compatible socket or surface mount assembly.
- Full 32 kbyte of loadable programming space that overlays (replaces) the on-chip ROM or EPROM. On-chip RAM and I/O blocks are used directly or recreated on the probe as necessary.
- Full 4k frame synchronous trace memory. Address, instruction, and 8 unspecified, circuit connectable trace lines. Display can be HLL source (e.g., C source), assembly or mixed.
- A full 64k hardware configurable break, trace on, trace off control, and pass count increment events.
- Tool set integrated interactive symbolic debugger—supports both assembler (COFF) and C Compiler (.COD) linked object formats.
- Real time performance profiling analysis; selectable bucket definition.
- Watch windows, content updated automatically at each execution break.
- Instruction by instruction memory/register changes displayed on source window when in single step operation.
- Single base unit and debugger software reconfigurable to support the entire COP8 family; only the probe personality needs to change. Debugger software is processor customized, and reconfigured from a master model file.
- Processor specific symbolic display of registers and bit level assignments, configured from master model file.
- Halt/Idle mode notification.
- On-line HELP customized to specific processor using master model file.
- Includes a copy of COP8-DEV-IBMA assembler and linker SDK.

IM Order Information

Base Unit	
IM-COP8/400-1	iceMASTER base unit, 110V power supply
IM-COP8/400-2	iceMASTER base unit, 220V power supply
iceMASTER Probe	
MHW-880C20DWPC	20 DIP
MHW-880C28DWPC	28 DIP
MHW-880CJ40DWPC	40 DIP
MHW-880CJ44PWPC	44 PLCC
DIP to SO Adapters	
MHW-SOIC20	20 SO
MHW-SOIC28	28 DIP



TL/DD/10802–24

FIGURE 12. COP8 iceMASTER Environment

Development Support (Continued)

iceMASTER DEBUG MODULE (DM)

The iceMASTER Debug Module is a PC based, combination in-circuit emulation tool and COP8 based OTP/EPROM programming tool developed and marketed by MetaLink Corporation to support the whole COP8 family of products. National is a resale vendor for these products.

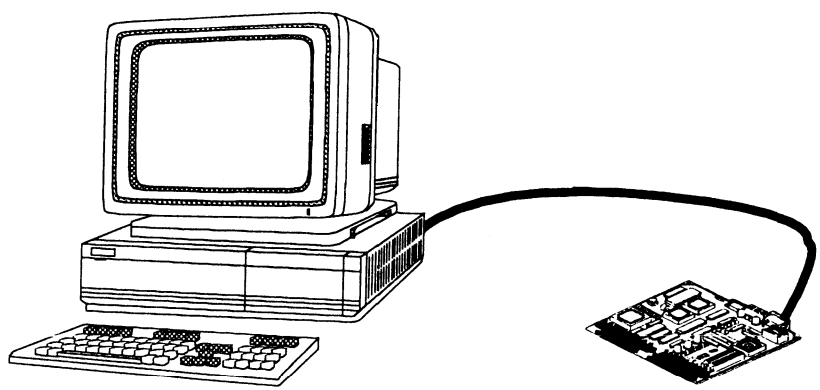
See *Figure 13* for configuration.

The iceMASTER Debug Module is a moderate cost development tool. It has the capability of in-circuit emulation for a specific COP8 microcontroller and in addition serves as a programming tool for COP8 OTP and EPROM product families. Summary of features is as follows:

- Real-time in-circuit emulation; full operating voltage range operation, full DC-10 MHz clock.
- All processor I/O pins can be cabled to an application development board with package compatible cable to socket and surface mount assembly.
- Full 32 kbyte of loadable programming space that overlays (replaces) the on-chip ROM or EPROM. On-chip RAM and I/O blocks are used directly or recreated as necessary.
- 100 frames of synchronous trace memory. The display can be HLL source (C source), assembly or mixed. The most recent history prior to a break is available in the trace memory.
- Configured break points; uses INTR instruction which is modestly intrusive.
- Software—only supported features are selectable.
- Tool set integrated interactive symbolic debugger—supports both assembler (COFF) and C Compiler (.COD) SDK linked object formats.
- Instruction by instruction memory/register changes displayed when in single step operation.
- Debugger software is processor customized, and reconfigured from a master model file.
- Processor specific symbolic display of registers and bit level assignments, configured from master model file.
- Halt/Idle mode notification.
- Programming menu supports full product line of programmable OTP and EPROM COP8 products. Program data is taken directly from the overlay RAM.
- Programming of 44 PLCC and 68 PLCC parts requires external programming. adapters.
- Includes wallmount power supply.
- On-board VPP generator from 5V input or connection to external supply supported. Rquires VPP level adjustment per the family programming specification (correct level is provided on an on-screen pop-down display).
- On-line HELP customized to specific processor using master model file.
- Includes a copy of COP8-DEV-IBMA assembler and linker SDK.

DM Order Information

Debug Model Unit	
COP8-DM/880C	
Cable Adapters	
DM-COP8/20D	20 DIP
DM-COP8/28D	28 DIP
DM-COP8/40D	40 DIP
DM-COP8/44P	44 PLCC
DIP to SO Adapters	
DM-COP8/20D-SO	20 SO
DM-COP8/28D-SO	28 SO



TL/DD/10802–25

FIGURE 13. COP8-DM Environment

Development Support (Continued)

iceMASTER EVALUATION PROGRAMMING UNIT (EPU)

The iceMASTER EPU-COP880C is a PC based, in-circuit simulation tool to support the feature family COP8 products.

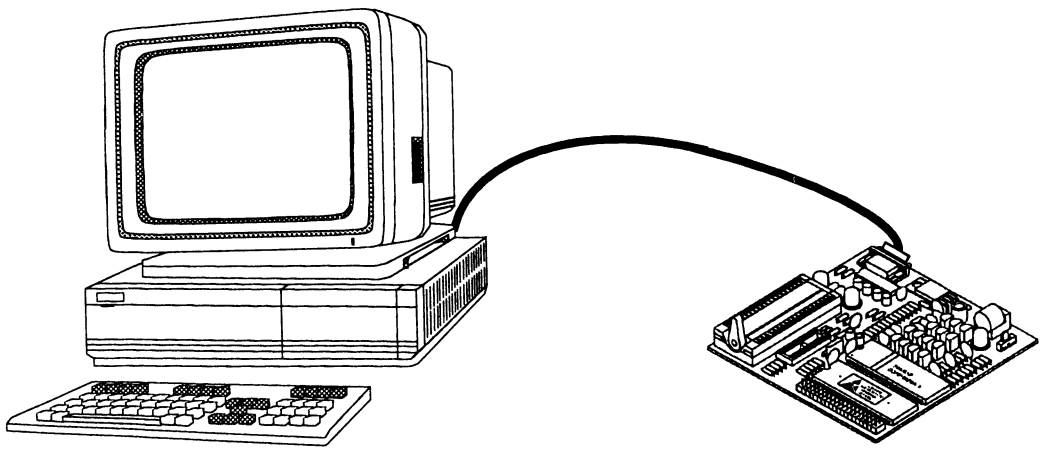
See *Figure 14* for configuration.

The simulation capability is a very low cost means of evaluating the general COP8 architecture. In addition, the EPU has programming capability, with added adapters, for programming the whole COP8 product family of OTP and EPROM products. The product includes the following features:

- Non-real-time in-circuit simulation. Program overlay memory is PC resident; instructions are downloaded over RS-232 as executed. Approximate performance is 20 kHz.
- Includes a 40 pin DIP cable adapter. Other target packages are not supported. All processor I/O pins are cabled to the application development environment.
- Full 32 kbyte of loadable programmable space that overlays (replaces) the on-chip ROM or EPROM. On-chip RAM and I/O blocks are used directly or recreated as necessary.
- On-chip timer and WATCHDOG execution are not well synchronized to the instruction simulation.
- 100 frames of synchronous trace memory. The display can be HLL source (e.g., C source), assembly or mixed. The most recent history prior to a break is available in the trace memory.
- Up to eight software configured break points; uses INTR instruction which is modestly intrusive.
- Common look-feel debugger software across all MetaLink products—only supported features are selectable.
- Tool set integrated interactive symbolic debugger—supports both assembler (COFF) and C Compiler (.COD) SDK linked object formats.
- Instruction by instruction memory/register changes displayed when in single step operation.
- Processor specific symbolic display of registers and bit level assignments, configured from master model file.
- Halt/Idle mode notification. Restart requires special handling.
- Programming menu supports full product line of programmable OTP and EPROM COP8 products. Only a 40 ZIF socket is available on the EPU unit. Adapters are available for other part package configurations.
- Integral wall mount power supply provides 5V and develops the required V_{PP} to program parts.
- Includes a copy of COP8-DEV-IBMA assembler, linker SDK.

EPU Order Information

Evaluation Programming Unit	
EPU-COP880C	Evaluation Programming Unit with debugger and programmer control software with 40 ZIF programming socket.
General Programming Adapters	
COP8-PGMA-DS	28 and 20 DIP and SOIC adapter
COP8-PGMA-DS44P	28 and 20 DIP and SOIC plus 44 PLCC adapter



TL/DD/10802-26

FIGURE 14. EPU-COP8 Tool Environment

Development Support (Continued)

COP8 ASSEMBLER/LINKER SOFTWARE DEVELOPMENT TOOL KIT

National Semiconductor offers a relocateable COP8 macro cross assembler, linker, librarian and utility software development tool kit. Features are summarized as follows:

- Basic and Feature Family instruction set by "device" type.
- Nested macro capability.
- Extensive set of assembler directives.
- Supported on PC/DOS platform.
- Generates National standard COFF output files.
- Integrated Linker and Librarian.
- Integrated utilities to generate ROM code file outputs.
- DUMPCOFF utility.

This product is integrated as a part of MetaLink tools as a development kit, fully supported by the MetaLink debugger. It may be ordered separately or it is bundled with the MetaLink products at no additional cost.

Order Information

Assembler SDK:	
COP8-DEV-IBMA	Assembler SDK on installable 3.5" PC/DOS Floppy Disk Drive format. Periodic upgrades and most recent version is available on National's BBS and Internet.

COP8 C COMPILER

A C Compiler is developed and marketed by Byte Craft Limited. The COP8C compiler is a fully integrated development tool specifically designed to support the compact embedded configuration of the COP8 family of products.

Features are summarized as follows:

- ANSI C with some restrictions and extensions that optimize development for the COP8 embedded application.
- BITS data type extension. Register declaration #pragma with direct bit level definitions.
- C language support for interrupt routines.
- Expert system, rule based code geration and optimization.
- Performs consistency checks against the architectural definitions of the target COP8 device.
- Generates program memory code.
- Supports linking of compiled object or COP8 assembled object formats.
- Global optimization of linked code.
- Symbolic debug load format fully sourced level supported by the MetaLink debugger.

INDUSTRY WIDE OTP/EPROM PROGRAMMING SUPPORT

Programming support, in addition to the MetaLink development tools, is provided by a full range of independent approved vendors to meet the needs from the engineering laboratory to full production.

Approved List

Manufacturer	North America	Europe	Asia
BP Microsystems	(800) 225-2102 (713) 688-4600 Fax: (713) 688-0920	+49-8152-4183 +49-8856-932616	+852-234-16611 +852-2710-8121
Data I/O	(800) 426-1045 (206) 881-6444 Fax: (206) 882-1043	+44-0734-440011	Call North America
HI-LO	(510) 623-8860	Call Asia	+886-2-764-0215 Fax: +886-2-756-6403
ICE Technology	(800) 624-8949 (919) 430-7915	+44-1226-767404 Fax: 0-1226-370-434	
MetaLink	(800) 638-2423 (602) 926-0797 Fax: (602) 693-0681	+49-80 9156 96-0 Fax: +49-80 9123 86	+852-737-1800
Systems General	(408) 263-6667	! 41 1 9450300	! 886 2 917 3005 Fax: +886-2-911-1283
Needhams	(916) 924-8037 Fax: (916) 924-8065		

Development Support (Continued)

AVAILABLE LITERATURE

For more information, please see the COP8 Basic Family User's Manual, Literature Number 620895, COP8 Feature Family User's Manual, Literature Number 620897 and National's Family of 8-bit Microcontrollers COP8 Selection Guide, Literature Number 630009.

DIAL-A-HELPER SERVICE

Dial-A-Helper is a service provided by the Microcontroller Applications group. The Dial-A-Helper is an Electronic Information System that may be accessed as a Bulletin Board System (BBS) via data modem, as an FTP site on the Internet via standard FTP client application or as an FTP site on the Internet using a standard Internet browser such as Netscape or Mosaic.

The Dial-A-Helper system provides access to an automated information storage and retrieval system . The system capabilities include a MESSAGE SECTION (electronic mail, when accessed as a BBS) for communications to and from the Microcontroller Applications Group and a FILE SECTION which consists of several file areas where valuable application software and utilities could be found.

DIAL-A-HELPER BBS via a Standard Modem

Modem:	CANADA/U.S.:	(800) NSC-MICRO (800) 672-6427	
	EUROPE:	(+49) 0-8141-351332	
	Baud:	14.4k	
	Set-Up:	Length:	8-Bit
		Parity:	None
		Stop Bit:	1
	Operation:	24 Hours, 7 Days	

DIAL-A-HELPER via FTP

ftp nscmicro.nsc.com
user: anonymous
password: username@yourhost.site.domain

DIAL-A-HELPER via WorldWide Web Browser

ftp://nscmicro.nsc.com

National Semiconductor on the WorldWide Web

See us on the WorldWide Web at: http://www.national.com

CUSTOMER RESPONSE CENTER

Complete product information and technical support is available from National's customer response centers.

CANADA/U.S.:	Tel:	(800)272-9959
	email:	support@tevm2.nsc.com
EUROPE:	email:	europe.support@nsc.com
	Deutsch Tel:	+49 (0) 180-530 85 85
	English Tel:	+49 (0) 180-532 78 32
	Français Tel:	+49 (0) 180-532 93 58
	Italiano Tel:	+49 (0) 180-534 16 80
JAPAN:	Tel:	+81-043-299-2309
S.E. ASIA:	Beijing Tel:	(+86) 10-6856-8601
	Shanghai Tel:	(+86) 21-6415-4092
	Hong Kong Tel:	(+852) 2737-1600
	Korea Tel:	(+82) 2-3771-6909
	Malaysia Tel:	(+60-4) 644-9061
	Singapore Tel:	(+65) 255-2226
	Taiwan Tel:	+886-2-521-3288
AUSTRALIA:	Tel:	(+61) 3-9558-9999
INDIA:	Tel:	(+91) 80-559-9467

Section 5

COP8™ Feature Family Products

Section 5 Contents

COP884BC/COP684BC 8-Bit Microcontrollers with CAN Interface 5-3
COP888EB/COP889EB/COP688EB/COP689EB 8-Bit Microcontroller with CAN Interface, A/D, and UART 5-59
COP688CL/COP684CL/COP888CL/COP884CL/COP988CL/COP984CL 8-Bit Microcontroller 5-130
COP888CF/COP884CF/COP988CF/COP984CF 8-Bit CMOS Microcontroller with A/D Converter 5-167
COP8ACC5 8-Bit Microcontroller with High Resolution A/D Conversion 5-202
COP688EK/COP684EK/COP888EK/COP884EK/COP988EK/COP984EK 8-Bit Microcontroller with Analog Function Block 5-240
COP688GD/COP888GD/COP988GD 8-Bit Microcontroller with A/D Converter 5-279
COP888CG/COP884CG 8-Bit Microcontroller with UART and Three Multi-Function Timers .. 5-281
COP688CS/COP684CS/COP888CS/COP884CS/COP988CS/COP984CS 8-Bit Microcontroller with UART and One Multi-Function Timer 5-319
COP688EG/COP684EG/COP888EG/COP884EG/COP988EG/COP984EG 8-Bit Microcontroller with UART and Three Multi-Function Timers 5-362
COP688FH/COP684FH/COP888FH/COP884FH/COP988FH/COP984FH 8-Bit Microcontroller with UART, Three Multi-Function Timers and Multiply/Divide Block 5-406
COP688GG/COP888GG 8-Bit Microcontroller with UART, and Three Multi-Function Timers .. 5-449
COP688HG/COP888HG 8-Bit Microcontroller with UART, and Three Multi-Function Timers .. 5-492
COP688KG/COP888KG 8-Bit Microcontroller with UART, and Three Multi-Function Timers .. 5-535
COP888GW 8-Bit Microcontroller with Pulse Train Generators and Capture Modules 5-576

National Semiconductor

PRELIMINARY

COP684BC/COP884BC
8-Bit Microcontrollers with CAN Interface

General Description

The COP684BC/COP884BC are members of the COP8™ feature family of microcontrollers which uses an 8-bit single chip core architecture fabricated with National Semiconductor's M²CMOS™ process technology. Each device is a member of this expandable 8-bit core processor family of microcontrollers. (Continued)

Key Features

- CAN Interface
- On chip reset
- One 16-bit timer, with two 16-bit registers supporting:
  - — Processor Independent PWM mode
  - — External Event counter mode
  - — Input Capture mode
- High speed, constant resolution 8-bit PWM/frequency monitor timer with 2 output pins
- 2048 bytes on-board ROM
- 64 bytes on-board RAM

Additional Peripheral Features

- Idle Timer
- Multi-Input Wake Up (MIWU) with optional interrupts (7)
- Two analog comparators
- MICROWIRE/PLUS™ serial I/O

I/O Features

- Memory mapped I/O
- Software selectable I/O options (TRI-STATE® Output, Push-Pull Output, Weak Pull-Up Input, High Impedance Input)
- Schmitt trigger inputs on ports G and L
- Package: 28 SO with 18 I/O pins

CPU/Instruction Set Features

- 1 μs instruction cycle time
- Eleven multi-source vectored interrupts servicing
  - — External Interrupt
  - — Idle Timer T0
  - — Timer T1 (with 2 Interrupts)
  - — MICROWIRE/PLUS
  - — Multi-Input Wake Up
  - — Software Trap
  - — PWM Timer
  - — CAN Interface (with 3 interrupts)
- Versatile and easy to use instruction set
- 8-bit Stack Pointer (SP)—stack in RAM
- Two 8-bit Register Indirect Data Memory Pointers (B and X)

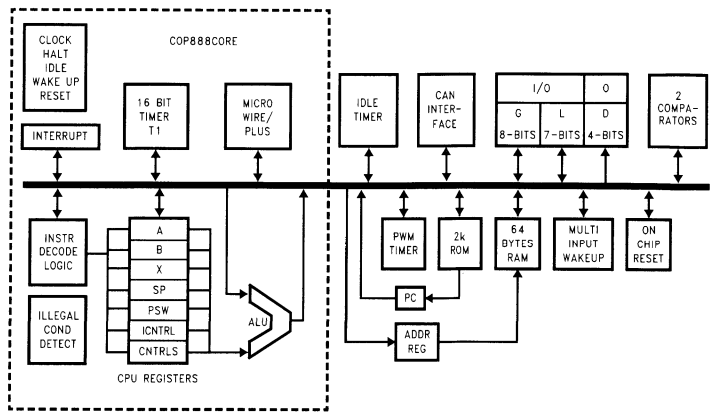
Fully Static CMOS

- Two power saving modes: HALT and IDLE
- Low current drain (typically <1 μA)
- Single supply operation: 4.5V–5.5V
- Temperature ranges: $-40°C$ to $+85°C$, $-55°C$ to $+125°C$

Development Support

- Emulation and OTP devices
- Real time emulation and full program debug offered by MetaLink Development Systems

Block Diagram



TL/DD/12067–1

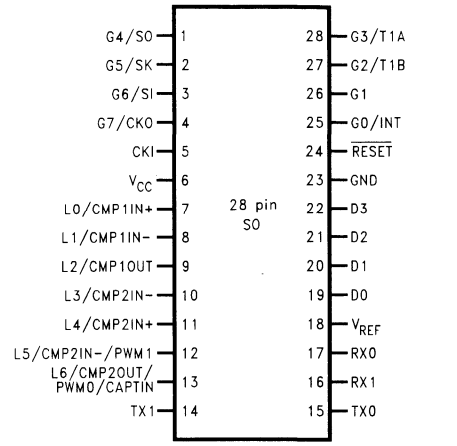
FIGURE 1

5

General Description (Continued)

It is a fully static part, fabricated using double-metal silicon gate microCMOS technology. Features include an 8-bit memory mapped architecture, MICROWIRE/PLUS serial I/O, a 16-bit timer/counter supporting three modes (Processor Independent PWM generation, External Event counter, and Input Capture mode capabilities), a CAN interface, two comparators, 8-bit, high speed, constant resolution PWM/ frequency monitor timer, and two power savings modes (HALT and IDLE), both with a multi-sourced wake up/ interrupt capability. This multi-sourced interrupt capability may also be used independent of the HALT or IDLE modes. Each I/O pin has software selectable configurations. The device operates over a voltage range of 4.5V to 5.5V. High throughput is achieved with an efficient, regular instruction set operating at a maximum of 1 μs per instruction rate. The device has low EMI emissions. Low radiated emissions are achieved by gradual turn-on output drivers and internal I_{CC} smoothing filters on the chip logic and crystal oscillator.

Connection Diagram



TL/DD/12067-2

Top View

Order Number COP884BC-xxx/M or COP684BC-xxx/M
See NS Package Number M28B

FIGURE 2

Pinouts for 28-Pin SO Package

Port Pin	Type	Alt. Function	28-Pin SO
G0	I/O	INTR	25
G1	I/O		26
G2	I/O	T1B	27
G3	I/O	T1A	28
G4	I/O	SO	1
G5	I/O	SK	2
G6	I	SI	3
G7	I	CKO	4
L0	I/O	CMP1IN+/MIWU	7
L1	I/O	CMP1IN−/MIWU	8
L2	I/O	CMP1OUT/MIWU	9
L3	I/O	CMP2IN−/MIWU	10
L4	I/O	CMP2IN+/MIWU	11
L5	I/O	CMP2IN−/PWM1/MIWU	12
L6	I/O	CMP2OUT/PWM0/ CAPTIN/MIWU	13
D0	O		19
D1	O		20
D2	O		21
D3	O		22
CAN V_{REF}			18
CAN Tx0	O		15
CAN Tx1	O		14
CAN Rx0	I	MIWU (Note A)	17
CAN Rx1	I	MIWU	16
V_{CC}			6
GND			23
CKI	I		5
RESET	I		24

Note A: The MIWU function for the CAN interface is internal (see CAN interface block diagram)

Absolute Maximum Ratings (Note)

If Military/Aerospace specified devices are required, please contact the National Semiconductor Sales Office/Distributors for availability and specifications.

Supply Voltage (V_{CC})	6V
Voltage at Any Pin	−0.3V to V_{CC} +0.3V
Total Current into V_{CC} Pin (Source)	90 mA
Total Current out of GND Pin (Sink)	100 mA
Storage Temperature Range	−65°C to +150°C

Note: *Absolute maximum ratings indicate limits beyond which damage to the device may occur. DC and AC electrical specifications are not ensured when operating the device at absolute maximum ratings.*

DC Electrical Characteristics COP884BC: $-40°C \le T_A \le +85°C$

Parameter	Conditions	Min	Typ	Max	Units
Operating Voltage		4.5		5.5	V
Power Supply Ripple (Note 1)	Peak-to-Peak			0.1 V_{CC}	V
Supply Current					
CKI = 10 MHz (Note 2)	V_{CC} = 5.5V, t_c = 1 μs			15	mA
HALT Current (Notes 3, 4)	V_{CC} = 5.5V, CKI = 0 MHz				
	Power-On Reset Enabled		<300	480	μA
	Power-On Reset Disabled		<250	380	μA
IDLE Current (Note 4)					
CKI = 10 MHz	V_{CC} = 5.5V, t_c = 1 μs			5.5	mA
Input Levels (V_{IH}, V_{IL})					
Reset, CKI					
Logic High		0.8 V_{CC}			V
Logic Low				0.2 V_{CC}	V
All Other Inputs					
Logic High		0.7 V_{CC}			V
Logic Low				0.2 V_{CC}	V
Hi-Z Input Leakage	V_{CC} = 5.5V			±2	μA
Input Pull-up Current	V_{CC} = 5.5V, V_{IN} = 0V	−40		−250	μA
G and L Port Input Hysteresis	(Notes 6, 7)		0.05 V_{CC}		V
Output Current Levels D Outputs					
Source	V_{CC} = 4.5V, V_{OH} = 3.3V	−0.4			mA
Sink	V_{CC} = 4.5V, V_{OL} = 1.0V	10			mA
Comparator Output (L2, L6)					
Source (Push-Pull)	V_{CC} = 4.5V, V_{OH} = 3.3V	−1.6			mA
Sink (Push-Pull)	V_{CC} = 4.5V, V_{OL} = 0.4V	1.6			mA
CAN Transmitter Outputs					
Source (Tx1)	V_{CC} = 4.5V, V_{OH} = V_{CC} − 0.1V	−1.5			mA
	V_{CC} = 4.5V, V_{OH} = V_{CC} − 0.6V	−10			mA
Sink (Tx0)	V_{CC} = 4.5V, V_{OL} = 0.1V	1.5			mA
	V_{CC} = 4.5V, V_{OL} = 0.6V	10			mA
All Others					
Source (Weak Pull-Up)	V_{CC} = 4.5V, V_{OH} = 2.7V	−10		−110	μA
Source (Push-Pull)	V_{CC} = 4.5V, V_{OH} = 3.3V	−0.4			mA
Sink (Push-Pull)	V_{CC} = 4.5V, V_{OL} = 0.4V	1.6			mA
TRI-STATE Leakage	V_{CC} = 5.5V			±2.0	μA

DC Electrical Characteristics COP884BC: $-40°C \leq T_A \leq +85°C$ (Continued)

Parameter	Conditions	Min	Typ	Max	Units
Allowable Sink/Source Current per Pin					
D Outputs (Sink)				15	mA
Tx0 (Sink) (Note 7)				30	mA
Tx1 (Source) (Note 7)				30	mA
All Other				3	mA
Maximum Input Current without Latchup (Notes 5, 7)	Room Temp			±100	mA
RAM Retention Voltage, V_r (Note 6)	500 ns Rise and Fall Time	2.0			V
Input Capacitance	(Note 7)			7	pF
Load Capacitance on D2				1000	pF

Note 1: Maximum rate of voltage change must be less than 0.5 V/ms

Note 2: Supply current is measured after running 2000 cycles with a square wave CKI input, CKO open, inputs at V_{CC} or GND, and outputs open.

Note 3: The HALT mode will stop CKI from oscillating in the Crystal configurations. Halt test conditions: All inputs tied to V_{CC}; L, and G port I/Os configured as outputs and programmed low; D outputs programmed low. Parameter refers to HALT mode entered via setting bit 7 of the G Port data register. Part will pull up CKI during HALT in crystal clock mode.

Note 4: HALT and IDLE current specifications assume CAN block and comparators are disabled.

Absolute Maximum Ratings (Note)

If Military/Aerospace specified devices are required, please contact the National Semiconductor Sales Office/Distributors for availability and specifications.

Supply Voltage (V_{CC})	7V
Voltage at Any Pin	−0.3V to V_{CC} +0.3V
Total Current into V_{CC} Pin (Source)	100 mA
Total Current out of GND Pin (Sink)	110 mA
Storage Temperature Range	−65°C to +150°C

Note: *Absolute maximum ratings indicate limits beyond which damage to the device may occur. DC and AC electrical specifications are not ensured when operating the device at absolute maximum ratings.*

DC Electrical Characteristics COP684BC: $-55°C \le T_A \le +125°C$

Parameter	Conditions	Min	Typ	Max	Units
Operating Voltage		4.5		5.5	V
Power Supply Ripple (Note 1)	Peak-to-Peak			0.1 V_{CC}	V
Supply Current					
CKI = 10 MHz (Note 2)	V_{CC} = 5.5V, t_c = 1 μs			15	mA
HALT Current (Notes 3, 4)	V_{CC} = 5.5V, CKI = 0 MHz				
	Power-On Reset Enabled		<300	480	μA
	Power-On Reset Disabled		<250	380	μA
IDLE Current (Note 4)					
CKI = 10 MHz	V_{CC} = 5.5V, t_c = 1 μs			5.5	mA
Input Levels (V_{IH}, V_{IL})					
Reset, CKI					
Logic High		0.8 V_{CC}			V
Logic Low				0.2 V_{CC}	V
All Other Inputs					
Logic High		0.7 V_{CC}			V
Logic Low				0.2 V_{CC}	V
Hi-Z Input Leakage	V_{CC} = 5.5V			±5	μA
Input Pull-up Current	V_{CC} = 5.5V, V_{IN} = 0V	−35		−250	μA
G and L Port Input Hysteresis	(Note 6)		0.05 V_{CC}		V
Output Current Levels D Outputs					
Source	V_{CC} = 4.5V, V_{OH} = 3.3V	−0.4			mA
Sink	V_{CC} = 4.5V, V_{OL} = 1.0V	9.0			mA
Comparator Output (L2, L6)					
Source (Push-Pull)	V_{CC} = 4.5V, V_{OH} = 3.3V	−1.6			mA
Sink (Push-Pull)	V_{CC} = 4.5V, V_{OL} = 0.4V	1.6			mA
CAN Transmitter Outputs					
Source (Tx1)	V_{CC} = 4.5V, V_{OH} = V_{CC} − 0.1V	−1.5			mA
	V_{CC} = 4.5V, V_{OH} = V_{CC} − 0.6V	−10			mA
Sink (Tx0)	V_{CC} = 4.5V, V_{OL} = 0.1V	1.5			mA
	V_{CC} = 4.5V, V_{OL} = 0.6V	10			mA
All Others					
Source (Weak Pull-Up)	V_{CC} = 4.5V, V_{OH} = 2.7V	−9.0		−100	μA
Source (Push-Pull)	V_{CC} = 4.5V, V_{OH} = 3.3V	−0.4			mA
Sink (Push-Pull)	V_{CC} = 4.5V, V_{OL} = 0.4V	1.4			mA
TRI-STATE Leakage	V_{CC} = 5.5V			±5.0	μA

DC Electrical Characteristics COP684BC: $-55°C \leq T_A \leq +125°C$ (Continued)

Parameter	Conditions	Min	Typ	Max	Units
Allowable Sink/Source Current per Pin					
D Outputs (Sink)				12	mA
Tx0 (Sink) (Note 7)				30	mA
Tx1 (Source) (Note 7)				30	mA
All Other				2.5	mA
Maximum Input Current without Latchup (Notes 5, 7)	Room Temp			±100	mA
RAM Retention Voltage, V_r (Note 6)	500 ns Rise and Fall Time	2.0			V
Input Capacitance	(Note 7)			7	pF
Load Capacitance on D2				1000	pF

Note 5: Pins G6 and $\overline{\text{RESET}}$ are designed with a high voltage input network. These pins allow input voltages greater than V_{CC} and the pins will have sink current to V_{CC} when biased at voltages greater than V_{CC} (the pins do not have source current when biased at a voltage below V_{CC}). The effective resistance to V_{CC} is 750Ω (typical). These two pins will not latch up. The voltage at the pins must be limited to less than 14V.

Note 6: Condition and parameter valid only for part in HALT mode.

Note 7: Parameter characterized but not tested.

AC Electrical Characteristics: COP684BC and COP884BC: $-55°C \le T_A \le +125°C$

Parameter	Conditions	Min	Typ	Max	Units
Instruction Cycle Time (t_c)					
Crytal/Resonator	$V_{CC} \ge 4.5V$	1.0		DC	μs
Inputs					
t_{SETUP}	$V_{CC} \ge 4.5V$	200			ns
t_{HOLD}	$V_{CC} \ge 4.5V$	60			ns
PWM Capture Input					
t_{SETUP}	$V_{CC} \ge 4.5V$	30			ns
t_{HOLD}	$V_{CC} \ge 4.5V$	70			ns
Output Propagation Delay					
(t_{PD1}, t_{PD0}) (Note 8)	$C_L = 100$ pF, $R_L = 2.2$ kΩ				
SK, SO	$V_{CC} \ge 4.5V$			0.7	μs
PWM Outputs	$V_{CC} \ge 4.5V$			75	ns
All Others	$V_{CC} \ge 4.5V$			1	μs
MICROWIRE					
Setup Time (t_{UWS}) (Note 9)		20			ns
Hold Time (t_{UWH}) (Note 9)		56			ns
Output Prop Delay (t_{UPD})				220	ns
Input Pulse Width (Note 10)					
Interrupt High Time		1			t_c
Interrupt Low Time		1			t_c
Timer 1,2 High Time		1			t_c
Timer 1,2 Low Time		1			t_c
Reset Pulse Width (Note 9)		1.0			μs
Power Supply Rise Time for Proper Operation of On-Chip RESET		50 μs		$256*t_c$	

Note: For device testing purposes of all AC parameters, V_{OH} will be tested at $0.5*V_{CC}$.

Note 8: The output propagation delay is referenced to the end of the instruction cycle where the output change occurs.

Note 9: Parameter not tested.

Note 10: t_c = Instruction Cycle Time.

On-Chip Voltage Reference: $-55°C \le T_A \le +125°C$

Parameter	Conditions	Min	Max	Units
Reference Voltage V_{REF}	$I_{OUT} < 80$ μA, $V_{CC} = 5V$	$0.5\ V_{CC} - 0.12$	$0.5\ V_{CC} + 0.12$	V
Reference Supply Current, I_{DD}	$I_{OUT} = 0A$, (No Load) $V_{CC} = 5V$ (Note A)		120	μA

Note A: Reference supply I_{DD} is supplied for information purposes only, it is not tested.

Comparator DC/AC Characteristics: $4.5V \le V_{CC} \le 5.5V$, $-55°C \le T_A \le +125°C$

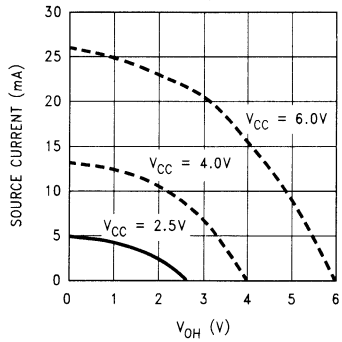
Parameter	Conditions	Min	Typ	Max	Units
Input Offset Voltage	$0.4V < V_{IN} < V_{CC} - 1.5V$		±10	±25	mV
Input Common Mode Voltage Range		0.4		$V_{CC} - 1.5$	V
Voltage Gain			300k		V/V
Outputs Sink/Source	See I/O-Port DC Specifications				
DC Supply Current (when enabled)	$V_{CC} = 6.0V$			250	μA
Response Time	TBD mV Step, TBD mV Overdrive, 100 pF Load		1		μs

CAN Comparator DC and AC Characteristics: $4.8V \le V_{CC} \le 5.2V$, $-40°C \le T_A \le +125°C$

Parameters	Conditions	Min	Typ	Max	Units
Differential Input Voltage				±25	mV
Input Offset Voltage	$1.5V < V_{IN} < V_{CC} - 1.5V$			±10	mV
Input Common Mode Voltage Range		1.5		$V_{CC} - 1.5$	V
Input Hysteresis		8			mV

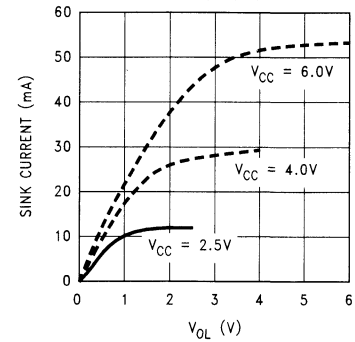
Typical Performance Characteristics $-55°C < T_A < +125°C$

Port D Source Current



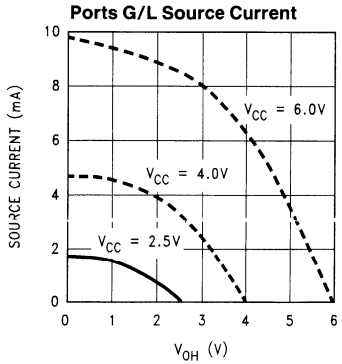
TL/DD/12067-39

Port D Sink Current



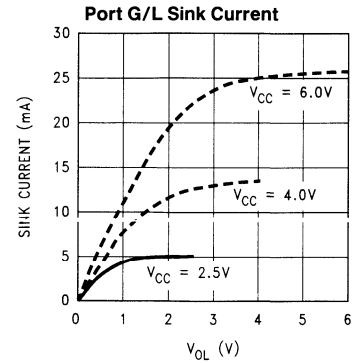
TL/DD/12067-40

Ports G/L Source Current



TL/DD/12067-41

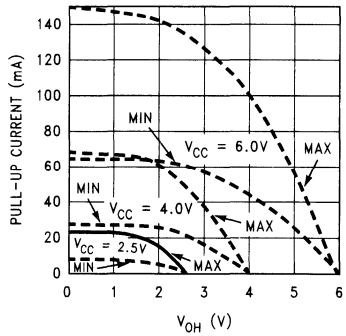
Port G/L Sink Current



TL/DD/12067-42

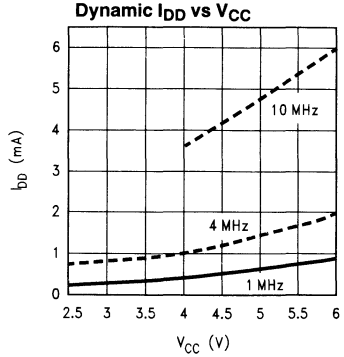
Typical Performance Characteristics $-55°C \le T_A \le +125°C$ (Continued)

Ports G/L Weak Pull-Up Source Current



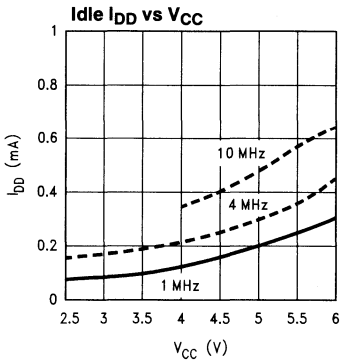
TL/DD/12067-43

Dynamic I_{DD} vs V_{CC}



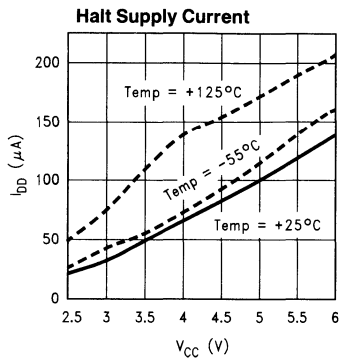
TL/DD/12067-44

Idle I_{DD} vs V_{CC}



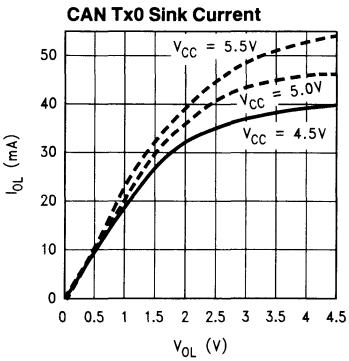
TL/DD/12067-45

Halt Supply Current



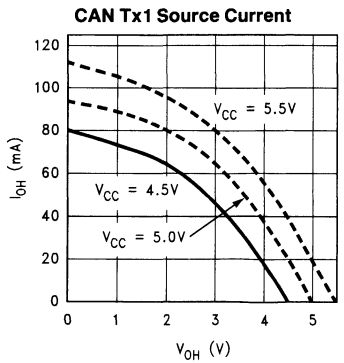
TL/DD/12067-46

CAN Tx0 Sink Current



TL/DD/12067-47

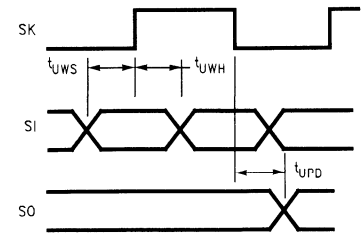
CAN Tx1 Source Current



TL/DD/12067-48

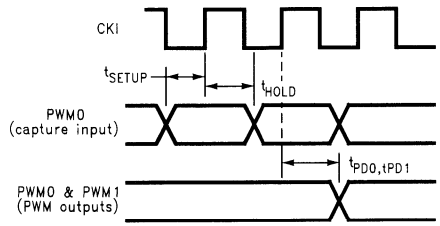
5

AC Electrical Characteristics (Continued)



TL/DD/12067–3

FIGURE 3. MICROWIRE/PLUS Timing Diagram



TL/DD/12067–4

FIGURE 4. PWM/CAPTURE Timer Input/Output Timing Diagram

Pin Descriptions

V_{CC} and GND are the power supply pins.

CKI is the clock input. The clock can come from a crystal oscillator (in conjunction with CKO). See Oscillator Description section.

$\overline{\text{RESET}}$ is the master reset input. See Reset Description section.

The device contains one bidirectional 8-bit I/O port (G), and one 7-bit bidirectional I/O port (L) where each individual bit may be independently configured as an input (Schmitt trigger inputs on ports G and L), output or TRI-STATE under program control. Three data memory address locations are allocated for each of these I/O ports. Each I/O port has two associated 8-bit memory mapped registers, the CONFIGURATION register and the output DATA register. A memory mapped address is also reserved for the input pins of each I/O port. (See the memory map for the various addresses associated with the I/O ports.) *Figure 5* shows the I/O port configurations for the device. The DATA and CONFIGURATION registers allow for each port bit to be individually configured under software control as shown below:

Configuration Register	Data Register	Port Set-Up
0	0	Hi-Z Input (TRI-STATE Output)
0	1	Input with Weak Pull-Up
1	0	Push-Pull Zero Output
1	1	Push-Pull One Output

PORT L is a 7-bit I/O port. All L-pins have Schmitt triggers on the inputs.

Port L supports Multi-Input Wake Up (MIWU) on all seven pins.

Port L has the following alternate features:

L0 MIWU or CMP1IN+
L1 MIWU or CMP1IN−
L2 MIWU or CMP1OUT
L3 MIWU or CMP2IN−
L4 MIWU or CMP2IN+
L5 MIWU or CMP2IN− or PWM1
L6 MIWU or CMP2OUT or PWM0 or CAPTIN

Port G is an 8-bit port with 5 I/O pins (G0–G5), an input pin (G6), and one dedicated output pin (G7). Pins G0–G6 all have Schmitt Triggers on their inputs. G7 serves as the dedicated output pin for the CKO clock output. There are two registers associated with the G Port, a data register and a configuration register. Therefore, each of the 6 I/O bits (G0–G5) can be individually configured under software control.

Since G6 is an input only pin and G7 is the dedicated CKO clock output pin the associated bits in the data and configuration registers for G6 and G7 are used for special purpose functions as outlined below. Reading the G6 and G7 data bits will return zeroes.

Note that the chip will be placed in the HALT mode by writing a "1" to bit 7 of the Port G Data Register. Similarly the chip will be placed in the IDLE mode by writing a "1" to bit 6 of the Port G Data Register.

Writing a "1" to bit 6 of the Port G Configuration Register enables the MICROWIRE/PLUS to operate with the alternate phase of the SK clock.

	Config. Register	Data Register
G7		HALT
G6	Alternate SK	IDLE

CAN pins: For the on-chip CAN interface this device has five dedicated pins with the following features:

V_{REF} On-chip reference voltage with the value of $V_{CC}/2$

Rx0 CAN receive data input pin.

Rx1 CAN receive data input pin.

Tx0 CAN transmit data output pin. This pin may be put in the TRI-STATE mode with the TXEN0 bit in the CAN Bus control register.

Tx1 CAN transmit data output pin. This pin may be put in the TRI-STATE mode with the TXEN1 bit in the CAN Bus control register.

Port G has the following alternate features:

G0 INTR (External Interrupt Input)
G2 T1B (Timer T1 Capture Input)
G3 T1A (Timer T1 I/O)
G4 SO (MICROWIRE Serial Data Output)
G5 SK (MICROWIRE Serial Clock)
G6 SI (MICROWIRE Serial Data Input)

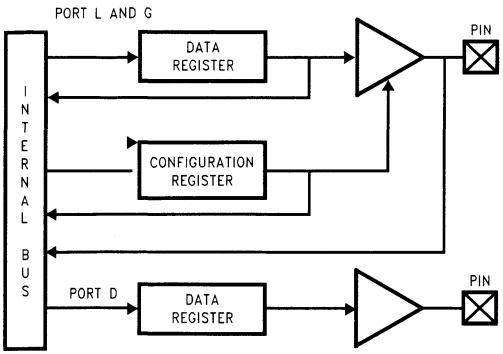
Port G has the following dedicated function:

G7 CKO Oscillator dedicated output

Port D is a 4-bit output port that is preset high when RESET goes low. The user can tie two or more D port outputs (except D2) together in order to get a higher drive.

Note: Care must be exercised with the D2 pin operation. At RESET, the external loads on this pin must ensure that the output voltages stay above 0.8 V_{CC} to prevent the chip from entering special modes. Also keep the external loading on D2 to less than 1000 pF.

Pin Descriptions (Continued)



TL/DD/12067–5

FIGURE 5. I/O Port Configurations

Functional Description

The architecture of the device utilizes a modified Harvard architecture. With the Harvard architecture, the control store program memory (ROM) is separated from the data store memory (RAM). Both ROM and RAM have their own separate addressing space with separate address buses. The architecture, though based on Harvard architecture, permits transfer of data from ROM to RAM.

CPU REGISTERS

The CPU can do an 8-bit addition, subtraction, logical or shift operation in one instruction (t_C) cycle time.

There are five CPU registers:

A is the 8-bit Accumulator Register

PC is the 15-bit Program Counter Register

PU is the upper 7 bits of the program counter (PC)

PL is the lower 8 bits of the program counter (PC)

B is an 8-bit RAM address pointer, which can be optionally post auto incremented or decremented.

X is an 8-bit alternate RAM address pointer, which can be optionally post auto incremented or decremented.

SP is the 8-bit stack pointer, which points to the subroutine/interrupt stack (in RAM). The SP is initialized to RAM address 02F with reset.

All the CPU registers are memory mapped with the exception of the Accumulator (A) and the Program Counter (PC).

PROGRAM MEMORY

Program memory for the device consists of 2048 bytes of ROM. These bytes may hold program instructions or constant data (data tables tor the LAID instruction, jump vectors for the JID instruction, and interrupt vectors for the VIS instruction). The program memory is addressed by the 15-bit program counter (PC). All interrupts in the device vector to program memory location 0FF Hex.

DATA MEMORY

The data memory address space includes the on-chip RAM and data registers, the I/O registers (Configuration, Data and Pin), the control registers, the MICROWIRE/PLUS SIO shift register, and the various registers, and counters associated with the timers (with the exception of the IDLE timer). Data memory is addressed directly by the instruction or indirectly by the B, X and SP pointers.

The device has 64 bytes of RAM. Sixteen bytes of RAM are mapped as "registers" at addresses 0F0 to 0FF Hex. These registers can be loaded immediately, and also decremented and tested with the DRSZ (decrement register and skip if zero) instruction. The memory pointer registers X, SP, and B are memory mapped into this space at address locations 0FC to 0FE Hex respectively, with the other registers (other than reserved register 0FF) being available for general usage.

The instruction set permits any bit in memory to be set, reset or tested. All I/O and registers (except A and PC) are memory mapped; therefore, I/O bits and register bits can be directly and individually set, reset and tested. The accumulator (A) bits can also be directly and individually tested.

Note: RAM contents are undefined upon power-up.

RESET

The $\overline{\text{RESET}}$ input when pulled low initializes the microcontroller. Initialization will occur whenever the $\overline{\text{RESET}}$ input is pulled low. Upon initialization, the data and configuration registers for Ports L and G, are cleared, resulting in these Ports being initialized to the TRI-STATE mode. Port D is initialized high with $\overline{\text{RESET}}$. The PC, PSW, CNTRL, and ICNTRL control registers are cleared. The Multi-Input Wake Up registers WKEN, WKEDG, and WKPND are cleared. The Stack Pointer, SP, is initialized to 02F Hex.

The following initializations occur with $\overline{\text{RESET}}$:

Port L: TRI-STATE

Port G: TRI-STATE

Port D: HIGH

PC: CLEARED

PSW, CNTRL and ICNTRL registers: CLEARED

Accumulator and Timer 1:

RANDOM after RESET with power already applied

RANDOM after RESET at power-on

SP (Stack Pointer): Loaded with 2F Hex

CMPSL (Comparator control register): CLEARED

PWMCON (PWM control register): CLEARED

B and X Pointers:

UNAFFECTED after RESET with power already applied

RANDOM after RESET at power-up

RAM:

UNAFFECTED after RESET with power already applied

RANDOM after RESET at power-up

CAN:

The CAN Interface comes out of external reset in the "error-active" state and waits until the user's software sets either one or both of the TXEN0, TXEN1 bits to "1". After that, the device will not start transmission or reception of a frame until eleven consecutive "recessive" (undriven) bits have been received. This is done to ensure that the output drivers are not enabled during an active message on the bus.

CSCAL, CTIM, TCNTL, TEC, REC: CLEARED

RTSTAT: CLEARED with the exception of the TBE bit which is set to 1

RID, RIDL, TID, TDLC: RANDOM

Functional Description (Continued)

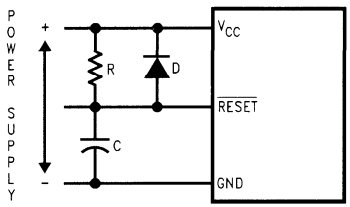
ON-CHIP POWER-ON RESET

The device is designed with an on-chip power-on reset circuit which will trigger a 256 t_C delay as V_{CC} rises above the minimum RAM retention voltage (V_r). This delay allows the oscillator to stabilize before the device exits the reset state. The contents of data registers and RAM are unknown following an on-chip power-on reset. The external reset takes priority over the on-chip reset and will deactivate the 256 t_C delay if in progress.

When using external reset, the external RC network shown in *Figure 6* should be used to ensure that the $\overline{\text{RESET}}$ pin is held low until the power supply to the chip stabilizes.

Under no circumstances should the $\overline{\text{RESET}}$ pin be allowed to float. If the on-chip power-on reset feature is being used, $\overline{\text{RESET}}$ should be connected directly to V_{CC}. Be aware of the Power Supply Rise Time requirements specified in the DC Specifications Table. These requirements must be met for the on-chip power-on reset to function properly.

The on-chip power-on reset circuit may reset the device if the operating voltage (V_{CC}) goes below V_r.



TL/DD/12067–6

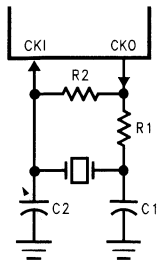
RC > 5 x Power Supply Rise Time

FIGURE 6. Recommended Reset Circuit

Oscillator Circuits

The chip can be driven by a clock input on the CKI input pin which can be between DC and 10 MHz. The CKO output clock is on pin G7. The CKI input frequency is divided by 10 to produce the instruction cycle clock ($1/t_C$).

Figure 7 shows the Crystal diagram.



TL/DD/12067–7

FIGURE 7. Crystal Oscillator Diagram

CRYSTAL OSCILLATOR

CKI and CKO can be connected to make a closed loop crystal (or resonator) controlled oscillator.

Table I shows the component values required for various standard crystal values.

TABLE I. Crystal Oscillator Configuration, T_A = 25°C

R1 (kΩ)	R2 (MΩ)	C1 (pF)	C2 (pF)	CKI Freq. (MHz)	Conditions
0	1	30	30–36	10	V_{CC} = 5V
0	1	30	30–36	4	V_{CC} = 5V
0	1	200	100–150	0.455	V_{CC} = 5V

Control Registers

CNTRL Register (Address X'00EE)

The Timer1 (T1) and MICROWIRE/PLUS control register contains the following bits:

SL1 & SL0 Select the MICROWIRE/PLUS clock divide by (00 = 2, 01 = 4, 1x = 8)

IEDG External interrupt edge polarity select (0 = Rising edge, 1 = Falling edge)

MSEL Selects G5 and G4 as MICROWIRE/PLUS signals SK and SO respectively

T1C0 Timer T1 Start/Stop control in timer

Timer T1 Underflow Interrupt Pending Flag in timer mode 3

T1C1 Timer T1 mode control bit

T1C2 Timer T1 mode control bit

T1C3 Timer T1 mode control bit

T1C3	T1C2	T1C1	T1C0	MSEL	IEDG	SL1	SL0

Bit 7 Bit 0

Control Registers (Continued)

PSW Register (Address X'00EF)

The PSW register contains the following select bits:

- GIE Global interrupt enable (enables interrupts)
- EXEN Enable external interrupt
- BUSY MICROWIRE/PLUS busy shifting flag
- EXPND External interrupt pending
- T1ENA Timer T1 Interrupt Enable for Timer Underflow or T1A Input capture edge
- T1PNDA Timer T1 Interrupt Pending Flag (Autoreload RA in mode 1, T1 Underflow in Mode 2, T1A capture edge in mode 3)
- C Carry Flag
- HC Half Carry Flag

HC	C	T1PNDA	T1ENA	EXPND	BUSY	EXEN	GIE

Bit 7 Bit 0

The Half-Carry bit is also affected by all the instructions that affect the Carry flag. The SC (Set Carry) and RC (Reset Carry) instructions will respectively set or clear both the carry flags. In addition to the SC and RC instructions, ADC, SUBC, RRC and RLC instructions affect the carry and Half Carry flags.

ICNTRL Register (Address X'00E8)

The ICNTRL register contains the following bits:

- T1ENB Timer T1 Interrupt Enable for T1B Input capture edge
- T1PNDB Timer T1 Interrupt Pending Flag for T1B capture edge
- μWEN Enable MICROWIRE/PLUS interrupt
- μWPND MICROWIRE/PLUS interrupt pending
- T0EN Timer T0 Interrupt Enable (Bit 12 toggle)
- T0PND Timer T0 Interrupt pending
- LPEN L Port Interrupt Enable (Multi-Input Wake Up/ Interrupt)

  Bit 7 could be used as a flag

Unused	LPEN	T0PND	T0EN	μWPND	μWEN	T1PNDB	T1ENB

Bit 7 Bit 0

Timers

The device contains a very versatile set of timers (T0, T1, and an 8-bit PWM timer). All timers and associated autoreload/capture registers power up containing random data.

Figure 8 shows a block diagram for timers T1 and T0 on the device.

TIMER T0 (IDLE TIMER)

The device supports applications that require maintaining real time and low power with the IDLE mode. This IDLE mode support is furnished by the IDLE timer T0, which is a 16-bit timer. The Timer T0 runs continuously at the fixed rate of the instruction cycle clock, t_C. The user cannot read or write to the IDLE Timer T0, which is a count down timer.

The Timer T0 supports the following functions:

Exit out of the Idle Mode (See Idle Mode description)

Start up delay out of the HALT mode

The IDLE Timer T0 can generate an interrupt when the thirteenth bit toggles. This toggle is latched into the T0PND pending flag, and will occur every 4.096 ms at the maximum clock frequency (t_C = 1 μs). A control flag T0EN allows the interrupt from the thirteenth bit of Timer T0 to be enabled or disabled. Setting T0EN will enable the interrupt, while resetting it will disable the interrupt.

TIMER T1

The device has a powerful timer/counter block, T1.

The timer block consists of a 16-bit timer, T1, and two supporting 16-bit autoreload/capture registers, R1A and R1B. The timer block has two pins associated with it, T1A and T1B. The pin T1A supports I/O required by the timer block, while the pin T1B is an input to the timer block. The powerful and flexible timer block allows the device to easily perform all timer functions with minimal software overhead. The timer block has three operating modes: Processor Independent PWM mode, External Event Counter mode, and Input Capture mode.

The control bits T1C3, T1C2, and T1C1 allow selection of the different modes of operation.

Mode 1. Processor Independent PWM Mode

As the name suggests, this mode allows the device to generate a PWM signal with very minimal user intervention.

The user only has to define the parameters of the PWM signai (ON time and OFF time). Once begun, the timer block will continuously generate the PWM signal completely independent of the microcontroller. The user software services the timer block only when the PWM parameters require updating.

In this mode the timer T1 counts down at a fixed rate of t_C. Upon every underflow the timer is alternately reloaded with the contents of supporting registers, R1A and R1B. The very first underflow of the timer causes the timer to reload from the register R1A. Subsequent underflows cause the timer to be reloaded from the registers alternately beginning with the register R1B.

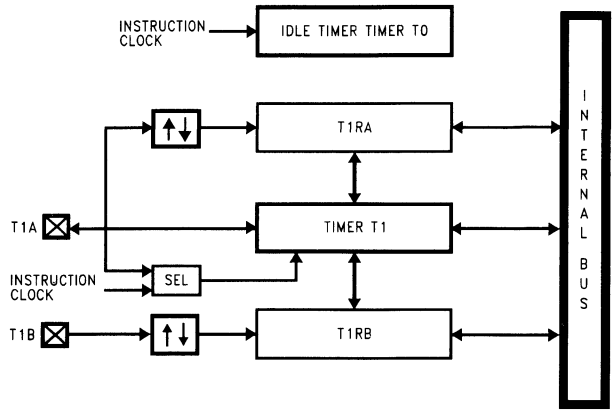
The T1 Timer control bits, T1C3, T1C2 and T1C1 set up the timer for PWM mode operation.

Figure 9 shows a block diagram of the timer in PWM mode. The underflows can be programmed to toggle the T1A output pin. The underflows can also be programmed to generate interrupts.

Underflows from the timer are alternately latched into two pending flags, T1PNDA and T1PNDB. The user must reset these pending flags under software control. Two control enable flags, T1ENA and T1ENB, allow the interrupts from the timer underflow to be enabled or disabled. Setting the timer enable flag T1ENA will cause an interrupt when a timer underflow causes the R1A register to be reloaded into the timer. Setting the timer enable flag T1ENB will cause an interrupt when a timer underflow causes the R1B register to be reloaded into the timer. Resetting the timer enable flags will disable the associated interrupts.

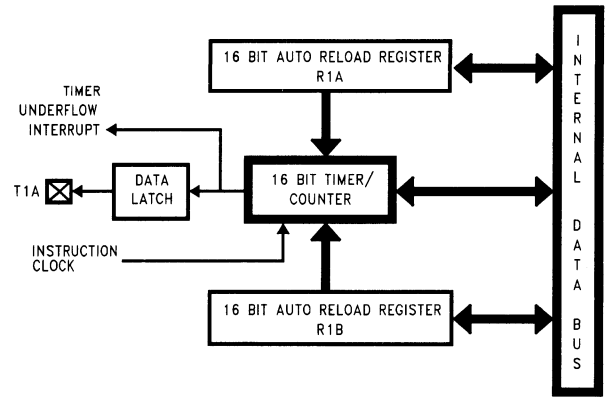
Either or both of the timer underflow interrupts may be enabled. This gives the user the flexibility of interrupting once per PWM period on either the rising or falling edge of the PWM output. Alternatively, the user may choose to interrupt on both edges of the PWM output.

Timers (Continued)



TL/DD/12067–8

FIGURE 8. Timers T1 and T0



TL/DD/12067–9

FIGURE 9. Timer 1 in PWM MODE

Mode 2. External Event Counter Mode

This mode is quite similar to the processor independent PWM mode described above. The main difference is that the timer, T1, is clocked by the input signal from the T1A pin. The T1 timer control bits, T1C3, T1C2 and T1C1 allow the timer to be clocked either on a positive or negative edge from the T1A pin. Underflows from the timer are latched into the T1PNDA pending flag. Setting the T1ENA control flag will cause an interrupt when the timer underflows.

In this mode the input pin T1B can be used as an independent positive edge sensitive interrupt input if the T1ENB control flag is set. The occurrence of a positive edge on the T1B input pin is latched into the T1PNDB flag.

Figure 10 shows a block diagram of the timer in External Event Counter mode.

Note: The PWM output is not available in this mode since the T1A pin is being used as the counter input clock.

Mode 3. Input Capture Mode

The device can precisely measure external frequencies or time external events by placing the timer block, T1, in the input capture mode.

In this mode, the timer T1 is constantly running at the fixed t_C rate. The two registers, R1A and R1B, act as capture registers. Each register acts in conjunction with a pin. The register R1A acts in conjunction with the T1A pin and the register R1B acts in conjunction with the T1B pin.

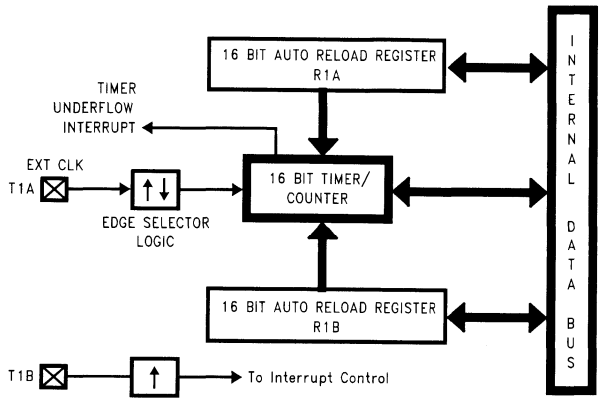
The timer value gets copied over into the register when a trigger event occurs on its corresponding pin. Control bits, T1C3, T1C2 and T1C1, allow the trigger events to be specified either as a positive or a negative edge. The trigger condition for each input pin can be specified independently.

Timers (Continued)

The trigger conditions can also be programmed to generate interrupts. The occurrence of the specified trigger condition on the T1A and T1B pins will be respectively latched into the pending flags, T1PNDA and T1PNDB. The control flag T1ENA allows the interrupt on T1A to be either enabled or disabled. Setting the T1ENA flag enables interrupts to be generated when the selected trigger condition occurs on the T1A pin. Similarly, the flag T1ENB controls the interrupts from the T1B pin.

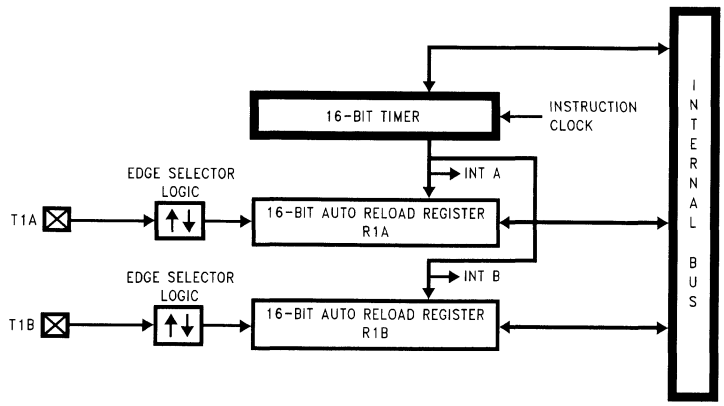
Underflows from the timer can also be programmed to generate interrupts. Underflows are latched into the timer T1C0 pending flag (the T1C0 control bit serves as the timer underflow interrupt pending flag in the Input Capture mode). Consequently, the T1C0 control bit should be reset when entering the Input Capture mode. The timer underflow interrupt is enabled with the T1ENA control flag. When a T1A interrupt occurs in the Input Capture mode, the user must check both the T1PNDA and T1C0 pending flags in order to determine whether a T1A input capture or a timer underflow (or both) caused the interrupt.

Figure 11 shows a block diagram of the timer in Input Capture mode.



TL/DD/12067-10

FIGURE 10. Timer 1 in External Event Counter Mode



TL/DD/12067-11

FIGURE 11. Timer 1 in Input Capture Mode

5

Timers (Continued)

TIMER CONTROL FLAGS

The control bits and their functions are summarized below.

T1C0 Timer Start/Stop control in Modes 1 and 2 (Processor Independent PWM and External Event Counter), where 1 = Start, 0 = Stop
Timer Underflow Interrupt Pending Flag in Mode 3 (Input Capture)
T1PNDA Timer Interrupt Pending Flag
T1PNDB Timer Interrupt Pending Flag
T1ENA Timer Interrupt Enable Flag
T1ENB Timer Interrupt Enable Flag
1 = Timer Interrupt Enabled
0 = Timer Interrupt Disabled
T1C3 Timer mode control
T1C2 Timer mode control
T1C1 Timer mode control

The timer mode control bits (T1C3, T1C2 and T1C1) are detailed below:

T1C3	T1C2	T1C1	Timer Mode	Interrupt A Source	Interrupt B Source	Timer Counts On
0	0	0	MODE 2 (External Event Counter)	Timer Underflow	Positive T1B Edge	T1A Positive Edge
0	0	1	MODE 2 (External Event Counter)	Timer Underflow	Positive T1B Edge	T1A Negative Edge
1	0	1	MODE 1 (PWM) T1A Toggle	Autoreload RA	Autoreload RB	t_C
1	0	0	MODE 1 (PWM) No T1A Toggle	Autoreload RA	Autoreload RB	t_C
0	1	0	MODE 3 (Capture) Captures: T1A Positive Edge T1B Positive Edge	Positive T1A Edge or Timer Underflow	Positive T1B Edge	t_C
1	1	0	MODE 3 (Capture) Captures: T1A Positive Edge T1B Negative Edge	Positive T1A Edge or Timer Underflow	Negative T1B Edge	t_C
0	1	1	MODE 3 (Capture) Captures: T1A Negative Edge T1B Positive Edge	Negative T1A Edge or Timer Underflow	Positive T1B Edge	t_C
1	1	1	MODE 3 (Capture) Captures: T1A Negative Edge T1B Negative Edge	Negative T1A Edge or Timer Underflow	Negative T1B Edge	t_C

HIGH SPEED, CONSTANT RESOLUTION PWM TIMER

The device has one processor independent PWM timer. The PWM timer operates in two modes: PWM mode and capture mode. In PWM mode the timer outputs can be programmed to two pins PWM0 and PWM1. In capture mode, pin PWM0 functions as the capture input. *Figure 12* shows a block diagram for this timer in capture mode and *Figure 13* shows a block diagram for the timer in PWM mode.

PWM Timer Registers

The PWM Timer has three registers: PWMCON, the PWM control register, RLON, the PWM on-time register and PSCAL, the prescaler register.

PWM Prescaler Register (PSCAL) (Address X'00A0)

The prescaler is the clock source for the counter in both PWM mode and in frequency monitor mode.

PSCAL is a read/write register that can be used to program the prescaler. The clock source to the timer in both PWM and capture modes can be programmed to CKI/N where N = PSCAL + 1, so the maximum PWM clock frequency = CKI and the minimum PWM clock frequency = CKI/256. The processor is able to modify the PSCAL register regardless of whether the counter is running or not and the change in frequency occurs with the next underflow of the prescaler (CK-PWM).

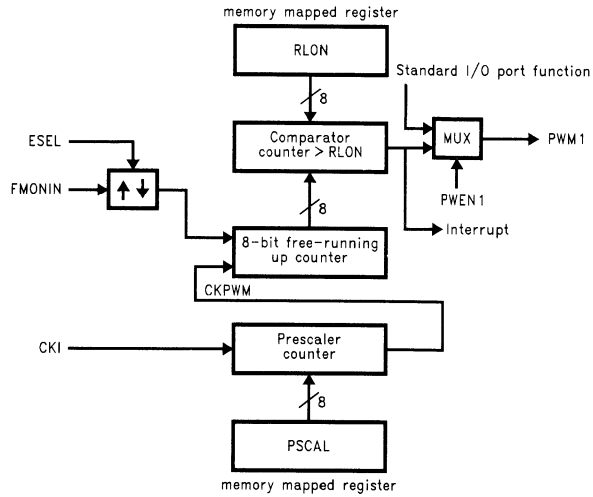
PWM On-time Register (RLON) (Address X'00A1)

RLON is a read/write register. In PWM mode the timer output will be a "1" for RLON counts out of a total cycle of 255 PWM clocks. In capture mode it is used to program the threshold frequency.

The PWM timer is specially designed to have a resolution of 255 PWM clocks. This allows the duty cycle of the PWM output to be selected between 1/255 and 254/255. A value of 0 in the RLON register will result in the PWM output being continuously low and a value of 255 will result in the PWM output being continuously high.

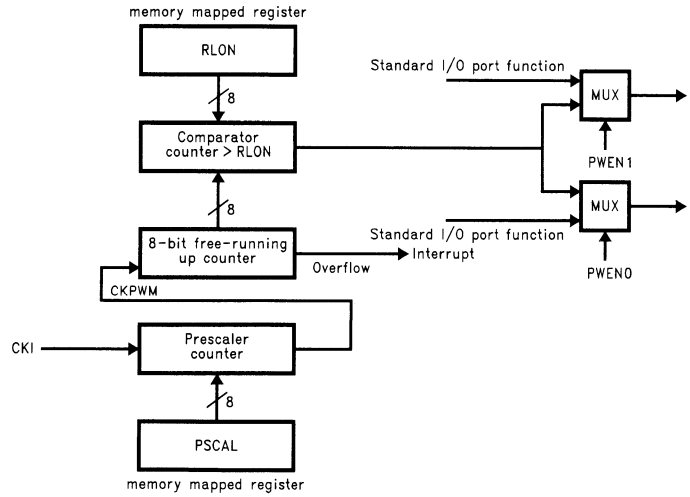
Note: The effect of changing the RLON register during active PWM mode operation is delayed until the boundary of a PWM cycle. In capture mode the effect takes place immediately.

Timers (Continued)



TL/DD/12067–12

FIGURE 12. PWM Timer Capture Mode Block Diagram



TL/DD/12067–13

FIGURE 13. PWM Timer PWM Mode Block Diagram

5

Timers (Continued)

PWM Control Register (PWMCON) (Address X'00A2)

The PWMCON Register Bits are:

PWEN0 Enable PWM0 output/input function on I/O port.

PWEN1 Enable PWM1 output function on I/O port.

Note: The associated bits in the configuration and data register of the I/O-port have to be setup as outputs and/or inputs in addition to setting the PWEN bits.

PWON PWM start Bit, "1" to start timer, "0" to stop timer.

PWMD PWM Mode bit, "1" for PWM mode, "0" for frequency monitor mode.

PWIE PWM interrupt enable bit.

PWPND PWM interrupt pending bit.

ESEL Edge select bit, "1" for falling edge, "0" for rising edge.

unused	ESEL	PWPND	PWIE	PWMD	PWON	PWEN1	PWEN0

Bit 7 Bit 0

PWM Mode

The PWM timer can generate PWM signals at frequencies up to 39 kHz (@ t_C = 1 μs) with a resolution of 255 parts. Lower PWM frequencies can be programmed via the prescaler.

If the PWM mode bit (PWMD) in the PWM configuration register (PWMCON) is set to "1" the timer operates in PWM mode. In this mode, the timer generates a PWM signal with a fixed, non-programmable repetition rate of 255 PWM clock cycles. The timer is clocked by the output of an 8-bit, programmable prescaler, which is clocked with the chip's CKI frequency. Thus the PWM signal frequency can be calculated with the formula:

$$fpwm = \frac{CKI}{(1 + (PSCAL\text{-}contents)) \times 255}$$

Selecting the PWM mode by setting PWMD to "1", but not yet starting the timer (PWON is "0"), will set the timer output to "1".

The contents of an 8-bit register, RLON, multiplied by the clock cycle of the prescaler output defines the time between overflow (or starting) and the falling edge of the PWM output.

Once the timer is started, the timer output goes low after RLON cycles and high after a total of 255 cycles. The procedure is continually repeated. In PWM mode the timer is available at pins PWM0 and/or PWM1, provided the port configuration bits for those pins are defined as outputs and the PWEN0 and/or PWEN1 bits in the PWMCON register are set.

The PWM timer is started by the software setting the PWON bit to "1". Starting the timer initializes the timer register. From this point, the timer will continually generate the PWM signal, independent of any processor activity, until the timer is stopped by software setting the PWON bit to "0". The processor is able to modify the RLON register regardless of whether the timer is running. If RLON is changed while the timer is running, the previous value of RLON is used for comparison until the next overflow occurs, when the new value of RLON is latched into the comparator inputs.

When the timer overflows, the PWM pending flag (PWPND) is set to "1". If the PWM interrupt enable bit (PWIE) is also set to "1", timer overflow will generate an interrupt. The PWPND bit remains set until the user's software writes a "0" to it. If the software writes a "1" to the PWPND bit, this has no effect. If the software writes a "0" to the PWPND bit at the same time as the hardware writes to the bit, the hardware has precedence.

Note: The software controlling the duty cycle is able to change the PWM duty cycle without having to wait for the timer overflow.

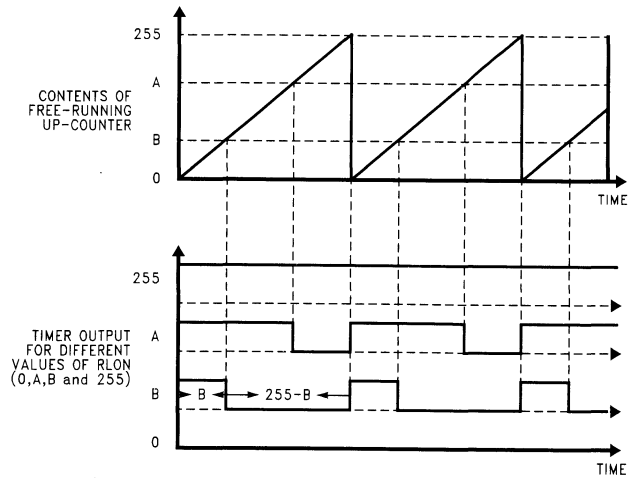
Figure 14 shows how the PWM output is implemented. The PWM Timer output is set to "1" on an overflow of the timer and set to "0" when the timer is greater than RLON. The output can be multiplexed to two pins.

Capture Mode

If the PWM mode bit (PWMD) is set to "0" the PWM Timer operates in capture mode. Capture mode allows the programmer to test whether the frequency of an external source exceeds a certain threshold.

If PWMD is "0" and PWON is "0", the timer output is set to "0". In capture mode the timer output is available at pin PWM1, provided the port configuration register bit for that pin is set up as an output and the PWEN1 bit in the PWMCON register is set. Setting PWON to "1" will initialize the timer register and start the counter. A rising edge, or if selected, a falling edge, on the FMONIN input pin will initialize the timer register and clear the timer output. The counter continues to count up after being initialized. The ESEL bit determines whether the active edge is a rising or a falling edge.

Timers (Continued)



TL/DD/12067-14

FIGURE 14. PWM Mode Operation

If, in capture mode PWM0 is configured incorrectly as an output and is enabled via the PWEN0 bit, the timer output will feedback into the PWM block as the timer input.

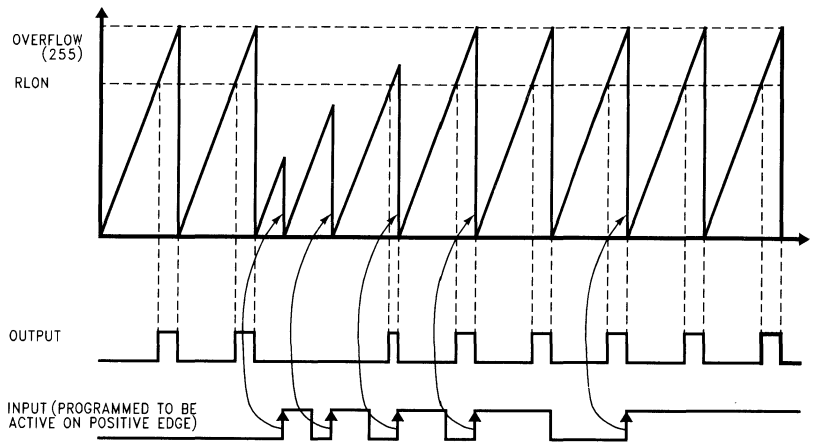
The contents of the counter are continually compared with the RLON register. If the frequency of the input edges is sufficiently high, the contents of the counter will always be less than the value in RLON. However, if the frequency of the input edges is too low, the free-running counter value will count up beyond the value in RLON.

When the counter is greater than RLON, the PWM timer output is set to "1". It is set to "0" by a detected edge on the timer input or when the counter overflows. When the counter becomes greater than RLON, the PWPND bit in the PWM control register is set to "1". If the PWIE bit is also set to "1", the PWPND bit is enabled to request an interrupt.

It should be noted that two other conditions could also set the PWPND bit:

1. If the mode of operation is changed on the fly the timer output will toggle. If frequency monitor mode is entered on the fly such that the timer output changes from 0 to 1, PWPND will be set.
2. If the timer is operating in frequency monitor mode and the RLON value is changed on the fly so that RLON becomes less than the current timer value, PWPND will be set.

The PWPND bit remains set until the user's software writes a "0" to it. If the software writes a "1" to the PWPND bit, this has no effect. If the software writes a "0" to the PWPND bit at the same time as the hardware writes to the bit, the hardware has precedence. (See *Figure 15* for Frequency Monitor Mode Operation.)



TL/DD/12067-15

FIGURE 15. Frequency Monitor Mode Operation

Power Save Modes

The device offers the user two power save modes of operation: HALT and IDLE. In the HALT mode, all microcontroller activities are stopped. In the IDLE mode, the on-board oscillator circuitry and timer T0 are active but all other microcontroller activities are stopped. In either mode, all on-board RAM, registers, I/O states, and timers (with the exception of T0) are unaltered.

HALT MODE

The contents of all PWM Timer registers are frozen during HALT mode and are left unchanged when exiting HALT mode. The PWM timer resumes its previous mode of operation when exiting HALT mode.

The device is placed in the HALT mode by writing a "1" to the HALT flag (G7 data bit). All microcontroller activities, including the clock, and timers, are stopped. In the HALT mode, the power requirements of the device are minimal and the applied voltage (V_{CC}) may be decreased to V_r (V_r = 2.0V) without altering the state of the machine.

The device supports two different ways of exiting the HALT mode. The first method of exiting the HALT mode is with the Multi-Input Wake Up feature on the L port. The second method of exiting the HALT mode is by pulling the RESET pin low.

Since a crystal or ceramic resonator may be selected as the oscillator, the Wake Up signal is not allowed to start the chip running immediately since crystal oscillators and ceramic resonators have a delayed start up time to reach full amplitude and frequency stability. The IDLE timer is used to generate a fixed delay to ensure that the oscillator has indeed stabilized before allowing instruction execution. In this case, upon detecting a valid Wake Up signal, only the oscillator circuitry is enabled. The IDLE timer is loaded with a value of 256 and is clocked with the t_c instruction cycle clock. The t_c clock is derived by dividing the oscillator clock down by a factor of 10. The Schmitt trigger following the CKI inverter on the chip ensures that the IDLE timer is clocked only when the oscillator has a sufficiently large amplitude to meet the Schmitt trigger specifications. This Schmitt trigger is not part of the oscillator closed loop. The start-up time-out from the IDLE timer enables the clock signals to be routed to the rest of the chip.

The device has two mask options associated with the HALT mode. The first mask option enables the HALT mode feature, while the second mask option disables the HALT mode. With the HALT mode enable mask option, the device will enter and exit the HALT mode as described above. With the HALT disable mask option, the device cannot be placed in the HALT mode (writing a "1" to the HALT flag will have no effect).

IDLE MODE

The device is placed in the IDLE mode by writing a "1" to the IDLE flag (G6 data bit). In this mode, all activities, except the associated on-board oscillator circuitry, and the IDLE Timer T0, are stopped. The power supply requirements of the microcontroller in this mode of operation are typically around 30% of normal power requirement of the microcontroller.

As with the HALT mode, the device can be returned to normal operation with a reset, or with a Multi-Input Wake Up from the L Port or CAN Interface. Alternately, the microcontroller resumes normal operation from the IDLE mode when the thirteenth bit (representing 4.096 ms at internal clock frequency of 1 MHz, t_c = 1 μs) of the IDLE Timer toggles.

This toggle condition of the thirteenth bit of the IDLE Timer T0 is latched into the T0PND pending flag.

The user has the option of being interrupted with a transition on the thirteenth bit of the IDLE Timer T0. The interrupt can be enabled or disabled via the T0EN control bit. Setting the T0EN flag enables the interrupt and vice versa.

The user can enter the IDLE mode with the Timer T0 interrupt enabled. In this case, when the T0PND bit gets set, the device will first execute the Timer T0 interrupt service routine and then return to the instruction following the "Enter Idle Mode" instruction.

Alternatively, the user can enter the IDLE mode with the IDLE Timer T0 interrupt disabled. In this case, the device will resume normal operation with the instruction immediately following the "Enter IDLE Mode" instruction.

Note: It is necessary to program two NOP instructions following both the set HALT mode and set IDLE mode instructions. These NOP instructions are necessary to allow clock resynchronization following the HALT or IDLE modes.

Multi-Input Wake Up

The Multi-Input Wake Up feature is used to return (wake up) the device from either the HALT or IDLE modes. Alternately, the Multi-Input Wake Up/Interrupt feature may also be used to generate up to 7 edge selectable external interrupts.

Figure 16 shows the Multi-Input Wake Up logic for the microcontroller. The Multi-Input Wake Up feature utilizes the L Port. The user selects which particular L port bit (or combination of L Port bits) will cause the device to exit the HALT or IDLE modes. The selection is done through the Reg: WKEN. The Reg: WKEN is an 8-bit read/write register, which contains a control bit for every L port bit. Setting a particular WKEN bit enables a Wake Up from the associated port pin.

The user can select whether the trigger condition on the selected L Port pin is going to be either a positive edge (low to high transition) or a negative edge (high to low transition). This selection is made via the Reg: WKEDG, which is an 8-bit control register with a bit assigned to each L Port pin. Setting the control bit will select the trigger condition to be a negative edge on that particular L Port pin. Resetting the bit selects the trigger condition to be a positive edge. Changing an edge select entails several steps in order to avoid a pseudo Wake Up condition as a result of the edge change. First, the associated WKEN bit should be reset, followed by the edge select change in WKEDG. Next, the associated WKPND bit should be cleared, followed by the associated WKEN bit being re-enabled.

An example may serve to clarify this procedure. Suppose we wish to change the edge select from positive (low going high) to negative (high going low) for L Port bit 5, where bit 5 has previously been enabled for an input interrupt. The program would be as follows:

```
RBIT  5, WKEN
SBIT  5, WKEDG
RBIT  5, WKPND
SBIT  5, WKEN
```

Multi-Input Wake Up (Continued)

If the L port bits have been used as outputs and then changed to inputs with Multi-Input Wake Up/Interrupt, a safety procedure should also be followed to avoid inherited pseudo wake up conditions. After the selected L port bits have been changed from output to input but before the associated WKEN bits are enabled, the associated edge select bits in WKEDG should be set or reset for the desired edge selects, followed by the associated WKPND bits being cleared.

This same procedure should be used following reset, since the L port inputs are left floating as a result of reset. The occurrence of the selected trigger condition for Multi-Input Wake Up is latched into a pending register called WKPND. The respective bits of the WKPND register will be set on the occurrence of the selected trigger edge on the corresponding Port L pin. The user has the responsibility of clearing these pending flags. Since WKPND is a pending register for the occurrence of selected wake up conditions, the device will not enter the HALT mode if any Wake Up bit is both enabled and pending. Consequently, the user has the responsibility of clearing the pending flags before attempting to enter the HALT mode.

The WKEN, WKPND and WKEDG are all read/write registers, and are cleared at reset.

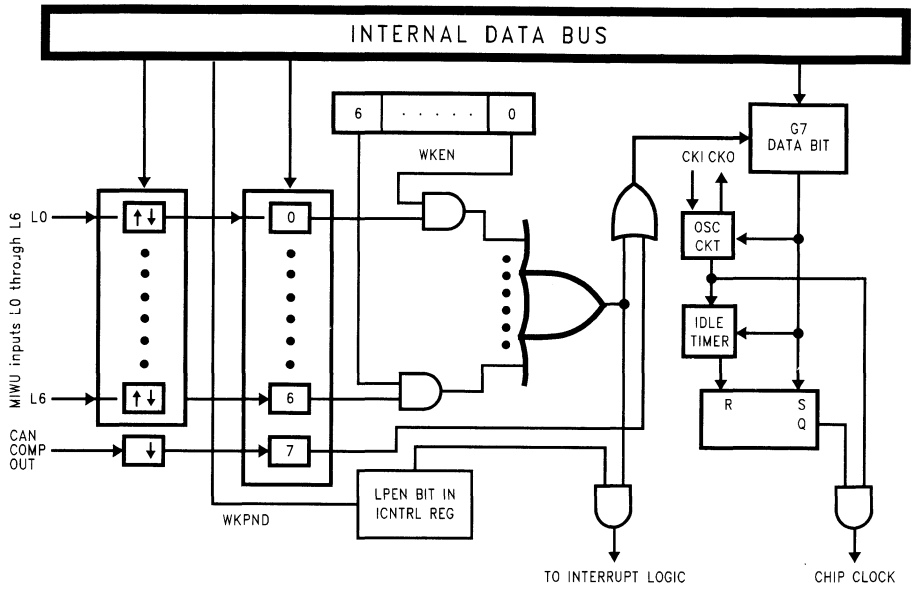


FIGURE 16. Multi-Input Wake Up Logic

Multi-Input Wake Up (Continued)

CAN RECEIVE WAKE UP

The CAN Receive Wake Up source is always enabled and is always active on a falling edge of the CAN comparator output. There is no specific enable bit for the CAN Wake Up feature. Although the wake up feature on pins L0..L6 can be programmed to generate an interrupt (L-port interrupt), no interrupt is generated upon a CAN receive wake up condition. The CAN block has its own, dedicated receiver interrupt upon receive buffer full.

PORT L INTERRUPTS

Port L provides the user with an additional seven fully selectable, edge sensitive interrupts which are all vectored into the same service subroutine.

The interrupt from Port L shares logic with the wake up circuitry. The register WKEN allows interrupts from Port L to be individually enabled or disabled. The register WKEDG specifies the trigger condition to be either a positive or a negative edge. Finally, the register WKPND latches in the pending trigger conditions.

The GIE (global interrupt enable) bit enables the interrupt function. A control flag, LPEN, functions as a global interrupt enable for Port L interrupts. Setting the LPEN flag will enable interrupts and vice versa. A separate global pending flag is not needed since the register WKPND is adequate.

Since Port L is also used for waking the device out of the HALT or IDLE modes, the user can elect to exit the HALT or IDLE modes either with or without the interrupt enabled. If he elects to disable the interrupt, then the device will restart execution from the instruction immediately following the instruction that placed the microcontroller in the HALT or IDLE modes. In the other case, the device will first execute the interrupt service routine and then revert to normal operation.

The Wake Up signal will not start the chip running immediately since crystal oscillators or ceramic resonators have a finite start up time. The IDLE Timer (T0) generates a fixed delay to ensure that the oscillator has indeed stabilized before allowing the device to execute instructions. In this case, upon detecting a valid Wake Up signal, only the oscillator circuitry and the IDLE Timer T0 are enabled. The IDLE Timer is loaded with a value of 256 and is clocked from the t_C instruction cycle clock. The t_C clock is derived by dividing down the oscillator clock by a factor of 10. A Schmitt trigger following the CKI on-chip inverter ensures that the IDLE timer is clocked only when the oscillator has a sufficiently large amplitude to meet the Schmitt trigger specifications. This Schmitt trigger is not part of the oscillator closed loop. The start-up time-out from the IDLE timer enables the clock signals to be routed to the rest of the chip.

Interrupts

The device supports a vectored interrupt scheme. It supports a total of eleven interrupt sources. The following table lists all the possible device interrupt sources, their arbitration ranking and the memory locations reserved for the interrupt vector for each source.

Two bytes of program memory space are reserved for each interrupt source. All interrupt sources except the software interrupt are maskable. Each of the maskable interrupts have an Enable bit and a Pending bit. A maskable interrupt is active if its associated enable and pending bits are set. If GIE = 1 and an interrupt is active, then the processor will be interrupted as soon as it is ready to start executing an instruction except if the above conditions happen during the Software Trap service routine. This exception is described in the Software Trap sub-section.

The interruption process is accomplished with the INTR instruction (opcode 00), which is jammed inside the Instruction Register and replaces the opcode about to be executed. The following steps are performed for every interrupt:

1. The GIE (Global Interrupt Enable) bit is reset.
2. The address of the instruction about to be executed is pushed into the stack.
3. The PC (Program Counter) branches to address 00FF. This procedure takes 7 t_C cycles to execute.

At this time, since GIE = 0, other maskable interrupts are disabled. The user is now free to do whatever context switching is required by saving the context of the machine in the stack with PUSH instructions. The user would then program a VIS (Vector Interrupt Select) instruction in order to branch to the interrupt service routine of the highest priority interrupt enabled and pending at the time of the VIS. Note that this is not necessarily the interrupt that caused the branch to address location 00FF Hex prior to the context switching.

Thus, if an interrupt with a higher rank than the one which caused the interruption becomes active before the decision of which interrupt to service is made by the VIS, then the interrupt with the higher rank will override any lower ones and will be acknowledged. The lower priority interrupt(s) are still pending, however, and will cause another interrupt immediately following the completion of the interrupt service routine associated with the higher priority interrupt just serviced. This lower priority interrupt will occur immediately following the RETI (Return from Interrupt) instruction at the end of the interrupt service routine just completed.

Inside the interrupt service routine, the associated pending bit has to be cleared by software. The RETI (Return from Interrupt) instruction at the end of the interrupt service routine will set the GIE (Global Interrupt Enable) bit, allowing the processor to be interrupted again ff another interrupt is active and pending.

The VIS instruction looks at all the active interrupts at the time it is executed and performs an indirect jump to the beginning of the service routine of the one with the highest rank.

The addresses of the different interrupt service routines, called vectors, are chosen by the user and stored in ROM in a table starting at 01E0 (assuming that VIS is located between 00FF and 01DF). The vectors are 15 bit wide and therefore occupy 2 ROM locations.

VIS and the vector table must be located in the same 256-byte block (0y00 to 0yFF) except if VIS is located at the last address of a block. In this case, the table must be in the next block. The vector table cannot be inserted in the first 256-byte block.

Interrupts (Continued)

The vector of the maskable interrupt with the lowest rank is located at 0yE0 (Hi-Order byte) and 0yE1 (Lo-Order byte) and so forth in increasing rank number. The vector of the maskable interrupt with the highest rank is located at 0yFA (Hi-Order byte) and 0yFB (Lo-Order byte).

The Software Trap has the highest rank and its vector is located at 0yFE and 0yFF.

Arbitration Ranking	Source	Vector Address Hi-Low Byte
1	Software Trap	0yFE–0yFF
2	Reserved	0yFC–0yFD
3	CAN Receive	0yFA–0yFB
4	CAN Error (transmit/receive)	0yF8–0yF9
5	CAN Transmit	0yF6–0yF7
6	Pin G0 Edge	0yF4–0yF5
7	IDLE Timer Underflow	0yF2–0yF3
8	Timer T1A/Underflow	0yF0–0yF1
9	Timer T1B	0yEE–0yEF
10	MICROWIRE/PLUS	0yEC–0yED
11	PWM timer	0YEA–0yEB
12	Reserved	0yE8–0yE9
13	Reserved	0yE6–0yE7
14	Reserved	0yE4–0yE5
15	Port L/Wake Up	0yE2–0yE3
16	Default VIS Interrupt	0yE0–0yE1

y is VIS page, y ≠ 0

If, by accident, a VIS gets executed and no interrupt is active, then the PC (Program Counter) will branch to a vector located at 0yE0-0yE1.

WARNING

A Default VIS interrupt handler routine must be present. As a minimum, this handler should confirm that the GIE bit is cleared (this indicates that the interrupt sequence has been taken), take care of any required housekeeping, restore context and return. Some sort of Warm Restart procedure should be implemented. These events can occur without any error on the part of the system designer or programmer.

Note: There is always the possibility of an interrupt occurring during an instruction which is attempting to reset the GIE bit or any other interrupt enable bit. If this occurs when a single cycle instruction is being used to reset the interrupt enable bit, the interrupt enable bit will be reset but an interrupt may still occur. This is because interrupt processing is started at the same time as the interrupt bit is being reset. To avoid this scenario, the user should always use a two-, three-, or four-cycle instruction to reset interrupt enable bits.

Figure 17 shows the Interrupt Block diagram.

SOFTWARE TRAP

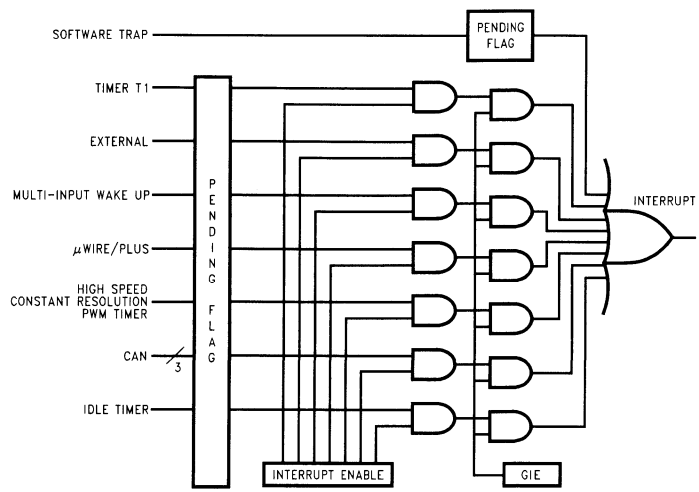
The Software Trap (ST) is a special kind of non-maskable interrupt which occurs when the INTR instruction (used to acknowledge interrupts) is fetched from ROM and placed inside the instruction register. This may happen when the PC is pointing beyond the available ROM address space or when the stack is over-popped.

When an ST occurs, the user can re-initialize the stack pointer and do a recovery procedure (similar to RESET, but not necessarily containing all of the same initialization procedures) before restarting.

The occurrence of an ST is latched into the ST pending bit. The GIE bit is not affected and the ST pending bit (**not accessible by the user**) is used to inhibit other interrupts and to direct the program to the ST service routine with the VIS instruction. The RPND instruction is used to clear the software interrupt pending bit. This bit is also cleared on reset.

The ST has the highest rank among all interrupts.

Nothing (except another ST) can interrupt an ST being serviced.



TL/DD/12067–17

FIGURE 17. Interrupt Block Diagram

CAN Block Description *

This device contains a CAN serial bus interface as described in the CAN Specification Rev. 2.0 part B.

*Patents Pending.

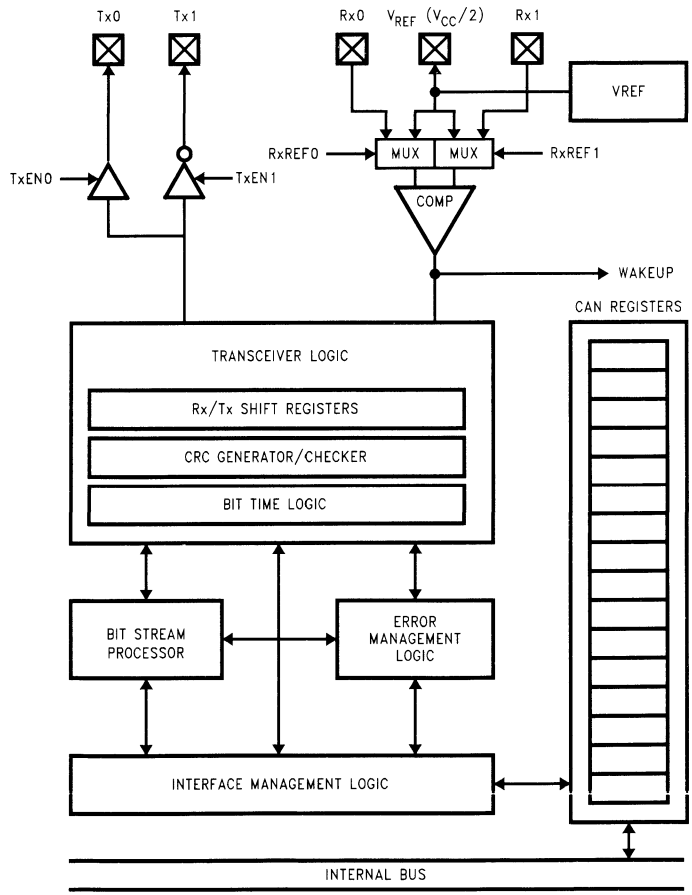
CAN Interface Block

This device supports applications which require a low speed CAN interface. It is designed to be programmed with two transmit and two receive registers. The user's program may check the status bytes in order to get information of the bus state and the received or transmitted messages. The device has the capability to generate an interrupt as soon as one byte has been transmitted or received. Care must be taken if more than two bytes in a message frame are to be transmitted/received. In this case the user's program must poll the transmit buffer empty (TBE)/receive buffer full (RBF) bits or enable their respective interrupts and perform a data exchange between the user data and the Tx/Rx registers.

Fully automatic transmission on error is supported for messages not longer than two bytes. Messages which are longer than two bytes have to be processed by software.

The interface is compatible with CAN Specification 2.0 part B, without the capability to receive/transmit extended frames. Extended frames on the bus are checked and acknowledged according to the CAN specification.

The maximum bus speed achievable with the CAN interface is a function of crystal frequency, message length and software overhead. The device can support a bus speed of up to 1 Mbit/s with a 10 MHz oscillator and 2 byte messages. The 1 Mbit/s bus speed refers to the rate at which protocol and data bits are transferred on the bus. Longer messages require slower bus speeds due to the time required for software intervention between data bytes. The device will support a maximum of 125k bit/s with eight byte messages and a 10 MHz oscillator.



TL/DD/12067–49

FIGURE 15. CAN Interface Block Diagram

Functional Block Description of the CAN Interface

Interface Management Logic (IML)

The IML executes the CPU's transmission and reception commands and controls the data transfer between CPU, Rx/Tx and CAN registers. It provides the CAN Interface with Rx/Tx data from the memory mapped Register Block. It also sets and resets the CAN status information and generates interrupts to the CPU.

Bit Stream Processor (BSP)

The BSP is a sequencer controlling the data stream between The Interface Management Logic (parallel data) and the bus line (serial data). It controls the transceive logic with regard to reception and arbitration, and creates error signals according to the bus specification

Transceive Logic (TCL)

The TCL is a state machine which incorporates the bit stuff logic and controls the output drivers, CRC logic and the Rx/Tx shift registers. It also controls the synchronization to the bus with the CAN clock signal generated by the BTL.

Error Management Logic (EML)

The EML is responsible for the fault confinement of the CAN protocol. It is also responsible for changing the error counters, setting the appropriate error flag bits and interrupts and changing the error status (passive, active and bus off).

Cyclic Redundancy Check (CRC) Generator and Register

The CRC Generator consists of a 15-bit shift register and the logic required to generate the checksum of the destuffed bit-stream. It informs the EML about the result of a receiver checksum.

The checksum is generated by the polynomial:

$$X^{15} + X^{14} + X^{10} + X^{8} + X^{7} + X^{4} + X^{3} + 1$$

Receive/Transmit (Rx/Tx) Registers

The Rx/Tx registers are 8-bit shift registers controlled by the TCL and the BSP. They are loaded or read by the Interface Management Logic, which holds the data to be transmitted or the data that was received.

Bit Time Logic (BTL)

The bit time logic divider divides the CKI input clock by the value defined in the CAN prescaler (CSCAL) and bus timing register (CTIM). The resultig bit time (tcan) can be computed by the formula:

$$t_{can} = \frac{CKI}{(1 + divider) \times (1 + 2 \times PS + PPS)}$$

Where *divider* is the value of the clock prescaler, *PS* is the programmable value of phase segment 1 and 2 (1..8) and *PPS* the programmed value of the propagation segment (1..8) (located in CTIM).

Bus Timing Considerations

The internal architecture of the CAN interface has been optimized to allow fast software response times within messages of more than two data bytes. The TBE (Transmit Buffer Empty) bit is set on the last bit of odd data bytes when CAN internal sample points are high.

It is the user's responsibility to ensure that the time between setting TBE and a reload of TxD2 is longer than the length of phase segment 2 as indicated in the following equation:

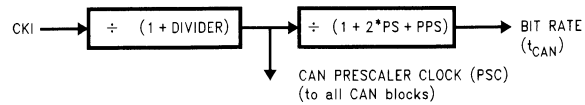
$$t_{LOAD} > \frac{(PS + 1) \times (CSCAL + 1)}{10} t_C = \text{absolute length of PS2}$$

Table V shows examples of the minimum required t_{LOAD} for different CSCAL settings based on a clock frequency of 10 MHz. Lower clock speeds require recalculation of the CAN bit rate and the mimimum t_{LOAD}.

TABLE V. CAN Timing (CKI = 10 MHz t_c = 1 μs)

PS	CSCAL	CAN Bit Rate (kbit/s)	Minimum t_{LOAD} (μs)
4	3	250	2.0
4	9	100	5.0
4	15	62	8.0
4	24	40	12.5
4	39	25	20
4	99	10	50
4	199	5	100

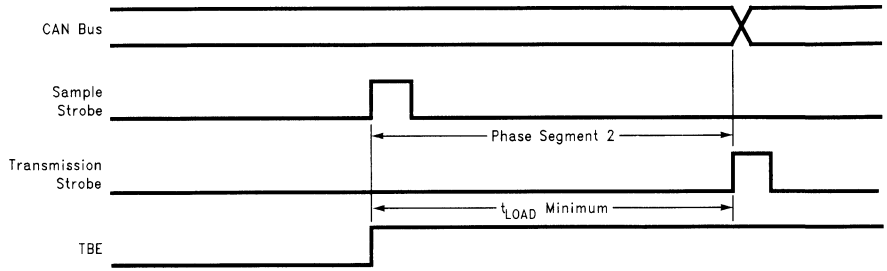
Functional Block Description of the CAN Interface (Continued)



TL/DD/12067-50

FIGURE 16. Bit Rate Generation

Figure 17 illustrates the minimum time required for t_{LOAD}.



TL/DD/12067-51

FIGURE 17. TBE Timing

In the case of an interrupt driven CAN interface, the calculation of the actual t_{LOAD} time would be done as follows:

```
INT:                                    ;Interrupt latency = 7tc = 7 µs
                PUSH        A           ;3tc = 3 µs
                LD          A,B         ;2tc = 2 µs
                PUSH        A           ;3tc = 3 µs
                VIS                     ;5tc = 5 µs
CANTX:                                  ;20tc = µs to this point
                •                       ;additional time for instructions which check
                •                       ;status prior to reloading the transmit data
                •                       ;registers with subsequent data bytes.
                LD          TXD2,DATA
                •
                •
                •
```

Functional Block Description of the CAN Interface (Continued)

Interrupt driven programs use more time than programs which poll the TBE flag, however programs which operate at lower baud rates (which are more likely to be sensitive to this issue) have more time for interrupt response.

Output Drivers/Input Comparators

The output drivers/input comparators are the physical interface to the bus. Control bits are provided to TRI-STATE the output drivers.

A dominant bit on the bus is represented as a "0" in the data registers and a recessive bit on the bus is represented as a "1" in the data registers.

TABLE VI. Bus Level Definition

Bus Level	Pin Tx0	Pin Tx1	Data
"dominant"	drive low (GND)	dirve high (V_{CC})	0
"recessive"	TRI-STATE	TRI-STATE	1

Register Block

The register block consists of fifteen 8-bit registers which are described in more detail in the following paragraphs.

Note: The contents of the receiver related registers RxD1, RxD2, RDLC, RIDH and RTSTAT are only changed if a received frame passes the acceptance filter or the Receive Identifier Acceptance Filter bit (RIAF) is set to accept all received messages.

TRANSMIT DATA REGISTER 1 (TXD1) (Address X'00B0)

The Transmit Data Register 1 contians the first data byte to be transmitted within a frame and then the successive odd byte numbers (i.e., bytes number 1,3,..,7).

TRANSMIT DATA REGISTER 2 (TXD2) (Address X'00B1)

The Transit Data Register 2 contains the second data byte to be transmitted within a frame and then the successive even byte numbers (i.e., bytes number 2,4,..,8).

TRANSMIT DATA LENGTH CODE AND IDENTIFIER LOW REGISTER (TDLC) (Address X'00B2)

TID3	TID2	TID1	TID0	TDLC3	TDLC2	TCLC1	TDLC0
Bit 7							Bit 0

This register is read/write.

TID3..TIDO Transmit Identifier Bits 3..0 (lower 4 bits)

The transmit identifier is composed of eleven bits in total, bits 3 to 0 of the TID are stored in bits 7 to 4 of this register.

TDLC3..TDLC0 Transmit Data Length Code

These bits determine the number of data bytes to be transmitted within a frame. The CAN specification allows a maximum of eight data bytes in any message.

TRANSMIT IDENTIFIER HIGH (TID) (Address X'00B3)

TRTR	TID10	TID9	TID8	TID7	TID6	TID5	TID4
Bit 7							Bit 0

This register is read/write.

TRTR Transmit Remote Frame Request

This bit is set if the frame to be transmitted is a remote frame request.

TID10..TID4 Transmit Identifier Bits 10 .. 4 (higher 7 bits)

Bits TID10..TID4 are the upper 7 bits of the 11 bit transmit identifier.

RECEIVER DATA REGISTER 1 (RXD1) (Address X'00A4)

The Receive Data Register 1 (RXD1) contains the first data byte received in a frame and then successive odd byte numbers (i.e., bytes 1, 3,..7). This register is read-only.

RECEIVE DATA REGISTER 2 (RXD2) (Address X'00A5)

The Receive Data Register 2 (RXD2) contains the second data byte received in a frame and then successive even byte numbers (i.e., bytes 2,4,..,8). This register is read-only.

REGISTER DATA LENGTH CODE AND IDENTIFIER LOW REGISTER (RIDL) (Address X'00B6)

RID3	RID2	RID1	RID0	RDLC3	RDLC2	RDLC1	RDLC0
Bit 7							Bit 0

This register is read only.

RID3..RID0 Receive Identifier bits (lower four bits)

The RID3..RID0 bits are the lower four bits of the eleven bit long Receive Identifier. Any received message that matches the upper 7 bits of the Receive Identifier (RID10..RID4) is accepted if the Receive Identifier Acceptance Filter (RIAF) bit is set to zero.

RDLC3..RDLC0 Receive Data Length Code bits

The RDLC3..RDLC0 bits determine the number of data bytes within a received frame.

RECEIVE IDENTIFIER HIGH (RID) (Address X'00B7)

unused	RID10	RID9	RID8	RID7	RID6	RID5	RID4
Bit 7							Bit 0

This register is read/write.

RID10..RID4 Receive Identifier bits (upper bits)

The RID10...RID4 bits are the upper 7 bits of the eleven bit long Receive Identifier. If the Receive Identifier Acceptance Filter (RIAF) bit (see CBUS register) is set to zero, bits 4 to 10 of the received identifier are compared with the mask bits of RID4..RID10. If the corresponding bits match, the message is accepted. If the RIAF bit is set to a one, the filter function is disabled and all messages, independent of identifier, will be accepted.

Functional Block Description of the CAN Interface (Continued)

CAN PRESCALER REGISTER (CSCAL) (Address X'00B8)

CKS7	CKS6	CKS5	CKS4	CKS3	CKS2	CKS1	CKS0
Bit 7							Bit 0

This register is read/write.

CKS7..0 Prescaler divider select.

The resulting clock value is the CAN Prescaler clock.

CAN BUS TIMING REGISTER (CTIM) (00B9)

PPS2	PPS1	PPS0	PPS0	PS2	PS1	PS0	Reserved
Bit 7							Bit 0

This register is read/write.

PPS2..PPS0 Propagation Segment, bits 2..0

The PPS2..PPS0 bits determine the length of the propagation delay in Prescaler clock cycles (PSC) per bit time. (For a more detailed discussion of propagation delay and phase segments, see SYNCHRONIZATION on page 41.)

PS2..PS0 Phase Segment 1, bits 2..0

The PS2..PS0 bits fix the number of Prescaler clock cycles per bit time for phase segment 1 and phase segment 2. The PS2..PS0 bits also set the synchronization Jump Width to a value equal to the lesser of the 4 PSC or the length of PS1/2 (Min: 4 I length of PS1/2).

TABLE VII. Synchronization Jump Width

PS2	PS1	PS0	Length of Phase Segment ½	Synchronization Jump Width
0	0	0	1 t_{can}	1 t_{can}
0	0	1	2 t_{can}	2 t_{can}
0	1	0	3 t_{can}	3 t_{can}
0	1	1	4 t_{can}	4 t_{can}
1	0	0	5 t_{can}	4 t_{can}
1	0	1	6 t_{can}	4 t_{can}
1	1	0	7 t_{can}	4 t_{can}
1	1	1	8 t_{can}	4 t_{can}

LENGTH OF TIME SEGMENTS (See *Figure 29*)

- The Synchronization Segment is 1 CAN Prescaler clock (PSC)
- The Propagation Segment can be programmed (PPS) to be 1,2...,8 PSC in length.
- Phase Segment 1 and Phase Segment 2 are programmable (PS) to be 1,2,..,8 PSC long.

Note: (BTL settings at high speed; PSC = 0) Due to the on-chip delay from the rx-pins through the receive comparator (worst case assumption: 3 clocks delay * 2 (devices on the bus) + 1 tx delay) the user needs to set the sample point to > (2*3 + 1) i.e., > 7 CKI clocks to ensure correct communication on the bus under all circumstances. With prescaler settings of > 0 this is a given (i.e., no caution has to be applied).

Example: for 1 Mbit CTIM = b'10000100 (PSS = 5; PS1 = 2). Example for 500 kbit CTIM = b'01011100 (PPS = 3; PS1 = 8). – all at 10 MHz CKI and CSCAL = 0.

CAN BUS CONTROL REGISTER (CBUS) (00BA)

Reserved	RIAF	TxEN1	TxEN0	RxREF1	RxREF0	Reserved	FMOD
Bit 7							Bit 0

Reserved This bit is reserved and should be zero.

RIAF Receive identifier acceptance filter bit

If the RIAF bit is set to zero, bits 4 to 10 of the received identifier are compared with the mask bits of RID4..RID10 and if the corresponding bits match, the message is accepted. If the RIAF bit is set to a one, the filter function is disabled and all messages independent of the identifier will be accepted.

TxEN0, TxEN1 TxD Output Driver Enable

TABLE VIII. Output Drivers

TxEN1	TxEN0	Output
0	0	Tx0, Tx1 TRI-STATE, CAN input comparator disabled
0	1	Tx0 enabled
1	0	Tx1 enabled
1	1	Tx0 and Tx1 enabled

Bus synchronization of the device is done in the following way:

If the output was disabled (TxEN1, TxEN0 = "0") and either TxEN1 or TxEN0, or both are set to 1, the device will not start transmission or reception of a frame until eleven consecutive "recessive" bits have been received. Resetting the TxEN1 and TxEN0 bits will disable the output drivers and the CAN input comparator. All other CAN related registers and flags will be unaffected. It is recommended that the user reset the TxEN1 and TxEN0 bits before switching the device into the HALT mode (the CAN receive wakeup will still work) in order to reduce current consumption and to assure a proper resychronization to the bus after exiting the HALT mode.

Note: A "bus off" condition will also cause Tx0 and Tx1 to be at TRI-STATE (independent of the values of the TxEN1 and TxEN0 bits).

RXREF1 Reference voltage applied to Rx1 if bit is set

RXREF0 Reference voltage applied to Rx0 if bit is set

FMOD Fault Confinement Mode select

Setting the FMOD bit to "0" (default after power on reset) will select the Standard Fault Confinement mode. In this mode the device goes from "bus off" to "error active" after monitoring 128*11 recessive bits (including bus idle) on the bus. This mode has been implemented for compatibility with existing solutions. Setting the FMOD bit to "1" will select the Enhanced Fault Confinement mode. In this mode the device goes from "bus off" to "error active" after monitoring 128 "good" messages, as indicated by the reception of 11 consecutive "recessive" bits including the End of Frame, whereas the standard mode may time out after 128 x 11 recessive bits (e.g., bus idle).

Functional Block Description of the CAN Interface (Continued)

TRANSMIT CONTROL/STATUS (TCNTL) (00BB)

NS1	NS0	TERR	RERR	CEIE	TIE	RIE	TXSS
Bit 7							Bit 0

NS1..NS0 Node Status, i.e., Error Status.

TABLE IX. Node Status

NS1	NS0	Output
0	0	Error active
0	1	Error passive
1	0	Bus off
1	1	Bus off

The Node Status bits are read only.

TERR Transmit Error

This bit is automatically set when an error occurs during the transmission of a frame. TERR can be programmed to generate an interrupt by setting the Can Error Interrupt Enable bit (CEIE). This bit must be cleared by the user's software.

Note: This is used for messages for more than two bytes. If an error occurs during the transmission of a frame with more than 2 data bytes, the user's software has to handle the correct reloading of the data bytes to the TxD registers for retransmission of the frame. For frames with 2 or less data bytes the interface logic of this chip does an automatic retransmission. Regardless of the number of data bytes, the user's software must reset this bit if CEIE is enabled. Otherwise a new interrupt will be generated immediately after return from the interrupt service routine.

RERR Receiver Error

This bit is automatically set when an error occurred during the reception of a frame. RERR can be programmed to generate an interrupt by setting the Can Error Interrupt Enable bit (CEIE). This bit has to be cleared by the user's software.

CEIE CAN Error Interrupt Enable

If set by the user's software, this bit enables the tansmit and receive error interrupts. The interrupt pending flags are TERR and RERR. Resetting this bit with a pending error interrupt will inhibit the interrupt, but will not clear the cause of the interrupt (RERR or TERR). If the bit is then set without clearing the cause of the interrupt, the interrupt will reoccur.

TIE Transmit Interrupt Enable

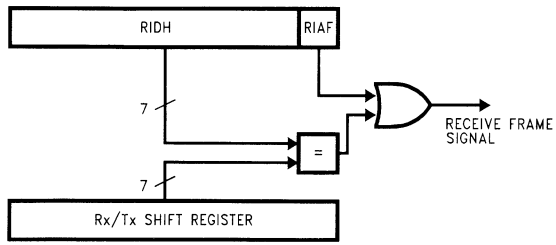
If set by the user's software, this bit enables the transmit interrupt. (See TBE and TXPND.) Resetting this bit with a pending transmit interrupt will inhibit the interrupt, but will not clear the cause of the interrupt. If the bit is then set without clearing the cause of the interrupt, the interrupt will reoccur.

RIE Receive Interrupt Enable

If set by the user's software, this bit enables the receive interrupt or a remote transmission request interrupt (see RBF, RFV and RRTR). Resetting this bit with a pending receive interrupt will inhibit the interrupt, but will not clear the cause of the interrupt. If the bit is then set without clearing the cause of the interrupt, the interrupt will reoccur.

TXSS Transmission Start/Stop

This bit is set by the user's software to initiate the transmission of a frame. Once this bit is set, a transmission is pending, as indicated by the TXPND flag being set. It can be reset by software to cancel a pending transmission. Resetting the TXSS bit will only cancel a transmission, if the transmission of a frame hasn't been started yet (bus idle), if arbitration has been lost (receiving) or if an error occurs during transmission. If the device has already started transmission (won arbitration) the TXPND and TXSS flags will stay set until the transmission is completed, even if the user's software has written zero to the TXSS bit. If one or more data bytes are to be transmitted, care must be taken by the user, that the Transmit Data Register(s) have been loaded before the TXSS bit is set. TXSS will be cleared on three conditions only: Successful completion of a transmitted message; successful cancellation of a pending transmision; Transition of the CAN interface to the bus-off state.



TL/DD/12067–52

FIGURE 18. Acceptance Filter Block-Diagram

Functional Block Description of the CAN Interface (Continued)

Writing a zero to the TXSS bit will request cancellation of a pending transmission but TXSS will not be cleared until completion of the operation. If an error occurs during transmission of a frame, the logic will check for cancellation requests prior to restarting transmission. If zero has been written to TXSS, retransmission will be canceled.

RECEIVE/TRANSMIT STATUS (RTSTAT) (Address X'00BC)

TBE	TXPND	RRTR	ROLD	RORN	RFV	RCV	RBF
1	0	0	0	0	0	0	0

Bit 7 Bit 0

This register is read only.

TBE Transmit Buffer Empty

This bit is set as soon as the TxD2 register is copied into the Rx/Tx shift register, i.e., the 1st data byte of each pair has been transmitted. The TBE bit is automatically reset if the TxD2 register is written (the user should write a dummy byte to the TxD2 register when transmitting an odd number of bytes of zero bytes). TBE can be programmed to generate an interrupt by setting the Transmit Interrupt Enable bit (TIE). When servicing the interrupt the user has to make sure that TBE gets cleared by executing a WRITE instruction on the TxD2 register, otherwise a new interrupt will be generated immediately after return from the interrupt service routine. The TBE bit is read only. It is set to 1 upon reset. TBE is also set upon completion of transmission of a valid message.

TXPND Transmission Pending

This bit is set as soon as the Transmit Start/Stop (TXSS) bit is set by the user. It will stay set until the frame was successfully transmitted, until the transmission was successfully canceled by writing zero to the Transmission Start/Stop bit (TXSS), or the device enters the bus-off state. Resetting the TXSS bit will only cancel a transmission if the transmission of a frame hasn't been started yet (bus idle) or if arbitration has been lost (receiving). If the device has already started transmission (won arbitration) the TXPND flag will stay set until the transmission is completed, even if the user's software has requested cancellation of the message. If an error occurs during transmission, a requested cancellation may occur prior to the begining of retransmission.

RRTR Received Remote Transmission Request

This bit is set when the remote transmission request (RTR) bit in a received frame was set. It is automatically reset through a read of the RXD1 register.

To detect RRTR the user can either poll this flag or enable the receive interrupt (the reception of a remote transmission request will also cause an interrupt if the receive interrupt is enabled). If the receive interrupt is enabled, the user should check the RRTR flag in the service routine in order to distinguish between a RRTR interrupt and a RBF interrupt. It is the responsibility of the user to clear this bit by reading the RXD1 register, before the next frame is received.

ROLD Received Overload Frame

This bit is automatically set when an Overload Frame was received on the bus. It is automatically reset through a read of the Receive/Transmit Status register. It is the responsibility of the user to clear this bit by reading the Receive/Transmit Status register, before the next frame is received.

RORN Receiver Overrun

This bit is automatically set on an overrun of the receive data register, i.e., if the user's program does not maintain the RxDn registers when receiving a frame. It it automatically reset through a read of the Receive/Transmit Status register. It is the responsibility of the user to clear this bit by reading the Receive/Transmit Status register before the next frame is received.

RFV Received Frame Valid

This bit is set if the received frame is valid, i.e., after the penultimate bit of the End of Frame is received. It is automatically reset through a read of the Receive/Transmit Status register. It is the responsibility of the user to clear this bit by reading the receive/transmit status register (RTSTAT), before the next frame is received. RFV will cause a Receive Interrupt if enabled by RIE. The user should be careful to read the last data byte (RxD1) of odd length messages (1, 3, 5 or 7 data bytes) on receipt of RFV. RFV is the only indication that the last byte of the message has been received.

RCV Receive Mode

This bit is set after the data length code of a message that passes the device's acceptance filter has been received. It is automatically reset after the CRC-delimiter of the same frame has been received. It indicates to the user's software that arbitration is lost and that data is coming in for that node.

RBF Receive Buffer Full

This bit is set if the second Rx data byte was received. It is reset automatically, after the RxD1-Register has been read by the software. RBF can be programmed to generate an interrupt by setting the Receive Interrupt Enable bit (RIE). When servicing the interrupt, the user has to make sure that RBF gets cleared by executing a LD instruction from the RxD1 register, otherwise a new interrupt will be generated immediately after return from the interrupt service routine. The RBF bit is read only.

TRANSMIT ERROR COUNTER (TEC) (Address X'00BD)

TEC7	TEC6	TEC5	TEC4	TEC3	TEC2	TEC1	TEC0

Bit 7 Bit 0

This register is read/write.

Functional Block Description of the CAN Interface (Continued)

For test purposes and to identify the node status, the transmit error counter, an 8-bit error counter, is mapped into the data memory. If the lower seven bits of the counter overflow, i.e., TEC7 is set, the device is error passive.

CAUTION

To prevent interference with the CAN fault confinement, the user must not write to the REC/TEC registers. Both counters are automatically updated following the CAN specification.

RECEIVE ERROR COUNTER (REC) (00BE)

ROVL	REC6	REC5	REC4	REC3	REC2	REC1	REC0
Bit 7							Bit 0

This register is read/write.

ROVL receive error counter overflow

For test purposes and to identify the node status the receive error counter, a 7-bit error counter, is mapped into the data memory. If the counter overflows the ROVL bit is set to indicate that the device is error passive and won't transmit any active error frames. If ROVL is set then the counter is frozen.

MESSAGE IDENTIFICATION

a. Transmitted Message

The user can select all 11 Transmit Identifier Bits to transmit any message whigh fulfills the CAN2.0, part B spec without an extended identifier (see note below). Fully automatic retransmission is supported for messages no longer than 2 bytes.

b. Received Messages

The lower four bits of the Receive Identifier are don't care, i.e., the controller will receive all messages that fit in that window (16 messages). The upper 7 bits can be defined by the user in the Receive Identifier High Register to mask out groups of messages. If the RIAF bit is set, all messages will be received.

Note: The CAN interface tolerates the extended CAN frame format of 29 identifier bits and gives an acknowledgment. If an error occurs the receive error counter will be increased, and decreased if the frame is valid.

BUS SYNCHRONIZATION DURING OPERATION

Resetting the TxEN1 and TxEN0 bits in Bus Control Register will disable the output drivers and do a resynchronization to the bus. All other CAN related registers and flags will be unaffected.

Bus synchronization of the device is this case is done in the following way:

If the output was disabled (TxEN1, TxEN0 = "0") and either TxEN1 or TxEN0, or both are set to 1, the device will not start transmission or reception of a frame until eleven consecutive "recessive" bits have been received.

A "bus off" condition will also cause the output drivers Tx1 and Tx0 to be at TRI-STATE (independent of the status of TxEN1 and TxEN0). The device will switch from "bus off" to "error active" mode as described under the FMOD-bit description (see Can Bus Control register). This will ensure that the device is synchronized to the bus, before starting to transmit or receive.

For information on bus synchronization and status of the CAN related registers after external reset refer to the RESET section.

ON-CHIP VOLTAGE REFERENCE

The on-chip voltage reference is a ratiometric reference. For electrical characteristics of the voltage reference refer to the electrical specifications section.

ANALOG SWITCHES

Analog switches are used for selecting between Rx0 and V_{REF} and between Rx1 and V_{REF}.

Basic CAN Concepts

The following paragraphs provide a generic overview of the basic concepts of the Controller Area Network (CAN) as described in *Chapter 4 of ISO/DIS11519-1.* Implementation related issues of the National Semiconductor device will be discussed as well.

This device will process standard frame format only. Extended frame formats will be acknowledged, however the data will be discarded. For this reason the description of frame formats in the following section will cover only the standard frame format.

The following section provides some more detail on how the device will handle received extended frames:

If the device's remote identifier acceptance filter bit (RIAF) is set to "1", extended frame messages will be acknowledged. However, the data will be discarded and the device will not reply to a remote transmission request received in extended frame format. If the device's RIAF bit is set to "0", the upper 7 received ID bits of an extended frame that match the device's receive identifier (RID) acceptance filtler bits, are stroed in the device's RID register. However, the device does not reply to an RTR and any data is discarded. The device will only acknowledge the message.

MULTI-MASTER PRIORITY BASED BUS ACCESS

The CAN protocol is message based protocol that allows a total of 2032 ($= 2^{11} - 16$) different messages in the standard format and 512 million ($= 2^{29} - 16$) different messages in the extended frame format.

MULTICAST FRAME TRANSFER BY ACCEPTANCE FILTERING

Every CAN Frame is put on the common bus. Each module receives every frame and filters out the frames which are not required for the module's task.

REMOTE DATA REQUEST

A CAN master module has the ability to set a specific bit called the "remote transmission request bit" (RTR) in a frame. This causes another module, either another master or a slave, to transmit a data frame after the current frame has been completed.

Basic CAN Concepts (Continued)

SYSTEM FLEXIBILITY

Additional modules can be added to an existing network without a configuration change. These modules can either perform completely new functions requiring new data or process existing data to perform a new function.

SYSTEM WIDE DATA CONSISTENCY

As the CAN network is message oriented, a message can be used like a variable which is automatically updated by the controlling processor. If any module cannot process information it can send an overload frame. The device is incapable of initiating an overload frame, but will join a overload frame initiated by another device as required by CAN specifications.

NON-DESTRUCTIVE CONTENTION-BASED ARBITRATION

The CAN protocol allows several transmitting modules to start a transmission at the same time as soon as they monitor the bus to be idle. During the start of transmission every node monitors the bus line to detect whether its message is overwritten by a message with a higher priority. As soon as a transmitting module detects another module with a higher priority accessing the bus, it stops transmitting its own frame and switches to receive mode. For illustration see *Figure 19*.

AUTOMATIC RETRANSMISSION OF FRAMES

If a data or remote frame is overwritten by either a higher-prioritized data frame, remote frame or an error frame, the transmitting module will automatically retransmit it. This device will handle the automatic retransmission of up to two data bytes automatically. Messages with more than 2 data bytes require the user's software to update the transmit registers.

ERROR DETECTION AND ERROR SIGNALING

All messages on the bus are checked by each CAN node and acknowledge if they are correct. If any node detects an error it starts the transmission of an error frame.

Switching Off Defective Nodes

There are two error counters, one for transmitted data and one for received data, which are incremented, depending on the error type, as soon as an error occurs. If either counter goes beyond a specific value the node goes to an error state. A valid frame causes the error counters to decrease.

The device can be in one of three states with respect to error handling:

- Error active
  An error active unit can participate in bus communication and sends an active ("dominant") error flag.
- Error passive
  An error passive unit can participate in bus communication. However, if the unit detects an error it is not allowed to send an active error flag. The unit sends only a passive ("recessive") error flag.
- Bus off

A unit that is "bus off" has the output drivers disabled, i.e., it does not participate in any bus activity.

(See ERROR MANAGEMENT AND DETECTION for more detailed information.)

Frame Formats

INTRODUCTION

There are basically two different types of frames used in the CAN protocol.

The data transmission frames are: data/remote frame

The control frames are: error/overload frame

Note: This device cannot send an overload frame as a result of not being able to process all information. However, the device is able to recognize an overload condition and join overload frames initiated by other devices.

If no message is being transmitted, i.e., the bus is idle, the bus is kept at the "recessive" level. *Figure 20* and *Figure 21* give an overview of the various CAN frame formats.

DATA AND REMOTE FRAME

Data frames consist of seven bit fields and remote frames consist of six different bit fields:

1. Start of Frame (SOF)
2. Arbitration field
3. Control field (IDE bit, R0 bit, and DLC field)
4. Data field (not in remote frame)
5. CRC field
6. ACK field
7. End of Frame (EOF)

A remote frame has no data field and is used for requesting data from other (remote) CAN nodes. *Figure 22* shows the format of a CAN data frame.

FRAME CODING

Remote and Data Frames are NRZ codes with bit-stuffing in every bit field which holds computable information for the interface, i.e., Start of Frame arbitration field, control field, data field (if present) and CRC field.

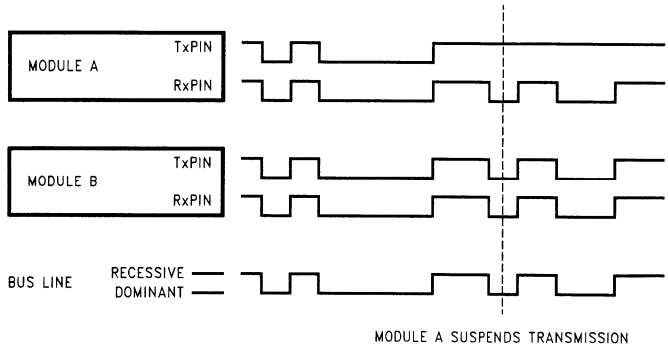
Error and overload frames are NRZ coded without bit stuffing.

BIT STUFFING

After five consecutive bits of the same value, a stuff bit of the inverted value is inserted by the transmitter and deleted by the receiver.

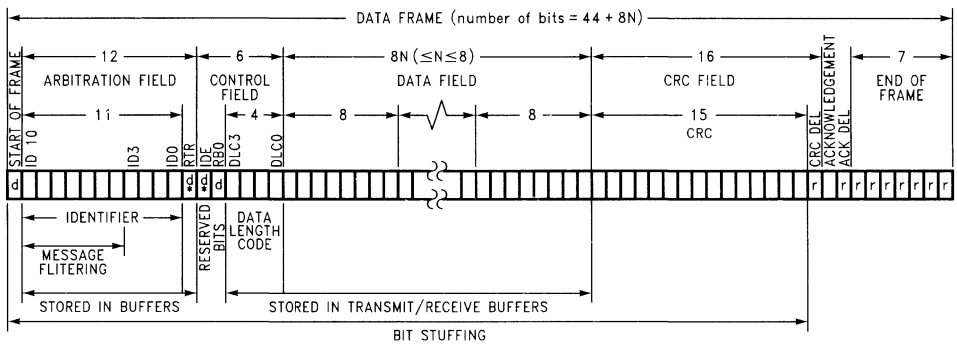
Destuffed Bit Stream	100000x	011111x
Stuffed Bit Stream	1000001x	0111110x
		x = {0,1}

Basic CAN Concepts (Continued)

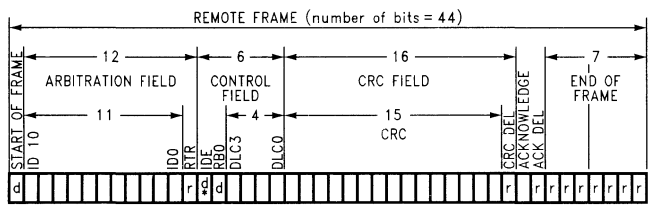


TL/DD/12067–53

FIGURE 19. CAN Message Arbitration



TL/DD/12067–54



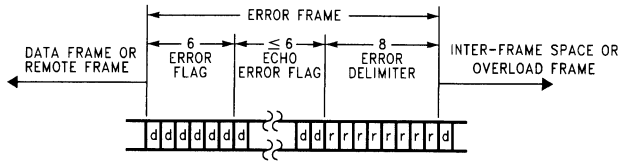
TL/DD/12067–55

A remote frame is identical to a data frame, except that the RTR bit is "recessive", and there is no data field.

IDE = Identifier Extension Bit
The IDE bit in the standard format is transmitted "dominant", whereas in the extended format the IDE bit is "recessive" and the id is expanded to 29 bits.
r = recessive
d = dominant

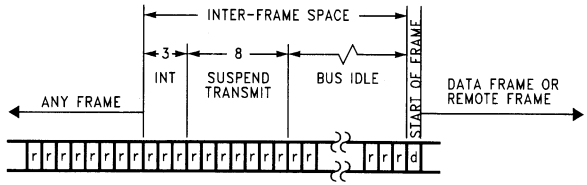
FIGURE 20. CAN Data Transmission Frames

Basic CAN Concepts (Continued)



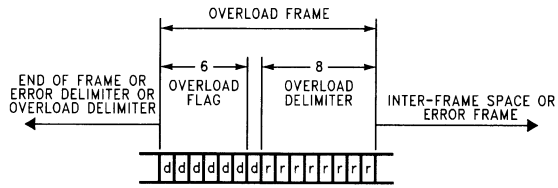
TL/DD/12067–56

An error frame can start anywhere in the middle of a frame.



TL/DD/12067–57

INT = Intermission
Suspend Transmission is only for error passive nodes.

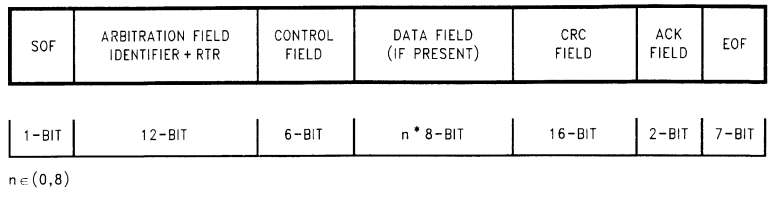


TL/DD/12067–58

An overload frame can only start at the end of a frame.

FIGURE 21. CAN Control Frames

Basic CAN Concepts (Continued)



TL/DD/12067–59

FIGURE 22. CAN Frame Format

START OF FRAME (SOF)

The Start of Frame indicates the beginning of data and remote frames. It consists of a single "dominant" bit. A node is only allowed to start transmission when the bus is idle. All nodes have to synchronize to the leading edge (first edge after the bus was idle) caused by SOF of the node which starts transmission first.

ARBITRATION FIELD

The arbitration field is composed of the identifier field and the RTR (Remote Transmission Request) bit. The value of the RTR bit is "dominant" in a data frame and "recessive" in a remote frame.

CONTROL FIELD

The control field consists of six bits. It starts with two bits reserved for future expansion followed by the four-bit Data Length Code. Receivers must accept all possible combinations of the two reserved bits. Until the function of these reserved bits is defined, the transmitter only sends "0" (dominant) bits. The first reserved bit (IDE) is actually defined to indicate an extended frame with 29 Identifier bits if set to "1". CAN chips must tolerate extended frames, even if they can only understand standard frames, to prevent the destruction of an extended frames on an existing network.

The Data Length Code indicates the number of bytes in the data field. This Data Length Code consists of four bits. The data field can be of length zero. The permissible number of data bytes for a data frame ranges from 0 to 8.

DATA FIELD

The Data field consists of the data to be transferred within a data frame. It can contain 0 to 8 bytes and each byte contains 8 bits. A remote frame has no data field.

CRC FIELD

The CRC field consists of the CRC sequence followed by the CRC delimiter. The CRC sequence is derived by the transmitter from the modulo 2 division of the preceding bit fields, starting with the SOF up to the end of the data field, excluding stuff-bits, by the generator polynomial:

$$\chi^{15} + \chi^{14} + \chi^{10} + \chi^{8} + \chi^{7} + \chi^{4} + \chi^{3} + 1$$

The remainder of this division is the CRC sequence transmitted over the bus. On the receiver side the module divides all bit fields up to the CRC delimiter, excluding stuff-bits, and checks if the result is zero. This will then be interpreted as a valid CRC. After the CRC sequence a single "recessive" bit is transmitted as the CRC delimiter.

ACK FIELD

The ACK field is two bits long and contains the ACK slot and the ACK delimiter. The ACK slot is filled with a "recessive" bit by the transmitter. This bit is overwritten with a "dominant" bit by every receiver that has received a correct CRC sequence. The second bit of the ACK field is a "recessive" bit called the acknowledge delimiter. As a consequence the acknowledge flag of a valid frame is surrounded by two "recessive" bits, the CRC-delimiter and the ACK delimiter.

EOF FIELD

The End of Frame Field closes a data and a remote frame. It consists of seven "recessive" bits.

INTERFRAME SPACE

Data and remote frames are separate from every preceding frame (data, remote, error and overload frames) by the interframe space see *Figure 23* and *Figure 24* for details. Error and overload frames are not preceded by an interframe space. They can be transmitted as soon as the condition occurs. The interframe space consists of a minimum of three bit fields depending on the error state of the node.

These bit fields are coded as follows:

The intermission has the fixed form of three "recessive" bits. While this bit field is active, no node is allowed to start a transmission of a data or a remote frame. The only action to be taken is signaling an overload condition. This means that an error in this bit field would be interpreted as an overload condition. Suspend transmission has to be inserted by error-passive nodes that were transmitter for the last message. This bit field has the form of eight "recessive" bits. However, it may be overwritten by a "dominant" start-bit from another non error passive node which starts transmission. The bus idle field consists of "recessive" bits. Its length is not specified and depends on the bus load.

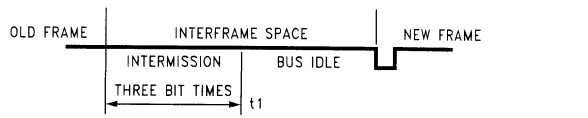
Basic CAN Concepts (Continued)

ERROR FRAME

The Error Frame consists of two bit fields: the error flag and the error delimiter. The error field is built up from the various error flags of the different nodes. Therefore, its length may vary from a minimum of six bits up to a maximum of twelve bits depending on when a module detects the error. Whenever a bit error, stuff error, form error, or acknowledgment error is detected by a node, this node starts transmission of the error flag at the next bit. If a CRC error is detected, transmission of the error flag starts at the bit following the acknowledge delimiter, unless an error flag for a previous error condition has already been started. *Figure 25* shows how a local fault at one module (module 2) leads to a 12-bit error frame on the bus.

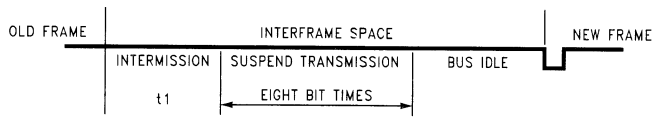
The bus level may either be "dominant" for an error-active node or "recessive" for an error-passive node. An error active node detecting an error, starts transmitting an active error flag consisting of six "dominant" bits. This causes the destruction of the actual frame on the bus. The other nodes detect the error flag as either a violation of the rule of bit-stuffing or the value of a fixed bit field is destroyed. As a consequence all other nodes start transmission of their own error flag. This means, that the error sequence which can be monitored on the bus as a maximum length of twelve bits. If an error passive node detects an error it transmits six "recessive" bits on the bus. This sequence does not destroy a message sent by another node and is not detected by other nodes. However, if the node detecting an error was the transmitter of the frame the other modules will get an error condition by a violation of the fixed bit or stuff rule. *Figure 24* shows how an error passive transmitter transmits a passive error frame and when it is detected by the receivers.

After any module has transmitted its active or passive error flag it waits for the error delimiter which consists of eight "recessive" bits before continuing.



TL/DD/12067–60

FIGURE 23. Interframe Space for Nodes Which Are Not Error Passive or Have Been Receiver for the Last Frame



TL/DD/12067–61

FIGURE 24. Interframe Space for Nodes Which Are Error Passive and Have Been Transmitter for the Last Frame

Basic CAN Concepts (Continued)

OVERLOAD FRAME

Like an error frame, an overload frame consists of two bit fields: the overload flag and the overload delimiter. The bit fields have the same length as the error frame field: six bits for the overload flag and eight bits for the delimiter. The overload frame can only be sent after the end of frame (EOF) field and in they way destroys the fixed form of the intermission field.

ORDER OF BIT TRANSMISSION

A frame is transmitted starting with the Start of Frame, sequentially followed by the remaining bit fields. In every bit field the MSB is transmitted first.

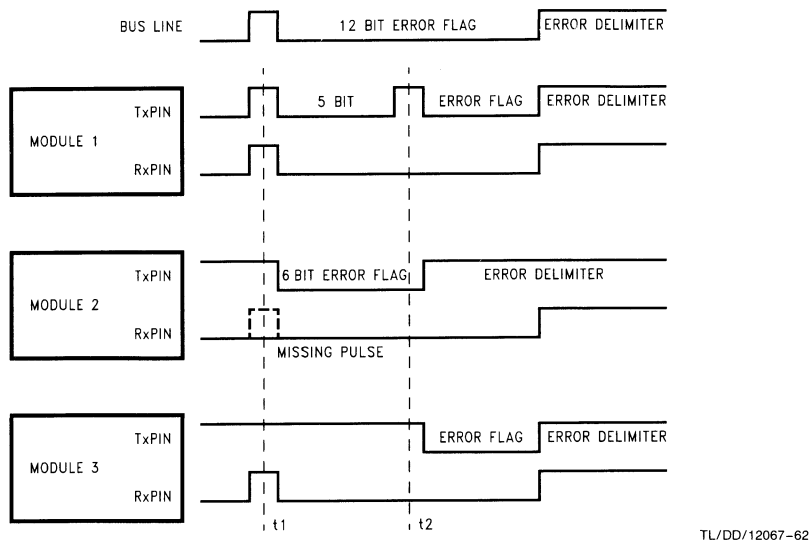
FRAME VALIDATION

Frames have a different validation point for transmitters and receivers. A frame is valid for the transmitter of a message, if there is no error until the end of the last bit of the End of Frame field. A frame is valid for a receiver, if there is no error until and including the end of the penultimate bit of the End of Frame.

FRAME ARBITRATION AND PRIORITY

Except for an error passive node which transmitted the last frame, all nodes are allowed to start transmission of a frame after the intermission, which can lead to two or more nodes starting transmission at the same time. To prevent a node from destroying another node's frame, it monitors the bus during transmission of the identifier field and the RTR-bit. As soon as it detects a "dominant" bit while transmitting a "recessive" bit it releases the bus, immediately stops transmission and starts receiving the frame. This causes no data or remote frame to be destroyed by another. Therefore the highest priority message with the identifier 0x000 out of 0x7EF (including the remote data request (RTR) bit) always gets the bus. This is only valid for standard CAN frame format. Note that while the CAN specification allows valid standard identifiers only in the range 0x000 to 0x7EF, the device will allow identifiers to 0x7FF.

There are three more items that should be taken into consideration to avoid unrecoverable collisions on the bus:



TL/DD/12067–62

module 1 = error active transmitter detects bit error at t2
module 2 = error active receiver with a local fault at t1
module 3 = error active receiver detects stuff error at t2

FIGURE 25. Error Frame—Error Active Transmitter

Basic CAN Concepts (Continued)

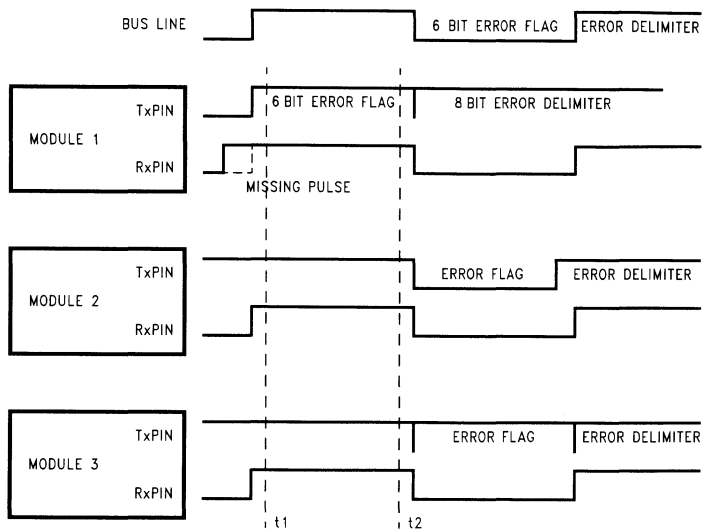
- Within one system each message must be assigned a unique identifier. This is to prevent bit errors, as one module may transmit a "dominant" data bit while the other is transmitting a "recessive" data bit. This could happen if two or more modules start transmission of a frame at the same time and all win arbitration.
- Data frames with a given identifier and a non-zero data length code may be initiated by one node only. Otherwise, in worst case, two nodes would count up to the bus-off state, due to bit errors, if they always start transmitting the same ID with different data.
- Every remote frame should have a system-wide data length code (DLC). Otherwise two modules starting transmission of a remote frame at the same time will overwrite each other's DLC which result in bit errors.

ACCEPTANCE FILTERING

Every node may perform acceptance filtering on the identifier of a data or a remote frame to filter out the messages which are not required by the node. In they way only the data of frames which match the acceptance filter is stored in the corresponding data buffers. However, every node which is not in the bus-off state and has received a correct CRC-sequence acknowledges each frame.

ERROR MANAGEMENT AND DETECTION

There are multiple mechanisms in the CAN protocol, to detect errors and to inhibit erroneous modules from disabling all bus activities.



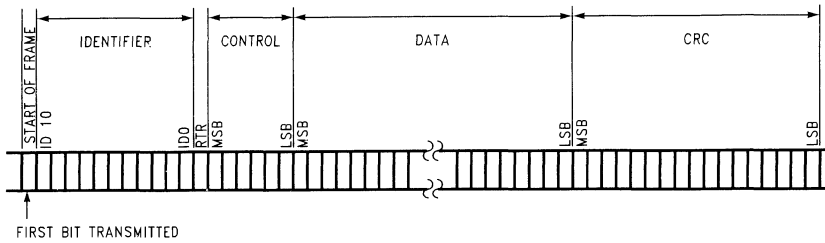
TL/DD/12067–63

module 1 = error active receiver with a local fault at t1
module 2 = error passive transmitter detects bit error at t2
module 3 = error passive receiver detects stuff error at t2

FIGURE 26. Error Frame—Error Passive Transmitter

5

Basic CAN Concepts (Continued)



TL/DD/12067–64

FIGURE 27. Order of Bit Transmission within a CAN Frame

The following errors can be detected:

- Bit Error
  A CAN device that is sending also monitors the bus. If the monitored bit value is different from the bit value that is sent, a bit error is detected. The reception of a "dominant" bit instead of a "recessive" bit during the transmission of a passive error flag, during the stuffed bit stream of the arbitration field or during the acknowledge slot, is not interpreted as a bit error.
- Stuff error
  A stuff error is detected, if the bit level after 6 consecutive bit times has not changed in a message field that has to be coded according to the bit stuffing method.
- Form Error
  A form error is detected, if a fixed frame bit (e.g., CRC delimiter, ACK delimiter) does not have the specified value. For a receiver a "dominant" bit during the last bit of End of Frame does NOT constitute a form error.
- Bit CRC Error
  A CRC error is detected if the remainder of the CRC calculation of a received CRC polynomial is non-zero.
- Acknowledgment Error
  An acknowledgment error is detected whenever a transmitting node does not get an acknowledgment from any other node (i.e., when the transmitter does not receive a "dominant" bit during the ACK frame).

The device can be in one of three states with respect to error handling:

- Error active
  An error active unit can participate in bus communication and sends an active ("dominant") error flag.
- Error passive
  An error passive unit can participate in bus communication. However, if the unit detects an error it is not allowed to send an active error flag. The unit sends only a passive ("recessive") error flag. A device is error passive when the transmit error counter is greater than 127 or when the receive error counter is greater than 127. A device becoming error passive sends an active error flag. An error passive device becomes error active again when both transmit and receive error counter are less than 128.
- Bus off
  A unit that is "bus off" has the output drivers disabled, i.e., it does not participate in any bus activity. A device is bus off when the transmit error counter is greater than 255. A bus off device will become error active again in one of two ways depending on which mode is selected by the user through the Fault Confinement Mode select bit (FMOD) in the CAN Bus Control Register (CBUS). Setting the FMOD bit to "0" (default after power on reset) will select the Standard Fault Confinement mode. In this mode the device goes from "bus off" to "error active" after monitoring 128*11 recessive bits (including bus idle) on the bus. This mode has been implemented for compatibility reasons with existing solutions. Setting the FMOD bit to "1" will select the Enhanced Fault Confinement mode. In this mode the device goes from "bus off" to "error active" after monitoring 128 "good" messages, as indicated by the reception of 11 consecutive "recessive" bits including the End of Frame. The enhanced mode offers the advantage that a "bus off" device (i.e., a device with a serious fault) is not allowed to destroy any messages on the bus until other devices can transmit at least 128 messages. This is not guaranteed in the standard mode, where a defective device could seriously impact bus communication. When the device goes from "bus off" to "error active", both error counters will have the value "0".

In each CAN module there are two error counters to perform a sophisticated error management. The receive error counter (REC) is 7 bits wide and switches the device to the error passive state if it overflows. The transmit error counter (TEC) is 8 bits wide. If it is greater than 127, the device is switched to the error passive state. As soon as the TEC overflows, the device is switched bus-off, i.e., it does not participate in any bus activity.

Basic CAN Concepts (Continued)

The counters are modified by the device's hardware according to the following rules:

TABLE X. Receive Error Counter Handling

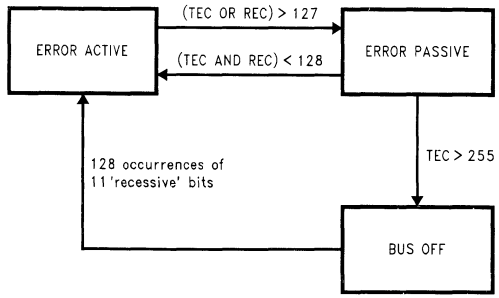
Condition	Receive Error Counter
A receiver detects a Bit Error during sending an active error flag.	Increment by 8
A receiver detects a "dominant" bit as the first bit after sending an error flag.	Increment by 8
After detecting the 14th consecutive "dominant" bit following an active error flag or overload flag or after detecting the 8th consecutive "dominant" bit following a passive error flag. After each sequence of additional 8 consecutive "dominant" bits.	Increment by 8
Any other error condition (stuff, frame, CRC, ACK).	Increment by 1
A valid reception or transmission.	Decrement by 1 if Counter is not 0

TABLE XI. Transmit Error Counter Handling

Condition	Transmit Error Counter
A transmitter detects a Bit Error during sending an active error flag.	Increment by 8
After detecting the 14th consecutive "dominant" bit following an active error flag or overload flag or after detecting the 8th consecutive "dominant" bit following a passive error flag. After each sequence of additional 8 consecutive "dominant" bits.	Increment by 8
Any other error condition (stuff, frame, CRC, ACK).	Increment by 8
A valid reception or transmission.	Decrement by 1 if Counter is not 0

Special error handling for the TEC counter is performed in the following situations:

- A stuff error occurs during arbitration, when a transmitted "recessive" stuff bit is received as a "dorminant" bit. This does not lead to an incrementation of the TEC.
- An ACK-error occurs in an error passive device and no "dominant" bits are detected while sending the passive error flag. This does not lead to an incrementation of the TEC.
- If only one device is on the bus and this device transmits a message, it will get no acknowledgment. This will be detected as an error and message will be repeated. When the device goes "error passive" and detects an acknowledge error, the TEC counter is not incremented. Therefore the device will not go from "error passive" to the "bus off" state due to such a condition.



TL/DD/12067–65

FIGURE 28. CAN Bus States

Figure 28 shows the connection of different bus states according to the error counters.

SYNCHRONIZATION

Every receiver starts with a "hard synchronization" on the falling edge of the SOF bit. One bit time consists of four bit segments: Synchronization segment, propagation segment, phase segment 1 and phase segment 2.

A falling edge of the data signal should be in the synchronization segment. This segment has the fixed length of one time quanta. To compensate for the various delays within a network, the propagation segment is used. Its length is programmable from 1 to 8 time quanta. Phase segment 1 and phase segment 2 are used to resynchronize during an active frame. The length of these segments is from 1 to 8 time quanta long.

Two types of synchronization are supported:

Hard synchronization is done with the falling edge on the bus while the bus is idle, which is then interpreted as the SOF. It restarts the internal logic.

Soft synchronization is used to lengthen or shorten the bit time while a data or remote frame is received. Whenever a falling edge is detected in the propagation segment or in phase segment 1, the segment is lengthened by a specific value, the resynchronization jump width (see *Figure 30*).

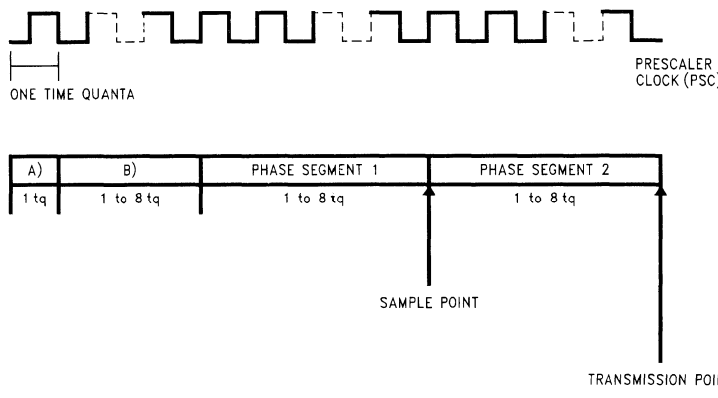
If a falling edge lies in the phase segment 2 (as shown in *Figure 30*) it is shortened by the resynchronization jump width. Only one resynchronization is allowed during one bit time. The sample point lies between the two phase segments and is the point where the received data is supposed to be valid. The transmission point lies at the end of phase segment 2 to start a new bit time with the synchronization segment.

Note 1: The resynchronization jump width (RJW) is automatically determined from the programmed value of PS. If a soft resynchronization is done during phase segment 1 or the propagation segment, then RJW will either be equal to 4 internal CAN clocks (CKI/(1 + *divider)*) or the programmed value of PS, whichever is less. PS2 will never be shorter than 1 internal CAN clock.

Note 2: (PS1—BTL settings any PSC setting) The PS1 of the BTL should always be programmed to values greater than 1. To allow device resynchronization for positive and negative phase errors on the bus. (if PS1 is programmed to one, a bit time could only be lengthened and never shortened which basically disables half of the synchronization).

5

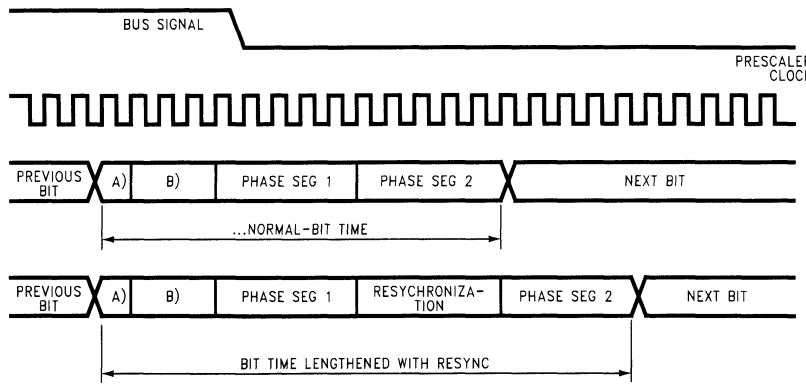
Basic CAN Concepts (Continued)



TL/DD/12067–66

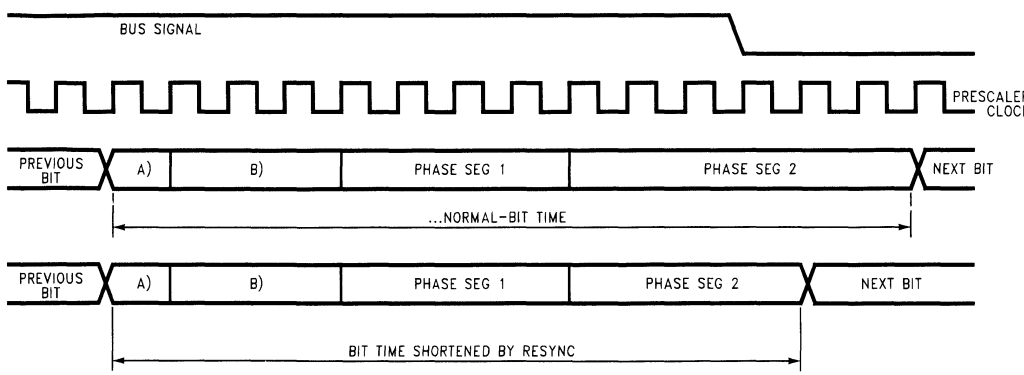
A) Synchronization segment
B) Propagation segment

FIGURE 29. Bit Timing



TL/DD/12067–67

FIGURE 30. Resynchronization 1



TL/DD/12067–68

FIGURE 31. Resynchronization 2

Comparators

The device has two differential comparators. Port L is used for the comparators. The output of the comparators is multiplexed out to two pins. The following are the Port L assignments:

L0 Comparator 1 positive input
L1 Comparator 1 negative input
L2 Comparator 1 output
L3 Comparator 2 negative input
L4 Comparator 2 positive input
L5 Comparator 2 negative input
L6 Comparator 2 output

Additionally the comparator output can be connected internally to the L-Port pin of the respective positive input and thereby generate an interrupt using the L-Port interrupt structure (neg/pos. edge, enable/disable).

Note that in *Figure 32*, pin L6 has a second alternate function of supporting the PWM0 output. The comparator 2 output MUST be disabled in order to use PWM0 output on L6.

Figure 32 shows the Comparator Block Diagram.

COMPARATOR CONTROL REGISTER (CMPLSS) (00D3)

These bits reside in the Comparator Register

CMP2 SEL	CMP2 OE	CMP2 RD	CMP2 EN	CMP1 OE	CMP1 RD	CMP1 EN	un- used

Bit 7 Bit 0

The register contains the following bits:

CMP1EN Enables comparator 1 ("1" = enable). If comparator 1 is disabled the associated L-pins can be used as standard I/O.

CMP1RD Reads comparator 1 output internally (CMP1EN = 1) Read-only, reads as a "0" if comparator not enabled.

CMP1OE Enables comparator 1 output ("1" = enable), CMP1EN bit must be set to enable this function.

CMP2EN Enables comparator 2 ("1" = enable). If comparator 2 is disabled the associated L-pins can be used as standard 1/O.

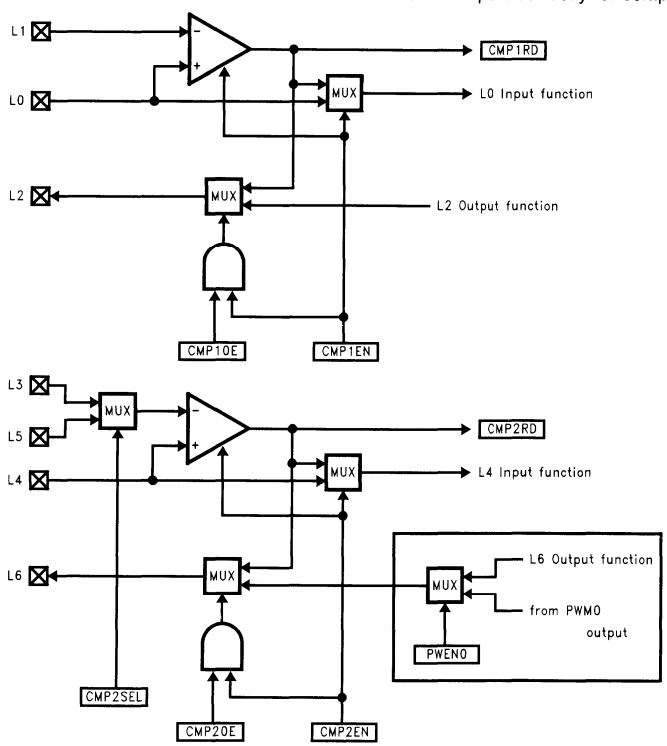
CMP2RD Reads comparator 2 output internally (CMP2EN = 1) Read-only, reads as a "0" if comparator not enabled.

CMP2OE Enables comparator 2 output ("1" = enable), CMP2EN bit must be set to enable this function.

CMP2SEL Selects which L port pin to use for comparator2 negative input. (CMP2SEL = 0 selects L5; CMP2SEL = 1 selects pin L3).

The Comparator Select/Control bits are cleared on RESET (the comparator is disabled). To save power, the program should also disable the comparator before the device enters the HALT mode.

The Comparator rise and fall times are symmetrical. The user program must set up the Configuration and Data registers of the L port correctly for comparator Inputs/Output.



TL/DD/12067–36

Note: The BOXED area shows logic from PWM Timer. Comparator 2 output (CMP2OE) must be disabled in order to use PWM0 output.

FIGURE 32. Comparator Block

Detection of Illegal Conditions

The device can detect various illegal conditions resulting from coding errors, transient noise, power supply voltage drops, runaway programs, etc.

Reading of undefined ROM gets zeroes. The opcode for software interrupt is zero. If the program fetches instructions from undefined ROM, this will force a software interrupt, thus signaling that an illegal condition has occurred.

The subroutine stack grows down for each call (jump to subroutine), interrupt, or PUSH, and grows up for each return or POP. The stack pointer is initialized to RAM location 02F Hex during reset. Consequently, if there are more returns than calls, the stack pointer will point to addresses 030 and 031 Hex (which are undefined RAM). Undefined RAM from addresses 030 to 03F Hex is read as all 1's, which in turn will cause the program to return to address 7FFF Hex. This is an undefined ROM location and the instruction fetched (all 0's) from this location will generate a software interrupt signaling an illegal condition.

Thus, the chip can detect the following illegal conditions:

1. Executing from undefined ROM.
2. Over "POP"ing the stack by having more returns than calls.

When the software interrupt occurs, the user can re-initialize the stack pointer and do a recovery procedure before restarting (this recovery program is probably similar to that following reset, but might not contain the same program initialization procedures).

MICROWIRE/PLUS

MICROWIRE/PLUS is a serial synchronous communications interface. The MICROWIRE/PLUS capability enables the device to interface with any of National Semiconductor's MICROWIRE peripherals (i.e., A/D converters, display drivers, E2PROMs etc.) and with other microcontrollers which support the MICROWIRE interface. It consists of an 8-bit serial shift register (SIO) with serial data input (SI), serial data output (SO) and serial shift clock (SK). *Figure 33* shows a block diagram of the MICROWIRE/PLUS logic.

The shift clock can be selected from either an internal source or an external source. Operating the MICROWIRE/PLUS arrangement with the internal clock source is called the Master mode of operation. Similarly, operating the MICROWIRE arrangement with an external shift clock is called the Slave mode of operation.

The CNTRL register is used to configure and control the MICROWIRE/PLUS mode. To use the MICROWIRE/PLUS, the MSEL bit in the CNTRL register is set to one. In the master mode the SK clock rate is selected by the two bits, SL0 and SL1, in the CNTRL register. Table XI details the different clock rates that may be selected.

MICROWIRE/PLUS OPERATION

Setting the BUSY bit in the PSW register causes the MICROWIRE/PLUS to start shifting the data. It gets reset when eight data bits have been shifted. The user may reset the BUSY bit by software to allow less than 8 bits to shift. If enabled, an interrupt is generated when eight data bits have been shifted. The device may enter the MICROWIRE/PLUS mode either as a Master or as a Slave. *Figure 34* shows how two COP888 family microcontrollers and several peripherals may be interconnected using the MICROWIRE/PLUS arrangements.

Warning:

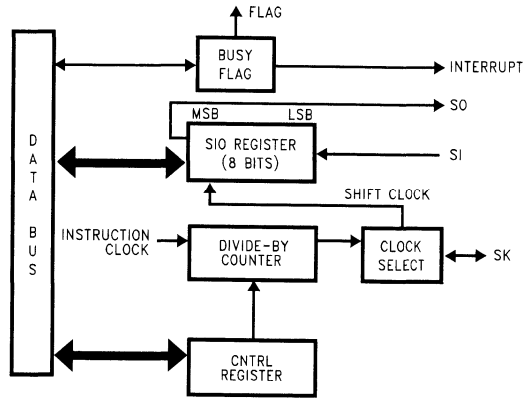
The SIO register should only be loaded when the SK clock is low. Loading the SIO register while the SK clock is high will result in undefined data in the SIO register. SK clock is normally low when not shifting.

Setting the BUSY flag when the input SK clock is high in the MICROWIRE/PLUS slave mode may cause the current SK clock for the SIO shift register to be narrow. For safety, the BUSY flag should only be set when the input SK clock is low.

MICROWIRE/PLUS Master Mode Operation

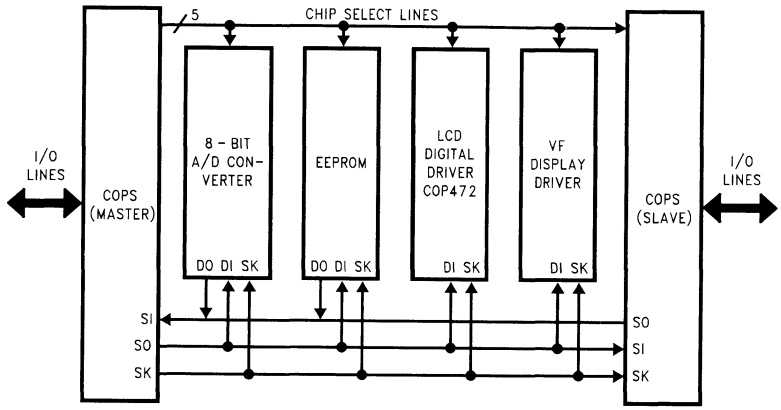
In the MICROWIRE/PLUS Master mode of operation the shift clock (SK) is generated internally. The MICROWIRE Master always initiates all data exchanges. The MSEL bit in the CNTRL register must be set to enable the SO and SK functions onto the G Port. The SO and SK pins must also be selected as outputs by setting appropriate bits in the Port G configuration register. Table XII summarizes the bit settings required for Master or Slave mode of operation.

MICROWIRE/PLUS (Continued)



TL/DD/12067–37

FIGURE 33. MICROWIRE/PLUS Block Diagram



TL/DD/12067–38

FIGURE 34. MICROWIRE/PLUS Application

MICROWIRE/PLUS (Continued)

TABLE XI. MICROWIRE/PLUS Master Mode Clock Selection

SL1	SL0	SK
0	0	$2 \times t_C$
0	1	$4 \times t_C$
1	x	$8 \times t_C$

Where t_C is the instruction cycle clock

MICROWIRE/PLUS Slave Mode Operation

In the MICROWIRE/PLUS Slave mode of operation the SK clock is generated by an external source. Setting the MSEL bit in the CNTRL register enables the SO and SK functions onto the G Port. The SK pin must be selected as an input and the SO pin is selected as an output pin by setting and resetting the appropriate bit in the Port G configuration register. Table V summarizes the settings required to enter the Slave mode of operation.

The user must set the BUSY flag immediately upon entering the Slave mode. This will ensure that all data bits sent by the Master will be shifted properly. After eight clock pulses the BUSY flag will be cleared and the sequence may be repeated.

Alternate SK Phase Operation

The device allows either the normal SK clock or an alternate phase SK clock to shift data in and out of the SIO register. In both the modes the SK is normally low. In the normal mode data is shifted in on the rising edge of the SK clock and the data is shifted out on the falling edge of the SK clock. The SIO register is shifted on each falling edge of the SK clock in the normal mode. In the alternate SK phase mode the SIO register is shifted on the rising edge of the SK clock.

A control flag, SKSEL, allows either the normal SK clock or the alternate SK clock to be selected. Resetting SKSEL causes the MICROWIRE/PLUS logic to be clocked from the normal SK signal. Setting the SKSEL flag selects the alternate SK clock. The SKSEL is mapped into the G6 configuration bit. The SKSEL flag will power up in the reset condition, selecting the normal SK signal.

TABLE XII. MICROWIRE/PLUS Mode Selection

G4 (SO) Config. Bit	G5 (SK) Config. Bit	G4 Fun.	G5 Fun.	Operation
1	1	SO	Int. SK	MICROWIRE/PLUS Master
0	1	TRI-STATE	Int. SK	MICROWIRE/PLUS Master
1	0	SO	Ext. SK	MICROWIRE/PLUS Slave
0	0	TRI-STATE	Ext. SK	MICROWIRE/PLUS Slave

This table assumes that the control flag MSEL is set.

Memory Map

All RAM, ports and registers (except A and PC) are mapped into data memory address space.

Address	Contents
00 to 2F	On-Chip RAM bytes (48 bytes)
30 to 7F	Unused RAM Address Space (Reads As All Ones)
80 to 9F	Unused RAM Address Space (Reads Undefined Data)
A0	PSCAL, PWM timer Prescaler Register
A1	RLON, PWM timer On-Time Register
A2	PWMCON, PWM Control Register
B0	TXD1, Transmit 1 Data
B1	TXD2, Transmit 2 Data
B2	TDLC, Transmit Data Length Code and Identifier Low
B3	TID, Transmit Identifier High
B4	RXD1, Receive Data 1
B5	RXD2, Receive Data 2
B6	RIDL, Receive Data Length Code
B7	RID, Receive Identify High
B8	CSCAL, CAN Prescaler
B9	CTIM, Bus Timing Register
BA	CBUS, Bus Control Register
BB	TCNTL, Transmit/Receive Control Register
BC	RTSTAT Receive/Transmit Status Register
BD	TEC, Transmit Error Count Register
BE	REC, Receive Error Count Register
BF	Reserved
C0 to C7	Reserved
C8	WKEDG, MIWU Edge Select Register
C9	WKEN, MIWU Enable Register
CA	WKPND, MIWU Pending Register
CB	Reserved
CC	Reserved
CD to CF	Reserved

Memory Map (Continued)

Address	Contents
D0	PORTLD, Port L Data Register
D1	PORTLC, Port L Configuration Register
D2	PORTLP, Port L Input Pins (Read Only)
D3	CMPSL, Comparator control register
D4	PORTGD, Port G Data Register
D5	PORTGC, Port G Configuration Register
D6	PORTGP, Port G Input Pins (Read Only)
D7 to DB	Reserved
DC	PORTD, Port D output register
DD to DF	Reserved for Port D
E0–E5	Reserved
E6	T1RBLO, Timer T1 Autoload Register Lower Byte
E7	T1RBHI, Timer T1 Autoload Register Upper Byte
E8	ICNTRL, Interrupt Control Register
E9	SIOR, MICROWIRE/PLUS Shift Register
EA	TMR1LO, Timer T1 Lower Byte
EB	TMR1HI, Timer T1 Upper Byte
EC	T1RALO, Timer T1 Autoload Register Lower Byte
ED	T1RAHI, Timer T1 Autoload Register T1RA Upper Byte
EE	CNTRL, Control Register
EF	PSW, Processor Status Word Register
F0 to FB	On-Chip RAM Mapped as Registers
FC	X Register
FD	SP Register
FE	B Register
FF	Reserved (Note A)

Note: Reading memory locations 30–7F Hex will return all ones. Reading other unused memory locations will return undefined data.

Note A: In devices with more than 128 bytes of RAM, location 0FF is used as the Segment register to switch between different Segments of RAM memory. In this device location 0FF can be used as a general purpose, on-chip RAM mapped register. However, the user is advised that caution should be taken in porting software utilizing this memory location to a chip with more than 128 bytes of RAM.

Addressing Modes

There are ten addressing modes, six for operand addressing and four for transfer of control.

OPERAND ADDRESSING MODES

Register Indirect

This is the "normal" addressing mode. The operand is the data memory addressed by the B pointer or X pointer.

Register Indirect (with auto post Increment or decrement of pointer)

This addressing mode is used with the LD and X instructions. The operand is the data memory addressed by the B pointer or X pointer. This is a register indirect mode that automatically post increments or decrements the B or X register after executing the instruction.

Direct

The instruction contains an 8-bit address field that directly points to the data memory for the operand.

Immediate

The instruction contains an 8-bit immediate field as the operand.

Short Immediate

This addressing mode is used with the Load B Immediate instruction. The instruction contains a 4-bit immediate field as the operand.

Indirect

This addressing mode is used with the LAID instruction. The contents of the accumulator are used as a partial address (lower 8 bits of PC) for accessing a data operand from the program memory.

TRANSFER OF CONTROL ADDRESSING MODES

Relative

This mode is used for the JP instruction, with the instruction field being added to the program counter to get the new program location. JP has a range from −31 to +32 to allow a 1-byte relative jump (JP + 1 is implemented by a NOP instruction). There are no "pages" when using JP, since all 15 bits of PC are used.

Absolute

This mode is used with the JMP and JSR instructions, with the instruction field of 12 bits replacing the lower 12 bits of the program counter (PC). This allows jumping to any location in the current 4k program memory segment.

Absolute Long

This mode is used with the JMPL and JSRL instructions, with the instruction field of 15 bits replacing the entire 15 bits of the program counter (PC). This allows jumping to any location up to 32k in the program memory space.

Indirect

This mode is used with the JID instruction. The contents of the accumulator are used as a partial address (lower 8 bits of PC) for accessing a location in the program memory. The contents of this program memory location serve as a partial address (lower 8 bits of PC) for the jump to the next instruction.

Note: The VIS is a special case of the Indirect Transfer of Control addressing mode, where the double byte vector associated with the interrupt is transferred from adjacent addresses in the program memory into the program counter (PC) in order to jump to the associated interrupt service routine.

Instruction Set

Register and Symbol Definition

Registers	
A	8-Bit Accumulator Register
B	8-Bit Address Register
X	8-Bit Address Register
SP	8-Bit Stack Pointer Register
PC	15-Bit Program Counter Register
PU	Upper 7 Bits of PC
PL	Lower 8 Bits of PC
C	1-Bit of PSW Register for Carry
HC	1-Bit of PSW Register for Half Carry
GIE	1-Bit of PSW Register for Global Interrupt Enable
VU	Interrupt Vector Upper Byte
VL	Interrupt Vector Lower Byte

Symbols	
[B]	Memory Indirectly Addressed by B Register
[X]	Memory Indirectly Addressed by X Register
MD	Direct Addressed Memory
Mem	Direct Addressed Memory or [B]
Meml	Direct Addressed Memory or [B] or Immediate Data
Imm	8-Bit Immediate Data
Reg	Register Memory: Addresses F0 to FF (Includes B, X and SP)
Bit	Bit Number (0 to 7)
←	Loaded with
↔	Exchanged with

Instruction Set (Continued)

INSTRUCTION SET

ADD	A,Meml	ADD	A ← A + Meml
ADC	A,Meml	ADD with Carry	A ← A + Meml + C, C ← Carry, HC ← Half Carry
SUBC	A,Meml	Subtract with Carry	A ← A − Meml + C, C ← Carry, HC ← Half Carry
AND	A,Meml	Logical AND	A ← A and $\overline{\text{Meml}}$
ANDSZ	A,Imm	Logical AND Immed., Skip if Zero	Skip next if (A and Imm) = 0
OR	A,Meml	Logical OR	A ← A or Meml
XOR	A,Meml	Logical EXclusive OR	A ← A xor Meml
IFEQ	MD,Imm	IF EQual	Compare MD and Imm, Do next if MD = Imm
IFEQ	A,Meml	IF EQual	Compare A and Meml, Do next if A = Meml
IFNE	A,Meml	IF Not Equal	Compare A and Meml, Do next if A ≠ Meml
IFGT	A,Meml	IF Greater Than	Compare A and Meml, Do next if A > Meml
IFBNE	#	If B Not Equal	Do next if lower 4 bits of B ≠ Imm
DRSZ	Reg	Decrement Reg., Skip if Zero	Reg ← Reg − 1, Skip if Reg = 0
SBIT	#,Mem	Set BIT	1 to bit, Mem (bit = 0 to 7 immediate)
RBIT	#,Mem	Reset BIT	0 to bit, Mem
IFBIT	#,Mem	IF BIT	If bit in A or Mem is true do next instruction
RPND		Reset PeNDing Flag	Reset Software Interrupt Pending Flag
X	A,Mem	EXchange A with Memory	A ⟷ Mem
X	A,[X]	EXchange A with Memory [X]	A ⟷ [X]
LD	A,Meml	LoaD A with Memory	A ← Meml
LD	A,[X]	LoaD A with Memory [X]	A ← [X]
LD	B,Imm	LoaD B with Immed.	B ← Imm
LD	Mem,Imm	LoaD Memory Immed.	Mem ← Imm
LD	Reg,Imm	LoaD Register Memory Immed.	Reg ← Imm
X	A, [B±]	EXchange A with Memory [B]	A ⟷ [B], (B ← B ± 1)
X	A, [X±]	EXchange A with Memory [X]	A ⟷ [X], (X ← ± 1)
LD	A, [B±]	LoaD A with Memory [B]	A ← [B], (B ← B ± 1)
LD	A, [X±]	LoaD A with Memory [X]	A ← [X], (X ← X ± 1)
LD	[B±],Imm	LoaD Memory [B] Immed.	[B] ← Imm, (B ← B ± 1)
CLR	A	CLeaR A	A ← 0
INC	A	INCrement A	A ← A + 1
DEC	A	DECrementA	A ← A − 1
LAID		Load A InDirect from ROM	A ← ROM (PU,A)
DCOR	A	Decimal CORrect A	A ← BCD correction of A (follows ADC, SUBC)
RRC	A	Rotate A Right thru C	C → A7 → . . . → A0 → C
RLC	A	Rotate A Left thru C	C ← A7 ← . . . ← A0 ← C
SWAP	A	SWAP nibbles of A	A7 . . . A4 ⟷ A3 . . . A0
SC		Set C	C ← 1, HC ← 1
RC		Reset C	C ← 0, HC ← 0
IFC		IF C	IF C is true, do next instruction
IFNC		IF Not C	If C is not true, do next instruction
POP	A	POP the stack into A	SP ← SP + 1, A ← [SP]
PUSH	A	PUSH A onto the stack	[SP] ← A, SP ← SP − 1
VIS		Vector to Interrupt Service Routine	PU ← [VU], PL ← [VL]
JMPL	Addr.	Jump absolute Long	PC ← ii (ii = 15 bits, 0k to 32k)
JMP	Addr.	Jump absolute	PC9 . . . 0 ← i (i = 12 bits)
JP	Disp.	Jump relative short	PC ← PC + r (r is −31 to +32, except 1)
JSRL	Addr.	Jump SubRoutine Long	[SP] ← PL, [SP−1] ← PU,SP−2, PC ← ii
JSR	Addr.	Jump SubRoutine	[SP] ← PL, [SP−1] ← PU,SP−2, PC9 . . . 0 ← i
JID		Jump InDirect	PL ← ROM (PU,A)
RET		RETurn from subroutine	SP + 2, PL ← [SP], PU ← [SP−1]
RETSK		RETurn and SKip	SP + 2, PL ← [SP],PU ← [SP−1]
RETI		RETurn from Interrupt	SP + 2, PL ← [SP],PU ← [SP−1],GIE ← 1
INTR		Generate an Interrupt	[SP] ← PL, [SP−1] ← PU, SP−2, PC ← 0FF
NOP		No OPeration	PC ← PC + 1

5

Instruction Execution Time

Most instructions are single byte (with immediate addressing mode instructions taking two bytes).

Most single byte instructions take one cycle time to execute.

Skipped instructions require x number of cycles to be skipped, where x equals the number of bytes in the skipped instruction opcode.

See the BYTES and CYCLES per INSTRUCTION table for details.

Bytes and Cycles per Instruction

The following table shows the number of bytes and cycles for each instruction in the format of byte/cycle.

Arithmetic and Logic Instructions

	[B]	Direct	Immed.
ADD	1/1	3/4	2/2
ADC	1/1	3/4	2/2
SUBC	1/1	3/4	2/2
AND	1/1	3/4	2/2
OR	1/1	3/4	2/2
XOR	1/1	3/4	2/2
IFEQ	1/1	3/4	2/2
IFGT	1/1	3/4	2/2
IFBNE	1/1		
DRSZ		1/3	
SBIT	1/1	3/4	
RBIT	1/1	3/4	
IFBIT	1/1	3/4	

RPND	1/1

Instructions Using A and C

CLRA	1/1
INCA	1/1
DECA	1/1
LAID	1/3
DCORA	1/1
RRCA	1/1
RLCA	1/1
SWAPA	1/1
SC	1/1
RC	1/1
IFC	1/1
IFNC	1/1
PUSHA	1/3
POPA	1/3
ANDSZ	2/2

Transfer of Control Instructions

JMPL	3/4
JMP	2/3
JP	1/3
JSRL	3/5
JSR	2/5
JID	1/3
VIS	1/5
RET	1/5
RETSK	1/5
RETI	1/5
INTR	1/7
NOP	1/1

Memory Transfer Instructions

	Register Indirect		Direct	Immed.	Register Indirect Auto Incr. and Decr.		
	[B]	[X]			[B+, B−]	[X+, X−]	
X A,*	1/1	1/3	2/3		1/2	1/3	
LD A,*	1/1	1/3	2/3	2/2	1/2	1/3	
LD B, Imm				1/1			(IF B < 16)
LD B, Imm				2/3			(IF B > 15)
LD Mem, Imm	2/2		3/3		2/2		
LD Reg, Imm			2/3				
IFEQ MD, Imm			3/3				

* = > Memory location addressed by B or X or directly.

Opcode Table

UPPER NIBBLE																
F	**E**	**D**	**C**	**B**	**A**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**	
JP − 15	JP −31	LD 0F0, #i	DRSZ 0F0	RRCA	RC	ADC A, #i	ADC A,[B]	IFBIT 0,[B]	ANDSZ A, #i	LD B,#0F	IFBNE 0	JSR x000−x0FF	JMP x000−x0FF	JP + 17	INTR	0
JP − 14	JP −30	LD 0F1, #i	DRSZ 0F1	*	SC	SUBC A, #i	SUB A,[B]	IFBIT 1,[B]	*	LD B,#0E	IFBNE 1	JSR x100−x1FF	JMP x100−x1FF	JP + 18	JP + 2	1
JP − 13	JP −29	LD 0F2, #i	DRSZ 0F2	X A, [X+]	X A, [B+]	IFEQ A, #i	IFEQ A,[B]	IFBIT 2,[B]	*	LD B,#0D	IFBNE 2	JSR x200−x2FF	JMP x200−x2FF	JP + 19	JP + 3	2
JP − 12	JP −28	LD 0F3, #i	DRSZ 0F3	X A, [X−]	X A, [B−]	IFGT A, #i	IFGT A,[B]	IFBIT 3,[B]	*	LD B,#0C	IFBNE 3	JSR x300−x3FF	JMP x300−x3FF	JP + 20	JP + 4	3
JP − 11	JP −27	LD 0F4, #i	DRSZ 0F4	VIS	LAID	ADD A, #i	ADD A,[B]	IFBIT 4,[B]	CLRA	LD B,#0B	IFBNE 4	JSR x400−x4FF	JMP x400−x4FF	JP + 21	JP + 5	4
JP − 10	JP −26	LD 0F5, #i	DRSZ 0F5	RPND	JID	AND A, #i	AND A,[B]	IFBIT 5,[B]	SWAPA	LD B,#0A	IFBNE 5	JSR x500−x5FF	JMP x500−x5FF	JP + 22	JP + 6	5
JP − 9	JP −25	LD 0F6, #i	DRSZ 0F6	X A,[X]	X A,[B]	XOR A, #i	XOR A,[B]	IFBIT 6,[B]	DCORA	LD B,#09	IFBNE 6	JSR x600−x6FF	JMP x600−x6FF	JP + 23	JP + 7	6
JP − 8	JP −24	LD 0F7, #i	DRSZ 0F7	*	*	OR A,#i	OR A,[B]	IFBIT 7,[B]	PUSHA	LD B,#08	IFBNE 7	JSR x700−x7FF	JMP x700−x7FF	JP + 24	JP + 8	7
JP − 7	JP −23	LD 0F8, #i	DRSZ 0F8	NOP	RLCA	LD A,#i	IFC	SBIT 0,[B]	RBIT 0,[B]	LD B,#07	IFBNE 8	JSR x800−x8FF	JMP x800−x8FF	JP + 25	JP + 9	8
JP − 6	JP −22	LD 0F9, #i	DRSZ 0F9	IFNE A,[B]	IFEQ Md,#i	IFNE A,#i	IFNC	SBIT 1,[B]	RBIT 1,[B]	LD B,#06	IFBNE 9	JSR x900−x9FF	JMP x900−x9FF	JP + 26	JP + 10	9
JP − 5	JP −21	LD 0FA, #i	DRSZ 0FA	LD A, [X+]	LD A, [B+]	LD [B+], #i	INCA	SBIT 2,[B]	RBIT 2,[B]	LD B,#05	IFBNE 0A	JSR xA00−xAFF	JMP xA00−xAFF	JP + 27	JP + 11	A
JP − 4	JP −20	LD 0FB, #i	DRSZ 0FB	LD A, [X−]	LD A, [B−]	LD [B−], #i	DECA	SBIT 3,[B]	RBIT 3,[B]	LD B,#04	IFBNE 0B	JSR xB00−xBFF	JMP xB00−xBFF	JP + 28	JP + 12	B
JP − 3	JP −19	LD 0FC, #i	DRSZ 0FC	LD Md,#i	JMPL	X A,Md	POPA	SBIT 4,[B]	RBIT 4,[B]	LD B,#03	IFBNE 0C	JSR xC00−xCFF	JMP xC00−xCFF	JP + 29	JP + 13	C
JP − 2	JP −18	LD 0FD, #i	DRSZ 0FD	DIR	JSRL	LD A,Md	RETSK	SBIT 5,[B]	RBIT 5,[B]	LD B,#02	IFBNE 0D	JSR xD00−xDFF	JMP xD00−xDFF	JP + 30	JP + 14	D
JP − 1	JP −17	LD 0FE, #i	DRSZ 0FE	LD A,[X]	LD A,[B]	LD [B],#i	RET	SBIT 6,[B]	RBIT 6,[B]	LD B,#01	IFBNE 0E	JSR xE00−xEFF	JMP xE00−xEFF	JP + 31	JP + 15	E
JP − 0	JP −16	LD 0FF, #i	DRSZ 0FF	*	*	LD B,#i	RETI	SBIT 7,[B]	RBIT 7,[B]	LD B,#00	IFBNE 0F	JSR xF00−xFFF	JMP xF00−xFFF	JP + 32	JP + 16	F

LOWER NIBBLE

where,

i is the immediate data

Md is a directly addressed memory location

* is an unused opcode

Note: The opcode 60 Hex is also the opcode for IFBIT #i,A

Mask Options

The COP684BC and COP884BC mask programmable options are shown below. The options are programmed at the same time as the ROM pattern submission.

OPTION 1: CLOCK CONFIGURATION

= 1 Crystal Oscillator (CKI/10)

G7 (CKO) is clock generator output to crystal/resonator

CKI is the clock input

OPTION 2: HALT

= 1 Enable HALT mode

= 2 Disable HALT mode

OPTION 3: BOND OUT

= 1 28-Pin SO

OPTION 4: ON-CHIP POWER-ON RESET

= 1 Enable ON-CHIP POWER-ON RESET

= 2 Disable ON-CHIP POWER-ON RESET

The chip can be driven by a clock input on the CKI input pin which can be between DC and 10 MHz. The CKO output clock is on pin G7. The CKI input frequency is divided down by 10 to produce the instruction cycle clock ($1/t_c$).

Development Support

SUMMARY

- iceMASTER: IM-COP8/400—Full feature in-circuit emulation for all COP8 products. A full set of COP8 Basic and Feature Family device and package specific probes are available.
- COP8 Debug Module: Moderate cost in-circuit emulation and development programming unit.
- COP8 Evaluation and Programming Unit: EPU-COP888GG—low cost in-circuit simulation and development programming unit.
- Assembler: COP8-DEV-IBMA. A DOS installable cross development Assembler, Linker, Librarian and Utility Software Development Tool Kit.
- C Compiler: COP8C. A DOS installable cross development Software Tool Kit.
- OTP/EPROM Programmer Support: Covering needs from engineering prototype, pilot production to full production environments.

Development Support (Continued)

IceMASTER (IM) IN-CIRCUIT EMULATION

The iceMASTER IM-COP8/400 is a full feature, PC based, in-circuit emulation tool development and marketed by MetaLink Corporation to support the whole COP8 family of products. National is a resale vendor for these products.

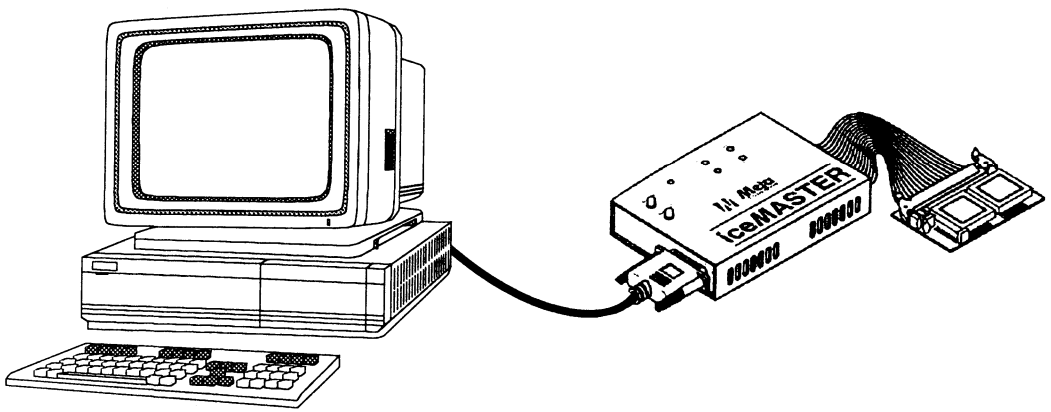
See *Figure 35* for configuration.

The iceMASTER IM-COP8/400 with its device specific COP8 Probe provides a rich feature set for developing, testing and maintaining product:

- Real-time in-circuit emulation; full 2.4V–5.5V operation range, full DC-10 MHz clock. Chip options are programmable or jumper selectable.
- Direct connection to application board by package compatible socket or surface mount assembly.
- Full 32 kbytes of loadable programming space that overlays (replaces) the on-chip ROM or EPROM. On-chip RAM and I/O blocks are used directly or recreated on the probe as necessary.
- Full 4k frame synchronous trace memory. Address, instruction, and eight unspecified, circuit connectable trace lines. Display can be HLL source (e.g., C source), assembly or mixed.
- A full 64k hardware configurable break, trace on, trace off control, and pass count increment events.
- Tool set integrated interactive symbolic debugger—supports both assembler (COFF) and C Compiler (.COD) linked object formats.
- Real time performance profiling analysis; selectable bucket definition.
- Watch windows, content updated automatically at each execution break.
- Instruction by instruction memory/register changes displayed on source window when in single step operation.
- Single base unit and debugger software reconfigurable to support the entire COP8 family; only the probe personality needs to change. Debugger software is processor customized, and reconfigured from a master model file.
- Processor specific symbolic display of registers and bit level assignments, configured from master model file.
- Halt/Idle mode notification.
- On-Line HELP customized to specific processor using master model file.
- Includes a copy of COP8-DEV-IBMA assembler and linker SDK.

IM Order Information

Base Unit	
IM-COP8/400-1	iceMASTER Base Unit, 110V Power Supply
IM-COP8/400-2	iceMASTER Base Unit, 220V Power Supply
iceMASTER Probe	
MHW-888BC28P5PC	28 DIP
28 DIP to 28 SO Adapter	
DM-SOIC28	28 SO



TL/DD/12067–70

FIGURE 35. COP8 iceMASTER Environment

Development Support (Continued)

iceMASTER DEBUG MODULE (DM)

The iceMASTER Debug Module is a PC based, combination in-circuit emulation tool and COP8 based OTP/EPROM programming tool developed and marketed by MetaLink Corporation to support the whole COP8 family of products. National is a resale vendor for these products.

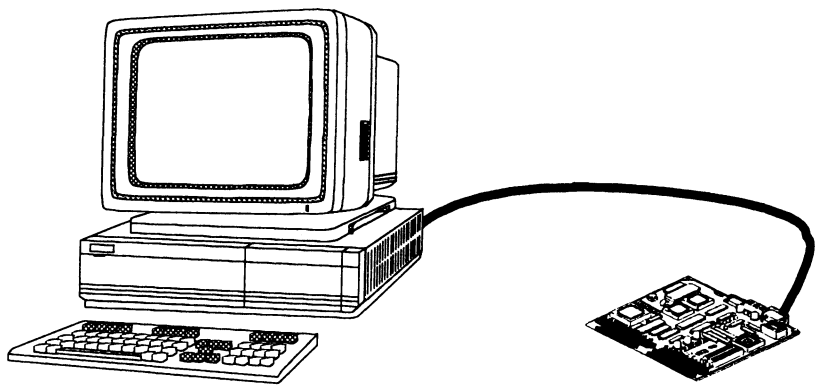
See *Figure 36* for configuration.

The iceMASTER Debug Module is a moderate cost development tool. It has the capability of in-circuit emulation for a specific COP8 microcontroller and in addition serves as a programming tool for COP8 OTP and EPROM product families. Summary of features is as follows:

- Real-time in-circuit emulation; full operating voltage range operation, full DC-10 MHz clock.
- All processor I/O pins can be cabled to an application development board with package compatible cable to socket and surface mount assembly.
- Full 32 kbytes of loadable programming space that overlays (replaces) the on-chip ROM or EPROM. On-chip RAM and I/O blocks are used directly or recreated as necessary.
- 100 frames of synchronous trace memory. The display can be HLL source (C source), assembly or mixed. The most recent history prior to a break is available in the trace memory.
- Configured break points; uses INTR instruction which is modestly intrusive.
- Software—only supported features are selectable.
- Tool set integrated interactive symbolic debugger—supports both assembler (COFF) and C Compiler (.COD) SDK linked object formats.
- Instruction by instruction memory/register changes displayed when in single step operation.
- Debugger software is processor customized, and reconfigured from a master model file.
- Processor specific symbolic display of registers and bit level assignments, configured from master model file.
- Halt/Idle mode notification.
- Programming menu supports full product line of programmable OTP and EPROM COP8 products. Program data is taken directly from the overlay RAM.
- Programming of 44 PLCC and 68 PLCC parts requires external programming adapters.
- Includes wallmount power supply.
- On-board V_{PP} generator from 5V input or connection to external supply supported. Requires V_{PP} level adjustment per the family programming specification (correct level is provided on an on-screen pop-down display).
- On-line HELP customized to specific processor using master model file.
- Includes a copy of COP8-DEV-IBMA assembler and linker SDK.

DM Order Information

Debug Module Unit	
COP8-DM/888BC	
Cable Adapters	
DM-COP8/28D	28 DIP
28 DIP to 28 SO Adapter	
DM-SOIC28	28 SO



TL/DD/12067–71

FIGURE 36. COP8-DM Environment

Development Support (Continued)

COP8 ASSEMBLER/LINKER SOFTWARE DEVELOPMENT TOOL KIT

National Semiconductor offers a relocatable COP8 macro cross assembler, linker, librarian and utility software development tool kit. Features are summarized as follows:

- Basic and Feature Family instruction set by "device" type.
- Nested macro capability.
- Extensive set of assembler directives.
- Supported on PC/DOS platform.
- Generates National standard COFF output files.
- Integrated Linker and Librarian.
- Integrated utilities to generate ROM code file outputs.
- DUMPCOFF utility.

This product is integrated as a part of MetaLink tools as a development kit, fully supported by the MetaLink debugger. It may be ordered separately or it is bundled with the MetaLink products at no additional cost.

Order Information

Assembler SDK:	
COP8-DEV-IBMA	Assembler SDK on installable 3.5" PC/DOS Floppy Disk Drive format. Periodic upgrades and most recent version is available on National's BBS and Internet.

COP8 C COMPILER

A C Compiler is developed and marketed by Byte Craft Limited. The COP8C compiler is a fully integrated development tool specifically designed to support the compact embedded configuration of the COP8 family of products.

Features are summarized as follows:

- ANSI C with some restrictions and extensions that optimize development for the COP8 embedded application.
- BITS data type extension. Register declaration #pragma with direct bit level definitions.
- C language support for interrupt routines.
- Expert system, rule based code generation and optimization.
- Performs consistency checks against the architectural definitions of the target COP8 device.
- Generates program memory code.
- Supports linking of compiled object or COP8 assembled object formats.
- Global optiomization of linked code.
- Symbolic debug load format fully source level supported by the MetaLink debugger.

SINGLE CHIP EMULATOR SUPPORT

The COP8 family is fully supported by single chip form, fit and function emulators. For more detailed information refer to the emulation device specific datasheets and the form, fit, function emulator selection table below.

OTP Ordering Information

Device Number	Clock Option	Package	Emulates
COP87L84BC	Crystal	28 SO	COP884BC

INDUSTRY WIDE OTP/EPROM PROGRAMMING SUPPORT

Programming support, in addition to the MetaLink development tools, is provided by a full range of independent approved vendors to meet the needs from the engineering laboratory to full production.

Approved List

Manufacturer	North America	Europe	Asia
BP Microsystems	(800) 225-2102 (713) 688-4600 Fax: (713) 688-0920	+49-8152-4183 +49-8856-932616	+852-234-16611 +852-2710-8121
Data I/O	(800) 426-1045 (206) 881-6444 Fax: (206) 882-1043	+44-0734-440011	Call North America
HI-LO	(510) 623-8860	Call Asia	+886-2-764-0215 Fax: +886-2-756-6403
ICE Technology	(800) 624-8949 (919) 430-7915	+44-1226-767404 Fax: 0-1226-370-434	
MetaLink	(800) 638-2423 (602) 926-0797 Fax: (602) 693-0681	+49-80 9156 96-0 Fax: +49-80 9123 86	+852-737-1800
Systems General	(408) 263-6667	+41-1-9450300	+886-2-917-3005 Fax: +886-2-911-1283
Needhams	(916) 924-8037 Fax: (916) 924-8065		

Development Support (Continued)

AVAILABLE LITERATURE

For more information, please see the COP8 Basic Family User's Manual, Literature Number 620895, COP8 Feature Family User's Manual, Literature Number 620897 and National's Family of 8-bit Microcontrollers COP8 Selection Guide, Literature Number 630009.

DIAL-A-HELPER SERVICE

Dial-A-Helper is a service provided by the Microcontroller Applications group. The Dial-A-Helper is an Electronic Information System that may be accessed as a Bulletin Board System (BBS) via data modem, as an FTP site on the Internet via standard FTP client application or as an FTP site on the Internet using a standard Internet browser such as Netscape or Mosaic.

The Dial-A-Helper system provides access to an automated information storage and retrieval system. The system capabilities include a MESSAGE SECTION (electronic mail, when accessed as a BBS) for communications to and from the Microcontroller Applications Group and a FILE SECTION which consists of several file areas where valuable application software and utilities could be found.

DIAL-A-HELPER BBS via a Standard Modem

Modem:	CANADA/U.S.:	(800) NSC-MICRO
		(800) 672-6427
	EUROPE:	(+49) 0-814-135 13 32
	Baud:	14.4k
	Set-up:	Length: 8-Bit
		Parity: None
		Stop Bit: 1
	Operation:	24 Hrs., 7 Days

DIAL-A-HELPER via FTP

ftp nscmicro.nsc.com

user: anonymous

password: username@yourhost.site.domain

DIAL-A-HELPER via a WorldWide Web Browser

ftp://nscmicro.nsc.com

National Semiconductor on the WorldWide Web

See us on the WorldWide Web at: http://www.national.com

CUSTOMER RESPONSE CENTER

Complete product information and technical support is available from National's customer response centers.

CANADA/ U.S.:	Tel:		(800) 272-9959
	email:		support @tevm2.nsc.com
EUROPE:	email:		europe.support@nsc.com
	Deutsch	Tel:	+49 (0) 180-530 85 85
	English	Tel:	+49 (0) 180-532 78 32
	Français	Tel:	+49 (0) 180-532 93 58
	Italiano	Tel:	+49 (0) 180-534 16 80
JAPAN:	Tel:		+81-043-299-2309
S.E. ASIA:	Beijing	Tel:	(+86) 10-6856-8601
	Shanghai	Tel:	(+86) 21-6415-4092
	Hong Kong	Tel:	(+852) 2737-1600
	Korea	Tel:	(+82) 2-3771-6909
	Malaysia	Tel:	(+60-4) 644-9061
	Singapore	Tel:	(+65) 255-2226
	Taiwan	Tel:	+886-2-521-3288
AUSTRALIA:	Tel:		(+61) 3-9558-9999
INDIA:	Tel:		(+91) 80-559-9467

National Semiconductor

PRELIMINARY

COP888EB/COP889EB/COP688EB/COP689EB 8-Bit Microcontroller with CAN Interface, A/D, and UART

General Description

The COP888EB is a member of the COP8™ microcontroller feature family, which uses an 8-bit single chip core architecture fabricated with National Semiconductor's M2CMOS™ process technology. (Continued)

Key Features

- CAN bus interface, with Software Power save mode
- 8-bit A/D Converter with 8 channels
- Fully buffered UART
- Multi-input wake up (MIWU) on both Port L and M
- SPI Compatible Master/Slave Interface
- Quiet Design (Low Radiated Emissions)
- 8096 bytes of on-board ROM
- 192 bytes of on-board RAM

Additional Peripheral Features

- Idle timer (programmable)
- Two 16-bit timer, with two 16-bit registers supporting
  - — Processor independent PWM mode
  - — External Event counter mode
  - — Input capture mode
- WATCHDOG™ and Clock Monitor
- MICROWIRE/PLUS™ serial I/O

I/O Features

- Memory mapped I/O
- Software selectable I/O options (TRI-STATE® outputs, Push pull outputs, Weak pull up input, High impedance input)
- Schmitt trigger inputs on Port G, L and M
- Packages: 44 PLCC with 31 I/O pins
  68 PLCC with 58 I/O pins

CPU/Instruction Set Features

- 1 μs instruction cycle time
- Fourteen multi-sourced vectored interrupts servicing
  - — External interrupt
  - — Idle Timer T0
  - — Timers (T1 and T2) (4 Interrupts)
  - — MICROWIRE/PLUS and SPI
  - — Multi-input Wake up
  - — Software Trap
  - — CAN interface (3 interrupts)
  - — UART (2 Inputs)
- Versatile easy to use instruction set
- 8-bit stacker pointer (SP) (Stack in RAM)
- Two 8-bit RegisterR Indirect Memory Pointers (B, X)

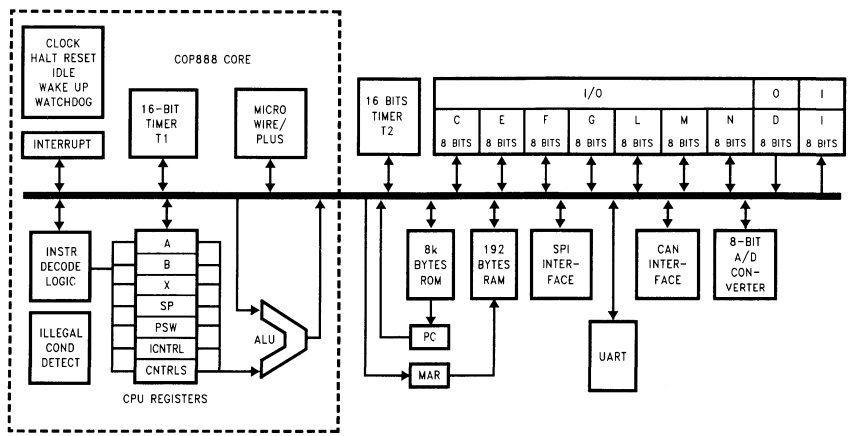
Fully Static CMOS

- Low current drain (typically < 1 μA)
- Two power saving modes: HALT, IDLE
- Single supply operation: 4.5V to 5.5V
- Temperature ranges: −40°C to +85°C and −55°C to +125°C

Development Support

- Emulation and OTP devices
- Real time emulation and full program debug offered by MetaLink Development System

Block Diagram



TL/DD/12837–1

FIGURE 1. Block Diagram

General Description (Continued)

The device is designed to perform complex embedded control applications such as those found in Automotive Control Applications, while providing control/diagnostic communications via the CAN bus interface. The device compiles to the basic CAN bus specification 2.0B (Passive). It is a fully static device fabricated using Nationals double metal silicon gate microCMOS technology. The device also includes design techniques to reduce radiated emissions by employing graduated I/O turn on, low current crystal oscillator and internal power supply filters. Efficient throughput is achieved through a regular efficient instruction set operating at a maximum of 1 μs instruction rate.

Basic Functional Description

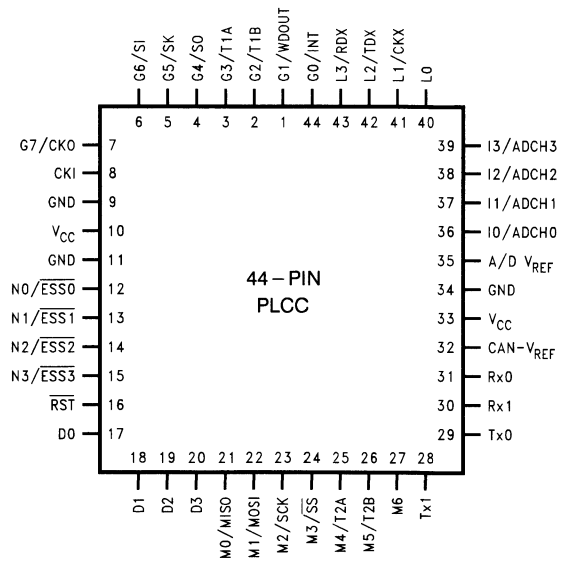
- CAN I/F—CAN serial bus interface block as described in the CAN specification part 2.0B (Passive)
  - — Interface rates up to 250k bit/s are supported utilizing standard message identifiers
- Programmable double buffered UART
- A/D—8-bit, 8 channel, 1-LSB Resolution, with improved Source Impedance and improved channel to channel cross talk immunity
- Multi-Input-Wake-Up (MIWU)—edge selectable wake-up and interrupt capability via input port and CAN interface (Port L, Port M and CAN I/F); supports Wake-Up capability on SPI, UART, and T2
- Port C—8-bit bi-directional I/O port
- Port D—8-bit Output port with high current drive capability (10 mA)
- Port E—8-bit bidirectional I/O
- Port F—8-bit bidirectional I/O
- Port G—8-bit bidirectional I/O port, including alternate functions for:
  - — MICROWIRE Input and Output
  - — Timer 1 Input or Output (Depending on mode selected)
  - — External Interrupt input
  - — WATCHDOG Output
- Port I—8-bit input port combining either digital input, or up to eight A/D input channels
- Port L—8-bit bidirectional I/O port, including alternate functions for:
  - — UART Transmit/Receive I/O
  - — Multi-input-wake up (MIWU on all pins)
- Port M—8-bit I/O port, with the following alternate function
  - — SPI Interface
  - — MIWU
  - — CAN Interface Wake-up (MSB)
  - — Timer 2 Input or Output (Depending on mode selected)
- Port N—8-bit bidirectional I/O
  - — SPI Slave Select Expander
- Two 16-bit multi-function Timer counters (T1 and T2) plus supporting registers
  - — (I/P Capture, PWM and Event Counting)
- Idle timer—Provides a basic time-base counter, (with interrupt) and automatic wake up from IDLE mode programmable
- MICROWIRE/PLUS—MICROWIRE serial peripheral interface, supporting both Master and Slave operation
- HALT and IDLE—Software programmable low current modes
  - — HALT—Processor stopped, Minimum current
  - — IDLE—Processor semi-active more than 60% power saving
- 8 kBytes ROM and 192 bytes of on board static RAM
- SPI Master/Slave interface includes 12 bytes Transmit and 12 bytes Receive FIFO Buffers. Operates up to 1M Bit/S
- On board programmable WATCHDOG and CLOCK Monitor

Applications

- Automobile Body Control and Comfort System
- Integrated Driver Informaiton Systems
- Steering Wheel Control
- Car Radio Control Panel
- Sensor/Actuator Applications in Automotive and Industrial Control

Connection Diagrams

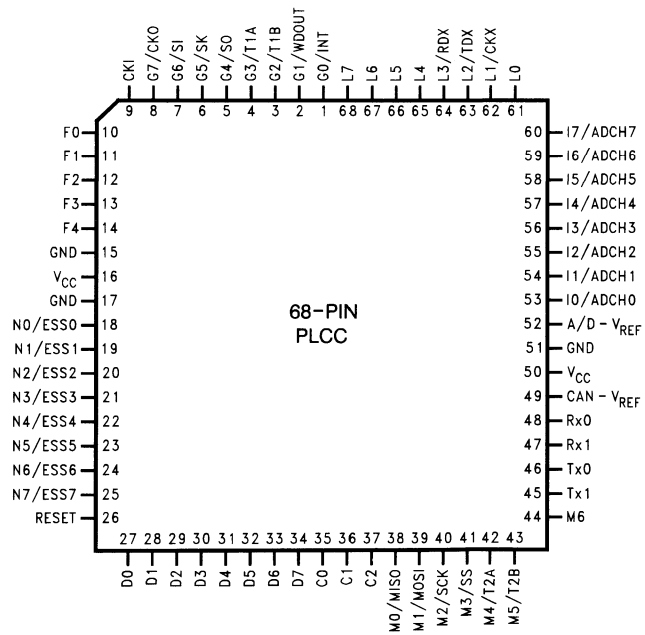
Plastic Chip Carrier



TL/DD/12837–2

Top View
Order Number COP888EB-XXX/V, COP688EB-XXX/V
See NS Plastic Chip Package Number V44A

Plastic Leaded Chip Carrier



TL/DD/12837–3

Top View
Order Number COP889EB-XXX/V, COP689EB-XXX/V
See NS Plastic Chip Package Number V68A

FIGURE 2. Connection Diagrams

Connection Diagrams (Continued)

TABLE I. Pinouts for 44-Pin and 68-Pin Packages

Port Pin	Type	ALT Function	44-Pin PLCC	68-Pin PLCC
G0	I/O	INT	44	1
G1	I/O	WDOUT	1	2
G2	I/O	T1B	2	3
G3	I/O	T1A	3	4
G4	I/O	SO	4	5
G5	I/O	SK	5	6
G6	I	SI	6	7
G7	I	CKO	7	8
D0	O		17	27
D1	O		18	28
D2	O		19	29
D3	O		20	30
D4	O			31
D5	O			32
D6	O			33
D7	O			34
I0	I	ADCH0	36	53
I1	I	ADCH1	37	54
I2	I	ADCH2	38	55
I3	I	ADCH3	39	56
I4	I	ADCH4		57
I5	I	ADCH5		58
I6	I	ADCH6		59
I7	I	ADCH7		60
L0	I/O	MIWU	40	61
L1	I/O	MIWU;CKX	41	62
L2	I/O	MIWU;TDX	42	63
L3	I/O	MIWU;RDX	43	64
L4	I/O	MIWU		65
L5	I/O	MIWU		66
L6	I/O	MIWU		67
L7	I/O	MIWU		68
E4	I/O			
E5	I/O			
E6	I/O			
E7	I/O			
M0	I/O	MIWU;MISO	21	38
M1	I/O	MIWU;MOSI	22	39
M2	I/O	MIWU;SCK	23	40

TABLE I. Pinouts for 44-Pin and 68-Pin Packages (Continued)

Port Pin	Type	ALT Function	44-Pin PLCC	68-Pin PLCC
M3	I/O	MIWU;$\overline{SS}$	24	41
M4	I/O	MIWU;T2A	25	42
M5	I/O	MIWU;T2B	26	43
M6	I/O	MIWU	27	44
M7	I/O			
N0	I/O	ESS0	12	18
N1	I/O	ESS1	13	19
N2	I/O	ESS2	14	20
N3	I/O	ESS3	15	21
N4	I/O	ESS4		22
N5	I/O	ESS5		23
N6	I/O	ESS6		24
N7	I/O	ESS7		25
F0	I/O			10
F1	I/O			11
F2	I/O			12
F3	I/O			13
F4	I/O			14
F5	I/O			
F6	I/O			
F7	I/O			
C0	I/O			35
C1	I/O			36
C2	I/O			37
C3	I/O			
C4	I/O			
C5	I/O			
C6	I/O			
RX0	I		31	48
RX1	I		30	47
TX0	O		29	46
TX1	O		28	45
$CANV_{REF}$			32	49
CKI			8	9
RESET			16	26
DV_{CC}			10, 33	16, 50
GND			9, 11, 34	15, 17, 51
A/D V_{REF}			35	52

Absolute Maximum Ratings

If Military/Aerospace specified devices are required, please contact the National Semiconductor Sales Office/Distributors for availability and specifications.

Supply Voltage (V_{CC})	6V
Voltage at Any Pin	−0.3V to V_{CC} +0.3V
Total Current into V_{CC} Pins (Source)	90 mA
Total Currnet out of GND Pins (Sink)	100 mA
Storage Temperature Range	−65°C to +150°C

Note: *Absolute maximum ratings indicate limits beyond which damage to the device may occur. DC and AC electrical specifications are not ensured when operating the device at absolute maximum ratings.*

DC Electrical Characteristics COP88xEB $-40°C \le T_A \le +85°C$

Parameter	Conditions	Min	Typ	Max	Units
Operating Voltage		4.5		5.5	V
Power Supply Ripple (Note 1)	Peak-to-Peak			0.1 V_{CC}	V
Supply Current CKI = 10 MHz (Note 2)	V_{CC} = 5.5V, t_c = 1 μs			16	mA
HALT Current (Notes 3, 4)	V_{CC} = 5.5V, CKI = 0 MHz		< 1		μA
IDLE Current (Note 4) CKI = 10 MHz	V_{CC} = 5.5V, t_c = 1 μs			5.5	mA
Input Levels (V_{IH}, V_{IL})					
Reset, CKI					
Logic High		0.8V_{CC}			V
Logic Low				0.2V_{CC}	V
All Other Inputs					
Logic High		0.7V_{CC}			V
Logic Low				0.2V_{CC}	V
Hi-Z Input Leakage	V_{CC} = 5.5V			±2	μA
Input Pull-Up Current	V_{CC} = 5.5V, V_{IN} = 0V	−40		−250	μA
Port G, L and M Input Hysteresis	(Note 7)		0.05V_{CC}		V
Output Current Levels					
D Outputs					
Source	V_{CC} = 4.5V, V_{OH} = 3.3V	−0.4			mA
Sink	V_{CC} = 4.5V, V_{OL} = 1.0V	10			mA
CAN Transmitter Outputs					
Source (Tx1)	V_{CC} = 4.5V, V_{OH} = V_{CC} −0.1V	−1.5			mA
	V_{CC} = 4.5V, V_{OH} = V_{CC} − 0.6V	−10		+5.0	mA
Sink (Tx0)	V_{CC} = 4.5V, V_{OL} = 0.1V	1.5			mA
	V_{CC} = 4.5V, V_{OL} = 0.6V	10			mA
All Others					
Source (Weak Pull-Up)	V_{CC} = 4.5V, V_{OH} = 2.7V	−10		−110	μA
Source (Push-Pull)	V_{CC} = 4.5V, V_{OH} = 3.3V	−0.4			mA
Sink (Push-Pull)	V_{CC} = 4.5V, V_{OL} = 0.4V	1.6			mA
TRI-STATE Leakage	V_{CC} = 5.5V			±2.0	μA
Allowable Sink/Source Current per Pin					
D Outputs (sink)				15	mA
Tx0 (Sink) (Note 7)				30	mA
Tx1 (Source) (Note 7)				30	mA
All Other				3	mA
Maximum Input Current without Latchup (Notes 5, 7)	Room Temp			±100	mA
RAM Retention Voltage, V_r (Note 6)	500 ns Rise and Fall Time	2.0			V
Input Capacitance	(Note 7)			7	pF
Load Capacitance on D2				1000	pF

Note 1: Maxiumum rate of voltage change must be < 0.5V/ms

Note 2: Supply current is measured after running 2000 cycles with a square wave CKI input, CKO open, inputs at V_{CC} or GND, and outputs open.

Note 3: The HALT mode will stop CKI from oscillating in the Crystal configurations. Halt test conditions: All inputs tied to V_{CC}; Port C, G, E, F, L, M and N I/Os configured as outputs and programmed low; D outputs programmed high. Parameter refers to HALT mode entered via setting bit 7 of the Port G data register. Part will pull up CKI during HALT in crystal clock mode. Both CAN main comparator and the CAN Wakeup comparator need to be disabled.

Note 4:. HALT and IDLE current specifications assume CAN block comparators are disabled.

Note 5: Pins G6 and $\overline{\text{RESET}}$ are designed with a high voltage input network. These pins allow input voltages greater than V_{CC} and the pins will have sink current to V_{CC} when biased at voltages greater than V_{CC} (the pins do not have source current when biased at a voltage below V_{CC}). The effective resistance to V_{CC} is 750Ω (typical). These two pins will not latch up. The voltage at the pins must be limited to less than 14V.

Note 6: Condition and parameter valid only for part in HALT mode.

Note 7: Parameter characterized but not tested.

5

Absolute Maximum Ratings

If Military/Aerospace specified devices are required, please contact the National Semiconductor Sales Office/Distributors for availability and specifications.

Supply Voltage (V_{CC})	7V
Voltage at Any Pin	−0.3V to V_{CC} +0.3V
Total Current into V_{CC} Pins (Source)	100 mA
Total Current out of GND Pins (Sink)	110 mA
Storage Temperature Range	−65°C to +150°C

Note: *Absolute maximum ratings indicate limits beyond which damage to the device may occur. DC and AC electrical specifications are not ensured when operating the device at absolute maximum ratings.*

DC Electrical Characteristics COP68xEB $-55°C \le T_A \le +125°C$

Parameter	Conditions	Min	Typ	Max	Units
Operating Voltage		4.5		5.5	V
Power Supply Ripple (Note 1)	Peak-to-Peak			0.1 V_{CC}	V
Supply Current CKI = 10 MHz (Note 2)	V_{CC} = 5.5V, t_c = 1 μs			16	mA
HALT Current (Notes 3, 4)	V_{CC} = 5.5V, CKI = 0 MHz		10		μA
IDLE Current (Note 4) CKI = 10 MHz	V_{CC} = 5.5V, t_c = 1 μs			5.5	mA
Input Levels (V_{IH}, V_{IL})					
$\overline{\text{Reset}}$, CKI					
Logic High		0.8V_{CC}			V
Logic Low				0.2V_{CC}	V
All Other Inputs					
Logic High		0.7V_{CC}			V
Logic Low				0.2V_{CC}	V
Hi-Z Input Leakage	V_{CC} = 5.5V			±5	μA
Input Pull-Up Current	V_{CC} = 5.5V, V_{IN} = 0V	−35		−250	μA
Port G, L and M Input Hysteresis	(Note 7)		0.05V_{CC}		V
Output Current Levels					
D Outputs					
Source	V_{CC} = 4.5V, V_{OH} = 3.3V	−0.4			mA
Sink	V_{CC} = 4.5V, V_{OL} = 1.0V	9.0			mA
CAN Transmitter Outputs					
Source (Tx1)	V_{CC} = 4.5V, V_{OH} = V_{CC} −0.1V	−1.5			mA
	V_{CC} = 4.5V, V_{OH} = V_{CC} − 0.6V	−10			mA
Sink (Tx0)	V_{CC} = 4.5V, V_{OL} = 0.1V	1.5			mA
	V_{CC} = 4.5V, V_{OL} = 0.6V	10			mA
All Others					
Source (Weak Pull-Up)	V_{CC} = 4.5V, V_{OH} = 2.7V	−9.0		−100	μA
Source (Push-Pull)	V_{CC} = 4.5V, V_{OH} = 3.3V	−0.4			mA
Sink (Push-Pull)	V_{CC} = 4.5V, V_{OL} = 0.4V	1.4			mA
TRI-STATE Leakage	V_{CC} = 5.5V			±5.0	μA
Allowable Sink/Source Current per Pin					
D Outputs (sink)				12	mA
Tx0 (Sink) (Note 7)				30	mA
Tx1 (Source) (Note 7)				30	mA
All Other				2.5	mA
Maximum Input Current without Latchup (Notes 5, 7)	Room Temp			±100	mA
RAM Retention Voltage, V_r (Note 6)	500 ns Rise and Fall Time	2.0			V
Input capacitance	(Note 7)			7	pF
Load Capacitance on D2				1000	pF

Note 1: Maxiumum rate of voltage change must be < 0.5V/ms

Note 2: Supply current is measured after running 2000 cycles with a square wave CKI input, CKO open, inputs at V_{CC} or GND, and outputs open.

Note 3: The HALT mode will stop CKI from oscillating in the Crystal configurations. Halt test conditions: All inputs tied to V_{CC}; Port C, G, E, F, L, M and N I/Os configured as outputs and programmed low; D outputs programmed high. Parameter refers to HALT mode entered via setting bit 7 of the Port G data register. Part will pull up CKI during HALT in crystal clock mode. Both CAN main comparator and the CAN Wakeup comparator need to be disabled.

Note 4: HALT and IDLE current specifications assume CAN block comparators are disabled.

Note 5: Pins G6 and $\overline{\text{RESET}}$ are designed with a high voltage input network. These pins allow input voltages greater than V_{CC} and the pins will have isnk current to V_{CC} when biased at voltages greater than V_{CC} (the pins do not have source current when biased at a voltage below V_{CC}). The effective resistance to V_{CC} is 750Ω (typical). These two pins will not latch up. The voltage at the pins must be limited to less than 14V.

Note 6: Condition and parameter valid only for part in HALT mode.

Note 7: Parameter characterized but not tested.

AC Electrical Characteristics COP68xEB and COP88xEB $-55°C \le T_A \le +125°C$

Parameter	Conditions	Min	Typ	Max	Units
Instruction Cycle Time (t_c)					
Crystal/Resonator	$V_{CC} \ge 4.5V$	1.0		DC	μs
Inputs					
t_{SETUP}	$V_{CC} \ge 4.5V$	200			ns
t_{HOLD}	$V_{CC} \ge 4.5V$	60			ns
Output Propagation Delay (t_{PD1}, t_{PD0}) (Note 8)	$C_L = 100$ pF, $R_L = 2.2$ kΩ				
SK, SO	$V_{CC} \ge 4.5V$			0.7	μs
All others	$V_{CC} \ge 4.5V$			1	μs
MICROWIRE					
Setup Time (tUWS) (Note 9)		20			ns
Hold Time (tUWH) (Note 9)		56			ns
Output Pop Delay (tUPD)				220	ns
Input Pulse Width					
Interrupt High Time		1			t_c
Interrupt Low Time		1			t_c
Timer 1, 2 High Time		1			t_c
Timer 1, 2 Low Time		1			t_c
Reset Pulse Width (Note 9)		1.0			μs

t_c = Instruction Cycle Time

The maximum bus speed achievable with the CAN interface is a function of crystal frequency, message length and software overhead. The device can support a bus speed of up to 1 Mbit/S with a 10 MHz oscillator and 2 byte messages. The 1M bus speed refers to the rate at which protocol and data bits are transferred on the bus. Longer messages require slower bus speeds due to the time required for software intervention between data bytes. The device will support a maximum of 125k bits/s with eight byte messages and a 10 MHz oscillator.

Note: For device testing purpose of all AC parameters, V_{OH} will be tested at $0.5*V_{CC}$.

Note 8: The output propagation delay is referenced to the end of the instruction cycle where the output change occurs.

Note 9: Parameter not tested.

On-Chip Voltage Reference $-55°C \le T_A \le +125°C$

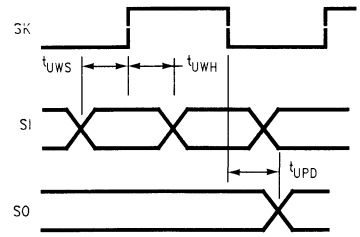
Parameter	Conditions	Min	Max	Units
Reference Voltage V_{REF}	$I_{OUT} < 80\ \mu A$, $V_{CC} = 5V$	$0.5V_{CC} - 0.12$	$0.5V_{CC} + 0.12$	V
Reference Supply Current, I_{DD}	$I_{OUT} = 0A$, (No Load) $V_{CC} = 5V$ (Note 1)		120	μA

Note 1: Reference supply I_{DD} is supplied for information purposes only, it is not tested.

CAN Comparator DC and AC Characteristics $4.8V \le V_{CC} \le 5.2V$, $-40°C \le T_A \le +125°C$

Parameter	Conditions	Min	Typ	Max	Units
Differential Input Voltage				± 25	mV
Input Offset Voltage	$1.5V < V_{IN} < V_{CC} - 1.5V$			± 10	mV
Input Common Mode Voltage Range		1.5		$V_{CC} - 1.5$	V
Input Hysteresis		8			mV

5



TL/DD/12837–4

FIGURE 3. MICROWIRE/PLUS Timing Diagram

A/D Converter Specifications ($4.5V \leq V_{CC} \leq 5.5V$) ($V_{SS} - 0.050V$) $\leq$ Any Input $\leq$ ($V_{CC} + 0.050V$)

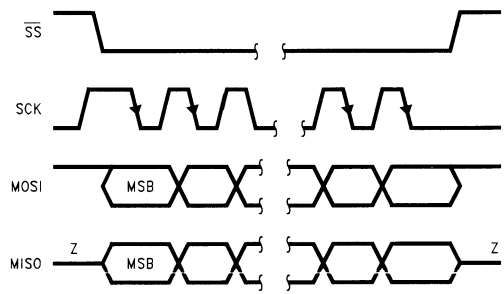
Parameter	Conditions	Min	Typ	Max	Units
Resolution				8	Bits
Absolute Accuracy	$V_{REF} = V_{CC}$			±2	LSB
Non-Linearity Deviation from the Best Straight Line				±1	LSB
Differential Non-Linearity				±1	LSB
Common Mode Input Range (Note 3)		GND		V_{CC}	V
DC Common Mode Error				±0.5	LSB
Off Channel Leakage Current			1	2.0	μA
On Channel Leakage Current			1	2.0	μA
A/D Clock Frequency (Note 2)		0.1		1.67	MHz
Conversion Time (Note 1)			17		A/D Clock Cycles
Internal Reference Resistance Turn-On Time (Note 4)				1	μs

Note 1: Conversion Time includes sample and hold time.

Note 2: See Prescaler description.

Note 3: For $V_{IN}(-) >= V_{IN}(+)$ the digital output code will be 0000 0000. Two on-chip doides are ties to each analog input. The diodes will forward conduct for analog input voltages below ground or above the V_{CC} supply. Be careful, during testing at low V_{CC} levels (4.5V), as high level analog inputs (5V) can cause this input diode to conduct—especially at elevated temperatures, and cause errors for analog inputs near full-scale. The spec allows 50 mV forward bias of either diode. This means that as long as the analog V_{IN} does not exceed the supply voltage by more than 50 mV, the output code will be correct. To achieve an absolute 0 V_{DC} to 5 V_{DC} input voltage range will therefore require a minimum supply voltage of 4.950 V_{DC} over temperature variations, initial tolerance and loading.

Note 4: Time for internal reference resistance to turn on after coming out of Halt or Idle Mode.



TL/DD/12837–5

FIGURE 4. SPI Timing Diagram

Pin Description

V_{CC} and GND are the power supply pins.

CKI is the clock input. The clock can come from a crystal oscillator (in conjunction with CKO). See Oscillator Description section.

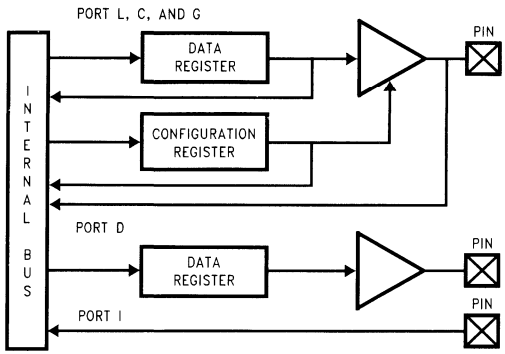
$\overline{\text{RESET}}$ is the master reset input. See Reset Description section.

The device contains seven bidirectional 8-bit I/O ports (C, E, F, G, L, M, N) where each individual bit may be independently configured as an input (Schmitt trigger inputs on all ports), output or TRI-STATE under program control. Three data memory address locations are allocated for each of these I/O ports. Each I/O port has two associated 8-bit memory mapped registers, the CONFIGURATION register and the output DATA register. A memory mapped address is also reserved for the input pins of each I/O port. (See the memory map for the various addresses associated with the I/O ports.) *Figure 5* shows the I/O port configurations for the device. The DATA and CONFIGURATION registers allow for each port bit to be individually configured under software control as shown below:

Configuration Register	Data Register	Port Set-Up
0	0	Hi-Z Input (TRI-STATE Output)
0	1	Input with Weak Pull-Up
1	0	Push-Pull Zero Output
1	1	Push-Pull One Output

Port L and M are 8-bit I/O ports, they support Multi-Input Wake-up (MIWU) on all eight pins. All L-pins and M-pins have Schmitt triggers on the inputs.

Port L and M only have one (1) interrupt vector.



TL/DD/12837–6

FIGURE 5. I/O Port Configurations

Port L has the following alternate features:

- L0 MIWU
- L1 MIWU or CKX
- L2 MIWU or TDX
- L3 MIWU or RDX
- L4 MIWU
- L5 MIWU
- L6 MIWU
- L7 MIWU

Port G is an 8-bit port with 5 I/O pins (G0–G5), an input pin (G6), and one dedicated output pin (G7). Pins G0–G6 all have Schmitt Triggers on their inputs. G7 serves as the dedicated output pin for the CKO clock output. There are two registers associated with the G Port, a data register and a configuration register. Therefore, each of the 6 I/O bits (G0–G5) can be individually configured under software control.

Since G6 is an input only pin and G7 is the dedicated CKO clock output pin the associated bits in the data and configuration registers for G6 and G7 are used for special purpose functions as outlined below. Reading the G6 and G7 data bits will return zeroes.

Note that the chip will be placed in the HALT mode by wirting a "1" to bit 7 of the Port G Data Register. Similarly the chip will be placed in the IDLE mode by writing a "1" to bit 6 of the Port G Data Register.

Writing a "1" to bit 6 of the Port G Configuration Register enables the MICROWIRE/PLUS to operate with the alternate phase of the SK clock

	Config. Reg.	Data Reg.
G7	CLKDLY	HALT
G6	Alternate SK	IDLE

Port G has the following alternate features:

- G0 INTR (External Interrupt Input)
- G1 Dedicated WATCHDOG output
- G2 (Timer T1 Capture Input)
- G3 T1A (Timer I/O)
- G4 SO (MICROWIRE Serial Data Output)
- G5 SK (MICROWIRE Serial Clock)
- G6 SI (MICROWIRE Serial Data Input)

Port G has the following dedicated function:

- G7 CKO Oscillator dedicated output

Port M is a bidirectional I/O, it may be configured in software as Hi-Z input, weak pull-up, or push-pull output. These pins may be used as general purpose input/output pins or for selected altlernate functions.

Port M pins have optional alternate functions. Each pin (M0–M5) has been assigned an alternate data, configuration, or wakeup source. If the respective alternate function is selected the content of the associated bits in the configuration and/or data register are ignored. If an alternate wakeup source is selected the input level at the respective pin will be ignored for the purpose of triggering a wakeup event, however it will still be possible to read that pin by accessing the input register. The SPI (Serial Peripheral Interface)

Pin Description (Continued)

block, for example, uses four of the Port M pins to automatically re-configure its MISO (Master Input, Slave Output), MOSI (Master Output, Slave Input), SCK (Serial Clock) and Slave-Select pins as inputs or outputs, depending on whether the interface has been configured as a Master or Slave. When the SPI interface is disabled those pins are available as general purpose I/O pins configurable by user software writing to the associated data and configuration bits. The CAN interface on the device makes use of one of the Port M's alternate wake-ups, to trigger a wakeup if such a condition has been detected on the CAN's dedicated receive pins.

Port M has the following alternate pin functions:

- M0 Multi-input Wakeup or MISO
- M1 Multi-input Wakeup or MOSI
- M2 Multi-input Wakeup or SCK
- M3 Multi-input Wakeup or $\overline{SS}$
- M4 Multi-input Wakeup or T2A
- M5 Multi-input Wakeup or T2B
- M6 Multi-input Wakeup
- M7 Multi-input Wakeup or CAN

Ports C, E, F and N are general-purpose, bidirectional I/O ports.

Any device package that has Port C, E, F, M, N but has fewer than eight pins, contains unbonded, floating pads internally on the chip. For these types of devices, the software should write a 1 to the configuration register bits corresponding to the non-existent port pins. This configures the port bits as outputs, thereby reducing leakage current of the device.

Port N is an 8-bit wide port with alternate function capability used for extending the slave select ($\overline{SS}$) lines of the on SPI interface. The SPI expander block provides mutually exclusive slave select extension signals ($\overline{ESS0}$ to $\overline{ESS7}$) according to the state of the $\overline{SS}$ line and specific contents of the SPI shift register. These slave select extension lines can be routed to the Port N I/O pins by enabling the alternate function of the port in the PORTNX register. If enabled, the internal signal on the $\overline{ESSx}$ line causes the ports state to change exactly like a change to the PORTND register. It is the user's responsibility to switch the port to an output when enabling the alternate function.

Port N has the following alternate pin functions:

- N0 $\overline{ESS0}$
- N1 $\overline{ESS1}$
- N2 $\overline{ESS2}$
- N3 $\overline{ESS3}$
- N4 $\overline{ESS4}$
- N5 $\overline{ESS5}$
- N6 $\overline{ESS6}$
- N7 $\overline{ESS7}$

CAN pins: For the on-chip CAN interface this device has five dedicated pins with the following features:

V_{REF} On-chip reference voltage with the value of $V_{CC}/2$

Rx0 CAN receive data input pin.

RX1 CAN receive data input pin.

Tx0 CAN transmit data output pin. This pin may be put in the TRI-STATE mode with the TXEN0 bit in the CAN Bus control register.

Tx1 CAN transmit data output pin. This pin may be put in the TRI-STATE mode with the TXEN1 bit in the CAN Bus control register.

ALTERNATE PORT FUNCTIONS

Many general-purpose pins have alternate functions. The software can program each pin to be used either for a general-purpose or for a specific function. The chip hardware determines which of the pins have alternate functions, and what those functions are. This section lists the alternate functions available on each of the pins.

Port D is an 8-bit output port that is preset high when RESET goes low. The user can tie two or more port D outputs (except D2) together in order to get a higher drive.

Note: Care must be exercised with D2 pin operation. At RESET, the external loads on this pin must ensure that the output voltages stay above 0.8 V_{CC} to prevent the chip from entering special modes. Also keep the external loading on D2 to < 1000 pF.

Port 1 is an 8-bit Hi-Z input port, and also provides the analog inputs to the A/D converter. If unterminated, Port 1 pins will draw power only when addressed.

Functional Description

The architecture of the device utilizes a modified Harvard architecture. With the Harvard architecture, the control store program memory (ROM) is separated from the data store memory (RAM). Both ROM and RAM have their own separate addressing space with separate address buses. The architecture, though based on Harvard architecture, permits transfer of data from ROM to RAM.

CPU REGISTERS

The CPU can do an 8-bit addition, subtraction, logical or shift operation in one instruction (t_C) cycle time.

There are five CPU registers:

A is the 8-bit Accumulator Register

PC is the 15-bit Program Counter Register

PU is the upper 7 bits of the program counter (PC)

PL is the lower 8 bits of the program counter (PC)

B is an 8-bit RAM address pointer, which can be optionally post auto incremented or decremented.

X is an 8-bit alternate RAM address pointer, which can be optionally post auto incremented or decremented.

SP is the 8-bit stack pointer, which points to the subroutine/interrupt stack (in RAM). The SP is initialized to RAM address 02F with reset.

All the CPU registers are memory mapped with the exception of the Accumulator (A) and the Program Counter (PC).

PROGRAM MEMORY

Program memory for the device consists of 8 kbytes of ROM. These bytes may hold program instructions or constant data (data tables for the LAID instruction, jump vectors for the JID instruction and interrupt vectors for the VIS instruction). The program memory is addressed by the 15-bit program counter (PC). All interrupts in the device vector to program memory location 0FF Hex.

DATA MEMORY

The data memory address space includes the on-chip RAM and data registers, the I/O registers (Configuration, Data and Pin), the control registers, the MICROWIRE/PLUS SIO shift register, and the various registers, and counters associated with the timers (with the exception of the IDLE timer). Data memory is addressed directly by the instruction or indirectly by the B, X and SP pointers.

The device has 192 bytes of RAM. Sixteen bytes of RAM are mapped as "registers" at addresses 0F0 to 0FF Hex. These registers can be loaded immediately, and also decremented and tested with the DRSZ (decrement register and skip if zero) instruction. The memory pointer registers X, SP, and B are memory mapped into this space at address locations 0FC to 0FE Hex respectively, with the other registers (other than reserved register 0FF) being available for general usage.

The instruction set permits any bit in memory to be set, reset or tested. All I/O and registers (except A and PC) are memory mapped; therefore I/O bits and register bits can be directly and individually set, reset and tested. The accumulator (A) bits can also be directly and individually tested.

Note: RAM contents are undefined upon power-up.

RESET

The $\overline{\text{RESET}}$ input when pulled low initializes the microcontroller. Initialization will occur whenever the $\overline{\text{RESET}}$ input is pulled low. Upon initialization, the data and configuration registers for Ports L and G, are cleared, resulting in these Ports being initialized to the TRI-STATE mode. Port D is initialized high with $\overline{\text{RESET}}$. The PC, CNTRL, and INCTRL control registers are cleared. The Multi-Input Wakeup registers WKEN, WKEDG, and WKPND are cleared. The Stack Pointer, SP, is initialized to 06F Hex.

The following initializations occur with $\overline{\text{RESET}}$:

SPI:
- SPICNTRL: Cleared
- SPISTAT: Cleared
- STBE Bit: Set

T1CNTRL & T2CNTRL: Cleared

ITMR: Cleared and IDLE timer period is reset to 4k Instr. CLK

ENAD: Cleared

ADDSLT: Random

SIOR: Unaffected after RESET with power already applied.
Random after RESET at power on.

Port L: TRI-STATE

Port G: TRI-STATE

Port D: HIGH

PC: CLEARED

PSW, CNTRL and ICNTRL registers: CLEARED

Accumulator and Timer 1:

RANDOM after RESET with power already applied
RANDOM after RESET at power-on

SP (Stack Pointer): Loaded with 6F Hex

B and X Pointers:

UNAFFECTED after RESET with power already applied
RANDOM after RESET at power-up

RAM:

UNAFFECTED after RESET with power already applied
RANDOM after RESET at power-up

CAN: The CAN Interface comes out of external reset in the "error-active" state and waits until the user's software sets either one or both of the TXEN0, TXEN1 bits to "1". After that, the device will not start transmission or reception of a frame util eleven consecutive "recessive" (undriven) bits have been received. This is done to ensure that the output drivers are not enamble during an active message on the bus.

CSCAL, CTIM, TCNTL, TEC, REC: CLEARED

RTSTAT: CLEARED with the exception of the TBE bit which is set to 1

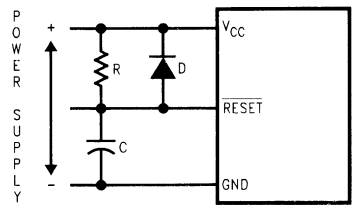
RID, RIDL, TID, TDLC: RANDOM

Functional Description (Continued)

WATCHDOG: The device comes out of reset with both the WATCHDOG logic and the Clock Monitor detector armed, with the WATCHDOG service window bits set and the Clock Monitor bit set. The WATCHDOG and Clock Monitor circuits are inhibited during reset. The WATCHDOG service window bits being initialized high default to the maximum WATCHDOG service window of 64k t_c clock cycles. The Clock Monitor bit being initialized high will cause a Clock Monitor bit being initialized high will cause a Clock Monitor error following reset if the clock has not reached the minimum specified frequency at the termination of reset. A Clock Monitor error will cause an active low error output on pin G1. This error output will continue until 16 t_c–32 t_c clock cycles following the clock frequency reaching the minimum specified value, at which time the G1 output will enter the TRI-STATE mode.

The $\overline{\text{RESET}}$ signal goes directly to the HALT latch to restart a halted chip.

When using external reset, the external RC network shown in *Figure 6* should be used to ensure that the $\overline{\text{RESET}}$ pin is held low until the power supply to the chip stabilizes. Under no circumstances should the $\overline{\text{RESET}}$ pin be allowed to float.



TL/DD/12837–7

RC > 5 x Power Supply Rise Time

FIGURE 6. Recommended Reset Circuit

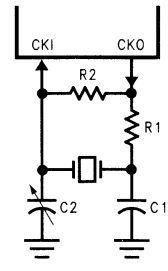
Oscillator Circuits

The chip can be driven by a clock input on the CKI input pin which can be between DC and 10 MHz. The CKO output clock is on pin G7. The CKI input frequency is divided by 10 to produce the instruction cycle clock ($1/t_c$).

Figure 7 shows the Crystal diagram.

CRYSTAL OSCILLATOR

CKI and CKO can be connected to make a closed loop crystal (or resonator) controlled oscillator.



TL/DD/12837–8

FIGURE 7. Crystal Oscillator Diagram

Table II shows the component values required for various standard crystal values.

TABLE II. Crystal Oscillator Configuration, T_A = 25°C

R1 (kΩ)	R2 (MΩ)	C1 (pF)	C2 (pF)	CKI Freq. (MHz)	Conditions
0	1	30	30–36	10	V_{CC} = 5V
0	1	30	30–36	4	V_{CC} = 5V
0	1	200	100–150	0.455	V_{CC} = 5V

Control Registers

CNTRL Register (Address X'00EE)

The Timer1 (T1 and MICROWIRE/PLUS control register contains the following bits:

SL1 & SL0 Select the MICROWIRE/PLUS clock divide by (00 = 2, 01 = 4, 1x = 8)

IEDG External interrupt edge polarity select (0 = Rising edge, 1 = Falling edge)

MSEL Selects G5 and G4 as MICROWIRE/PLUS signals SK and SO respectively

T1C0 Timer T1 Start/Stop control in timer Timer T1 Underflow Interrupt Pending Flag in timer mode 3

T1C1 Timer T1 mode control bit

T1C2 Timer T1 mode control bit

T1C3 Timer T1 mode control bit

T1C3	T1C2	T1C1	T1C0	MSEL	IEDG	SL1	SL0

Bit 7 Bit 0

PSW Register (Address X'00EF)

The PSW register contains the following select bits:

GIE Global interrupt enable (enables interrupts)

EXEN Enable external interrupt

BUSY MICROWIRE/PLUS busy shifting flag

EXPND External interrupt pending

T1ENA Timer T1 Interrupt Enable for Timer Underflow or T1A Input capture edge

T1PNDA Timer T1 Interrupt Pending Flag (Autoreload RA in mode 1, T1 Underflow in Mode 2, T1A capture edge in mode 3)

C Carry Flag

HC Half Carry Flag

HC	C	T1PNDA	T1ENA	EXPND	BUSY	EXEN	GIE

Bit 7 Bit 0

The Half-Carry bit is also affected by all the instructions that affect the Carry flag. The SC (Set Carry) and RC (Reset Carry) instructions will respectively set or clear both the carry flags. In addition to the SC and RC instructions, ADC, SUBC, RRC and RLC instructions affect the carry and Half Carry flags.

ICNTRL Register (Address X'00E8)

The ICNTRL register contains the following bits:

T1ENB Timer T1 Interrupt Enable for T1B Input capture edge

T1PNDB Timer T1 Interrupt Pending Flag for T1B capture edge

WEN Enable MICROWIRE/PLUS interrupt

WPND MICROWIRE/PLUS interrupt pending

T0EN Timer T0 Interrupt Enable (Bit 12 toggle)

T0PND Timer T0 Interrupt pending

LPEN Port L Interrupt Enable (Multi-Input Wakeup/ Interrupt)

Bit 7 could be used as a flag

Unused	LPEN	T0PND	T0EN	WPND	WEN	T1PNDB	T1ENB

Bit 7 Bit 0

T2CNTRL Register (Address X'00C6)

The T2CNTRL register contains the following bits:

T2ENB Timer T2 Interrupt Enable for T2B Input capture edge

T2PNDB Timer T2 Interrupt Pending Flag for T2B capture edge

T2ENA Timer T2 Interrupt Enable for Timer Underflow or T2 Input capture edge

T2PNDA Timer T2 Interrupt Pending Flag (Autoreload RA in mode 1, T2 Underflow in mode 2, T2A capture edge in mode 3)

T2C0 Timer T2 Start/Stop control in timer modes 1 and 2 Timer T2 Underflow Interrupt Pending Flag in timer mode 3

T2C1 Timer T2 mode control bit

T2C2 Timer T2 mode control bit

T2C3 Timer T2 mode control bit

T2C3	T2C2	T2C1	T2C0	T2PNDA	T2ENA	T2PNDB	T2ENB

Bit 7 Bit 0

Timers

The device contains a very versatile set of timers (T0, T1 and T2). All timers and associated autoreload/capture registers power up containing random data.

TIMER T0 (IDLE TIMER)

The device supports applications that require maintaining real time and low power with the IDLE mode. This IDLE mode support is furnished by the IDLE timer T0, which is a 16-bit timer. The Timer T0 runs continuously at the fixed rate of the instruction cycle clock, t_C. The user cannot read or write to the IDLE Timer T0, which is a count down timer.

The Timer T0 supports the following functions:

- Exit out of the Idle Mode (See Idle Mode description)
- WATCHDOG logic (See WATCHDOG description)
- Start up delay out of the HALT mode

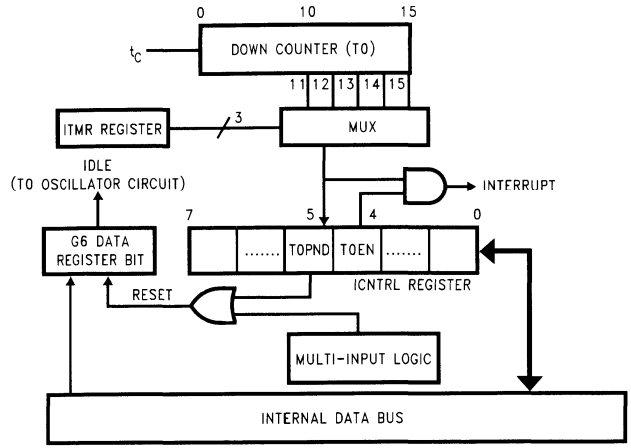
Figure 8 is a functional block diagram showing the structure of the IDLE Timer and its associated interrupt logic.

Bits 11 through 15 of the ITMR register can be selected for triggering the IDLE Timer interrupt. Each time the selected bit underflows (every 4k, 8k, 16k, 32k or 64k instruction cycles), the IDLE Timer interrupt pending bit T0PND is set, thus generating an interrupt (if enabled), and bit 6 of the Port G data register is reset, thus causing an exit from the IDLE mode if the device is in that mode.

In order for an interrupt to be generated, the IDLE Timer interrupt enable bit T0EN must be set, and the GIE (Global Interrupt Enable) bit must also be set. The T0PND flag and T0EN bit are bits 5 and 4 of the ICNTRL register, respectively. The interrupt can be used for any purpose. Typically, it is used to perform a task upon exit from the IDLE mode. For more information on the IDLE mode, refer to the Power Save Modes section.

5

Timers (Continued)



TL/DD/12837–9

FIGURE 8. Functional Block Diagram for Idle Timer T0

The Idle Timer period is selected by bits 0–2 of the ITMR register Bits 3–7 of the ITMR Register are reserved and should not be used as software flags.

TABLE III. Idle Timer Window Length

ITSEL2	ITSEL1	ITSEL0	Idle Timer Period (Instruction Cycles)
0	0	0	4,096
0	0	1	8,192
0	1	0	16,384
0	1	1	32,768
1	X	X	65,536

The ITMR register is cleared on Reset and the Idle Timer period is reset to 4,096 instruction cycles.

ITMR Register (Address X'0xCF)

Reserved	ITSEL2	ITSEL1	ITSLE0

Bit 7 Bit 0

Any time the IDLE Timer period is changed there is the possibility of generating a spurious IDLE Timer interrupt by setting the T0PND bit. The user is advised to disable IDLE Timer interrupts prior to changing the value of the ITSEL bits of the ITMR Register and then clear the T0PND bit before attempting to synchronize operation to the IDLE Timer.

TIMER T1 and TIMER T2

The device has a set of three powerful timer/counter blocks, T1 and T2. The associated features and functioning of a timer block are described by referring to the timer block Tx. Since the three timer blocks, T1 and T2 are identical, all comments are equally applicable to either of the three timer blocks.

Each timer block consists of a 16-bit timer, Tx, and two supporting 16-bit autoreload/capture registers, RxA and RxB. Each timer block has two pins associated with it, TxA and TxB. The pin TxA supports I/O required by the timer block, while the pin TxB is an input to the timer block. The powerful and flexible timer block allows the device to easily perform all timer functions with minimal software overhead. The timer block has three operating modes: Processor Independent PWM mode, External Event Counter mode, and Input Capture mode.

The control bits TxC3, TxC2, and TxC1 allow selection of the different modes of operation.

Timers (Continued)

Mode 1. Processor Independent PWM Mode

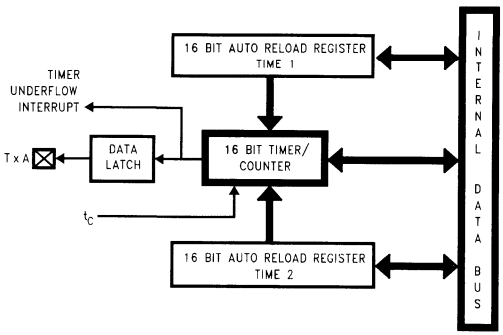
As the name suggests, this mode allows the device to generate a PWM signal with very minimal user intervention.

The user only has to define the parameters of the PWM signal (ON time and OFF time). Once begun, the timer block will continuously generate the PWM signal completely independent of the microcontroller. The user software services the timer block only when the PWM parameters require updating.

In this mode the timer Tx counts down at a fixed rate of t_C. Upon every underflow the timer is alternately reloaded with the contents of supporting registers, RxA and RxB. The very first underflow of the timer causes the timer to reload from the register RxA. Subsequent underflows cause the timer to be reloaded from the registers alternately beginning with the register RxB.

The Tx Timer control bits, TxC3, TxC2 and TxC1 set up the timer for PWM mode operation.

Figure 9 shows a block diagram of the timer in PWM mode.



TL/DD/12837–10

FIGURE 9. Timer in PWM Mode

The underflows can be programmed to toggle the TxA output pin. The underflows can also be programmed to generate interrupts.

Underflows from the timer are alternately latched into two pending flags, TxPNDA and TxPNDB. The user must reset these pending flags under software control. Two control enable flags, TxENA and TxENB, allow the interrupts from the timer underflow to be enabled or disabled. Setting the timer enable flag TxENA will cause an interrupt when a timer underflow causes the RxA register to be reloaded into the timer. Setting the timer enable flag TxENB will cause an interrupt when a timer underflow causes the RxB register to be reloaded into the timer. Resetting the timer enable flags will disable the associated interrupts.

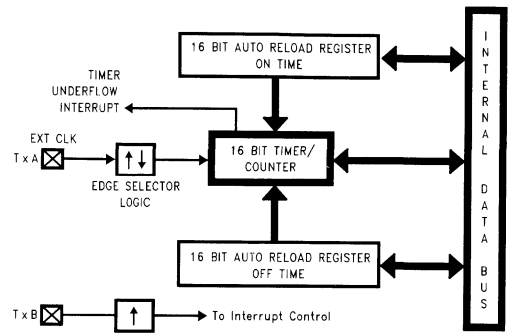
Either or both of the timer underflow interrupts may be enabled. This gives the user the flexibility of interrupting once per PWM period on either the rising or falling edge of the PWM output. Alternatively, the user may choose to interrupt on both edges of the PWM output.

Mode 2. External Event Counter Mode

This mode is quite similar to the processor independent PWM mode described above. The main difference is that the timer, Tx, is clocked by the input signal from the TxA pin. The Tx timer control bits, TxC3, TxC2 and TxC1 allow the timer to be clocked either on a positive or negative edge from the TxA pin. Underflows from the timer are latched into the TxPNDA pending flag. Setting the TxENA control flag will cause an interrupt when the timer underflows.

In this mode the input pin TxB can be used as an independent positive edge sensitive interrupt input if the TxENB control flag is set. The occurrence of the positive edge on the TxB input pin is latched to the TxPNDB flag.

Figure 10 shows a block diagram of the timer in External Event Counter mode.



TL/DD/12837–11

FIGURE 10. Timer in External Event Counter Mode

Note: The PWM output is not available in this mode since the TxA pin is being used as the counter input clock.

Mode 3. Input Capture Mode

The device can precisely measure external frequencies or time external events by placing the timer block, Tx, in the input capture mode.

In this mode, the timer Tx is constantly running at the fixed t_C rate. The two registers, RxA and RxB, act as capture registers. Each register acts in conjunction with a pin. The register RxA acts in conjunction with the TxA pin and the register RxB acts in conjunction with the TxB pin.

The timer value gets copied over into the register when a trigger event occurs on its corresponding pin. Control bits, TxC3, TxC2 and TxC1, allow the trigger events to be specified either as a positive or a negative edge. The trigger condition for each input pin can be specified independently.

Timers (Continued)

The trigger conditions can also be programmed to generate interrupts. The occurrence of the specified trigger condition on the TxA and TxB pins will be respectively latched into the pending flags, TxPNDA and TxPNDB. The control flag TxENA allows the interrupt on TxA to be either enabled or disabled. Setting the TxENA flag enables interrupts to be generated when the selected trigger condition occurs on the TxA pin. Similarly, the flag TxENB controls the interrupts from the TxB pin.

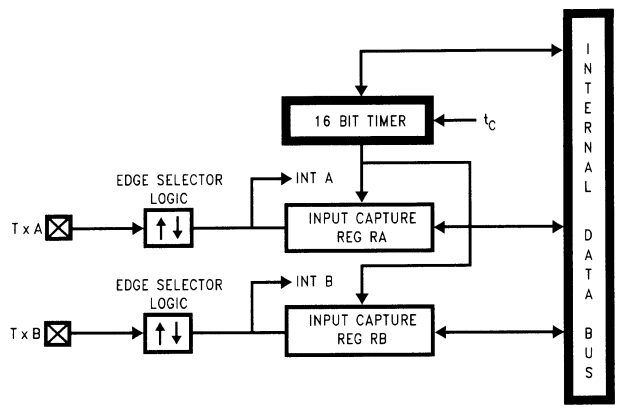
Underflows from the timer can also be programmed to generate interrupts. Underflows are latched into the timer TxC0 pending flag (the TxC0 control bit serves as the timer underflow interrupt pending flag in the Input Capture mode). Consequently, the TxC0 control bit should be reset when entering the Input Capture mode. The timer underflow interrupt is enabled with the TxENA control flag. When a TxA interrupt occurs in the Input Capture mode, the user must check both whether a TxA input capture or a timer underflow (or both) caused the interrupt.

Figure 11 shows a block diagram of the timer in Input Capture mode.

TIMER CONTROL FLAGS

The timers T1 and T2 have identical control structures. The control bits and their functions are summarized below.

TxC0	Timer Start/Stop control in Modes 1 and 2 (Processor Independent PWM and Extenral Event Counter), where 1 = State, 0 = Stop
	Timer Underflow Interrupt Pending Flag in Mode 3 (Input Capture)
TxPNDA	Timer Interrupt Pending Flag
TxPNDB	Timer Interrupt Pending Flag
TxENA	Timer Interrupt Enable Flag
TxENB	Timer Interrupt Enable Flag
	1 = Timer Interrupt Enabled
	0 = Timer Interrupt Disabled
TxC3	Timer mode control
TxC2	Timer mode control
TxC1	Timer mode control



TL/DD/12837–12

FIGURE 11. Timer in Input Capture Mode

Timers (Continued)

The timer mode control bits (TxC3, TxC2 and TxC1) are detailed below:

TxC3	TxC2	TxC1	Timer Mode	Interrupt A Source	Interrupt B Source	Timer Counts On
0	0	0	MODE 2 (External Event Counter)	Time Underflow	Pos. TxB Edge	TxA Pos. Edge
0	0	1	MODE 2 (External Event Counter)	Timer Underflow	Pos. TxB Edge	TxA Neg. Edge
1	0	1	MODE 1 (PWM) TxA Toggle	Autoreload RA	Autoreload RB	t_C
1	0	0	MODE 1 (PWM) No TxA Toggle	Autoreload RA	Autoreload RB	t_C
0	1	0	MODE 3 (Capture) Captures: TxA Pos. Edge TxB Pos. Edge	Pos. TxA Edge or Timer Underflow	Pos. TxB Edge	t_C
1	1	0	MODE 3 (Capture) Captures: TxA Pos. Edge TxB Neg. Edge	Pos. TxA Edge or Timer Underflow	Neg. TxB Edge	t_C
0	1	1	MODE 3 (Capture) Captures: TxA Neg. Edge TxB Pos. Edge	Neg. TxA Edge or Timer Underflow	Pos. TxB Edge	t_C
1	1	1	MODE 3 (Capture) Captures: TxA Neg. Edge TxB Neg. Edge	Neg. TxA Edge or Timer Underflow	Neg. TxB Edge	t_C

Power Save Modes

The device offer the user two power save modes of operation: HALT and IDLE. In the HALT mode, all microcontroller activities are stopped. In the IDLE mode, the on-board oscillator circuitry and timer T0 are active but all other microcontroller activities are stopped. In either mode, all on-board RAM, registers, I/O states, and timers (with the exception of T0) are unaltered.

HALT MODE

The device is placed in the HALT mode by writing a "1" to the HALT flag (G7 data bit). All microcontroller activities, including the clock, and timers, are stopped. In the HALT mode, the power requirements of the device are minimal and the applied voltage (V_{CC}) may be decreased to Vr (Vr = 2.0V) without altering the state of the machine.

CAN HALT/IDLE mode:

In order to reduce the device overall current consumption in HALT/IDLE mode a two step power save mechanism is implemented on the device:

Step 1: Disable main receive comparator. This is done by resetting both the TxEN0 and TxEN1 bits in the CBUS register. Note: These bits should always be reset before entering HALT/IDLE mode to allow proper resynchronization to the CAN bus after exiting HALT/IDLE mode.

Step 2: Disable the CAN wake-up comparators, this is done by resetting bit 7 in the port-m wakeup enable register (MWKEN) a transition on the CAN bus will then not wake the device up.

Note: If both the main receive comparator and the wake-up comparator are disabled the on chip CAN voltage reference is also disabled. The CAN-V_{REF} output is then High-Z

The following table shows the two CAN power save modes and the active CAN transceiver blocks:

Step 1	Step 2	Main-Comp	Wake-Up-Comp	CAN-V_{REF}	V_{REF} Pin
0	0	on	on	on	$V_{CC}/2$
0	1	on	off	on	$V_{CC}/2$
1	0	off	on	on	$V_{CC}/2$
1	1	off	off	off	High-Z

5

Power Save Modes (Continued)

The device supports two different ways of exiting the HALT mode. The first method of exiting the HALT mode is with the Multi-Input Wakeup feature on the L & M port. The second method of exiting the HALT mode is by pulling the RESET pin low.

Since a crystal or ceramic resonator may be selected as the oscillator, the Wakeup signal is not allowed to start the chip running immediately since crystal oscillators and ceramic resonators have a delayed start up time to reach full amplitude and frequency stability. The IDLE timer is used to generate a fixed delay to ensure that the oscillator has indeed stabilized before allowing instruction execution. In this case, upon detecting a valid Wakeup signal, only the oscillator circuitry is enabled. The IDLE timer is loaded with a value of 256 and is clocked with the t_c instruction cycle clock. The t_c clock is derived by dividing the oscillator clock down by a factor of 10. The Schmitt trigger following the CKI inverter on the chip ensures that the IDLE timer is clocked only when the oscillator has a sufficiently large amplitude to meet the Schmitt trigger specifications. This Schmitt trigger is not part of the oscillator closed loop. The start-up time-out from the IDLE timer enables the clock signals to be routed to the rest of the chip.

The device has two mask options associated with the HALT mode. The first mask option enables the HALT mode feature, while the second mask option disables the HALT mode. With the HALT mode enable mask option, the device will enter and exit the HALT mode as described above. With the HALT disable mask option, the device cannot be placed in the HALT mode (writing a "1" to the HALT flag will have to effect).

IDLE MODE

The device is placed in the IDLE mode by wirting a "1" to the IDLE flag (G6 data bit). In this mode, all activity, except the associated on-board oscillator circuitry, ad the IDLE Timer T0, is stopped. The power supply requirements of the microcontroller in this mode of operation are typically around 30% of normal power requirement of the microcontroller.

As with the HALT mode, the device can be returned to normal operation with a reset, or with a Multi-Input Wakeup from the Port L or CAN Interface. Alternately, the microcontroller resumes normal operation from the IDLE mode when the thirteenth bit (representing 4.096 ms at internal clock frequency of 1 MHz, t_c = 1 μs) of the IDLE Timer toggles.

This toggle condition of the thirteenth bit of the IDLE Timer T0 is latched into the T0PND pending flag.

The user has the option of being interrupted with a transition on the thirteenth bit of the IDLE Timer T0. The interrupt can be enabled or disabled via the T0EN control bit. Setting the T0EN flag enables the interrupt and vice versa.

The user can enter the IDLE mode with the Timer T0 interrupt enabled. In this case, when the T0PND bit gets set, the device will first execute the Timer T0 interrupt service routine and then return to the instruciton following the "Enter Idle Mode" instruction.

Alternatively, the user can enter the IDLE mode with the IDLE Timer T0 interrupt disabled. In this case, the device will resume normal operation with the instruction immediately following the "Enter IDLE Mode" instruction.

Note: It is necessary to program two NOP instructions following both the set HALT mode and set IDLE mode instructions. These NOP instructions are necessary to allow clock resynchronization following the HALT or IDLE modes.

Multi-Input Wakeup

The Multi-Input Wakeup feature is used to return (wakeup) the device from either the HALT or IDLE modes. Alternately, the Multi-Input Wakeup/Interrupt feature may also be used to generate up to 7 edge selectable external interrupts.

Note: The following description is for both the Port L and the M port. When the document refers to the registers WKEGD, WKEN or WKPND, the user will have to put either M (for M port) or L (for port) in front of the register, i.e., LWKEN (Port L WKEN), MWKEN (Port M WKEN).

Figures 12 and *13* shows the Multi-Input Wakeup logic for the microcontroller. The Multi-Input Wakeup feature utilizes the L Port. The user selects which particular Port L bit (or combination of Port L bits) will cause the device to exit the HALT or IDLE modes. The selection is done through the Reg: WKEN. The Reg: WKEN is an 8-bit read/write register, which contains a control bit for every Port L bit. Setting a particular WKEN bit enables a Wakeup from the associated Port L pin.

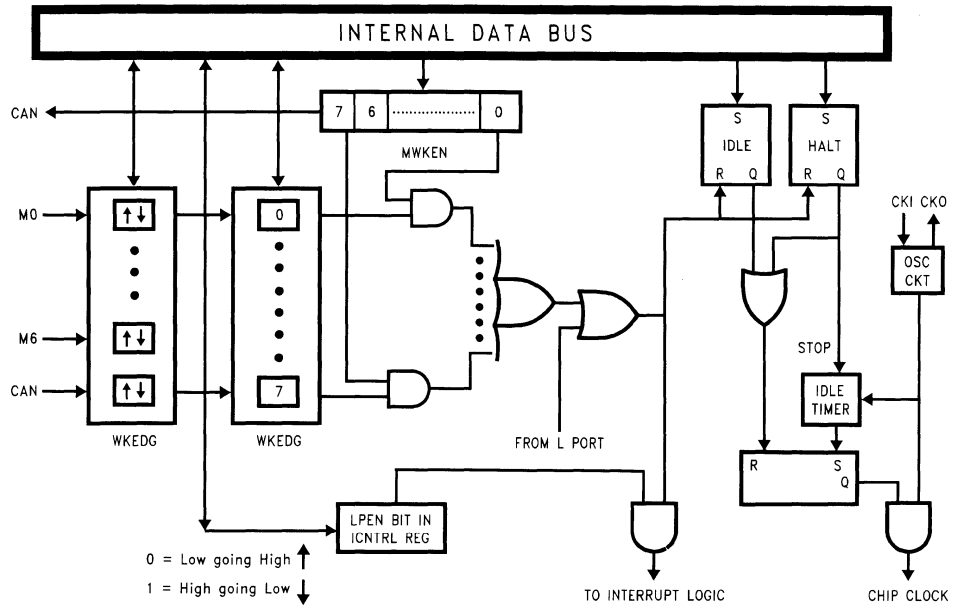
The user can select whether the trigger condition on the selected Port L pin is going to be either a positive edge (low to high transition) or a negative edge (high to low transition). This selection is made via the Reg: WKEDG, which is an 8-bit control register with a bit assigned to each Port L pin. Setting the control bit will select the trigger condition to be a negative edge on that particular Port L pin. Resetting the bit selects the trigger condition to be a positive edge. Changing an edge select entails several steps in order to avoid a pseudo Wakeup condition as a result of the edge change. First, the associated WKEN bit should be reset, followed by the edge select change in WKEDG. Next, the associated WKPND bit should be cleared, followed by the associated WKEN bit being re-enabled.

An example may serve to clarify this procedure. Suppose we wish to change the edge select from positive (low going high) to negative (high going low) for Port L bit 5, where bit 5 has previously been enabled for an input interrupt. The program would be as follows:

```
RBIT 5, WKEN   ;Disable Port bit 5 for
                    wake-up
SBIT 5, WKEDG  ;Select neg-rising edge
RBIT 5, WKPND  ;Clear pending bit
SBIT 5, WKEN   ;Re-enable the bit
```

If the Port L bits have been used as outputs and then changed to inputs with Multi-Input Wakeup/Interrupt, a safety procedure should also be followed to avoid inherited pseudo wakeup conditions. After the selected Port L bits have been changed from output to input but before the associated WKEN bits are enabled, the associated edge select bits in WKEDG should be set or reset for the desired edge selects, followed by the assoicated WKPND bits being cleared.

Multi-Input Wakeup (Continued)



TL/DD/12837–13

FIGURE 12. Port M Multi-Input Wake-up Logic

This same procedure should be used following reset, since the Port L inputs are left floating as a result of reset. The occurrence of the selected trigger condition for Multi-Input Wakeup is latched to a pending register called WKPND. The respective bits of the WKPND register will be set on the occurrence of the selected trigger edge on the corresponding Port L and Port M pin. The user has the responsibility of clearing these pending flags. Since WKPND is a pending register for the occurrence of selected wakeup conditions, the device will not enter the HALT mode if any wakeup bit is both enabled and pending. Consequently, the user has the responsibility of clearing the pending flags before attempting to enter the HALT mode.

The WKEN, WKPND and WKEDG are all read/write registers, and are cleared at reset.

Multi-Input Wakeup (Continued)

PORT L INTERRUPTS

Port L provides the user with additional eight fully selectable, edge sensitive interrupts which are all vectored into the same service subroutine.

The interrupt from Port L shares logic with the wake up circuitry. The register WKEN allows interrupts from Port L to be individually enabled or disabled. The register WKEDG specifies the trigger condition to be either a positive or a negative edge. Finally, the register WKPND latches in the pending trigger conditions.

The GIE (global interrupt enable) bit enables the interrupt function. A control flag, LPEN, functions as a global interrupt enable for Port L interrupts. Setting the LPEN flag will enable interrupts and vice versa. A separate global pending flag is not needed since the register WKPND is adequate.

Since Port L is also used for waking the device out of the HALT or IDLE modes, the user can elect to exit the HALT or IDLE modes either with or without the interrupt enabled. If he elects to disable the interrupt, then the device will restart execution from the instruction immediately following the instruction that placed the microcontroller in the HALT or IDLE modes. In the other case, the device will first execute the interrupt service routine and then revert to normal operation.

The Wakeup signal will not start the chip running immediately since crystal oscillators or ceramic resonators have a finite start up time. The IDLE Timer (T0) generates a fixed delay to ensure that the oscillator has indeed stabilized before allowing the device to execute instructions. In this case, upon detecting a valid Wakeup signal, only the oscillator circuitry and the IDLE Timer T0 are enabled. The IDLE Timer is loaded with a value of 256 and is clocked from the t_C instruction cycle clock. The t_C clock is derived by dividing down the oscillator clock by a factor of 10. A Schmitt trigger following the CKI on-chip inverter ensures that the IDLE timer is clocked only when the oscillator has a sufficiently large amplitude to meet the Schmitt trigger specifications. This Schmitt trigger is not part of the oscillator closed loop. The start-up time-out from the IDLE timer enables the clock signals to be routed to the rest of the chip.

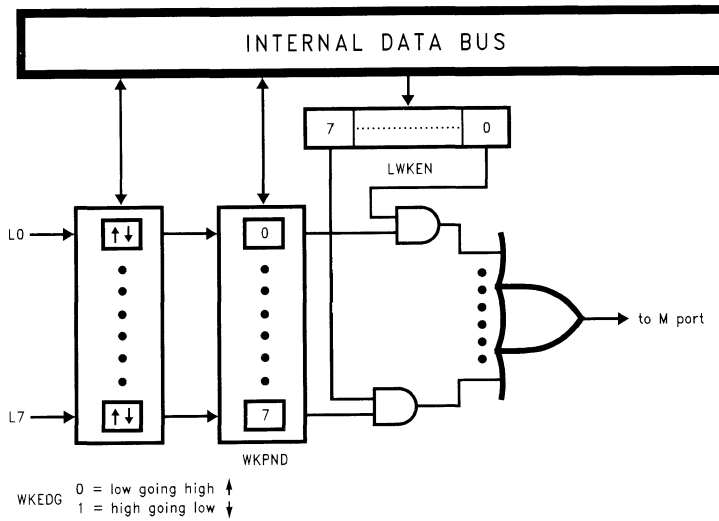


FIGURE 13. Port L Multi-Input Wake-Up Logic

Multi-Input Wakeup (Continued)

PORT M INTERRUPTS

Port M provides the user with seven fully selectable, edge sensitive interrupts which are all vectored into the same service subroutine.

The interrupt from Port M shares logic with the wake up circuitry. The MWKEN register allows interrupts from Port M to be individually enabled or disabled. The MWKEDG register specifies the trigger condition to be either a positive or a negative edge. The MWKPND register latches in the pending trigger conditions.

The LPEN control flag in the ICNTRL register functions as a global interrupt enable for Port M interrupts. Setting the LPEN flag enables interrupts. Note that the GIE bit in the PSW register must also be set to enable these Port L interrupts. A global pending flag is not needed since each pin has a corresponding pending flag in the MWKPND register.

Since Port M is also used for exiting the device from the HALT or IDLE mode, the user can elect to exit the HALT or IDLE mode either with or without the interrupt enabled. If the user elects to disable the interrupt, then the device restarts execution from the point at which it was stopped (first instruction cycle of the instruction following the enter HALT or IDLE mode instruction). In the other case, the device finishes the instruction which was being executed when the part was stopped (the NOP(1) instruction following the enter HALT or IDLE mode instruction), and then branches to the interrupt service routine. The device then reverts to normal operation.

Note 1: The user must plsce two NOPs after an enter HALT or IDLE mode instruction.

To prevent erroneous clearing of the SPI receive FIFO when entering HALT/IDLE mode, the user needs to enable the MIWU on port M3. ($\overline{SS}$) by setting bit 3 in the MWKEN register.

CAN RECEIVE WAKEUP

The CAN Receive Wakeup source can be enabled or disabled. There is no specific enable bit for the CAN Wakeup feature. Although the wakeup feature on pins L0..17 and M0..M7 can be programmed to generate an interrupt (Port L or Port M interrupt), no interrupt is generated upon a CAN receive wakeup condition. The CAN block has it's own, dedicated receiver interrupt upon receive buffer full (see CAN Section).

CAN Wake-Up:

The CAN interface can be programmed to wake the device from HALT/IDLE mode. This is done by setting bit 7 in the Port M wake-up enable register (MWKEN). A transition on the bus will cause the bit 7 of the Port M wake-up pending (MWKPND) to be set and thereby waking up the device. The frame on the CAN bus will be lost. The MWEDG (m port wake-up edge) register bit 7 can be programmed high or low (high will wake-up on the first falling edge on Rx0).

Resetting bit 7 in the MWKEN will disable the CAN wake-up.

The following sequence should be executed before entering HALT/IDLE mode:

```
RBIT 7, MWKPND; ;clear CAN wake-up pending
LD A, CBUS
AND A, #0CF    ; ;resetTxEN0 and TxEN1
X  A, CBUS     ; ;disable main receive
                 comparator
```

After the device woke-up the CBUS bits TxEN0 and/or TxEN1 need be set to allow synchronization on the bus and to enable transmission/reception of CAN frames.

Interrupts

The device supports a vectored interrupt scheme. It supports a total of fourteen interrupt sources. The following table lists all the possible device interrupt sources, their arbitration ranking and the memory locations reserved for the interrupt vector for each source.

Two bytes of program memory space are reserved for each interrupt source. All interrupt sources except the software interrupt are maskable. Each of the maskable interrupts have an Enable bit and a Pending bit. A maskable interrupt is active if its associated enable and pending bits are set. If GIE = 1 and an interrupt is active, then the processor will be interrupted as soon as it is ready to start executing an instruction except if the above conditions happen during the Software Trap service routine. This exception is described in the Software Trap sub-section.

The interruption process is accomplished with the INTR instruction (opcode 00), which is jammed inside the instruction Register and replaces the opcode about to be executed. The following steps are performed for every interrupt:

1. The GIE (Global Interrupt Enable) bit is reset.
2. The address of the instruction about to be executed is pushed into the stack.
3. The PC (Program Counter) branches to address 00FF. This procedure takes 7 t_C cycles to execute.

At this time, since GIE = 0, other maskable interrupts are disabled. The user is now free to do whatever context switching is required by saving the context of the machine in the stack with PUSH instructions. The user would then program a VIS (Vector Interrupt Select) instruction in order to branch to the interrupt service routine of the highest priority interrupt enabled and pending at the time of the VIS. Note that this is not necessarily the interrupt that caused the branch to address location 00FF Hex prior to the context switching.

Thus, if an interrupt with a higher rank than the one which caused the interruption becomes active before the decision of which interrupt to service is made by the VIS, then the interrupt with the higher rank will override any lower ones and will be acknowledged. The lower priority interrupt(s) are still pending, however, and will cause another interrupt immediately following the completion of the interrupt service routine associated with the higher priority interrupt just serviced. This lower priority interrupt will occur immediately following the RETI (Return from Interrupt) instruction at the end of the interrupt service routine just completed.

Inside the interrupt service routine, the associated pending bit has to be cleared by software. The RETI (Return from Interrupt) instruction at the end of the interrupt service routine will set the GIE (Global Interrupt Enable) bit, allowing the processor to be interrupted again if another interrupt is active and pending.

Interrupts (Continued)

The VIS instruction looks at all the active interrupts at the time it is executed and performs an indirect jump to the beginning of the service routine of the one with the highest rank.

The addresses of the different interrupt service routines, called vectors, are chosen by the user and stored in ROM in a table starting at 01E0 (assuming that VIS is located between 00FF and 01DF). The vectors are 15-bit wide and therefore occupy 2 ROM locations.

VIS and the vector table must be located in the same 256-byte block (0y00 to 0yFF) except if VIS is located at the last address of a block. In this case, the table must be in the next block. The vector table cannot be inserted in the first 256-byte block.

The vector of the maskable interrupt with the lowest rank is located at 0yE0 (Hi-Order byte) and 0yE1 (Lo-Order byte) and so forth in increasing rank number. The vector of the maskable interrupt with the highest rank is located at 0yFA (Hi-Order byte) and 0yFB (Lo-Order byte).

The Software Trap has the highest rank and its vector is located at 0yFE and 0yFF.

If, by accident, a VIS gets executed and no interrupt is active, then the PC (Program Counter) will branch to a vector located at 0yE0–0yE1.

WARNING:

A Default VIS interrupt handle routine must be present. As a minimum, this handler should confirm that the GIE bit is cleared (this indicates that the interrupt sequence has been taken), take care of any required housekeeping, restore context and return. Some sort of Warm Restart procedure should be implemented. These events can occur without any error on the part of the system designer or programmer.

Note: There is always the possibility of an interrupt occurring during an instruction which is attempting to reset the GIE bit or any other interrupt enable bit. If tis occurs when a single cycle instruction is being used to reset the interrupt enable bit, the interrupt enable bit will be reset but an interrupt may still occur. This is because interrupt processing is started at the same time as the interrupt bit is being reset. To avoid this scenario, the user should always use a two, three, or four cycle instruction to reset interrupt enable bits.

Figure 14 shows the Interrupt Block diagram.

TABLE IV. Interrupt Vector Table

Arbitration Rank	Interrupt Source	Description	Vector Address[a]
1	Software Trap	INTR Instruction	0yFE–0yFF
2	reserved	NMI	0yFC–0yFD
3	CAN Receive	RBF, RFV set	0yFA–0yFB
4	CAN Error (transmit/receive)	TERR, RERR set	0yF8–0yF9
5	CAN Transmit	TBE set	0yF6–0yF7
6	Pin G0 Edge	External	0yF4–0yF5
7	MICROWIRE/PLUS SPI Interface	BUSY Goes Low SRBF or STBE set	0yF2–0yF3
8	Timer T0	Idle Timer Underflow	0yF0–0yF1
9	UART	receive buffer full	0yEE–0yEF
10	UART	transmit buffer empty	0yEC–0yED
11	Timer T2	T2A/Underflow	0yEA–0yEB
12	Timer T2	T2B	0yE8–0yE9
13	Timer T1	T1A/Underflow	0yE6–0yE7
14	Timer T1	T1B	0yE4–0yE5
15	Port L, Port M; MIWU	Port L Edge or Port M Edge	0yE2–0yE3
16	Default VIS Interrupt	VIS Interrupt	0yE0–0yE1

a. y = 1 to 7F, depending on the location of the VIS instruction.

Interrupts (Continued)

SOFTWARE TRAP

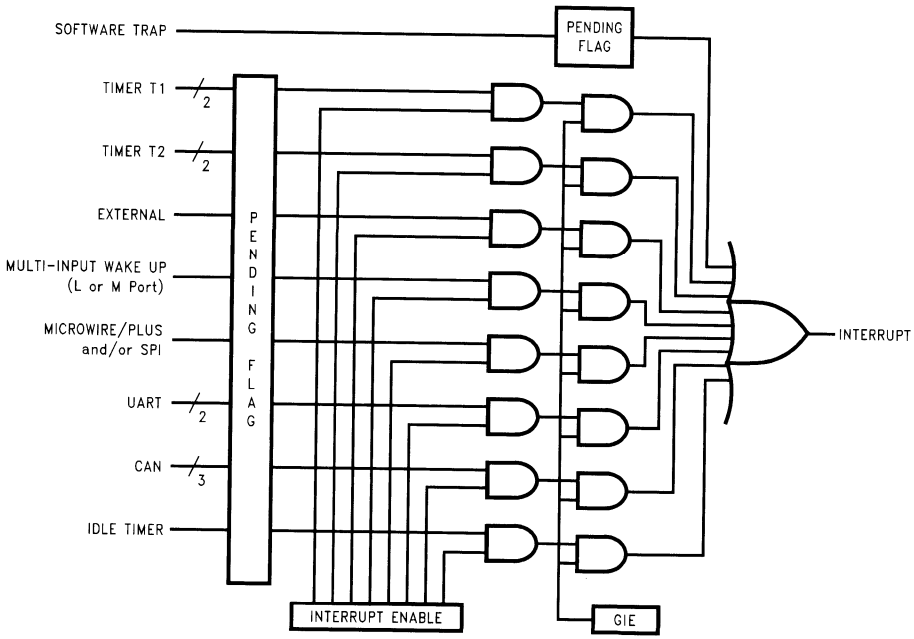
The Software Trap (ST) is a special kind of non-maskable interrupt which occurs when the INTR instruction (used to acknowledge interrupts) is fetched from ROM and placed inside the instruction register. This may happen when the PC is pointing beyond the available ROM address space or when the stack is over-popped.

When an ST occurs, the user can re-initialize the stack pointer and do a recovery procedure (similar to RESET, but not necessarily containing all of the same initialization procedures) before restarting.

The occurrence of an ST is latched into the ST pending bit. The GIE bit is not affected and the ST pending bit (**not accessible by the user**) is used to inhibit other interrupts and to direct the program to the ST service routine with the VIS instruciton. The RPND instruction is used to clear the software interrupt pending bit. This bit is also cleared on reset.

The ST as the highest rank among all interrupts.

Nothing (except another ST) can interrupt an ST being serviced.



TL/DD/12837–15

FIGURE 14. Interrupt Block Diagram

CAN Block Description *

This device contains a CAN serial bus interface as described in the CAN Specification Rev. 2.0 part B.

*Patents Pending.

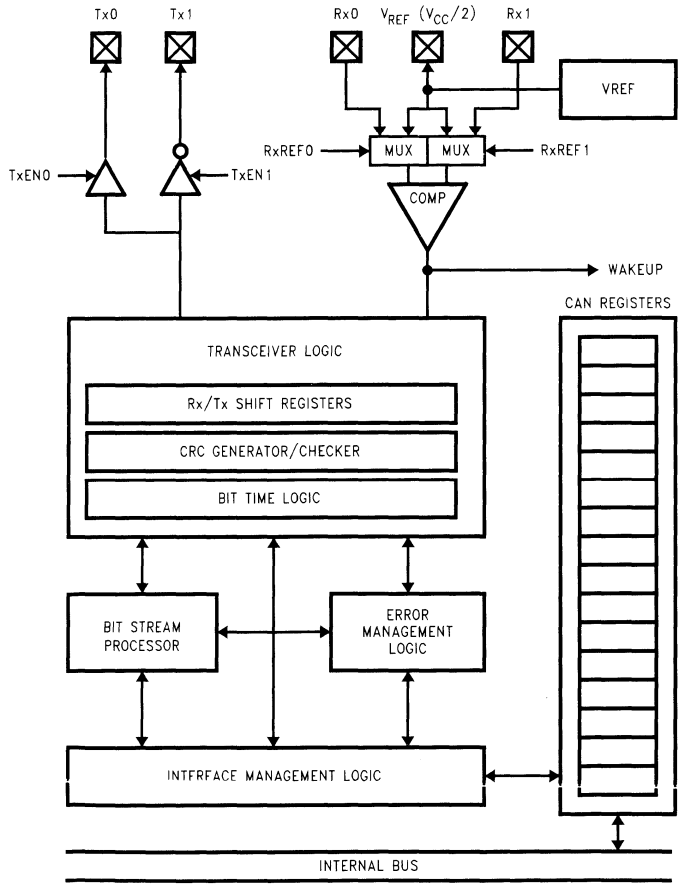
CAN Interface Block

This device supports applications which require a low speed CAN interface. It is designed to be programmed with two transmit and two receive registers. The user's program may check the status bytes in order to get information of the bus state and the received or transmitted messages. The device has the capability to generate an interrupt as soon as one byte has been transmitted or received. Care must be taken if more than two bytes in a message frame are to be transmitted/received. In this case the user's program must poll the transmit buffer empty (TBE)/receive buffer full (RBF) bits or enable their respective interrupts and perform a data exchange between the user data and the Tx/Rx registers.

Fully automatic transmission on error is supported for messages not longer than two bytes. Messages which are longer than two bytes have to be processed by software.

The interface is compatible with CAN Specification 2.0 part B, without the capability to receive/transmit extended frames. Extended frames on the bus are checked and acknowledged according to the CAN specification.

The maximum bus speed achievable with the CAN interface is a function of crystal frequency, message length and software overhead. The device can support a bus speed of up to 1 Mbit/s with a 10 MHz oscillator and 2 byte messages. The 1 Mbit/s bus speed refers to the rate at which protocol and data bits are transferred on the bus. Longer messages require slower bus speeds due to the time required for software intervention between data bytes. The device will support a maximum of 125k bit/s with eight byte messages and a 10 MHz oscillator.



TL/DD/12837–16

FIGURE 15. CAN Interface Block Diagram

Functional Block Description of the CAN Interface

Interface Management Logic (IML)

The IML executes the CPU's transmission and reception commands and controls the data transfer between CPU, Rx/Tx and CAN registers. It provides the CAN Interface with Rx/Tx data from the memory mapped Register Block. It also sets and resets the CAN status information and generates interrupts to the CPU.

Bit Stream Processor (BSP)

The BSP is a sequencer controlling the data stream between The Interface Management Logic (parallel data) and the bus line (serial data). It controls the transceive logic with regard to reception and arbitration, and creates error signals according to the bus specification

Transceive Logic (TCL)

The TCL is a state machine which incorporates the bit stuff logic and controls the output drivers, CRC logic and the Rx/Tx shift registers. It also controls the synchronization to the bus with the CAN clock signal generated by the BTL.

Error Management Logic (EML)

The EML is responsible for the fault confinement of the CAN protocol. It is also responsible for changing the error counters, setting the appropriate error flag bits and interrupts and changing the error status (passive, active and bus off).

Cyclic Redundancy Check (CRC) Generator and Register

The CRC Generator consists of a 15-bit shift register and the logic required to generate the checksum of the destuffed bit-stream. It informs the EML about the result of a receiver checksum.

The checksum is generated by the polynomial:

$$\chi^{15} + \chi^{14} + \chi^{10} + \chi^{8} + \chi^{7} + \chi^{4} + \chi^{3} + 1$$

Receive/Transmit (Rx/Tx) Registers

The Rx/Tx registers are 8-bit shift registers controlled by the TCL and the BSP. They are loaded or read by the Interface Management Logic, which holds the data to be transmitted or the data that was received.

Bit Time Logic (BTL)

The bit time logic divider divides the CKI input clock by the value defined in the CAN prescaler (CSCAL) and bus timing register (CTIM). The resultig bit time (tcan) can be computed by the formula:

$$t_{can} = \frac{CKI}{(1 + divider) \times (1 + 2 \times PS + PPS)}$$

Where *divider* is the value of the clock prescaler, *PS* is the programmable value of phase segment 1 and 2 (1..8) and *PPS* the programmed value of the propagation segment (1..8) (located in CTIM).

Bus Timing Considerations

The internal architecture of the CAN interface has been optimized to allow fast software response times within messages of more than two data bytes. The TBE (Transmit Buffer Empty) bit is set on the last bit of odd data bytes when CAN internal sample points are high.

It is the user's responsibility to ensure that the time between setting TBE and a reload of TxD2 is longer than the length of phase segment 2 as indicated in the following equation:

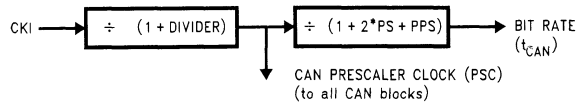
$$t_{LOAD} > \frac{(PS + 1) \times (CSCAL + 1)}{10} t_C = \text{absolute length of PS2}$$

Table V shows examples of the minimum required t_{LOAD} for different CSCAL settings based on a clock frequency of 10 MHz. Lower clock speeds require recalculation of the CAN bit rate and the mimimum t_{LOAD}.

TABLE V. CAN Timing (CKI = 10 MHz, t_c = 1 μs)

PS	CSCAL	CAN Bit Rate (kbit/s)	Minimum t_{LOAD} (μs)
4	3	250	2.0
4	9	100	5.0
4	15	62	8.0
4	24	40	12.5
4	39	25	20
4	99	10	50
4	199	5	100

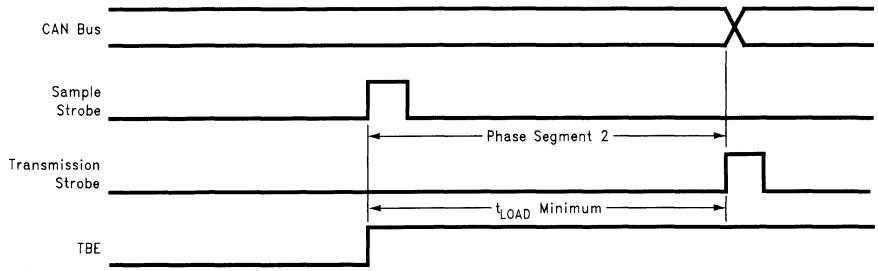
Functional Block Description of the CAN Interface (Continued)



TL/DD/12837–17

FIGURE 16. Bit Rate Generation

Figure 17 illustrates the minimum time required for t_{LOAD}.



TL/DD/12837–18

FIGURE 17. TBE Timing

In the case of an interrupt driven CAN interface, the calculation of the actual t_{LOAD} time would be done as follows:

```
INT:                                    ;Interrupt latency = 7tc = 7 µs
                PUSH        A           ;3tc = 3 µs
                LD          A,B         ;2tc = 2 µs
                PUSH        A           ;3tc = 3 µs
                VIS                     ;5tc = 5 µs
CANTX:                                  ;20tc = µs to this point
                •                       ;additional time for instructions which check
                •                       ;status prior to reloading the transmit data
                •                       ;registers with subsequent data bytes.
                LD          TXD2,DATA
                •
                •
                •
```

Functional Block Description of the CAN Interface (Continued)

Interrupt driven programs use more time than programs which poll the TBE flag, however programs which operate at lower baud rates (which are more likely to be sensitive to this issue) have more time for interrupt response.

Output Drivers/Input Comparators

The output drivers/input comparators are the physical interface to the bus. Control bits are provided to TRI-STATE the output drivers.

A dominant bit on the bus is represented as a "0" in the data registers and a recessive bit on the bus is represented as a "1" in the data registers.

TABLE VI. Bus Level Definition

Bus Level	Pin Tx0	Pin Tx1	Data
"dominant"	drive low (GND)	dirve high (V_{CC})	0
"recessive"	TRI-STATE	TRI-STATE	1

Register Block

The register block consists of fifteen 8-bit registers which are described in more detail in the following paragraphs.

Note: The contents of the receiver related registers RxD1, RxD2, RDLC, RIDH and RTSTAT are only changed if a received frame passes the acceptance filter or the Receive Identifier Acceptance Filter bit (RIAF) is set to accept all received messages.

TRANSMIT DATA REGISTER 1 (TXD1) (Address X'00A0)

The Transmit Data Register 1 contians the first data byte to be transmitted within a frame and then the successive odd byte numbers (i.e., bytes number 1,3,..,7).

TRANSMIT DATA REGISTER 2 (TXD2) (Address X'00A1)

The Transit Data Register 2 contains the second data byte to be transmitted within a frame and then the successive even byte numbers (i.e., bytes number 2,4,..,8).

TRANSMIT DATA LENGTH CODE AND IDENTIFIER LOW REGISTER (TDLC) (Address X'00A2)

TID3	TID2	TID1	TID0	TDLC3	TDLC2	TCLC1	TDLC0

Bit 7 Bit 0

This register is read/write.

TID3..TIDO Transmit Identifier Bits 3..0 (lower 4 bits)

The transmit identifier is composed of eleven bits in total, bits 3 to 0 of the TID are stored in bits 7 to 4 of this register.

TDLC3..TDLC0 Transmit Data Length Code

These bits determine the number of data bytes to be transmitted within a frame. The CAN specification allows a maximum of eight data bytes in any message.

TRANSMIT IDENTIFIER HIGH (TID) (Address X'00A3)

TRTR	TID10	TID9	TID8	TID7	TID6	TID5	TID4

Bit 7 Bit 0

This register is read/write.

TRTR Transmit Remote Frame Request

This bit is set if the frame to be transmitted is a remote frame request.

TID10..TID4 Transmit Identifier Bits 10 .. 4 (higher 7 bits)

Bits TID10..TID4 are the upper 7 bits of the 11 bit transmit identifier.

RECEIVER DATA REGISTER 1 (RXD1) (Address X'00A4)

The Receive Data Register 1 (RXD1) contains the first data byte received in a frame and then successive odd byte numbers (i.e., bytes 1, 3,..7). This register is read-only.

RECEIVE DATA REGISTER 2 (RXD2) (Address X'00A5)

The Receive Data Register 2 (RXD2) contains the second data byte received in a frame and then successive even byte numbers (i.e., bytes 2,4,..,8). This register is read-only.

REGISTER DATA LENGTH CODE AND IDENTIFIER LOW REGISTER (RIDL) (Address X'00A6)

RID3	RID2	RID1	RID0	RDLC3	RDLC2	RDLC1	RDLC0

Bit 7 Bit 0

This register is read only.

RID3..RID0 Receive Identifier bits (lower four bits)

The RID3..RID0 bits are the lower four bits of the eleven bit long Receive Identifier. Any received message that matches the upper 7 bits of the Receive Identifier (RID10..RID4) is accepted if the Receive Identifier Acceptance Filter (RIAF) bit is set to zero.

RDLC3..RDLC0 Receive Data Length Code bits

The RDLC3..RDLC0 bits determine the number of data bytes within a received frame.

RECEIVE IDENTIFIER HIGH (RID) (Address X'00A7)

unused	RID10	RID9	RID8	RID7	RID6	RID5	RID4

Bit 7 Bit 0

This register is read/write.

RID10..RID4 Receive Identifier bits (upper bits)

The RID10...RID4 bits are the upper 7 bits of the eleven bit long Receive Identifier. If the Receive Identifier Acceptance Filter (RIAF) bit (see CBUS register) is set to zero, bits 4 to 10 of the received identifier are compared with the mask bits of RID4..RID10. If the corresponding bits match, the message is accepted. If the RIAF bit is set to a one, the filter function is disabled and all messages, independent of identifier, will be accepted.

Functional Block Description of the CAN Interface (Continued)

CAN PRESCALER REGISTER (CSCAL) (Address X'00A8)

CKS7	CKS6	CKS5	CKS4	CKS3	CKS2	CKS1	CKS0

Bit 7 Bit 0

This register is read/write.

CKS7..0 Prescaler divider select.

The resulting clock value is the CAN Prescaler clock.

CAN BUS TIMING REGISTER (CTIM) (00A9)

PPS2	PPS1	PPS0	PPS0	PS2	PS1	PS0	Reserved

Bit 7 Bit 0

This register is read/write.

PPS2..PPS0 Propagation Segment, bits 2..0

The PPS2..PPS0 bits determine the length of the propagation delay in Prescaler clock cycles (PSC) per bit time. (For a more detailed discussion of propagation delay and phase segments, see SYNCHRONIZATION on page 41.)

PS2..PS0 Phase Segment 1, bits 2..0

The PS2..PS0 bits fix the number of Prescaler clock cycles per bit time for phase segment 1 and phase segment 2. The PS2..PS0 bits also set the synchronization Jump Width to a value equal to the lesser of the 4 PSC or the length of PS1/2 (Min: 4 I length of PS1/2).

TABLE VII. Synchronization Jump Width

PS2	PS1	PS0	Length of Phase Segment ½	Synchronization Jump Width
0	0	0	1 t_{can}	1 t_{can}
0	0	1	2 t_{can}	2 t_{can}
0	1	0	3 t_{can}	3 t_{can}
0	1	1	4 t_{can}	4 t_{can}
1	0	0	5 t_{can}	4 t_{can}
1	0	1	6 t_{can}	4 t_{can}
1	1	0	7 t_{can}	4 t_{can}
1	1	1	8 t_{can}	4 t_{can}

LENGTH OF TIME SEGMENTS (See *Figure 29*)

- The Synchronization Segment is 1 CAN Prescaler clock (PSC)
- The Propagation Segment can be programmed (PPS) to be 1,2...,8 PSC in length.
- Phase Segment 1 and Phase Segment 2 are programmable (PS) to be 1,2,..,8 PSC long.

Note: (BTL settings at high speed; PSC = 0) Due to the on-chip delay from the rx-pins through the receive comparator (worst case assumption: 3 clocks delay * 2 (devices on the bus) + 1 tx delay) the user needs to set the sample point to > (2*3 + 1) i.e., > 7 CKI clocks to ensure correct communication on the bus under all circumstances. With prescaler settings of > 0 this is a given (i.e., no caution has to be applied).

Example: for 1 Mbit CTIM = b'10000100 (PSS = 5; PS1 = 2). Example for 500 kbit CTIM = b'01011100 (PPS = 3; PS1 = 8). – all at 10 MHz CKI and CSCAL = 0.

CAN BUS CONTROL REGISTER (CBUS) (00AA)

Reserved	RIAF	TxEN1	TxEN0	RxREF1	RxREF0	Reserved	FMOD

Bit 7 Bit 0

Reserved This bit is reserved and should be zero.

RIAF Receive identifier acceptance filter bit

If the RIAF bit is set to zero, bits 4 to 10 of the received identifier are compared with the mask bits of RID4..RID10 and if the corresponding bits match, the message is accepted. If the RIAF bit is set to a one, the filter function is disabled and all messages independent of the identifier will be accepted.

TxEN0, TxEN1 TxD Output Driver Enable

TABLE VIII. Output Drivers

TxEN1	TxEN0	Output
0	0	Tx0, Tx1 TRI-STATE, CAN input comparator disabled
0	1	Tx0 enabled
1	0	Tx1 enabled
1	1	Tx0 and Tx1 enabled

Bus synchronization of the device is done in the following way:

If the output was disabled (TxEN1, TxEN0 = "0") and either TxEN1 or TxEN0, or both are set to 1, the device will not start transmission or reception of a frame until eleven consecutive "recessive" bits have been received. Resetting the TxEN1 and TxEN0 bits will disable the output drivers and the CAN input comparator. All other CAN related registers and flags will be unaffected. It is recommended that the user reset the TxEN1 and TxEN0 bits before switching the device into the HALT mode (the CAN receive wakeup will still work) in order to reduce current consumption and to assure a proper resychronization to the bus after exiting the HALT mode.

Note: A "bus off" condition will also cause Tx0 and Tx1 to be at TRI-STATE (independent of the values of the TxEN1 and TxEN0 bits).

RXREF1 Reference voltage applied to Rx1 if bit is set

RXREF0 Reference voltage applied to Rx0 if bit is set

FMOD Fault Confinement Mode select

Setting the FMOD bit to "0" (default after power on reset) will select the Standard Fault Confinement mode. In this mode the device goes from "bus off" to "error active" after monitoring 128*11 recessive bits (including bus idle) on the bus. This mode has been implemented for compatibility with existing solutions. Setting the FMOD bit to "1" will select the Enhanced Fault Confinement mode. In this mode the device goes from "bus off" to "error active" after monitoring 128 "good" messages, as indicated by the reception of 11 consecutive "recessive" bits including the End of Frame, whereas the standard mode may time out after 128 x 11 recessive bits (e.g., bus idle).

Functional Block Description of the CAN Interface (Continued)

TRANSMIT CONTROL/STATUS (TCNTL) (00AB)

NS1	NS0	TERR	RERR	CEIE	TIE	RIE	TXSS

Bit 7 Bit 0

NS1..NS0 Node Status, i.e., Error Status.

TABLE IX. Node Status

NS1	NS0	Output
0	0	Error active
0	1	Error passive
1	0	Bus off
1	1	Bus off

The Node Status bits are read only.

TERR Transmit Error

This bit is automatically set when an error occurs during the transmission of a frame. TERR can be programmed to generate an interrupt by setting the Can Error Interrupt Enable bit (CEIE). This bit must be cleared by the user's software.

Note: This is used for messages for more than two bytes. If an error occurs during the transmission of a frame with more than 2 data bytes, the user's software has to handle the correct reloading of the data bytes to the TxD registers for retransmission of the frame. For frames with 2 or less data bytes the interface logic of this chip does an automatic retransmission. Regardless of the number of data bytes, the user's software must reset this bit if CEIE is enabled. Otherwise a new interrupt will be generated immediately after return from the interrupt service routine.

RERR Receiver Error

This bit is automatically set when an error occurred during the reception of a frame. RERR can be programmed to generate an interrupt by setting the Can Error Interrupt Enable bit (CEIE). This bit has to be cleared by the user's software.

CEIE CAN Error Interrupt Enable

If set by the user's software, this bit enables the tansmit and receive error interrupts. The interrupt pending flags are TERR and RERR. Resetting this bit with a pending error interrupt will inhibit the interrupt, but will not clear the cause of the interrupt (RERR or TERR). If the bit is then set without clearing the cause of the interrupt, the interrupt will reoccur.

TIE Transmit Interrupt Enable

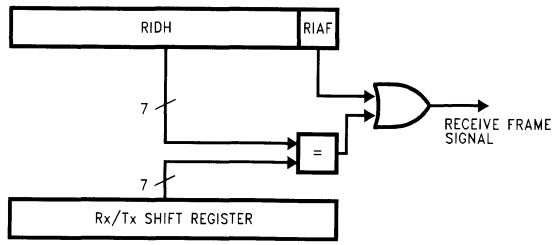
If set by the user's software, this bit enables the transmit interrupt. (See TBE and TXPND.) Resetting this bit with a pending transmit interrupt will inhibit the interrupt, but will not clear the cause of the interrupt. If the bit is then set without clearing the cause of the interrupt, the interrupt will reoccur.

RIE Receive Interrupt Enable

If set by the user's software, this bit enables the receive interrupt or a remote transmission request interrupt (see RBF, RFV and RRTR). Resetting this bit with a pending receive interrupt will inhibit the interrupt, but will not clear the cause of the interrupt. If the bit is then set without clearing the cause of the interrupt, the interrupt will reoccur.

TXSS Transmission Start/Stop

This bit is set by the user's software to initiate the transmission of a frame. Once this bit is set, a transmission is pending, as indicated by the TXPND flag being set. It can be reset by software to cancel a pending transmission. Resetting the TXSS bit will only cancel a transmission, if the transmission of a frame hasn't been started yet (bus idle), if arbitration has been lost (receiving) or if an error occurs during transmission. If the device has already started transmission (won arbitration) the TXPND and TXSS flags will stay set until the transmission is completed, even if the user's software has written zero to the TXSS bit. If one or more data bytes are to be transmitted, care must be taken by the user, that the Transmit Data Register(s) have been loaded before the TXSS bit is set. TXSS will be cleared on three conditions only: Successful completion of a transmitted message; successful cancellation of a pending transmision; Transition of the CAN interface to the bus-off state.



TL/DD/12837–19

FIGURE 18. Acceptance Filter Block-Diagram

Functional Block Description of the CAN Interface (Continued)

Writing a zero to the TXSS bit will request cancellation of a pending transmission but TXSS will not be cleared until completion of the operation. If an error occurs during transmission of a frame, the logic will check for cancellation requests prior to restarting transmission. If zero has been written to TXSS, retransmission will be canceled.

RECEIVE/TRANSMIT STATUS (RTSTAT) (Address X'00AC)

TBE	TXPND	RRTR	ROLD	RORN	RFV	RCV	RBF
1	0	0	0	0	0	0	0

Bit 7 Bit 0

This register is read only.

TBE Transmit Buffer Empty

This bit is set as soon as the TxD2 register is copied into the Rx/Tx shift register, i.e., the 1st data byte of each pair has been transmitted. The TBE bit is automatically reset if the TxD2 register is written (the user should write a dummy byte to the TxD2 register when transmitting an odd number of bytes of zero bytes). TBE can be programmed to generate an interrupt by setting the Transmit Interrupt Enable bit (TIE). When servicing the interrupt the user has to make sure that TBE gets cleared by executing a WRITE instruction on the TxD2 register, otherwise a new interrupt will be generated immediately after return from the interrupt service routine. The TBE bit is read only. It is set to 1 upon reset. TBE is also set upon completion of transmission of a valid message.

TXPND Transmission Pending

This bit is set as soon as the Transmit Start/Stop (TXSS) bit is set by the user. It will stay set until the frame was successfully transmitted, until the transmission was successfully canceled by writing zero to the Transmission Start/Stop bit (TXSS), or the device enters the bus-off state. Resetting the TXSS bit will only cancel a transmission if the transmission of a frame hasn't been started yet (bus idle) or if arbitration has been lost (receiving). If the device has already started transmission (won arbitration) the TXPND flag will stay set until the transmission is completed, even if the user's software has requested cancellation of the message. If an error occurs during transmission, a requested cancellation may occur prior to the begining of retransmission.

RRTR Received Remote Transmission Request

This bit is set when the remote transmission request (RTR) bit in a received frame was set. It is automatically reset through a read of the RXD1 register.

To detect RRTR the user can either poll this flag or enable the receive interrupt (the reception of a remote transmission request will also cause an interrupt if the receive interrupt is enabled). If the receive interrupt is enabled, the user should check the RRTR flag in the service routine in order to distinguish between a RRTR interrupt and a RBF interrupt. It is the responsibility of the user to clear this bit by reading the RXD1 register, before the next frame is received.

ROLD Received Overload Frame

This bit is automatically set when an Overload Frame was received on the bus. It is automatically reset through a read of the Receive/Transmit Status register. It is the responsibility of the user to clear this bit by reading the Receive/Transmit Status register, before the next frame is received.

RORN Receiver Overrun

This bit is automatically set on an overrun of the receive data register, i.e., if the user's program does not maintain the RxDn registers when receiving a frame. It it automatically reset through a read of the Receive/Transmit Status register. It is the responsibility of the user to clear this bit by reading the Receive/Transmit Status register before the next frame is received.

RFV Received Frame Valid

This bit is set if the received frame is valid, i.e., after the penultimate bit of the End of Frame is received. It is automatically reset through a read of the Receive/Transmit Status register. It is the responsibility of the user to clear this bit by reading the receive/transmit status register (RTSTAT), before the next frame is received. RFV will cause a Receive Interrupt if enabled by RIE. The user should be careful to read the last data byte (RxD1) of odd length messages (1, 3, 5 or 7 data bytes) on receipt of RFV. RFV is the only indication that the last byte of the message has been received.

RCV Receive Mode

This bit is set after the data length code of a message that passes the device's acceptance filter has been received. It is automatically reset after the CRC-delimiter of the same frame has been received. It indicates to the user's software that arbitration is lost and that data is coming in for that node.

RBF Receive Buffer Full

This bit is set if the second Rx data byte was received. It is reset automatically, after the RxD1-Register has been read by the software. RBF can be programmed to generate an interrupt by setting the Receive Interrupt Enable bit (RIE). When servicing the interrupt, the user has to make sure that RBF gets cleared by executing a LD instruction from the RxD1 register, otherwise a new interrupt will be generated immediately after return from the interrupt service routine. The RBF bit is read only.

TRANSMIT ERROR COUNTER (TEC) (Address X'00AD)

TEC7	TEC6	TEC5	TEC4	TEC3	TEC2	TEC1	TEC0

Bit 7 Bit 0

This register is read/write.

Functional Block Description of the CAN Interface (Continued)

For test purposes and to identify the node status, the transmit error counter, an 8-bit error counter, is mapped into the data memory. If the lower seven bits of the counter overflow, i.e., TEC7 is set, the device is error passive.

CAUTION

To prevent interference with the CAN fault confinement, the user must not write to the REC/TEC registers. Both counters are automatically updated following the CAN specification.

RECEIVE ERROR COUNTER (REC) (00AE)

ROVL	REC6	REC5	REC4	REC3	REC2	REC1	REC0
Bit 7							Bit 0

This register is read/write.

ROVL receive error counter overflow

For test purposes and to identify the node status the receive error counter, a 7-bit error counter, is mapped into the data memory. If the counter overflows the ROVL bit is set to indicate that the device is error passive and won't transmit any active error frames. If ROVL is set then the counter is frozen.

MESSAGE IDENTIFICATION

a. Transmitted Message

The user can select all 11 Transmit Identifier Bits to transmit any message whigh fulfills the CAN2.0, part B spec without an extended identifier (see note below). Fully automatic retransmission is supported for messages no longer than 2 bytes.

b. Received Messages

The lower four bits of the Receive Identifier are don't care, i.e., the controller will receive all messages that fit in that window (16 messages). The upper 7 bits can be defined by the user in the Receive Identifier High Register to mask out groups of messages. If the RIAF bit is set, all messages will be received.

Note: The CAN interface tolerates the extended CAN frame format of 29 identifier bits and gives an acknowledgment. If an error occurs the receive error counter will be increased, and decreased if the frame is valid.

BUS SYNCHRONIZATION DURING OPERATION

Resetting the TxEN1 and TxEN0 bits in Bus Control Register will disable the output drivers and do a resynchronization to the bus. All other CAN related registers and flags will be unaffected.

Bus synchronization of the device is this case is done in the following way:

If the output was disabled (TxEN1, TxEN0 = "0") and either TxEN1 or TxEN0, or both are set to 1, the device will not start transmission or reception of a frame until eleven consecutive "recessive" bits have been received.

A "bus off" condition will also cause the output drivers Tx1 and Tx0 to be at TRI-STATE (independent of the status of TxEN1 and TxEN0). The device will switch from "bus off" to "error active" mode as described under the FMOD-bit description (see Can Bus Control register). This will ensure that the device is synchronized to the bus, before starting to transmit or receive.

For information on bus synchronization and status of the CAN related registers after external reset refer to the RESET section.

ON-CHIP VOLTAGE REFERENCE

The on-chip voltage reference is a ratiometric reference. For electrical characteristics of the voltage reference refer to the electrical specifications section.

ANALOG SWITCHES

Analog switches are used for selecting between Rx0 and V_{REF} and between Rx1 and V_{REF}.

Basic CAN Concepts

The following paragraphs provide a generic overview of the basic concepts of the Controller Area Network (CAN) as described in *Chapter 4 of ISO/DIS11519-1.* Implementation related issues of the National Semiconductor device will be discussed as well.

This device will process standard frame format only. Extended frame formats will be acknowledged, however the data will be discarded. For this reason the description of frame formats in the following section will cover only the standard frame format.

The following section provides some more detail on how the device will handle received extended frames:

If the device's remote identifier acceptance filter bit (RIAF) is set to "1", extended frame messages will be acknowledged. However, the data will be discarded and the device will not reply to a remote transmission request received in extended frame format. If the device's RIAF bit is set to "0", the upper 7 received ID bits of an extended frame that match the device's receive identifier (RID) acceptance filtler bits, are stroed in the device's RID register. However, the device does not reply to an RTR and any data is discarded. The device will only acknowledge the message.

MULTI-MASTER PRIORITY BASED BUS ACCESS

The CAN protocol is message based protocol that allows a total of 2032 ($= 2^{11} - 16$) different messages in the standard format and 512 million ($= 2^{29} - 16$) different messages in the extended frame format.

MULTICAST FRAME TRANSFER BY ACCEPTANCE FILTERING

Every CAN Frame is put on the common bus. Each module receives every frame and filters out the frames which are not required for the module's task.

REMOTE DATA REQUEST

A CAN master module has the ability to set a specific bit called the "remote transmission request bit" (RTR) in a frame. This causes another module, either another master or a slave, to transmit a data frame after the current frame has been completed.

Basic CAN Concepts (Continued)

SYSTEM FLEXIBILITY

Additional modules can be added to an existing network without a configuration change. These modules can either perform completely new functions requiring new data or process existing data to perform a new function.

SYSTEM WIDE DATA CONSISTENCY

As the CAN network is message oriented, a message can be used like a variable which is automatically updated by the controlling processor. If any module cannot process information it can send an overload frame. The device is incapable of initiating an overload frame, but will join a overload frame initiated by another device as required by CAN specifications.

NON-DESTRUCTIVE CONTENTION-BASED ARBITRATION

The CAN protocol allows several transmitting modules to start a transmission at the same time as soon as they monitor the bus to be idle. During the start of transmission every node monitors the bus line to detect whether its message is overwritten by a message with a higher priority. As soon as a transmitting module detects another module with a higher priority accessing the bus, it stops transmitting its own frame and switches to receive mode. For illustration see *Figure 19*.

AUTOMATIC RETRANSMISSION OF FRAMES

If a data or remote frame is overwritten by either a higher-prioritized data frame, remote frame or an error frame, the transmitting module will automatically retransmit it. This device will handle the automatic retransmission of up to two data bytes automatically. Messages with more than 2 data bytes require the user's software to update the transmit registers.

ERROR DETECTION AND ERROR SIGNALING

All messages on the bus are checked by each CAN node and acknowledge if they are correct. If any node detects an error it starts the transmission of an error frame.

Switching Off Defective Nodes

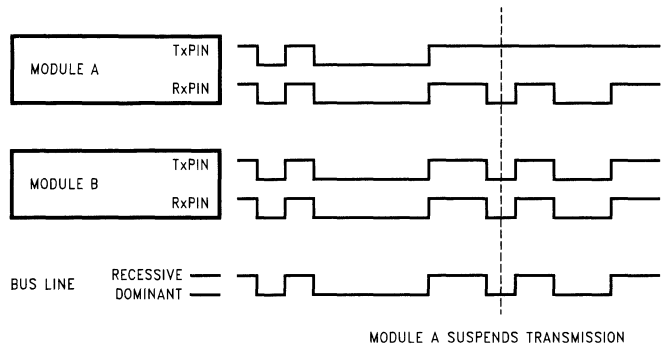
There are two error counters, one for transmitted data and one for received data, which are incremented, depending on the error type, as soon as an error occurs. If either counter goes beyond a specific value the node goes to an error state. A valid frame causes the error counters to decrease.

The device can be in one of three states with respect to error handling:

- Error active
  An error active unit can participate in bus communication and sends an active ("dominant") error flag.
- Error passive
  An error passive unit can participate in bus communication. However, if the unit detects an error it is not allowed to send an active error flag. The unit sends only a passive ("recessive") error flag.
- Bus off

A unit that is "bus off" has the output drivers disabled, i.e., it does not participate in any bus activity.

(See ERROR MANAGEMENT AND DETECTION for more detailed information.)



TL/DD/12837–20

FIGURE 19. CAN Message Arbitration

Frame Formats

INTRODUCTION

There are basically two different types of frames used in the CAN protocol.

The data transmission frames are: data/remote frame

The control frames are: error/overload frame

Note: This device cannot send an overload frame as a result of not being able to process all information. However, the device is able to recognize an overload condition and join overload frames initiated by other devices.

If no message is being transmitted, i.e., the bus is idle, the bus is kept at the "recessive" level. *Figure 20* and *Figure 21* give an overview of the various CAN frame formats.

DATA AND REMOTE FRAME

Data frames consist of seven bit fields and remote frames consist of six different bit fields:

1. Start of Frame (SOF)
2. Arbitration field
3. Control field (IDE bit, R0 bit, and DLC field)
4. Data field (not in remote frame)
5. CRC field
6. ACK field
7. End of Frame (EOF)

A remote frame has no data field and is used for requesting data from other (remote) CAN nodes. *Figure 22* shows the format of a CAN data frame.

FRAME CODING

Remote and Data Frames are NRZ codes with bit-stuffing in every bit field which holds computable information for the interface, i.e., Start of Frame arbitration field, control field, data field (if present) and CRC field.

Error and overload frames are NRZ coded without bit stuffing.

BIT STUFFING

After five consecutive bits of the same value, a stuff bit of the inverted value is inserted by the transmitter and deleted by the receiver.

Destuffed Bit Stream	100000x	011111x
Stuffed Bit Stream	1000001x	0111110x
		x = {0,1}

START OF FRAME (SOF)

The Start of Frame indicates the beginning of data and remote frames. It consists of a single "dominant" bit. A node is only allowed to start transmission when the bus is idle. All nodes have to synchronize to the leading edge (first edge after the bus was idle) caused by SOF of the node which starts transmission first.

ARBITRATION FIELD

The arbitration field is composed of the identifier field and the RTR (Remote Transmission Request) bit. The value of the RTR bit is "dominant" in a data frame and "recessive" in a remote frame.

CONTROL FIELD

The control field consists of six bits. It starts with two bits reserved for future expansion followed by the four-bit Data Length Code. Receivers must accept all possible combinations of the two reserved bits. Until the function of these reserved bits is defined, the transmitter only sends "0" (dominant) bits. The first reserved bit (IDE) is actually defined to indicate an extended frame with 29 Identifier bits if set to "1". CAN chips must tolerate extended frames, even if they can only understand standard frames, to prevent the destruction of an extended frames on an existing network.

The Data Length Code indicates the number of bytes in the data field. This Data Length Code consists of four bits. The data field can be of length zero. The permissible number of data bytes for a data frame ranges from 0 to 8.

DATA FIELD

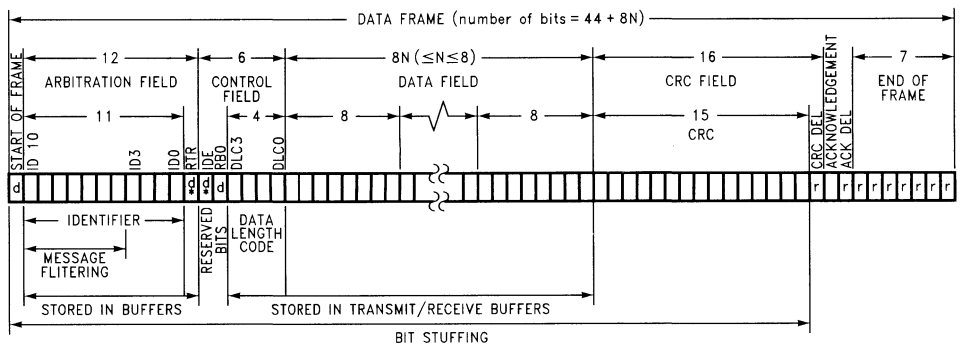
The Data field consists of the data to be transferred within a data frame. It can contain 0 to 8 bytes and each byte contains 8 bits. A remote frame has no data field.

CRC FIELD

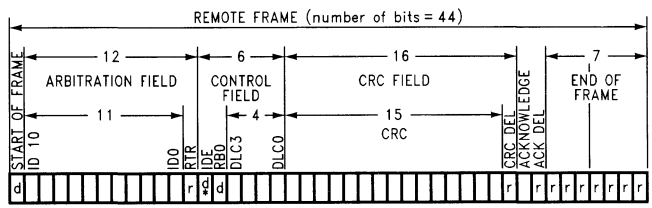
The CRC field consists of the CRC sequence followed by the CRC delimiter. The CRC sequence is derived by the transmitter from the modulo 2 division of the preceding bit fields, starting with the SOF up to the end of the data field, excluding stuff-bits, by the generator polynomial:

$$X^{15} + X^{14} + X^{10} + X^8 + X^7 + X^4 + X^3 + 1$$

The remainder of this division is the CRC sequence transmitted over the bus. On the receiver side the module divides all bit fields up to the CRC delimiter, excluding stuff-bits, and checks if the result is zero. This will then be interpreted as a valid CRC. After the CRC sequence a single "recessive" bit is transmitted as the CRC delimiter.



TL/DD/12837–21



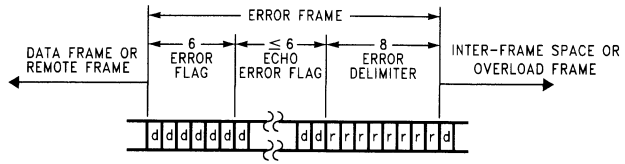
TL/DD/12837–22

A remote frame is identical to a data frame, except that the RTR bit is "recessive", and there is no data field.

IDE = Identifier Extension Bit
The IDE bit in the standard format is transmitted "dominant", whereas in the extended format the IDE bit is "recessive" and the id is expanded to 29 bits.
r = recessive
d = dominant

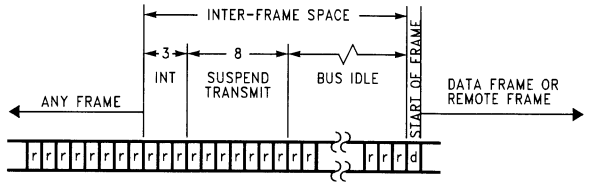
FIGURE 20. CAN Data Transmission Frames

Frame Formats (Continued)



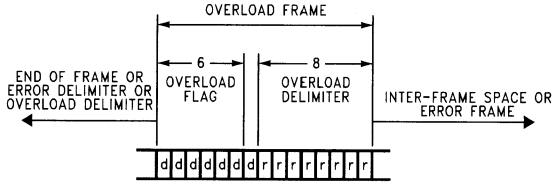
TL/DD/12837–23

An error frame can start anywhere in the middle of a frame.



TL/DD/12837–24

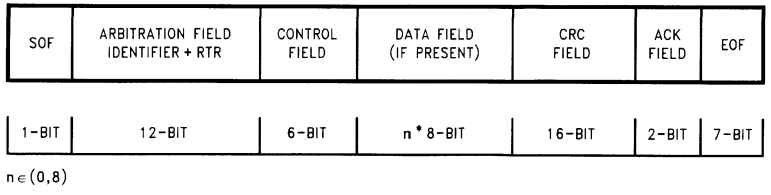
INT = Intermission
Suspend Transmission is only for error passive nodes.



TL/DD/12837–25

An overload frame can only start at the end of a frame.

FIGURE 21. CAN Control Frames



TL/DD/12837–26

FIGURE 22. CAN Frame Format

5

Frame Formats (Continued)

ACK FIELD

The ACK field is two bits long and contains the ACK slot and the ACK delimiter. The ACK slot is filled with a "recessive" bit by the transmitter. This bit is overwritten with a "dominant" bit by every receiver that has received a correct CRC sequence. The second bit of the ACK field is a "recessive" bit called the acknowledge delimiter. As a consequence the acknowledge flag of a valid frame is surrounded by two "recessive" bits, the CRC-delimiter and the ACK delimiter.

EOF FIELD

The End of Frame Field closes a data and a remote frame. It consists of seven "recessive" bits.

INTERFRAME SPACE

Data and remote frames are separate from every preceding frame (data, remote, error and overload frames) by the interframe space see *Figure 23* and *Figure 24* for details. Error and overload frames are not preceded by an interframe space. They can be transmitted as soon as the condition occurs. The interframe space consists of a minimum of three bit fields depending on the error state of the node.

These bit fields are coded as follows:

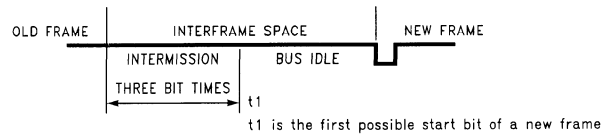
The intermission has the fixed form of three "recessive" bits. While this bit field is active, no node is allowed to start a transmission of a data or a remote frame. The only action to be taken is signaling an overload condition. This means that an error in this bit field would be interpreted as an overload condition. Suspend transmission has to be inserted by error-passive nodes that were transmitter for the last message. This bit field has the form of eight "recessive" bits. However, it may be overwritten by a "dominant" start-bit from another non error passive node which starts transmission. The bus idle field consists of "recessive" bits. Its length is not specified and depends on the bus load.

ERROR FRAME

The Error Frame consists of two bit fields: the error flag and the error delimiter. The error field is built up from the various error flags of the different nodes. Therefore, its length may vary from a minimum of six bits up to a maximum of twelve bits depending on when a module detects the error. Whenever a bit error, stuff error, form error, or acknowledgment error is detected by a node, this node starts transmission of the error flag at the next bit. If a CRC error is detected, transmission of the error flag starts at the bit following the acknowledge delimiter, unless an error flag for a previous error condition has already been started. *Figure 25* shows how a local fault at one module (module 2) leads to a 12-bit error frame on the bus.

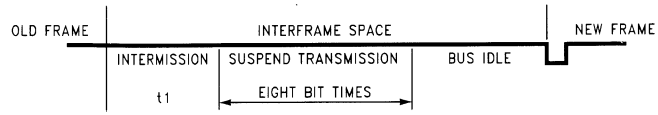
The bus level may either be "dominant" for an error-active node or "recessive" for an error-passive node. An error active node detecting an error, starts transmitting an active error flag consisting of six "dominant" bits. This causes the destruction of the actual frame on the bus. The other nodes detect the error flag as either a violation of the rule of bit-stuffing or the value of a fixed bit field is destroyed. As a consequence all other nodes start transmission of their own error flag. This means, that the error sequence which can be monitored on the bus as a maximum length of twelve bits. If an error passive node detects an error it transmits six "recessive" bits on the bus. This sequence does not destroy a message sent by another node and is not detected by other nodes. However, if the node detecting an error was the transmitter of the frame the other modules will get an error condition by a violation of the fixed bit or stuff rule. *Figure 26* shows how an error passive transmitter transmits a passive error frame and when it is detected by the receivers.

After any module has transmitted its active or passive error flag it waits for the error delimiter which consists of eight "recessive" bits before continuing.



TL/DD/12837-27

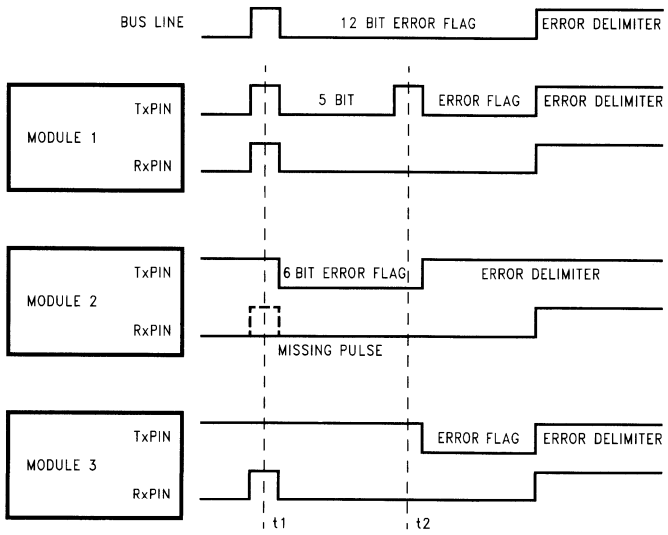
FIGURE 23. Interframe Space for Nodes Which Are Not Error Passive or Have Been Receiver for the Last Frame



TL/DD/12837-28

FIGURE 24. Interframe Space for Nodes Which Are Error Passive and Have Been Transmitter for the Last Frame

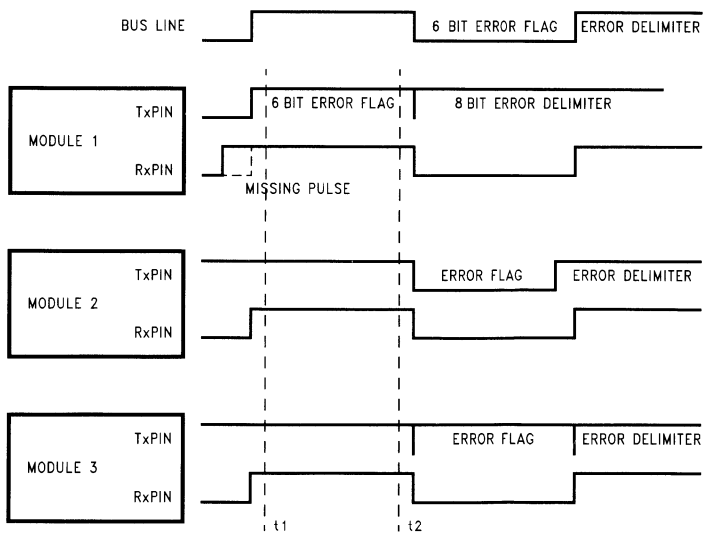
Frame Formats (Continued)



TL/DD/12837-29

module 1 = error active transmitter detects bit error at t2
module 2 = error active receiver with a local fault at t1
module 3 = error active receiver detects stuff error at t2

FIGURE 25. Error Frame—Error Active Transmitter



TL/DD/12837-30

module 1 = error active receiver with a local fault at t1
module 2 = error passive transmitter detects bit error at t2
module 3 = error passive receiver detects stuff error at t2

FIGURE 26. Error Frame—Error Passive Transmitter

5

Frame Formats (Continued)

OVERLOAD FRAME

Like an error frame, an overload frame consists of two bit fields: the overload flag and the overload delimiter. The bit fields have the same length as the error frame field: six bits for the overload flag and eight bits for the delimiter. The overload frame can only be sent after the end of frame (EOF) field and in they way destroys the fixed form of the intermission field.

ORDER OF BIT TRANSMISSION

A frame is transmitted starting with the Start of Frame, sequentially followed by the remaining bit fields. In every bit field the MSB is transmitted first.

FRAME VALIDATION

Frames have a different validation point for transmitters and receivers. A frame is valid for the transmitter of a message, if there is no error until the end of the last bit of the End of Frame field. A frame is valid for a receiver, if there is no error until and including the end of the penultimate bit of the End of Frame.

FRAME ARBITRATION AND PRIORITY

Except for an error passive node which transmitted the last frame, all nodes are allowed to start transmission of a frame after the intermission, which can lead to two or more nodes starting transmission at the same time. To prevent a node from destroying another node's frame, it monitors the bus during transmission of the identifier field and the RTR-bit. As soon as it detects a "dominant" bit while transmitting a "recessive" bit it releases the bus, immediately stops transmission and starts receiving the frame. This causes no data or remote frame to be destroyed by another. Therefore the highest priority message with the identifier 0x000 out of 0x7EF (including the remote data request (RTR) bit) always gets the bus. This is only valid for standard CAN frame format. Note that while the CAN specification allows valid standard identifiers only in the range 0x000 to 0x7EF, the device will allow identifiers to 0x7FF.

There are three more items that should be taken into consideration to avoid unrecoverable collisions on the bus:

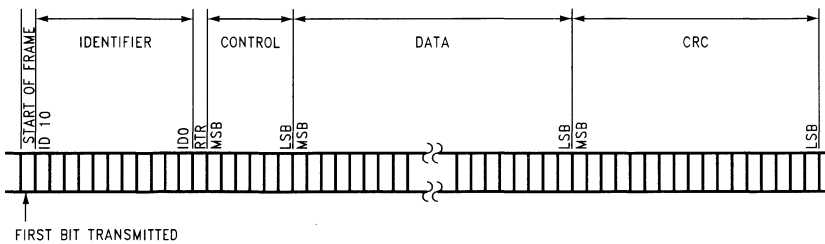
- Within one system each message must be assigned a unique identifier. This is to prevent bit errors, as one module may transmit a "dominant" data bit while the other is transmitting a "recessive" data bit. This could happen if two or more modules start transmission of a frame at the same time and all win arbitration.
- Data frames with a given identifier and a non-zero data length code may be initiated by one node only. Otherwise, in worst case, two nodes would count up to the bus-off state, due to bit errors, if they always start transmitting the same ID with different data.
- Every remote frame should have a system-wide data length code (DLC). Otherwise two modules starting transmission of a remote frame at the same time will overwrite each other's DLC which result in bit errors.

ACCEPTANCE FILTERING

Every node may perform acceptance filtering on the identifier of a data or a remote frame to filter out the messages which are not required by the node. In they way only the data of frames which match the acceptance filter is stored in the corresponding data buffers. However, every node which is not in the bus-off state and has received a correct CRC-sequence acknowledges each frame.

ERROR MANAGEMENT AND DETECTION

There are multiple mechanisms in the CAN protocol, to detect errors and to inhibit erroneous modules from disabling all bus activities.



TL/DD/12837–31

FIGURE 27. Order of Bit Transmission within a CAN Frame

Frame Formats (Continued)

The following errors can be detected:

- Bit Error
  A CAN device that is sending also monitors the bus. If the monitored bit value is different from the bit value that is sent, a bit error is detected. The reception of a "dominant" bit instead of a "recessive" bit during the transmission of a passive error flag, during the stuffed bit stream of the arbitration field or during the acknowledge slot, is not interpreted as a bit error.
- Stuff error
  A stuff error is detected, if the bit level after 6 consecutive bit times has not changed in a message field that has to be coded according to the bit stuffing method.
- Form Error
  A form error is detected, if a fixed frame bit (e.g., CRC delimiter, ACK delimiter) does not have the specified value. For a receiver a "dominant" bit during the last bit of End of Frame does NOT constitute a form error.
- Bit CRC Error
  A CRC error is detected if the remainder of the CRC calculation of a received CRC polynomial is non-zero.
- Acknowledgment Error
  An acknowledgment error is detected whenever a transmitting node does not get an acknowledgment from any other node (i.e., when the transmitter does not receive a "dominant" bit during the ACK frame).

The device can be in one of three states with respect to error handling:

- Error active
  An error active unit can participate in bus communication and sends an active ("dominant") error flag.
- Error passive
  An error passive unit can participate in bus communication. However, if the unit detects an error it is not allowed to send an active error flag. The unit sends only a passive ("recessive") error flag. A device is error passive when the transmit error counter is greater than 127 or when the receive error counter is greater than 127. A device becoming error passive sends an active error flag. An error passive device becomes error active again when both transmit and receive error counter are less than 128.
- Bus off
  A unit that is "bus off" has the output drivers disabled, i.e., it does not participate in any bus activity. A device is bus off when the transmit error counter is greater than 255. A bus off device will become error active again in one of two ways depending on which mode is selected by the user through the Fault Confinement Mode select bit (FMOD) in the CAN Bus Control Register (CBUS). Setting the FMOD bit to "0" (default after power on reset) will select the Standard Fault Confinement mode. In this mode the device goes from "bus off" to "error active" after monitoring 128*11 recessive bits (including bus idle) on the bus. This mode has been implemented for compatibility reasons with existing solutions. Setting the FMOD bit to "1" will select the Enhanced Fault Confinement mode. In this mode the device goes from "bus off" to "error active" after monitoring 128 "good" messages, as indicated by the reception of 11 consecutive "recessive" bits including the End of Frame. The enhanced mode offers the advantage that a "bus off" device (i.e., a device with a serious fault) is not allowed to destroy any messages on the bus until other devices can transmit at least 128 messages. This is not guaranteed in the standard mode, where a defective device could seriously impact bus communication. When the device goes from "bus off" to "error active", both error counters will have the value "0".

In each CAN module there are two error counters to perform a sophisticated error management. The receive error counter (REC) is 7 bits wide and switches the device to the error passive state if it overflows. The transmit error counter (TEC) is 8 bits wide. If it is greater than 127, the device is switched to the error passive state. As soon as the TEC overflows, the device is switched bus-off, i.e., it does not participate in any bus activity.

Frame Formats (Continued)

The counters are modified by the device's hardware according to the following rules:

TABLE X. Receive Error Counter Handling

Condition	Receive Error Counter
A receiver detects a Bit Error during sending an active error flag.	Increment by 8
A receiver detects a "dominant" bit as the first bit after sending an error flag.	Increment by 8
After detecting the 14th consecutive "dominant" bit following an active error flag or overload flag or after detecting the 8th consecutive "dominant" bit following a passive error flag. After each sequence of additional 8 consecutive "dominant" bits.	Increment by 8
Any other error condition (stuff, frame, CRC, ACK).	Increment by 1
A valid reception or transmission.	Decrement by 1 if Counter is not 0

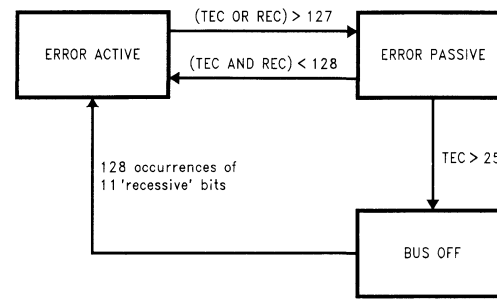
TABLE XI. Transmit Error Counter Handling

Condition	Transmit Error Counter
A transmitter detects a Bit Error during sending an active error flag.	Increment by 8
After detecting the 14th consecutive "dominant" bit following an active error flag or overload flag or after detecting the 8th consecutive "dominant" bit following a passive error flag. After each sequence of additional 8 consecutive "dominant" bits.	Increment by 8
Any other error condition (stuff, frame, CRC, ACK).	Increment by 8
A valid reception or transmission.	Decrement by 1 if Counter is not 0

Special error handling for the TEC counter is performed in the following situations:

- A stuff error occurs during arbitration, when a transmitted "recessive" stuff bit is received as a "dorminant" bit. This does not lead to an incrementation of the TEC.
- An ACK-error occurs in an error passive device and no "dominant" bits are detected while sending the passive error flag. This does not lead to an incrementation of the TEC.
- If only one device is on the bus and this device transmits a message, it will get no acknowledgment. This will be detected as an error and message will be repeated. When the device goes "error passive" and detects an acknowledge error, the TEC counter is not incremented. Therefore the device will not go from "error passive" to the "bus off" state due to such a condition.

Figure 28 shows the connection of different bus states according to the error counters.



TL/DD/12837–32

FIGURE 28. CAN Bus States

Frame Formats (Continued)

SYNCHRONIZATION

Every receiver starts with a "hard synchronization" on the falling edge of the SOF bit. One bit time consists of four bit segments: Synchronization segment, propagation segment, phase segment 1 and phase segment 2.

A falling edge of the data signal should be in the synchronization segment. This segment has the fixed length of one time quanta. To compensate for the various delays within a network, the propagation segment is used. Its length is programmable from 1 to 8 time quanta. Phase segment 1 and phase segment 2 are used to resynchronize during an active frame. The length of these segments is from 1 to 8 time quanta long.

Two types of synchronization are supported:

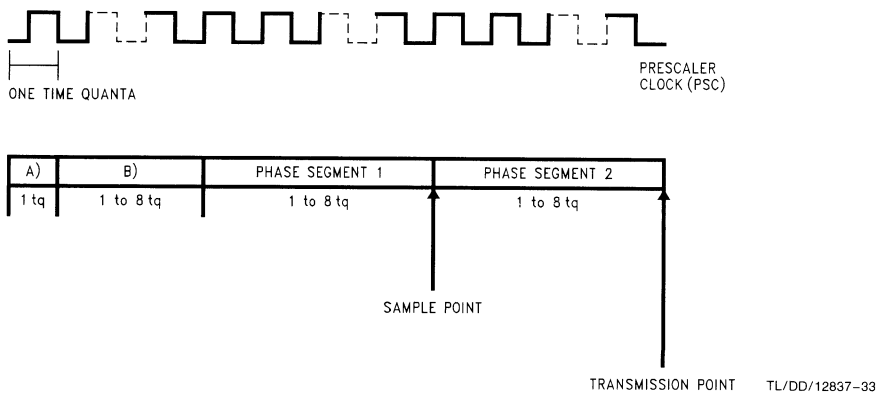
Hard synchronization is done with the falling edge on the bus while the bus is idle, which is then interpreted as the SOF. It restarts the internal logic.

Soft synchronization is used to lengthen or shorten the bit time while a data or remote frame is received. Whenever a falling edge is detected in the propagation segment or in phase segment 1, the segment is lengthened by a specific value, the resynchronization jump width (see *Figure 30*).

If a falling edge lies in the phase segment 2 (as shown in *Figure 30*) it is shortened by the resynchronization jump width. Only one resynchronization is allowed during one bit time. The sample point lies between the two phase segments and is the point where the received data is supposed to be valid. The transmission point lies at the end of phase segment 2 to start a new bit time with the synchronization segment.

Note 1: The resynchronization jump width (RJW) is automatically determined from the programmed value of PS. If a soft resynchronization is done during phase segment 1 or the propagation segment, then RJW will either be equal to 4 internal CAN clocks (CKI/(1 + *divider)*) or the programmed value of PS, whichever is less. PS2 will never be shorter than 1 internal CAN clock.

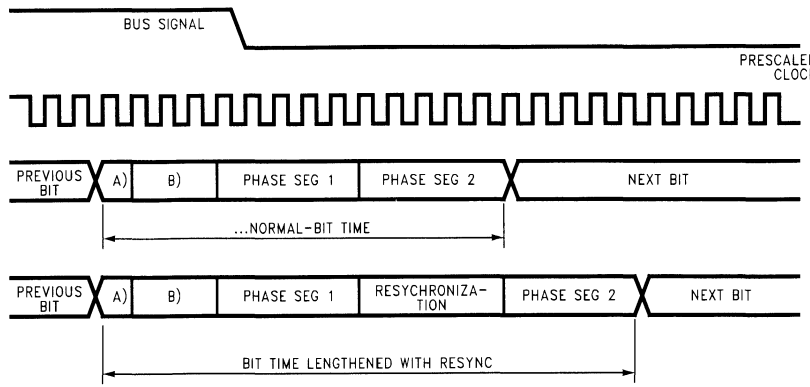
Note 2: (PS1—BTL settings any PSC setting) The PS1 of the BTL should always be programmed to values greater than 1. To allow device resynchronization for positive and negative phase errors on the bus. (if PS1 is programmed to one, a bit time could only be lengthened and never shortened which basically disables half of the synchronization).



A) Synchronization segment
B) Propagation segment

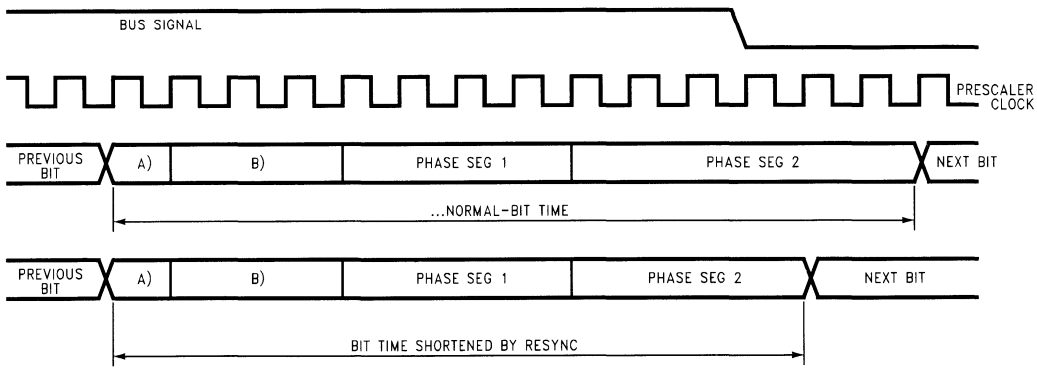
FIGURE 29. Bit Timing

Frame Formats (Continued)



TL/DD/12837-34

FIGURE 30. Resynchronization 1



TL/DD/12837-35

FIGURE 31. Resynchronization 2

Detection of Illegal Conditions

The device can detect various illegal conditions resulting from coding errors, transient noise, power supply voltage drops, runaway programs, etc.

Reading of underfined ROM gets zeros. The opcode for software interrupt is zero. If the program fetches instructions from undefined ROM, this will force a software interrupt, thus signaling that an illegal condition has occurred.

The subroutine stack grows down for each call (jump to subroutine), interrupt, or PUSH, and grows up for each return or POP. The stack pointer is initialized to RAM location 02F Hex during reset. Consequently, if there are more returns than calls, the stack pointer will point to addresses 030 and 031 Hex (which are undefined RAM). Undefined RAM from address 030 to 03F Hex is read as all 1's, which in turn will cause the program to return to address 7FFF Hex. This is an undefined ROM location and the instruction fetched (all 0's) from this location will generate a software interrupt signaling an illegal condition.

Thus, the chip can detect the following illegal conditions:

1. Executing from undefined ROM.
2. Over "POP"ing the stack by having more returns than calls.

When the software interrupt occurs, the user can re-initialize the stack pointer and do a recovery procedure before restarting (this recovery program is probably similar to that following reset, but might not contain the same program initialization procedures).

MICROWIRE/PLUS

MICROWIRE/PLUS is a serial synchronous communications interface. The MICROWIRE/PLUS capability enables the device to interface with any of National Semiconductor's MICROWIRE peripherals (i.e., A/D converters, display drivers, E2PROMs etc.) and with other microcontrollers which support the MICROWIRE interface. It consists of an 8-bit serial shift register (SIO) with serial data input (SI), serial data output (SO) and serial shift clock (SK). *Figure 32* shows a block diagram of the MICROWIRE/PLUS logic.

The shift clock can be selected from either an internal source or an external source. Operating the MICROWIRE/PLUS arrangement with the internal clock source is called the Master mode of operation. Similarly, operating the MICROWIRE/PLUS arrangement with an external shift clock is called the Slave mode of operation.

The CNTRL register is used to configure and control the MICROWIRE/PLUS mode. To use the MICROWIRE/PLUS, the MSEL bit in the CNTRL register is set to one. In the master mode the SK clock rate is selected by the two bits, SL0 and SL1, in the CNTRL register. Table XII details the different clock rates that may be selected.

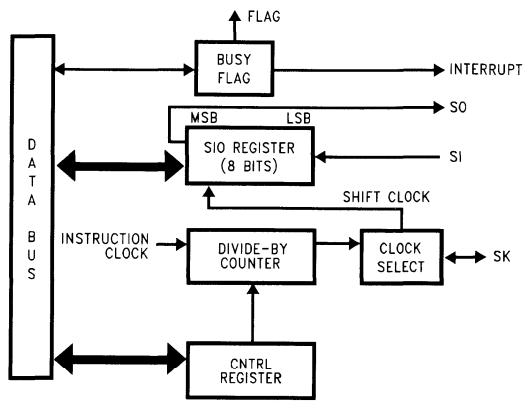
MICROWIRE/PLUS OPERATION

Setting the BUSY bit in the PSW register causes the MICROWIRE/PLUS to start shifting the data. It gets reset when eight data bits have been shifted. The user may reset the BUSY bit by software to allow less than 8 bits to shift. If enabled, an interrupt is generated when eight data bits have been shifted. The device may enter the MICROWIRE/PLUS mode either as a Master or as a Slave. *Figure 32* shows how two COP888 family microcontrollers and several peripherals may be interconnected using the MICROWIRE/PLUS arrangements.

WARNING:

The SIO register should only be loaded when the SK clock is low. Loading the SIO register while the SK clock is high will result in undefined data in the SIO register. SK clock is normally low when not shifting.

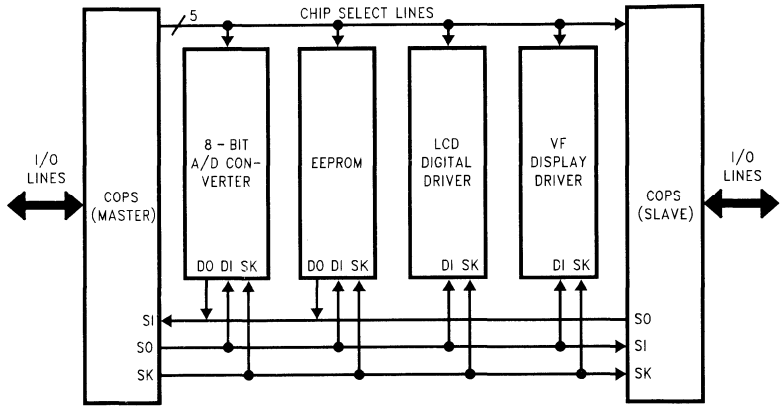
Setting the BUSY flag when the input SK clock is high in the MICROWIRE/PLUS slave mode may cause the current SK clock for the SIO shift register to be narrow. For safety, the BUSY flag should only be set when the input SK clock is low.



TL/DD/12837-36

FIGURE 32. MICROWIRE/PLUS Block Diagram

MICROWIRE/PLUS (Continued)



TL/DD/12837–37

FIGURE 33. MICROWIRE/PLUS Application

MICROWIRE/PLUS Master Mode Operation

In the MICROWIRE/PLUS Master mode of operation the shift clock (SK) is generated internally. The MICROWIRE Master always initiates all data exchanges. The MSEL bit in the CNTRL register must be set to enable the SO and SK functions onto the G Port. The SO and SK pins must also be selected as outputs by setting appropriate bits in the Port G configuraiton register. Table XIII summarizes the bit settings required for Master or Slave mode of operation.

TABLE XII. MICROWIRE/PLUS Master Mode Clock Selection

SL1	SL0	SK
0	0	$2 X t_C$
0	1	$4 X t_C$
1	x	$8 X t_C$

Where t_C is the instruction cycle clock

MICROWIRE/PLUS Slave Mode Operation

In the MICROWIRE/PLUS Slave mode of operation the SK clock is generated by an external source. Setting the MSEL bit in the CNTRL register enables the SO and SK functions onto the G Port. The SK pin must be selected as an input and the SO pin is selected as an output pin by setting and resetting the appropriate bit in the Port G configuration register. Table V summarizes the settings required to enter the Slave mode of operation.

The user must set the BUSY flag immediately upon entering the Salve mode. This will ensure that all data bits sent by the Master will be shifted properly. After eight clock pulses the BUSY flag will be cleared and the sequence may be repeated.

Alternate SK Phase Operation

The device allows either the normal SK clock or an alternate phase SK clock to shift data in and out of the SIO register. In both the modes the SK is normally low. In the normal mode data is shifted in on the rising edge of the SK clock and the data is shifted out on the falling edge of the SK clock. The SIO register is shifted on each falling edge of the SK clock in the normal mode. In the alternate SK phase mode the SIO register is shifted on the rising edge of the SK clock.

A control flag, SKSEL, allows either the normal SK clock or the alternate SK clock to be selected. Resetting SKSEL causes the MICROWIRE/PLUS logic to be clocked from the normal SK signal. Setting the SKSEL flag selects the alternate SK clock. The SKSEL is mapped into the G6 configuration bit. The SKSEL flag will power up in the reset condition, selecting the normal SK signal.

TABLE XIII. MICROWIRE/PLUS Mode Selection

G4 (SO) Config. Bit	G5 (SK) Config. Bit	G4 Fun.	G5 Fun.	Operation
1	1	SO	Int. SK	MICROWIRE/ PLUS Master
0	1	TRI-STATE	Int. SK	MICROWIRE/ PLUS Master
1	0	SO	Ext. SK	MICROWIRE/ PLUS Slave
0	0	TRI-STATE	Ext. SK	MICROWIRE/ PLUS Slave

This table assumes that the control flag MSEL is set.

Serial Peripheral Interface

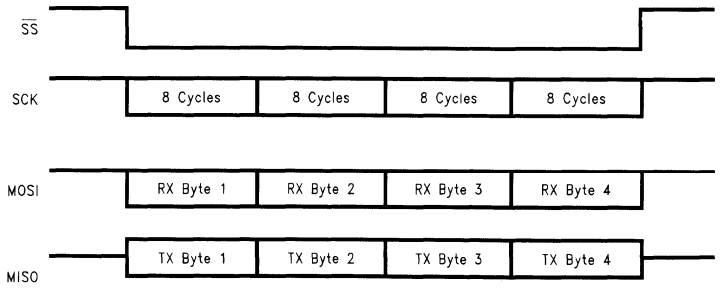


FIGURE 34. SPI Transmission Example

The Serial Peripheral Interface (SPI) is used in master-slave bus systems. It is a synchronous bidirectional serial communication interface with two data lines MISO and MOSI (**M**aster **I**n **S**lave **O**ut, **M**aster **O**ut **S**lave **I**n). A serial clock and a slave select ($\overline{SS}$) signal are always generated by the SPI Master. The interface receives/transmits protocol frames with up to 12 bytes length within a frame, where a frame is defined as the time between a falling edge and a rising edge of $\overline{SS}$.

THEORY OF OPERATION

Figure 36 shows a block diagram illustrating the basic operation of the SPI circuit. In the SPI interface, data is transmitted/received in packets of 8 bits length which are shifted into/out of a shift register with the active edge of the shift clock SCK. Two 12 byte FIFOs, which serve as a receive and a transmit buffer, allow a maximum message length of 12 x 8 bits in both transmit and receive direction without CPU intervention. With CPU intervention, many more bytes can be received. Two registers, the SPI Control Register (SPICNTL) and the SPI Status Register (SPISTAT), are used to control the SPI interface via the internal COP bus. Several different operation modes, such as master or slave operation, are possible.

An $\overline{SS}$-Expander allows the generation of up to 8 signals on the N-port, which can be used as additional $\overline{SS}$-signals ($\overline{ESS}$[7:0]) or as host programmable general purpose signals. The $\overline{SS}$-Expander is programmed with the content of the first MOSI-byte (i.e., the content of the 1st byte [7:0] appears at $\overline{ESS}$[7:0]) (N-port[7:0]), respectively), if the $\overline{ESS}$ programming mode is selected. The $\overline{ESS}$ programming mode is selected by the condition MOSI = L at the falling edge of $\overline{SS}$.

Use of the $\overline{ESS}$ expander requires the setup of four conditions by the user.

1. Set the SESSEN bit of SPICNTL.
2. Set PORTNX to select which bits are used for $\overline{SS}$ expansion.
3. Configure the PORTNC register to enable the desired $\overline{SS}$ expansion bits as outputs.
4. Have an $\overline{ESS}$ condition (MOSI = low at the falling edge of $\overline{SS}$).

Loop Back Mode

Setting the SLOOP bit enables the Loop Back mode, which can be used for test purposes. If the Loop Back mode is selected, TX FIFO data are communicated to the RX FIFO via the SPI Register. In the slave mode, MISO output is internally connected to the MOSI input. In the master mode, the MOSI output is internally connected to the MISO input.

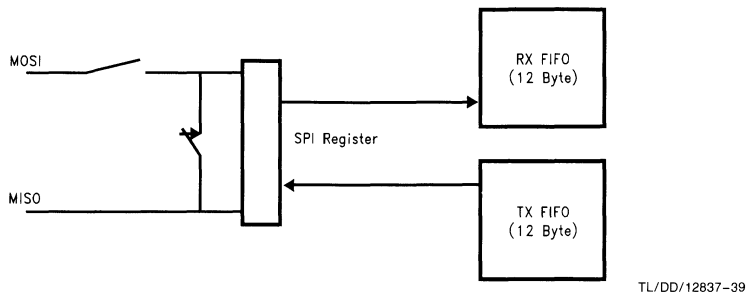
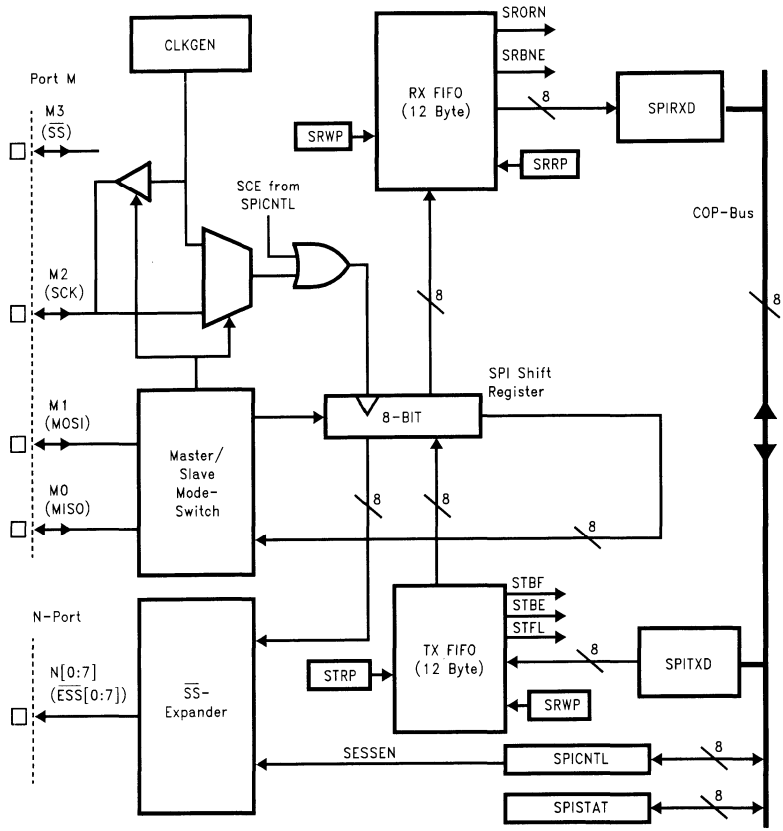


FIGURE 35. Loop Back Mode Block Diagram

Serial Peripheral Interface (Continued)



TL/DD/12837-40

FIGURE 36. SPI Block Diagram

The SPIU Control Register

TABLE XIV. SPI Control (SPICNTL) (0098)

Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
SRIE	STIE	SESSEN	SPIMOD[1:0]		SCE	SPIEN	SLOOP
0	0	0	0	0	0	0	0

Serial Peripheral Interface (Continued)

B7	SRIE	**S**PI **R**eceive **I**nterrupt **E**nable 0—disable receive interrupt 0—enable receive interrupt
B6	STIE	**S**PI **T**ransmit buffer **I**nterrupt **E**nable 0—disable transmit buffer interrupt 0—enable transmit buffer interrupt
B5	SESSEN	**S**PI $\overline{\textbf{SS}}$ Expander ($\overline{\text{ESS}}$) enable 0—The detection of the $\overline{\text{ESS}}$ programming mode is disabled, i.e., the value of MOSI at the falling edge of $\overline{\text{SS}}$ is "don't care". 1—$\overline{\text{ESS}}$ programming mode detection is enabled, i.e., if the condition "MOSI = 0 at the falling edge of $\overline{\text{SS}}$" occurs, the $\overline{\text{SS}}$-Expander is selected and bits [7:0] of the first transmitted byte determine the state of the N-port ($\overline{\text{ESS}}$[7:0]). $\overline{\text{ESS}}$[7:0] will go 1 at the positive edge of $\overline{\text{SS}}$.
B[4:3]	SPIMOD[1:0]	**SPI** operation **mode** select SPIMOD[1:0] 0 0: Slave mode, —SCK is SPI clock input —MISO is SPI data output —MOSI is SPI data input —$\overline{\text{SS}}$ is slave select input 1 0: Standard Master mode, —SCK is SPI clock output (CKI/40) —MISO is SPI data input —MOSI is SPI data output —$\overline{\text{SS}}$ is slave select output In the Master mode, 3 different SPI clock frequencies are available: 0 1: $f_{SCK} = 1/(t_c) = CKI/10$ 1 0: $f_{SCK} = 1/(4\ t_c) = CKI/40$ 1 1: $f_{SCK} = 1/(16\ t_c) = CKI/160$
B2	SCE	**S**PI **a**ctive clock **e**dge select 0: data are shifted out on the falling edge of SCK and are shifted in on the rising edge of SCK 1: data are shifted out on the rising edge of SCK and are shifted in on the falling edge of SCK
B1	SPIEN	**SPI en**able Enables the SPI interface and the alternate functions of the MISO, MOSI, SCK and $\overline{\text{SS}}$ pins. 0: disable SPI 1: enable SPI, all Port M $\overline{\text{ESS}}$ signals are set to 1
B0	SLOOP	**S**PI **loop** back mode 0: disable loop back mode 1: enable loop back mode, MISO and MOSI are internally connected (see *Figure 37*)

Serial Peripheral Interface (Continued)

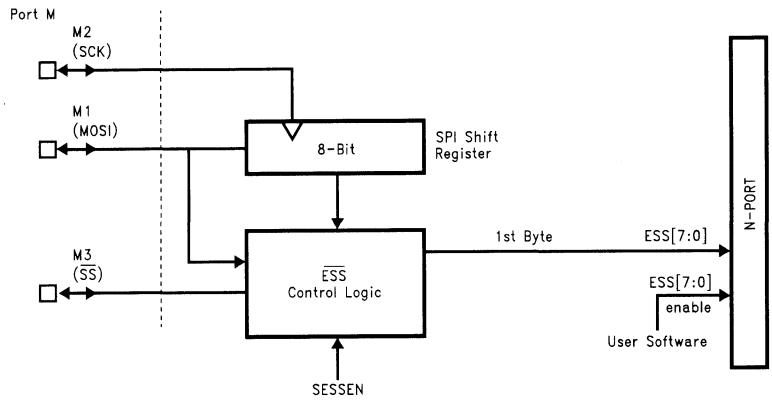
PROGRAMMING THE SPI EXPANDER

If the $\overline{SS}$ Expander is enabled by setting SESSEN = 1 in the SPI Control Register (SPICNTL), the N-port will be programmed with the content of the first MOSI-byte (i.e., the content of the 1st byte [7:0] appears at N-port[7:0] after complete reception of the first byte), if the $\overline{ESS}$ programming mode is detected. If any bytes follow after the 1st MOSI byte, all data will be ignored by the SPI.

The $\overline{ESS}$ programming mode is detected by the $\overline{ESS}$ control logic, which decodes the condition "MOSI = L at the falling edge of $\overline{SS}$. For further details, see *Figure 37*.

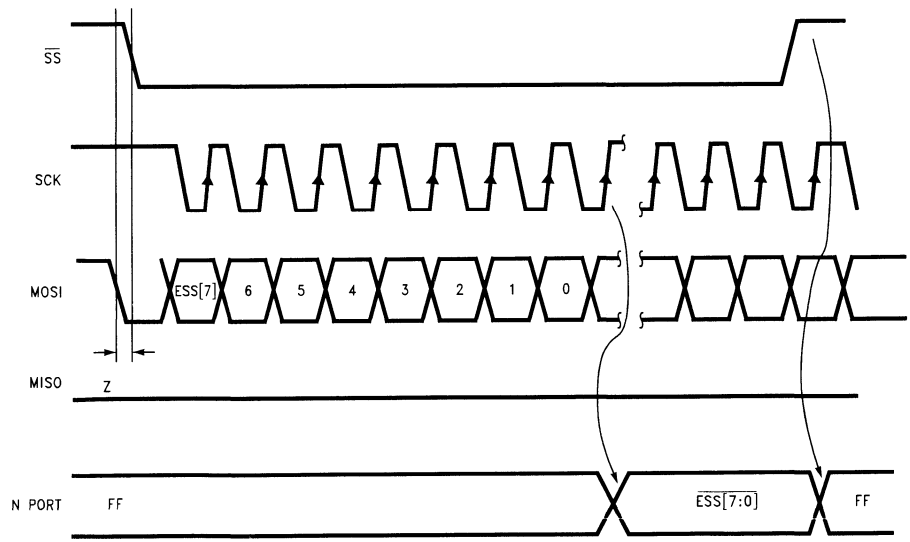
The selected N-port bits will be set to 1 after the positive edge of $\overline{SS}$.

Single N-port bits may be enabled for use as $\overline{SS}$ expansion, or disabled to allow for general purpose I/O, by the respective bits in the PORTNX register.



TL/DD/12837–41

FIGURE 37. Programming the SPI Expander

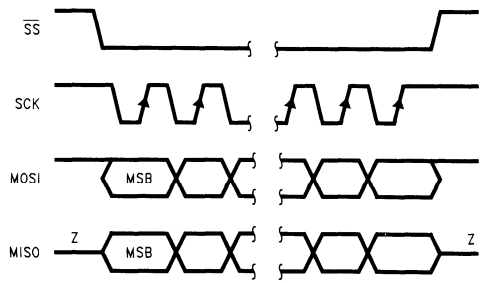


TL/DD/12837–42

SESSEN = 1, SCE = 0. If MOSI = 0 at the falling edge of $\overline{SS}$, the $\overline{ESS}$ programming mode is detected and all N-port alternate functions are enabled.

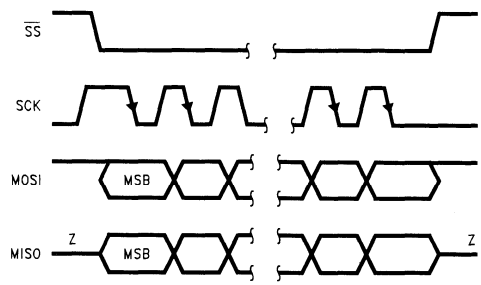
FIGURE 38. Programming the $\overline{SS}$ Expander

SPI Status Register



TL/DD/12837–43

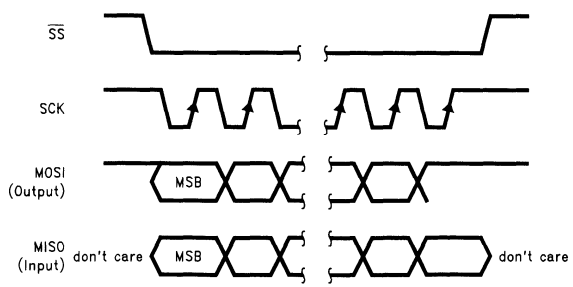
a) Slave mode; rising SCK edge is active edge. (SPIMOD[1,0] = [0,0], SCE = 0)



TL/DD/12837–44

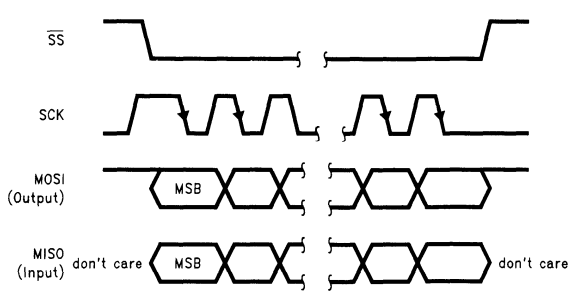
b) Slave mode; falling SCK edge is active edge. (SPIMOD[1,0] = [0,0], SCE = 1)

FIGURE 39. Slave Mode Communication



TL/DD/12837–45

a) Master mode; rising SCK edge is active edge. (SPIMOD[1,0] = [1,0], SCE = 0)



TL/DD/12837–46

b) Master mode; falling SCK edge is active edge. (SPIMOD[1,0] = [1,0], SCE = 1)

FIGURE 40. Master Mode Communication

SPI Status Register (Continued)

TABLE XV. SPI Status Register (SPISTAT) (0099)

Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
SRORN	SRBNE	STBF	STBE	STFL	SESSDET	x	x
0	0	1	1	0	0		0

The SPI Status Register is a read only register.

B7	SRORN	**S**PI **r**eceiver **o**verrun. This bit is set on the attempt to overwrite valid data in the RX FIFO by the SPI interface. (The condition to detect this is: SRWP = SRRP & COP has not read the data at SRRP and attempting to write to the RX FIFO by the SPI interface.) This bit can generate a receive interrupt if the receive interrupt is enabled (SRIE = 1). **Note 0:** At this condition the write operation will not be executed and all data get lost. **Note 1:** The SRORN bit stays set until the reset condition. This bit is reset with a dummy write to the SPISTAT register. (As the register is read only a dummy write does not have any effect on any other bits in this register.) As a result of the SRORN condition, the SRWP becomes frozen (i.e., does not change until the SRORN bit is reset) and the SPI will not store any new data in the RX FIFO. **Note 2:** With the SRRP being still available, the user can read the data in the RX FIFO before resetting the SRORN bit.
B6	SRBNE	**S**PI **R**eceive **b**uffer **n**ot **e**mpty This bit is set with a write to the SPI RX FIFO resulting in SRWP ! = SRRP (caution at rollover!). This bit is reset with the read of the SPIRXD register resulting in SRWP to be equal to SRRP.
B5	STBF	This bit can generate a receive interrupt if enabled with the RIE bit. **S**PI **T**ransmit **b**uffer **f**ull This bit is set after a write operation to the SPITXD register (from the COP side), which results in STRP = STWP. It gets reset as soon as the STRP gets incremented - by the SPI if reading data out of the TX FIFO.
B4	STBE	**S**PI transmit **b**uffer **e**mpty This bit is set after the last bit of the a read from the SPITXD register, which results in STRP = STWP. It gets reset as soon as the STWP gets incremented - by the COP if writing data into the TX FIFO. It is set on reset.
B3	STFL	**S**PI **T**ransmit **b**uffer **f**lush This bit indicates that the contents of the transmit buffer got discharged by the $\overline{SS}$ signal becoming high before all data in the transmit buffer could be transmitted. This bit gets set if the $\overline{SS}$ signal gets high and 1. STRP != STWP **or** 2. STRP = STWP and the current byte has not been completely transmitted from the SPI shift register These conditions will reset STRP and STWP to 0. These are virtual pointers and cannot be viewed. **Note:** STRP = STWP & STBE = 1 will generate an interrupt. This bit gets reset with a write to the SPITXD register.
B2	SESSDET	**S**PI $\overline{\textbf{SS}}$ **E**xpander **d**etection This bit indicates the detection of a $\overline{SS}$ expand condition (MOSI = 0 at the falling edge of $\overline{SS}$) immediately after the N-port has been programmed (8th SCK bit, 8 μs at SCK = 1 MHz). This bit is reset at the rising edge of $\overline{SS}$. 1: $\overline{SS}$ expand condition detected. 0: normal communication. **Note:** The SPI master must hold $\overline{SS}$ = 0 long enough to allow the device to read SESSDET. Otherwise the SESSDET information will get lost.
B1		unused
B0		unused

SPI Status Register (Continued)

SPI SYNCHRONIZATION

After the SPI is enabled (SPIEN = 1), the SPI internal receive and transmit shift clock is kept disabled until $\overline{SS}$ becomes inactive. This includes $\overline{SS}$ being active at the time SPIEN is set, i.e., no receive/transmit is possible until $\overline{SS}$ becomes inactive after enabling the SPI.

HALT/IDLE MODE

If the device enters the HALT/IDLE mode, both RX and TX FIFOs get reset (Flushed). If the device is exiting HALT/IDLE mode, and SPI synchronization takes place as described above. SPIRXD and SPITXD have the same state as after Reset, SPISTAT bits after HALT/IDLE mode are:

```
SRORN:    unchanged
SRBNE:    0
STBF:     0
STBE:     1
STFL:     1
SESSDET:  x (depending on SS
          and MOSI line)
```

TRANSMISSION START IN MASTER MODE

The transmission of data in the Master mode is started if the user controlled $\overline{SS}$ signal is switched active. No SCK will be generated in Master mode and thus no data is transmitted if the $\overline{SS}$ signal is kept high, i.e., $\overline{SS}$ must be switched low to generate SCK. Resetting the $\overline{SS}$ signal in the Master mode will immediately stop the transmission and flush the transmit FIFO. Thus, the user must only reset the $\overline{SS}$ if:

a) TBE is set or

b) SCK is high (SCE = 0) or low (SCE = 1)

TX AND RX FIFO

If the SPI is disabled (SPIEN = 0), all SPI FIFO related pointers are reset and kept at zero until the SPI is enabled again. Also, the Read/Write operation to both SPITXD and SPIRXD will not cause the pointers to change, if SPIEN is set, Read operations from the RXFIFO and Write operation to TXFIFO will increment the respective Read/Write pointers.

SPIRXD SPI Receive Data Register

SPIRXD is at address location "009A". It is a read/write register.

This register holds the receive data at the current SRRP location: a COP read operation from this register to the accumulator will read the RX FIFO at the SRRP location and increment SRRP afterwards. A write to this register (by the controllers SW) will write to the RX FIFO at the current SRRP location. The SRRP is not changed.

Note: During breakpoint the SRRP is not incremented.

A write to this register from the SPI interface side will write to the current SRWP location and increment SRWP afterwards.

SPITXD SPI Transmit Data Register

SPITXD is at address location "009B". It is a read/write register.

This register holds the transmit data at the current STWP location: a write from the controller to this register will write to the STWP location and increment the STWP afterwards. A read from the controller to this register will read the TX FIFO at the current STWP location. The pointer is not changed.

Writing data into this register will start a transmission of data in the master mode.

Note: No read modify write instructions should be used on this register.

Reading this register from the SPI side will read the byte at the current STRP location and afterwards increment STRP.

SPI RX FIFO

The SPI RX FIFO is a 12 byte first in first out buffer. SPI RX FIFO data are read from the controller by reading the SPIRXD register. A pointer (SRRP) controls the controller read location. Data is written to this register by the SPI interface. The write location is controlled by the SRWP. SRWP is incremented after data is stored to the FIFO SRWP is never decremented SRWP has a roll-over 10 → 11 → 0 → 1 → 2 → etc. It is a circularly linked list.

SRRP is incremented after data is read from the FIFO SRRP is never decremented SRRP has a roll-over 10 → 11 → 0 → 1 → 2 → etc. It is a circularly linked list.

Both pointers are cleared at reset.

The following bits indicate the status of the RX FIFO: SRBNE = (SRWP != SRRP) and !SRORN .SRORN is set at (SRWP = SRRP) and after a write from the SPI side, reset at write to SPISTAT.

Special conditions: if .SRORN is set, no writes to the RX FIFO are allowed from the SPI side. SRWP is frozen. Resetting .SRORN (after it was set) clears both SRWP and SRRP. To prevent erroneous clearing of the Receive FIFO when entering HALT/IDLE mode, the user needs to enable the MIWU or port M3 ($\overline{SS}$) by setting bit 3 in MWKEN register.

SPI TX FIFO

The SPI TX FIFO is a 12 byte first in first out buffer. Data is written to the FIFO by the controller executing a write instruction to the SPITXD register. A pointer (STWP) controls the controller write location. Data is read from this register by the SPI interface. The read location is controlled by the STRP. STRP is incremented after data is read from the FIFO STRP is never decremented STRP has a roll-over 10 → 11 → 0 → 1 → 2 → etc. It is a circularly linked list.

STWP is incremented after data is written to the FIFO STWP is never decremented STWP has a roll-over 10 → 11 → 0 → 1 → 2 → etc. It is a circularly linked list.

Both pointers are cleared at reset.

The following bits indicate the status of the TX FIFO: STBF = set at (STRP = STWP) after a write from the controller reset at ((STRP != STWP) I STBE) after a read from the SPI STBE = (STRP = STWP) after a read from the SPI.

Special conditions: If the $\overline{SS}$ signal becomes high before data the last bit of the last byte in the TX FIFO is transmitted both STRP and STWP will be set to 0. The STFL bit will be set. (STBE will be set as well.)

Note: The SRRP, SRWP, STRP and STWP registers are not available to the user. Their operation description is included for clarity and to enhance the user's understanding.

A/D Convertor

The device contains an 8-channel, multiplexed input, successive approximation, Analog-to-Digital convertor. The device contains AGND/AV_{CC} and ADV_{REF} for voltage reference.

OPERATING MODES

The A/D convertor supports ratiometric measurements. It supports both Single Ended and Differential modes of operation.

Four specific analog channel selection modes are supported. These are as follows:

Allow any specific channel to be selected at one time. The A/D convertor performs the specific conversion requested and stops.

Allow any specific channel to be scanned continuously. In other words, the user specifies the channel and the A/D convertor scans it continuously. At any arbitrary time the user can immediately read the result of the last conversion. The user must wait for only the first conversion to complete.

Allow any differential channel pair to be selected at one time. The A/D convertor performs the specific differential conversion requested and stops.

Allow any differential channel pair to be scanned continuously. In other words, the user specifies the differential channel pair and the A/D convertor scans it continuously. At any arbitrary time the user can immediately read the result of the last differential conversion. The user must wait for only the first conversion to complete.

The A/D convertor is supported by two memory mapped registers, the result register and the mode control register. When the device is reset, the mode control register (ENAD) is cleared, the A/D is powered down and the A/D result register has unknown data.

A/D Control Register

The ENAD control register contains 3 bits for channel selection, 2 bits for prescaler selection, 2 bits for mode selection and a Busy bit. An A/D conversion is initiated by setting the ADBSY bit and the ENAD control register. The result of the conversion is available to the user in the A/D result register, ADRSLT, when ADBSY is cleared by the hardware on completion of the conversion.

ENAD (address (0xCB)

CHANNEL SELECT			MODE SELECT		PRESCALER SELECT		BUSY
ADCH2	ADCH1	ADCH0	ADMOD1	ADMOD0	PSC1	PSC0	ADBSY

Bit 7 Bit 0

CHANNEL SELECT

This 3-bit field selects one of eight channels to be the $V_{IN}+$. The mode selection determines the $V_{IN}-$ input.

Single Ended mode:

Bit 7	Bit 6	Bit 5	Channel No.
0	0	0	0
0	0	1	1
0	1	0	2
0	1	1	3
1	0	0	4
1	0	1	5
1	1	0	6
1	1	1	7

Differential mode:

Bit 7	Bit 6	Bit 5	Channel Pairs (+, −)
0	0	0	0, 1
0	0	1	1, 0
0	1	0	2, 3
0	1	1	3, 2
1	0	0	4, 5
1	0	1	5, 4
1	1	0	6, 7
1	1	1	7, 6

MODE SELECT

This 2-bit field is used to select the mode of operation (single conversion, continuous conversions, differential, single ended) as shown in the following table.

Bit 4	Bit 3	Mode
0	0	Single Ended mode, single conversion
0	1	Single Ended mode, continuous scan of a single channel into the result register
1	0	Differential mode, single conversion
1	1	Differential mode, continuous scan of a channel pair into the result register

PRESCALER SELECT

This 2-bit field is used to select one of the four prescaler clocks for the A/D converter. The following table shows the various prescaler options.

A/D Convertor Clock Prescaler

Bit 2	Bit 1	Clock Select
0	0	Divide by 2
0	1	Divide by 4
1	0	Divide by 6
1	1	Divide by 12

BUSY BIT

The ADBSY bit of the ENAD register is used to control starting and stopping of the A/D conversion. When ADBSY is cleared, the prescale logic is disabled and the A/D clock is turned off. Setting the ADBSY bit starts the A/D clock and initiates a conversion based on the mode select value currently in the ENAD register. Normal completion of an A/D conversion clears the ADBSY bit and turns off the A/D convertor.

A/D Convertor (Continued)

The ADBSY bit remains a one during continuous conversion. The user can stop continuous conversion by writing a zero to the ADBSY bit.

If the user wishes to restart a conversion which is already in progress, this can be accomplished only by writing a zero to the ADBSY bit to stop the current conversion and then by writing a one to ADBSY to start a new conversion. This can be done in two consecutive instructions.

A/D Operation

The A/D convertor interface works as follows. Setting the ADBSY bit in the A/D control register ENAD initiates an A/D conversion. The conversion sequence starts at the beginning of the write to ENAD operation which sets ADBSY, thus powering up the A/D. At the first falling edge of the convertor clock following the write operation, the sample signal turns on for seven clock cycles. If the A/D is in single conversion mode, the conversion complete signal from the A/D will generate a power down for the A/D convertor and will clear the ADBSY bit in the ENAD register at the next instruction cycle boundary. If the A/D is in continuous mode, the conversion complete signal will restart the conversion sequence by deselecting the A/D for one convertor clock cycle before starting the next sample. The A/D 8-bit result is immediately loaded into the A/D result register (ADRSLT) upon completion. Internal logic prevents transient data (resulting from the A/D writing a new result over an old one) being read from ADRSLT.

Inadvertent changes to the ENAD register during conversion are prevented by the control logic of the A/D. Any attempt to write any bit of the ENAD Register except ADBSY, while ADBSY is a one, is ignored. ADBSY must be cleared either by completion or an A/D conversion or by the user before the prescaler, conversion mode or channel select values can be changed. After stopping the current conversion, the user can load different values for the prescaler, conversion mode or channel select and start a new conversion in one instruction.

It is important for the user to realize that, when used in differential mode, only the positive input to the A/D converter is sampled and held. The negative input is constantly connected and should be held stable for the duration of the conversion. Failure to maintain a stable negative input will result in incorrect conversion.

PRESCALER

The A/D Convertor (A/D) contains a prescaler option that allows four different clock selections. The A/D clock frequency is equal to CKI divided by the prescaler value. Note that the prescaler value must be chosen such that the A/D clock falls within the specified range. The maximum A/D frequency is 1.67 MHz. This equates to a 600 ns A/D clock cycle.

The A/D convertor takes 17 A/D clock cycles to complete a conversion. Thus the minimum A/D conversion time for the device is 10.2 μs when a prescaler of 6 has been selected. The 17 A/D clock cycles needed for conversion consist of 1 cycle at the beginning for reset, 7 cycles for sampling, 8 cycles for converting, and 1 cycle for loading the result into the A/D result register (ADRSLT). This A/D result register is a read-only register. The user cannot write into ADRSLT.

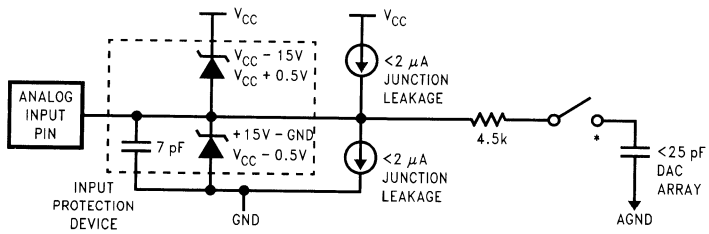
The ADBSY flag provides an A/D clock inhibit function, which saves power by powering down the A/D when it is not in use.

Note: The A/D convertor is also powered down when the device is in either the HALT or IDLE modes. If the A/D is running when the device enters the HALT or IDLE modes, the A/D powers down and then restarts the conversion with a corrupted sampled voltage (and thus an invalid result) when the device comes out of the HALT or IDLE modes.

Analog Input and Source Resistance Considerations

Figure 41 shows the A/D pin model in single ended mode. The differential mode has a similar A/D pin model. The leads to the analog inputs should be kept as short as possible. Both noise and digital clock coupling to an A/D input can cause conversion errors. The clock lead should be kept away from the analog input line to reduce coupling. The A/D channel input pins do not have any internal output driver circuitry connected to them because this circuitry would load the analog input signals due to output buffer leakage current.

Source impedances greater than 3 kΩ on the analog input lines will adversely affect the internal RC charging time during input sampling. As shown in *Figure 41*, the analog switch to the DAC array is closed only during the 7 A/D cycle sample time. Large source impedances on the analog inputs may result in the DAC array not being charged to the correct voltage levels, causing scale errors.



TL/DD/12837–47

* The analog switch is closed only during the sample time.

FIGURE 41. A/D Pin Model (Single Ended Mode)

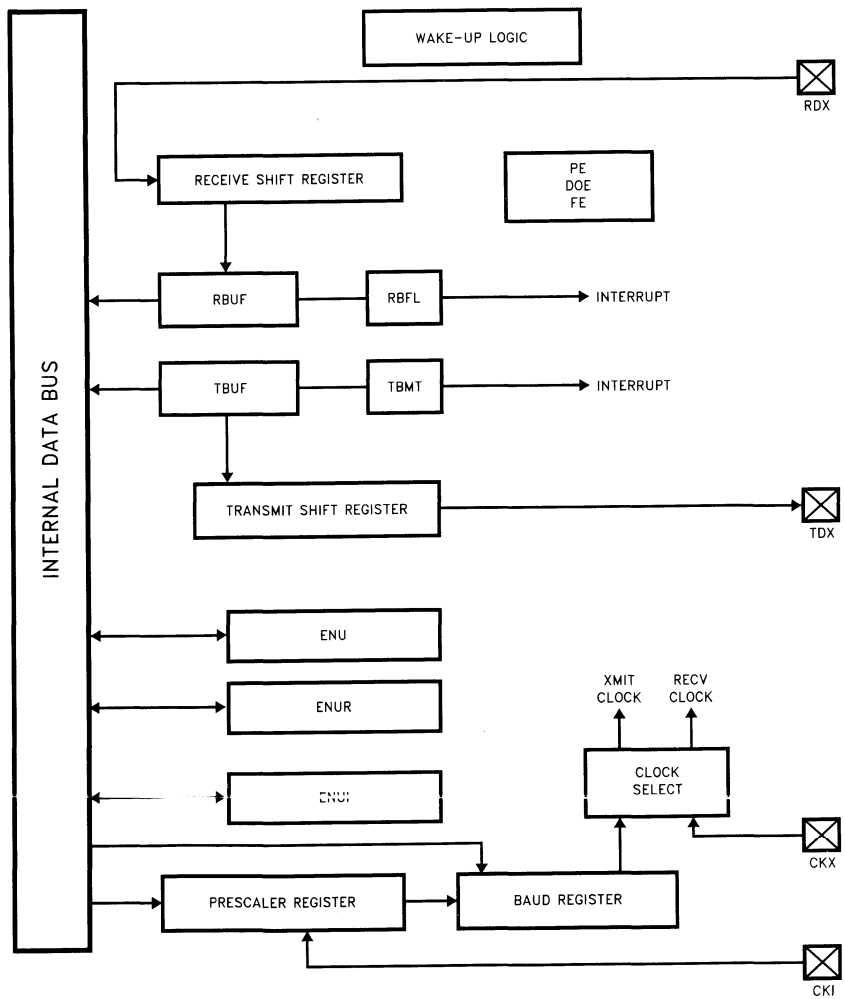
A/D Convertor (Continued)

If large source resistance is necessary, the recommended solution is to slow down the A/D clock speed in proportion to the source resistance. The A/D convertor may be operated at the maximum speed for $R_S < 3\ k\Omega$. For $R_S > 3\ k\Omega$, A/D clock speed needs to be reduced. For example, with $R_S = 6\ k\Omega$, the A/D convertor may be operated at half the maximum speed. A/D convertor clock speed may be slowed down by either increasing the A/D prescaler divide-by or decreasing the CKI clock frequency. The A/D minimum clock speed is 100 kHz.

UART

The device contains a full-duplex software programmable UART. The UART *(Figure 42)* consists of a transmit shift register, a receiver shift register and seven addressable registers, as follows: a transmit buffer register (TBUF), a receiver buffer register (RBUF), a UART control and status register (ENU), a UART receive control and status register (ENUR), a UART interrupt and clock source register (ENUI), a prescaler select register (PSR) and baud (BAUD) register. The ENU register contains flags for transmit and receive functions; this register also determines the length of the data frame (7, 8 or 9 bits), the value of the ninth bit in transmission, and parity selection bits. The ENUR register flags framing, data overrun and parity errors while the UART is receiving.

Other functions of the ENUR register include saving the ninth bit received in the data frame, enabling or disabling the UART's attention mode of operation and providing additional receiver/transmitter status information via RCVG and XMTG bits. The determination of an internal or external clock source is done by the ENUI register, as well as selecting the number of stop bits and enabling or disabling transmit and receive interrupts. A control flag in this register can also select the UART mode of operation: asynchronous or synchronous.



TL/DD/12837–48

FIGURE 42. UART Block Diagram

UART (Continued)

UART CONTROL AND STATUS REGISTERS

The operation of the UART is programmed through three registers: ENU, ENUR and ENUI. The function of the individual bits in these registers is as follows:

ENU-UART Control and Status Register (Address at 0BA)

PEN	PSEL1	XBIT9/ PSEL0	CHL1	CHL0	ERR	RBFL	TBMT
0RW	0RW	0RW	0RW	0RW	0R	0R	1R

Bit 7 Bit 0

ENU-UART Receive Control and Status Register (Address at 0BB)

DOE	FE	PE	SPARE	RBIT9	ATTN	XMTG	RCVG
0RD	0RD	0RD	0RW*	0R	0RW	0R	0R

Bit 7 Bit 0

ENUI-UART Interrupt and Clock Source Register (Address at 0BC)

STP2	STP78	ETDX	SSEL	XRCLK	XTCLK	ERI	ETI
0RW	0RW	0RW	0RW	0RW	0RW	0RW	0RW

Bit 7 Bit 0

* Bit is not used.

0 Bit is cleared on reset.

1 Bit is set to one on reset.

R Bit is read-only; it cannot be written by software.

R/W Bit is read/write.

D Bit is cleared on read; when read by software as a one, it is cleared automatically. Writing to the bit does not affect its state.

DESCRIPTION OF UART REGISTER BITS

ENU—UART CONTROL AND STATUS REGISTER

TBMT: This bit is set when the UART transfers a byte of data from the TBUF register into the TSFT register for transmission. It is automatically reset when software writes into the TBUF register.

RBFL: This bit is set when the UART has received a complete character and has copied it into the RBUF register. It is automatically reset when software reads the character from RBUF.

ERR: This bit is a global UART error flag which gets set if any or a combination of the errors (DOE, FE, PE) occur.

CHL1, CHL0: These bits select the character frame format. Parity is not included and is generated/verified by hardware.

CHL1 = 0, CHL0 = 0 The frame contains eight data bits.

CHL1 = 0, CHL0 = 1 The frame continues seven data bits.

CHL1 = 1, CHL0 = 0 The frame continues nine data bits.

CHL1 = 1, CHL0 = 1 Loopback Mode selected. Transmitter output internally looped back to receiver input. Nine bit framing format is used.

XBIT9/PSEL0: Programs the ninth bit for transmission when the UART is operating with nine data bits per frame. For seven or eight data bits per frame, this bit in conjunction with PSEL1 selects parity.

PSEL1, PSEL0: Parity select bits.

PSEL1 = 0, PSEL0 = 0 Odd Parity (if Parity enabled)

PSEL1 = 0, PSEL1 = 1 Odd Parity (if Parity enabled)

PSEL1 = 1, PSEL0 = 0 Mark(1) (if Parity enabled)

PSEL1 = 1, PSEL1 = 1 Space(0) (if Parity enabled)

PEN: This bit enables/disabled Parity (7- and 8-bit modes only).

PEN = 0 Parity disabled.

PEN = 1 Parity enabled.

ENUR—UART RECEIVE CONTROL AND STATUS REGISTER

RCVG: This bit is set high whenever a framing error occurs and goes low when RDX goes high.

XMTG: This bit is set to indicate that the UART is transmitting. It gets reset at the end of the last frame (end of last Stop bit).

ATTN: ATTENTION Mode is enabled while this bit is set. This bit is cleared automatically on receiving a character with data bit nine set.

RBIT9: Contains the ninth data bit received when the UART is operating with nine data bits per frame.

SPARE: Reserved for future use.

PE: Flags a Parity Error.

PE = 0 Indicates no Parity Error has been detected since the last time the ENUR register was read.

PE = 1 Indicates the occurrence of a Parity Error.

FE: Flags a Framing Error.

FE = 0 Indicates no Framing Error has been detected since the last time the ENUR register was read.

FE = 1 Indicates the occurrence of a Framing Error.

DOE: Flags a Data Overrun Error.

DOE = 0 Indicates no Data Overrun Error has been detected since the last time the ENUR register was read.

DOE = 1 Indicates the occurrence of a Data Overrun Error.

ENUE—UART INTERRUPT AND CLOCK SOURCE REGISTER

ETI: This bit enables/disables interrupt from the transmitter section.

ETI = 0 Interrupt from the transmitter is disabled.

ETI = 1 Interrupt from the transmitter is enabled.

ERI: This bit enables/disables interrupt from the receiver section.

ERI = 0 Interrupt from the receiver is disabled.

ERI = 1 Interrupt from the receiver is enabled.

XTCLK: This bit selects the clock source for the transmitter section.

UART (Continued)

XTCLK = 0 The clock source is selected through the PSR and BAUD registers.

XTCLK = 1 Signal on CKX (L1) pin is used as the clock.

XRCLK: This bit selects the clock source for the receiver section.

XRCLK = 0 The clock source is selected through the PSR and BAUD registers.

XRCLK = 1 Signal on CKX (L1) pin is used as the clock.

SSEL: UART mode select.

SSEL = 0 Asynchronous Mode.

SSEL = 1 Synchronous Mode.

ETDX: TDX (UART Transmit Pin) is the alternate function assigned to Port L pin L2; it is selected by setting EDTX bit. To simulate line break generation, software should reset ETDX bit and output logic zero to TDX pin through Port L data and configuration registers.

STP78: This bit is set to program the last Stop bit to be 7/8th of a bit in length.

STP2: This bit programs the number of Stop bits to be transmitted.

STP2 = 0 One Stop bit transmitted.

STP2 = 1 Two Stop bits transmitted.

Associated I/O Pins

Data is transmitted on the TDX pin and received on the RDX pin. TDX is the alternate function assigned to Port L pin L2; it is selected by setting ETDX (in the ENUI register) to one. RDX is an inherent function of Port L pin L3, requiring no setup.

The baud rate clock for the UART can be generated on-chip, or can be taken from an external source. Port L pin L1 (CKX) is the external clock I/O pin. The CKX pin can be either an input or an output, as determined by Port L Configuration and Data registers (Bit 1). As an input, it accepts a clock signal which may be selected to drive the transmitter and/or receiver. As an output, it presents the internal Baud Rate Generator output.

UART Operation

The UART has two modes of operation; asynchronous mode and synchronous mode.

ASYNCHRONOUS MODE

This mode is selected by resetting the SSEL (in the ENUI register) bit to zero. The input frequency to the UART is 16 times the baud rate.

The TSFT and TBUF registers double-buffer data for transmission. While TSFT is shifting out the current character on the TDX pin, the TBUF register may be loaded by software with the next byte to be transmitted. When TSFT finishes transmitting the current character the contents of TBUF are transferred to the TSFT register and the Transmit Buffer Empty Flag (TBMT in the ENU register) is set. The TBMT flag is automatically reset by the UART when software loads a new character into the TBUF register. There is also the XMTG bit which is set to indicate that the UART is transmitting. This bit gets reset at the end of the last frame (end of last Stop bit). TBUF is a read/write register.

The RSFT and RBUF registers double-buffer data being received. The UART receiver continually monitors the signal on the RDX pin for a low level to detect the beginning of a Start bit. Upon sensing this low level, it waits for half a bit time and samples again. If the RDX pin is still low, the receiver considers this to be a valid Start bit, and the remaining bits in the character frame are each sampled a single time, at the mid-bit position. Serial data input on the RDX pin is shifted into the RSFT register. Upon receiving the complete character, the contents of the RSFT register are copied into the RBUF register and the Received Buffer Full Flag (RBFL) is set. RBFL is automatically reset when software reads the character from the RBUF register. RBUF is a read only register. There is also the RCVG bit which is set high when a framing error occurs and goes low once RDX goes high. TBMT, XMTG, RBFL and RCVG are read only bits.

SYNCHRONOUS MODE

In this mode data is transferred synchronously with the clock. Data is transmitted on the rising edge and received on the falling edge of the synchronous clock.

This mode is selected by setting SSEL bit in the ENUI register. The input frequency to the UART is the same as the baud rate.

When an external clock input is selected at the CKX pin, data transmit and receive are performed synchronously with this clock through TDX/RDX pins.

If data transmit and receive are selected with the CKX pin as clock output, the device generates the synchronous clock output at the CKX pin. The internal baud rate generator is used to produce the synchronous clock. Data transmit and receive are performed synchronously with this clock.

FRAMING FORMATS

The UART supports several serial framing formats *(Figure 43)*. The format is selected using control bits in the ENU, ENUR and ENUI registers.

The first format (1, 1a, 1b, 1c) for data transmission (CHL0 = 1, CHL1 = 0) consists of Start bit, seven Data bits (excluding parity) and 7/8, one or two Stop bits. In applications using parity, the parity bit is generated and verified by hardware.

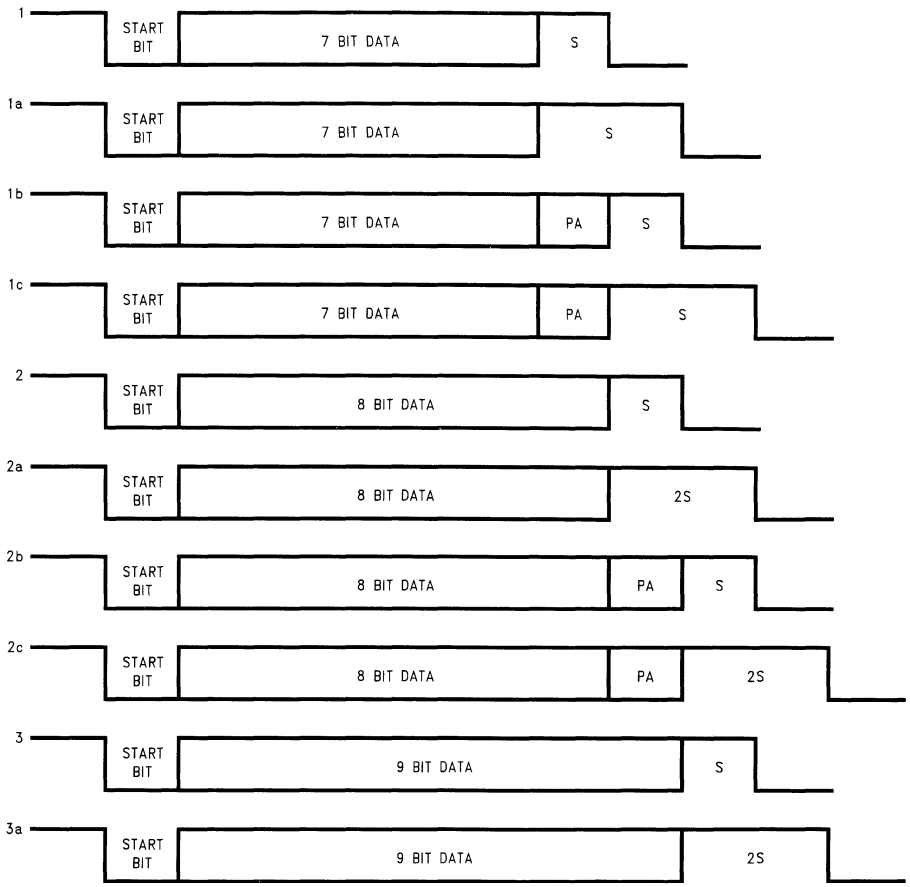
The second format (CHL0 = 0, CHL1 = 0) consists of one Start bit, eight Data bits (excluding parity) and 7/8, one or two Stop bits. Parity bit is generated and verified by hardware.

The third format for transmission (CHL0 = 0, CHL1 = 1) consists of one Start bit, nine Data bits and 7/8, one or two Stop bits. This format also supports the UART "ATTENTION" feature. When operating in this format, all eight bits of TBUF and RBUF are used for data. The ninth data bit is transmitted and received using two bits in the ENU and ENUR registers, called XBIT9 and RBIT9. RBIT9 is a read only bit. Parity is not generated or verified in this mode.

For any of the above framing formats, the last Stop bit can be programmed to be 7/8th of a bit in length. If two Stop bits are selected and the 7/8th bit is set (selected), the second Stop bit will be 7/8th of a bit in length.

The parity is enabled/disabled by PEN bit located in the ENU register. Parity is selected for 7-bit and 8-bit modes only. If parity is enabled (PEN = 1), the parity selection is then performed by PSEL0 and PSEL1 bits located in the ENU register.

UART (Continued)



TL/DD/12837–49

FIGURE 43. Framing Formats

Note that the XBIT9/PSEL0 bit located in the ENU register serves two mutually exclusive functions. This bit programs the ninth bit for transmission when the UART is operating with nine data bits per frame. There is no parity selection in this framing format. For other framing formats XBIT9 is not needed and the bit is PSEL0 used in conjunction with PSEL1 to select parity.

The frame formats for the receiver differ form the transmitter in the number to Stop bits required. The receiver only requires one Stop bit in a frame, regardless of the setting of the Stop bit selection bits in the control register. Note that an implicit assumption is made for full duplex UART operatioin that the framing formats are the same for the transmitter and receiver.

UART INTERRUPTS

The UART is capable of generating interrupts. Interrupts are generated on Receive Buffer Full and Transmit Buffer Empty. Both interrupts have individual interrupt vectors. Two bytes of program memory space are reserved for each interrupt vector. The two vectors are located at addresses 0xEC to 0xEF Hex in the program memory space. The interrupts can be individually enabled or disabled using Enable Transmit Interrupt (ETI) and Enable Receive Interrupt (ERI) bits in the ENUI register.

The interrupt from the Transmitter is set pending, and remains pending, as long as both the TBMT and ETI bits are set. To remove this interrupt, software must either clear the ETI bit or write to the TBUF register (thus clearing the TBMT bit).

The interrupt from the receiver is set pending, and remains pending, as long as both the RBFL and ERI bits are set. To remove this interrupt, software must either clear the ERI bit or read from the RBUF register (thus clearing the RBFL bit).

Baud Clock Generation

The clock inputs to the transmitter and receiver sections of the UART can be individually selected to come either from an external source at the CKX pin (port L, pin L1) or from a source selected in the PSR and BAUD registers. Internally, the basic baud clock is created from the oscillator frequency through a two-stage divider chain consisting of a 1–16

Baud Clock Generation (Continued)

(increments of 0.5) prescaler and an 11-bit binary counter. (*Figure 44*) The divide factors are specified through two read/write registers shown in *Figure 45*. Note that the 11-bit Baud Rate Divisor spills over into the Prescaler Select Register (PSR). PSR is cleared upon reset.

As shown in Table XVI, a Prescaler Factor of 0 corresponds to NO CLOCK. NO CLOCK condition is the UART power down mode where the UART clock is turned off for power saving purpose. The user must also turn the UART clock off when a different baud rate is chosen.

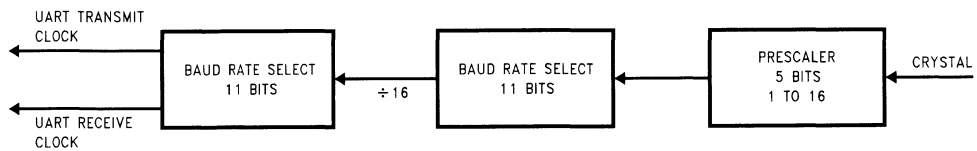
The correspondences between the 5-bit Prescaler Select and Prescaler factors are shown in Table XVI. There are many ways to calculate the two divisor factors, but one particularly effective method would be to achieve a 1.8432 MHz frequency coming out of the first stage. The 1.8432 MHz prescaler output is then used to drive the software programmable baud rate counter to create a x16 clock for the following baud rates: 110, 134.5, 150, 300, 600, 1200, 1800, 2400, 3600, 4800, 7200, 9600, 19200 and 38400 (Table XVII). Other baud rates may be created by using appropriate divisors. The x16 clock is then divided by 16 to provide the rate for the serial shift registers of the transmitter and receiver.

TABLE XVI. Prescaler Factors

Prescaler Select	Prescaler Factor
00000	NO CLOCK
00001	1
00010	1.5
00011	2
00100	2.5
00101	3
00110	3.5
00111	4
01000	4.5
01001	5

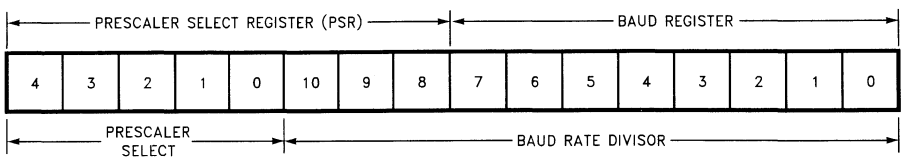
TABLE XVI. Prescaler Factors (Continued)

Prescaler Select	Prescaler Factor
01010	5.5
01011	6
01100	6.5
01101	7
01110	7.5
01111	8
100000	8.5
10001	9
10010	9.5
10011	10
10100	10.5
10101	11
10110	11.5
10111	12
11000	12.5
11001	13
11010	13.5
11011	14
11100	14.5
11101	15
11110	15.5
11111	16



TL/DD/12837–50

FIGURE 44. UART BAUD Clock Generation



TL/DD/12837–51

FIGURE 45. UART BAUD Clock Divisor Registers

Baud Clock Generation (Continued)

TABLE XVII. Baud Rate Divisors (1.8432 MHz Prescaler Output)

Baud Rate	Baud Rate Divisor − 1 (N-1)
110 (110.03)	1046
134.5 (134.58)	855
150	767
300	383
600	191
1200	95
1800	63
2400	47
3600	31
4800	23
7200	15
9600	11
19200	5
38400	2

Note: The entries in Table XVII assume a prescaler output of 1.8432 MHz. In the asynchronous mode the baud rate could be as high as 625k.

As an example, considering the Asynchronous Mode and a CKI clock of 4.608 MHz, the prescaler factor selected is:

$$4.608/1.8432 = 2.5$$

The 2.5 entry is available in Table XVI. The 1.8432 MHz prescaler output is then used with proper Baud Rate Divisor (Table II) to obtain different baud rates. For a baud rate of 19200 e.g., the entry in Table XVII is 5.

$$N - 1 = 5 \text{ (N} - 1 \text{ is the value from Table XVII)}$$

$$N = 6 \text{ (N is the Baud Rate Divisor)}$$

$$\text{Baud Rate} = 1.8432 \text{ MHz}/(16 \times 6) = 19200$$

The divide by 16 is performed because in the asynchronous mode, the input frequency to the UART is 16 times the baud rate. The equation to calculate baud rates is given below.

The actual Baud Rate may be found from:

$$BR = Fc/(16 \times N \times P)$$

Where:

BR is the Baud Rate

Fc is the CKI frequency

N is the Baud Rate Divisior (Table XVII).

P is the Prescaler Divide Factor selected by the value in the Prescaler Select Register (Table XVI)

Note: In the Synchronous Mode, the divisor 16 is replaced by two.

Example:

Asynchronous Mode:

Crystal Frequency = 5 MHz

Desired baud rate = 9600

Using the above equation N X P can be calculated first.

$$N \times P = (5 \times 106)/(16 \times 9600) = 32.552$$

Now 32.552 is divided by each Prescaler Factor (Table III) to obtain a value closet to an integer. This factor happens to be 6.5 (P = 6.5).

$$N = 32.552/6.5 = 5.008 \text{ (N} = 5)$$

The programmed value (from Table IV) should be 4 (N − 1).

Using the above values calculated for N and P:

$$BR = (5 \times 106)/(16 \times 5 \times 6.5) = 9615.384$$

$$\% \text{ error} = (9615.385 - 9600)/9600 = 0.16$$

Effect of HALT/IDLE

The UART logic is reinitialized when either the HALT or IDLE modes are entered. This reinitialization sets the TBMT flag and resets all read only bits in the UART control and status registers. Read/Write bits remain unchanged. The Transmit Buffer (TBUF) is not affected, but the Transmit Shift register (TSFT) bits are set to one. The receiver registers RBUF and RSFT are not affected.

The device will exit from the HALT/IDLE modes when the Start bit of a character is detected at the RDX (L3) pin. This feature is obtained by using the Multi-Input Wakeup scheme provided on the device.

Before entering the HALT or IDLE modes the user program must select the Wakeup source to be on the RXD pin. This selection is done by setting bit 3 of WKEN (Wakeup Enable) register. The Wakeup trigger condition is then selected to be high to low transition. This is done via the WKEDG register. (Bit 3 is one.)

If the device is halted and crystal oscillator is used, the Wakeup signal will not start the chip running immediately because of the finite start up time requirement of the crystal oscillator. The idle timer (T0) generates a fixed (256 t_c) delay to ensure that the oscillator has indeed stabilized before allowing the device to execute code. The user has to consider this delay when data transfer is expected immediately after exiting the HALT mode.

Diagnostic

Bits CHARL0 and CHARL1 in the ENU register provide a loopback feature for diagnostic testing of the UART. When these bits are set to one, the following occur: The receiver input pin (RDX) is internally connected to the transmitter output pin (TDX); the output of the Transmitter Shift Register is "looped back" into the Receive Shift Register input. In this mode, data that is transmitted is immediately received. This feature allows the processor to verify the transmit and receive data paths of the UART.

Diagnostic (Continued)

Note that the framing format for this mode is the nine bit format; one Start bit, nine data bits, and 7/8, one or two Stop bits. Parity is not generated or verified in this mode.

Attention Mode

The UART Receiver section supports an alternate mode of operation, referred to as ATTENTION Mode. This mode of operation is selected by the ATTN bit in the ENUR register. The data format for transmission must also be selected as having nine Data bits and either 7/8, one or two Stop bits.

The ATTENTION mode of operation is intended for use in networking the device with other processors, Typically in such environments the messages consists of device addresses, indicating which of several destinations should receive them, and the actual data. This Mode supports a scheme in which addresses are flagged by having the ninth bit of the data field set to a 1. If the ninth bit is reset to a zero the byte is a Data byte.

While in ATTENTION mode, the UART monitors the communication flow, but ignores all characters until an address character is received. Upon receiving an address character, the UART signals that the character is ready by setting the RBFL flag, which in turn interrupts the processor if UART Receiver interrupts are enabled. The ATTN bit is also cleared automatically at this point, so that data characters as well as address characters are recognized. Software examines the contents of the RBUF and responds by deciding either to accept the subsequent data stream (by leaving the ATTN bit reset) or to wait until the next address character is seen (by setting the ATTN bit again).

Operation of the UART Transmitter is not affected by selection of this Mode. The value of the ninth bit to be transmitted is programmed by setting XBIT9 appropriately. The value of the ninth bit received is obtained by reading RBIT9. Since this bit is located in ENUR register where the error flags reside, a bit operation on it will reset the error flags.

WATCHDOG

The device contains a WATCHDOG and clock monitor. The WATCHDOG is designed to detect the user program getting stuck in infinite loops resulting in loss of program control or "runaway" programs. The Clock Monitor is used to detect the absence of a clock or a very slow clock below a specified rate on the CKI pin.

The WATCHDOG consists of two independent logic blocks: WD UPPER and WD LOWER. WD UPPER establishes the upper limit on the service window and WD LOWER defines the lower limit of the service window.

Servicing the WATCHDOG consists of writing a specific value to a WATCHDOG Service Register named WDSVR which is memory mapped in the RAM. This value is composed of three fields, consisting of a 2-bit Window Select, a 5-bit Key Data field, and the 1-bit Clock Monitor Select field. Table XVIII shows the WDSVR register.

TABLE XVIII. WATCHDOG Service Register

Window Select		Key Data					Clock Monitor
X	X	0	1	1	0	0	Y

Bit 7 Bit 0

The lower limit of the service window is fixed at 2048 instruction cycles. Bits 7 and 6 of the WDSVR register allow the user to pick an upper limit of the service window.

Table XVIIII shows the four possible combinations of lower and upper limits for the WATCHDOG service window. This flexibility in choosing the WATCHDOG service window prevents any undue burden on the user software.

TABLE XIX. WATCHDOG Service Window Select

WDSVR Bit 7	WDSVR Bit 6	Service Window (Lower-Upper Limits)
0	0	2k–8k t_C Cycles
0	1	2k–16k t_C Cycles
1	0	2k–32k t_C Cycles
1	1	2k–64k t_C Cycles

Bits 5, 4, 3, 2 and 1 of the WDSVR register represent the 5-bit Key Data field. The key data is fixed at 01100. Bit 0 of the WDSVR Register is the Clock Monitor Select bit.

Clock Monitor

The Clock Monitor aboard the device can be selected or deselected under program control. The Clock Monitor is guaranteed not to reject the clock if the instruction cycle clock ($1/t_C$) is greater or equal to 10 kHz. This equates to a clock input rate on CKI of greater or equal to 100 kHz.

WATCHDOG Operation

The WATCHDOG and Clock Monitor are disabled during reset. The device comes out of reset with the WATCHDOG armed, the WATCHDOG Window Select (bits 6, 7 of the WDSVR Register) set, and the Clock Monitor bit (bit 0 of the WDSVR Register) enabled. Thus, a Clock Monitor error will occur after coming out of reset, if the instruction cycle clock frequency has not reached a minimum specified value, including the case where the oscillator fails to start.

The WDSVR register can be written to only once after reset and the key data (bits 5 through 1 of the WDSVR Register) must match to be a valid write. This write to the WDSVR register involves two irrevocable choices: (i) the selection of the WATCHDOG service window (ii) enabling or disabling of the Clock Monitor. Hence, the first write to WDSVR Register involves selecting or deselecting the Clock Monitor, select the WATCHDOG service window and match the WATCHDOG key data. Subsequent writes to the WDSVR register will compare the value being written by the user to the WATCHDOG service window value and the key data (bits 7 through 1) in the WDSVR Register. Table XX shows the sequence of events that can occur.

WATCHDOG Operation (Continued)

The user must service the WATCHDOG at least once before the upper limit of the service window expires. The WATCHDOG may not be serviced more than once in every lower limit of the service window. The user may service the WATCHDOG as many times as wished in the time period between the lower and upper limits of the service window. The first write to the WDSVR Register is also counted as a WATCHDOG service.

The WATCHDOG has an output pin associated with it. This is the WDOUT pin, on pin 1 of the Port G. WDOUT is active low. The WDOUT pin is in the high impedance state in the inactive state. Upon triggering the WATCHDOG, the logic will pull the WDOUT (G1) pin low for an additonal 16 t_C–32 t_C cycle after the signal level on WDOUT pin goes below the lower Schmitt trigger threshold. After this delay, the device will stop forcing the WDOUT output low.

The WATCHDOG service window will restart when the WDOUT pin goes high. It is recommended that the user tie the WDOUT pin back to V_{CC} through a resistor in order to pull WDOUT high.

A WATCHDOG service while the WDOUT signal is active will be ignored. The state of the WDOUT pin is not guaranteed on reset, but if the powers up low then the WATCHDOG will time out and WDOUT will enter high impedance state.

The Clock Monitor forces the G1 pin low upon detecting a clock frequency error. The Clock Monitor error will continue until the clock frequency has reached the minimum specified value, after which the G1 output will enter the high impedance TRI-STATE mode following 16 t_C–32 t_C clock cycles. The Clock Monitor generates a continual Clock Monitor error if the oscillator fails to start, or fails to reach the minimum specified frequency. The specification for the Clock Monitor is as follows:

$1/t_C > 10$ kHz—No clock rejection.

$1/t_C < 10$ Hz—Guaranteed clock rejection.

WATCHDOG AND CLOCK MONITOR SUMMARY

The following salient points regarding the COP888 WATCHDOG and CLOCK MONITOR should be noted:

- Both the WATCHDOG and Clock Monitor detector circuits are inhibited during RESET.
- Following RESET, the WATCHDOG and CLOCK MONITOR are both enabled, with the WATCHDOG having the maximum service window selected.
- The WATCHDOG service window and Clock Monitor enable/disable option can only be changed once, during the initial WATCHDOG service following RESET.
- The initial WATCHDOG service must match the key data value in the WATCHDOG Service register WDSVR in order to avoid a WATCHDOG error.
- Subsequent WATCHDOG services must match all three data fields in WDSVR in order to avoid WATCHDOG errors.
- The correct key data value cannot be read from the WATCHDOG Service register WDSVR. Any attempt to read this key data value of 01100 from WDSVR will read as key data value of all 0's.
- The WATCHDOG detector circuit is inhibited during both the HALT and IDLE modes.
- The Clock Monitor detector circuit is active during both the HALT and IDLE modes. Consequently, the device inadvertently entering the HALT mode will be detected as a Clock Monitor error (provided that the Clock Monitor enable option has been selected by the program).
- With the single-pin R/C oscillator mask option selected and the CLKDLY bit reset, the WATCHDOG service window will resume following HALT mode from where it left off before entering the HALT mode.
- With the crystal oscillator mask option selected, or with the single-pin R/C oscillator mask option selected and the CLKDLY bit set, the WATCHDOG service window will be set to its selected value from WDSVR following HALT. Consequently, the WATCHDOG should not be serviced for at least 2048 instruction cycles following HALT, but must be serviced within the selected window to avoid a WATCHDOG error.
- The IDLE timer T0 is not initialized with RESET.
- The user can sync in to the IDLE counter cycle with an IDLE counter (T0) interrupt or by monitoring the T0PND flag. The T0PND flag is set whenever the thirteenth bit of the IDLE counter toggles (every 4096 instruction cycles). The user is responsible for resetting the T0PND flag.
- A hardware WATCHDOG service occurs just as the device exits the IDLE mode. Consequently, the WATCHDOG should not be serviced for at least 2048 instruction cycles following IDLE, but must be serviced within the selected window to avoid a WATCHDOG error.
- Following RESET, the initial WATCHDOG service (where the service window and the CLOCK MONITOR enable/disable must be selected) may be programmed anywhere within the maximum service window (65,536 instruction cycles) initialized by RESET. Note that this initial WATCHDOG service may be programmed within the intitial 2048 instruction cycles without causing a WATCHDOG error.

TABLE XX. WATCHDOG Service Actions

Key Data	Window Data	Clock Monitor	Action
Match	Match	Match	Valid Service: Restart Service Window
Don't Care	Mismatch	Don't Care	Error: Generate WATCHDOG Output
Mismatch	Don't Care	Don't Care	Error: Generate WATCHDOG Output
Don't Care	Don't Care	Mismatch	Error: Generate WATCHDOG Output

Memory Map

All RAM, ports and registers (except A and PC) are mapped into data memory address space.

Address	Contents
0000 to 006F	On-Chip RAM bytes (112 bytes)
0070 to 007F	Unused RAM Address Space (Reads as All Ones)
0080	PORTMD, Port M Data Register
0081	PORTMC, Port M Configuration Register
0082	PORTMP, Port M Input Pins (Read Only)
0083	reserved for Port M
0084	MMIWU Edge Select Register (MWKEDG)
0085	MMIWU Enable Register (MWKEN)
0086	MMIWU Pending Register (MWKPND)
0087	reserved for MMIWU
0088	PORTND, Port N Data Register
0089	PORTNC, Port N Configuration Register
008A	PORTNP, Port N Input Pins (Read Only)
008B	PORTNX, Port N Alternate Function Enable
008C to 008F	Unused RAM Address Space (Reads Undefined Data)
0090	PORTED, Port E Data Register
0091	PORTEC, Port E Configuration Register
0092	PORTEP, Port E Input Pins (Read Only)
0093	reserved for Port E
0094	PORTFD, Port F Data Register
0095	PORTFC, Port F Configuration Register
0096	PORTFP, Port F Input Pins (Read Only)
0097	reserved for Port F
0098	SPICNTL, SPI Control Register
0099	SPISTAT, SPI Status Register
009A	SPIRXD, SPI Current Receive Data (Read Only)
009B	SPITXD, SPI Transmit Data
009c to 009F	unused
00A0	TXD1, Transmit 1 Data
00A1	TXD2, Transmit 2 Data
00A2	TDLC, Transmit Data Length Code and Identifier Low
00A3	TID, Transmit Identifier High
00A4	RXD1, Receive Data 1
00A5	RXD2, Receive Data 2
00A6	RIDL, Receive Data Length Code
00A7	RID, Receive Identify HIgh
00A8	CSCAL, CAN Prescaler
00A9	CTIM, Bus Timing Register
00AA	CBUS, Bus Control Register
00AB	TCNTL. Transmit/Receive Control Register
00AC	RTSTAT Receive/Transmit Status Register
00AD	TEC, Transmit Error Count Register
00AE	REC, Receive Error Count Register
00AF	PLATST, CAN Bit Stream Processor Test Register

Address	Contents
00B8	UART Transmit Buffer (TBUF)
00B9	UART Receive Buffer (RBUF)
00BA	UART Control Status (ENU)
00BB	UART Receive Control Status (ENUR)
00BC	UART Interrupt and Clock (ENUI)
00BD	UART Baud Register (BAUD)
00BE	UART Prescaler Register (PSR)
00BF	reserved for UART
00C0	Timer T2 Lower Byte (TMR2LO)
00C1	Timer T2 Upper Byte (TMR2HI)
00C2	Timer T2 Autoload Register T2RA Lower Byte (T2RALO)
00C3	Timer T2 Autoload Register T2RA Upper Byte (T2RAHI)
00C4	Timer T2 Autoload Register T2RB Lower Byte (T2RBLO)
00C5	Timer T2 Autoload Register T2RB Upper Byte (T2RBHI)
00C6	Timer T2 Control Register (T2CNTRL)
00C7	WATCHDOG Service Register (Reg:WDSVR)
00C8	LMIWU Edge Select Register (LWKEDG)
00C9	LMIWU Enable Register (LWKEN)
00CA	LLMIWU Pending Register (LWKPND)
00CB	A/D Converter Control Register (Reg:ENAD)
00CC	A/D Converter Result Register (Reg:ADRSLT)
00CD to 00CE	Reserved
00CF	IDLE Timer Control Register (Reg:ITMR)
00D0	PORTLD, Port L Data Register
00D1	PORTLC, Port L Configuration Register
00D2	PORTLP, Port L Input Pins (Read Only)
00D3	Reserved for Port L
00D4	PORTGD, Port G Data Register
00D5	PORTGC, Port G Configuration Register
00D6	PORTGP, Port G Input Pins (Read Only)
00D7	Port I Input Pins (Read Only)
00D8	Port CD, Port C Data Register
00D9	Port CC, Port C Configuration Register
00DA	Port CP, Port C Input Pins (Read Only)
00DB	Reserved for Port C
00DC	Port D
00DD to 00DF	Reserved for Port D
00E0 to 00E5	Reserved for EE Control Registers
00E6	Timer T1 Autoload Register T1RB Lower Byte (T1BRLO)
00E7	Timer T1 Autoload Register T1RB Upper Byte (T1BRHI)
00E8	ICNTRL Register
00E9	MICROWIRE/PLUS Shift Register (SOIR)
00EA	Timer T1 Lower Byte (TMR1LO)
00EB	Timer T1 Upper Byte (TMR1HI)
00EC	Timer T1 Autoload Register T1RA Lower Byte (T1RALO)

Memory Map (Continued)

Address	Contents
00ED	Timer T1 Autoload Register T1RA Upper Byte (T1RAHI)
00EE	CNTRL, Control Register
00EF	PSW, Processor Status Word Register
00F0 to 00FB	On-Chip RAM Mapped as Registers
00FC	X Register
00FD	SP Register
00FE	B Register
00FF	S Register
0100 to 013F	On-Chip RAM Bytes (64 Bytes)
Reading memory locations 0070H–007FH will return all ones. Reading unused memory locations 00xxH–00xxH will return undefined data.	

Addressing Modes

There are ten addressing modes, six for operand addressing and four for transfer of control.

OPERAND ADDRESSING MODES

Register Indirect

This is the "normal" addressing mode. The operand is the data memory addressed by the B pointer or X pointer.

Register Indirect (with auto post increment or decrement of pointer)

This addressing mode is used with the LD and X instructions. The operand is the data memory addressed by the B pointer or X pointer. This is a register indirect mode that automatically post increments or decrements the B or X register after executing the instruction.

Direct

The instruction contains an 8-bit address field that directly points to the data memory for the operand.

Immediate

The instruction contains an 8-bit immediate field as the operand.

Short Immediate

This addressing mode is used with the Load B Immediate instruction. The instruction contains a 4-bit immediate field as the operand.

Indirect

This addressing mode is used with the LAID instruction. The contents of the accumulator are used as a partial address (lower 8 bits of PC) for accessing a data operand from the program memory.

TRANSFER OF CONTROL ADDRESSING MODES

Relative

This mode is used for the JP instruction, with the instruction field being added to the program counter to get the new program location. JP has a range from −31 to +32 to allow a 1-byte relatie jump (JP + 1 is implemented by a NOP instruction). There are no "pages" when using JP, since all 15 bits of PC are used.

Absolute

The mode is used with the JMP and JSR instructions, with the instruction field of 12 bits replacing the lower 12 bits of the program counter (PC). This allows jumping to any location in the current 4k program memory segment.

Absolute Long

This mode is used with the JMPL and JSRL instructions, with the instruction field of 15 bits replacing the entire 15 bits of the program counter (PC). This allows jumping to any location up to 32k in the program memory space.

Indirect

This mode is used with the JID instruction. The contents of the accumulator are used as a partial address (lower 8 bits of PC) for accessing a location in the program memory. The contents of this program memory location serve as a partial address (lower 8 bits of PC) for the jump to the next instruction.

Note: The VIS is a special case of the Indirect Transfer of Control addressing mode, where the double byte vector associated with the interrupt is transferred from adjacent addresses in the program memory into the program counter (PC) in order to jump to the associated interrupt service routine.

Instruction Set

Register and Symbol Definition

Registers	
A	8-Bit Accumulator Register
B	8-Bit Address Register
X	8-Bit Address Register
SP	8-Bit Stack Pointer Register
PC	15-Bit Program Counter Register
PU	Upper 7 Bits of PC
PL	Lower 8 Bits of PC
C	1 Bit of PSW Register for Carry
HC	1 Bit of PSW Register for Half Carry
GIE	1 Bit of PSW Register for Global Interrupt Enable
VU	Interrupt Vector Upper Byte
VL	Interrupt Vector Lower Byte

Symbols	
[B]	Memory Indirectly Addressed by B Register
[X]	Memory Indirectly Addressed by X Register
MD	Direct Addressed Memory
Mem	Direct Addressed Memory or [B]
Meml	Direct Addressed Memory or [B] or Immediate Data
Imm	8-Bit Immediate Data
Reg	Register Memory: Addresses F0 to FF (Includes B, X and SP)
Bit	Bit Number (0 to 7)
←	Loaded with
⟷	Exchanged with

Instruction Set (Continued)

ADD	A,Meml	ADD	A ← A + Meml
ADC	A,Meml	ADD with Carry	A ← A + Meml + C, C ← Carry, HC ← Half Carry,
SUBC	A,Meml	Subtract with Carry	A ← A − Meml + C, C ← Carry, HC ← Half Carry
AND	A,Meml	Logical AND	A ← A and $\overline{\text{Meml}}$
ANDSZ	A,Imm	Logical AND Immed., Skip if Zero	Skip next if (A and Imm) = 0
OR	A,Meml	Logical OR	A ← A or Meml
XOR	A,Meml	Logical EXclusive OR	A ← A xor Meml
IFEQ	MD,Imm	IF EQual	Compare MD and Imm, Do next if MD = Imm
IFEQ	A,Meml	IF EQual	Compare A and Meml, Do next if A = Meml
IFNE	A,Meml	IF Not Equal	Compare A and Meml, Do next if A ≠ Meml
IFGT	A,Meml	IF Greater Than	Compare A and Meml, Do next if A > Meml
IFBNE	#	IF B Not Equal	Do next if lower 4 bits of B ≠ Imm
DRSZ	Reg	Decrement Reg., Skip if Zero	Reg ← Reg − 1, Skip if Reg = 0
SBIT	#,Mem	Set BIT	1 to bit, Mem (bit = 0 to 7 immediate)
RBIT	#,Mem	Reset BIT	0 to bit, Mem
IFBIT	#,Mem	IF BIT	If bit in A or Mem is true do next instruction
RPND		Reset PeNDing Flag	Reset Software Interrupt Pending Flag
X	A,Mem	EXchange A with Memory	A ⟷ Mem
X	A,[X]	EXchange A with Memory [X]	A ⟷ [X]
LD	A,Meml	LoaD A with Memory	A ← Meml
LD	A,[X]	LoaD A with Memory [X]	A ← [X]
LD	B,Imm	LoaD B with Immed.	B ← Imm
LD	Mem,Imm	LoaD Memory Immed.	Mem ← Imm
LD	Reg,Imm	LoaD Register Memory Immed.	Reg ← Imm
X	A, [B]	EXchange A with Memory [B]	A ⟷ [B], (B ← B 1)
X	A, [X]	EXchange A with Memory [X]	A ⟷ [X], (X ← X 1)
LD	A, [B]	LoaD A with Memory [B]	A ← [B], (B ← B 1)
LD	A, [X]	LoaD A with Memory [X]	A ← [X], (X ← X 1)
LD	[B],Imm	LoaD Memory [B] Immed.	[B] ← Imm, (B ← B 1)
CLR	A	CLeaR A	A ← 0
INC	A	INCrement A	A ← A + 1
DEC	A	DECrement A	A ← A − 1
LAID		Load A InDirect from ROM	A ← ROM (PU,A)
DCOR	A	Decimal CORrect A	A ← BCD correction of A (follows ADC, SUBC)
RRC	A	Rotate A Right thru C	C → A7 → . . . → A0 → C
RLC	A	Rotate A Left thru C	C ← A7 ← . . . ← A0 ← C
SWAP	A	SWAP nibbles of A	A7 . . . A4 ⟷ A3 . . . A0
SC		Set C	C ← 1, HC ← 1
RC		Reset C	C ← 0, HC ← 0
IFC		IF C	IF C is true, do next instruction
IFNC		IF Not C	If C is not true, do next instruction
POP	A	POP the stack into A	SP ← SP + 1, A ← [SP]
PUSH	A	PUSH A onto the stack	[SP] ← A, SP ← SP − 1
VIS		Vector to Interrupt Service Routine	PU ← [VU], PL ← [VL]
JMPL	Addr.	Jump absolute Long	PC ← ii (ii = 15 bits, 0 to 32k)
JMP	Addr.	Jump absolute	PC9 . . . 0 ← i (i = 12 bits)
JP	Disp.	Jump relative short	PC ← PC + r (r is −31 to +32, except 1)
JSRL	Addr.	Jump SubRoutine Long	[SP] ← PL, [SP−1] ← PU,SP−2, PC ← ii
JSR	Addr.	Jump SubRoutine	[SP] ← PL, [SP−1] ← PU,SP−2, PC9 . . . 0 ← i
JID		Jump InDirect	PL ← ROM (PU,A)
RET		RETurn from subroutine	SP + 2, PL ← [SP], PU ← [SP−1]
RETSK		RETurn and SKip	SP + 2, PL ← [SP],PU ← [SP−1]
RETI		RETurn from Interrupt	SP + 2, PL ← [SP],PU ← [SP−1],GIE ← 1
INTR		Generate an Interrupt	[SP] ← PL, [SP−1] ← PU, SP−2, PC ← 0FF
NOP		No OPeration	PC ← PC + 1

INSTRUCTION EXECUTION TIME

Most instructions are single byte (with immediate addressing mode instructions taking two bytes).

Most single byte instructions take one cycle time to execute.

Skipped instructions require x number of cycles to be skipped, where x equals the number of bytes in the skipped instruction opcode.

See the BYTES and CYCLES per INSTRUCTION table for details.

Bytes and Cycles per Instruction

The following table shows the number of bytes and cycles for each instruction in the format of byte/cycle.

Arithmetic and Logic Instructions

	[B]	Direct	Immed.
ADD	1/1	3/4	2/2
ADC	1/1	3/4	2/2
SUBC	1/1	3/4	2/2
AND	1/1	3/4	2/2
OR	1/1	3/4	2/2
XOR	1/1	3/4	2/2
IFEQ	1/1	3/4	2/2
IFGT	1/1	3/4	2/2
IFBNE	1/1		
DRSZ		1/3	
SBIT	1/1	3/4	
RBIT	1/1	3/4	
IFBIT	1/1	3/4	

RPND	1/1

Instructions Using A and C

CLRA	1/1
INCA	1/1
DECA	1/1
LAID	1/3
DCORA	1/1
RRCA	1/1
RLCA	1/1
SWAPA	1/1
SC	1/1
RC	1/1
IFC	1/1
IFNC	1/1
PUSHA	1/3
POPA	1/3
ANDSZ	2/2

Transfer of Control Instructions

JMPL	3/4
JMP	2/3
JP	1/3
JSRL	3/5
JSR	2/5
JID	1/3
VIS	1/5
RET	1/5
RETSK	1/5
RETI	1/5
INTR	1/7
NOP	1/1

Memory Transfer Instructions

	Register Indirect		Direct	Immed.	Register Indirect Auto Incr. and Decr.		
	[B]	[X]			[B+, B−]	[X+, X−]	
X A,*	1/1	1/3	2/3		1/2	1/3	
LD A,*	1/1	1/3	2/3	2/2	1/2	1/3	
LD B, Imm				1/1			(IF B < 16)
LD B, Imm				2/3			(IF B > 15)
LD Mem, Imm	2/2		3/3		2/2		
LD Reg, Imm			2/3				
IFEQ MD, Imm			3/3				

* = > Memory location addressed by B or X or directly.

5

COP888 Family Opcode Table

Upper Nibble																Lower Nibble
F	E	D	C	B	A	9	8	7	6	5	4	3	2	1	0	
JP−15	JP−31	LD 0F0, #i	DRSZ 0F0	RRCA	RC	ADC A,#i	ADC A,[B]	IFBIT 0,[B]	ANDSZ A, #i	LD B,#0F	IFBNE 0	JSR x000−x0FF	JMP x000−x0FF	JP+17	INTR	0
JP−14	JP−30	LD 0F1, #i	DRSZ 0F1	*	SC	SUBC A, #i	SUBC A,[B]	IFBIT 1,[B]	*	LD B,#0E	IFBNE 1	JSR x100−x1FF	JMP x100−x1FF	JP+18	JP+2	1
JP−13	JP−29	LD 0F2, #i	DRSZ 0F2	X A,[X+]	X A,[B+]	IFEQ A,#i	IFEQ A,[B]	IFBIT 2,[B]	*	LD B,#0D	IFBNE 2	JSR x200−x2FF	JMP x200−x2FF	JP+19	JP+3	2
JP−12	JP−28	LD 0F3, #i	DRSZ 0F3	X A,[X−]	X A,[B−]	IFGT A,#i	IFGT A,[B]	IFBIT 3,[B]	*	LD B,#0C	IFBNE 3	JSR x300−x3FF	JMP x300−x3FF	JP+20	JP+4	3
JP−11	JP−27	LD 0F4, #i	DRSZ 0F4	VIS	LAID	ADD A,#i	ADD A,[B]	IFBIT 4,[B]	CLRA	LD B,#0B	IFBNE 4	JSR x400−x4FF	JMP x400−x4FF	JP+21	JP+5	4
JP−10	JP−26	LD 0F5, #i	DRSZ 0F5	RPND	JID	AND A,#i	AND A,[B]	IFBIT 5,[B]	SWAPA	LD B,#0A	IFBNE 5	JSR x500−x5FF	JMP x500−x5FF	JP+22	JP+6	5
JP−9	JP−25	LD 0F6, #i	DRSZ 0F6	X A,[X]	X A,[B]	XOR A,#i	XOR A,[B]	IFBIT 6,[B]	DCORA	LD B,#09	IFBNE 6	JSR x600−x6FF	JMP x600−x6FF	JP+23	JP+7	6
JP−8	JP−24	LD 0F7, #i	DRSZ 0F7	*	*	OR A,#i	OR A,[B]	IFBIT 7,[B]	PUSHA	LD B,#08	IFBNE 7	JSR x700−x7FF	JMP x700−x7FF	JP+24	JP+8	7
JP−7	JP−23	LD 0F8, #i	DRSZ 0F8	NOP	RLCA	LD A,#i	IFC	SBIT 0,[B]	RBIT 0,[B]	LD B,#07	IFBNE 8	JSR x800−x8FF	JMP x800−x8FF	JP+25	JP+9	8
JP−6	JP−22	LD 0F9, #i	DRSZ 0F9	IFNE A,[B]	IFEQ Md,#i	IFNE A,#i	IFNC	SBIT 1,[B]	RBIT 1,[B]	LD B,#06	IFBNE 9	JSR x900−x9FF	JMP x900−x9FF	JP+26	JP+10	9
JP−5	JP−21	LD 0FA, #i	DRSZ 0FA	LD A,[X+]	LD A,[B+]	LD [B+],#i	INCA	SBIT 2,[B]	RBIT 2,[B]	LD B,#05	IFBNE 0A	JSR xA00−xAFF	JMP xA00−xAFF	JP+27	JP+11	A
JP−4	JP−20	LD 0FB, #i	DRSZ 0FB	LD A,[X−]	LD A,[B−]	LD [B−],#i	DECA	SBIT 3,[B]	RBIT 3,[B]	LD B,#04	IFBNE 0B	JSR xB00−xBFF	JMP xB00−xBFF	JP+28	JP+12	B
JP−3	JP−19	LD 0FC, #i	DRSZ 0FC	LD Md,#i	JMPL	X A,Md	POPA	SBIT 4,[B]	RBIT 4,[B]	LD B,#03	IFBNE 0C	JSR xC00−xCFF	JMP xC00−xCFF	JP+29	JP+13	C
JP−2	JP−18	LD 0FD, #i	DRSZ 0FD	DIR	JSRL	LD A,Md	RETSK	SBIT 5,[B]	RBIT 5,[B]	LD B,#02	IFBNE 0D	JSR xD00−xDFF	JMP xD00−xDFF	JP+30	JP+14	D
JP−1	JP−17	LD 0FE, #i	DRSZ 0FE	LD A,[X]	LD A,[B]	LD [B],#i	RET	SBIT 6,[B]	RBIT 6,[B]	LD B,#01	IFBNE 0E	JSR xE00−xEFF	JMP xE00−xEFF	JP+31	JP+15	E
JP−0	JP−16	LD 0FF, #i	DRSZ 0FF	*	*	LD B,#i	RETI	SBIT 7,[B]	RBIT 7,[B]	LD B,#00	IFBNE 0F	JSR xF00−xFFF	JMP xF00−xFFF	JP+32	JP+16	F

Where,

i is the immediate data

Md is a directly addressed memory location

* is an unused opcode

Note: The opcode 60 Hex is also the opcode for IFBIT #i,A

Mask Options

The COP684E and COP884EB mask programmable options are shown below. The options are programmed at the same time as the ROM pattern submission.

OPTION 1: CLOCK CONFIGURATION

= 1 Crystal Oscillator (CKI/10)

G7 (CKO) is clock generator output to crystal/resonator

CKI is the clock input

OPTION 2: HALT

= 1 Enable HALT mode

OPTION 3: BONDING OPTIONS

= 1 68-Pin PLCC

= 2 44-Pin PLCC

The chip can be driven by a clock input on the CKI input pin which can be between DC and 10 MHz. The CKO output clock is on pin G7. The CKI input frequency is divided down by 10 to produce the instruction cycle clock ($1/t_C$).

Development Support

SUMMARY

- iceMASTER: IM-COP8/400—Full feature in-circuit emulation for all COP8 products. A full set of COP8 Basic and Feature Family device and package specific probes are available.
- COP8 Debug Module: Moderate cost in-circuit emulation and development programming unit.
- COP8 Evaluation and Programming Unit: EPU-COP888GG—low cost in-circuit simulation and development programming unit.
- Assembler: COP8-DEV-IBMA. A DOS installable cross development Assembler, Linker, Librarian and Utility Software Development Tool Kit.
- C Compiler: COP8C. A DOS installable cross development Software Tool Kit.
- OTP/EPROM Programmer Support: Covering needs from engineering prototype, pilot production to full production environments.

Development Support (Continued)

IceMASTER (IM) IN-CIRCUIT EMULATION

The iceMASTER IM-COP8/400 is a full feature, PC based, in-circuit emulation tool development and marketed by MetaLink Corporation to support the whole COP8 family of products. National is a resale vendor for these products.

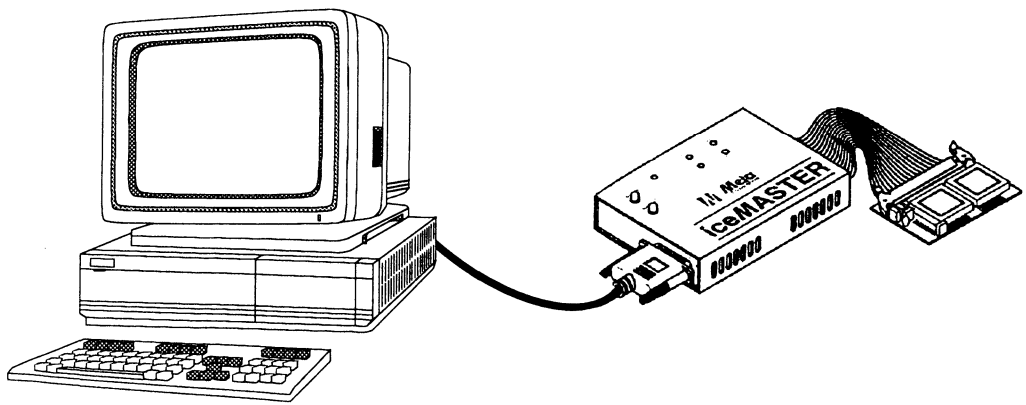
See *Figure 46* for configuration.

The iceMASTER IM-COP8/400 with its device specific COP8 Probe provides a rich feature set for developing, testing and maintaining product:

- Real-time in-circuit emulation; full 2.4V–5.5V operation range, full DC-10 MHz clock. Chip options are programmable or jumper selectable.
- Direct connection to application board by package compatible socket or surface mount assembly.
- Full 32-kbyte of loadable programming space that overlays (replaces) the on-chip ROM or EPROM. On-chip RAM and I/O blocks are used directly or recreated on the probe as necessary.
- Full 4k frame synchronous trace memory. Address, instruction, and eight unspecified, circuit connectable trace lines. Display can be HLL source (e.g., C source), assembly or mixed.
- A full 64k hardware configurable break, trace on, trace off control, and pass count increment events.
- Tool set integrated interactive symbolic debugger—supports both assembler (COFF) and C Compiler (.COD) linked object formats.
- Real time performance profiling analysis; selectable bucket definition.
- Watch windows, content updated automatically at each execution break.
- Instruction by instruction memory/register changes displayed on source window when in single step operation.
- Single base unit and debugger software reconfigurable to support the entire COP8 family; only the probe personality needs to change. Debugger software is processor customized, and reconfigured from a master model file.
- Processor specific symbolic display of registers and bit level assignments, configured from master model file.
- Halt/Idle mode notification.
- On-Line HELP customized to specific processor using master model file.
- Includes a copy of COP8-DEV-IBMA assembler and linker SDK.

IM Order-Information

Base Unit	
IM-COP8/400-1	iceMASTER Base Unit, 110V Power Supply
IM-COP8/400-2	iceMASTER Base Unit, 220V Power Supply
iceMASTER Probe	
MHW-888EB44PWPC	44 PLCC
MHW-888EB68PWPC	68 PLCC



TL/DD/12837–52

FIGURE 46. COP8 iceMASTER Environment

Development Support (Continued)

iceMASTER DEBUG MODULE (DM)

The iceMASTER Debug Module is a PC based, combination in-circuit emulation tool and COP8 based OTP/EPROM programming tool developed and marketed by MetaLink Corporation to support the whole COP8 family of products. National is a resale vendor for these products.

See *Figure 47* for configuration.

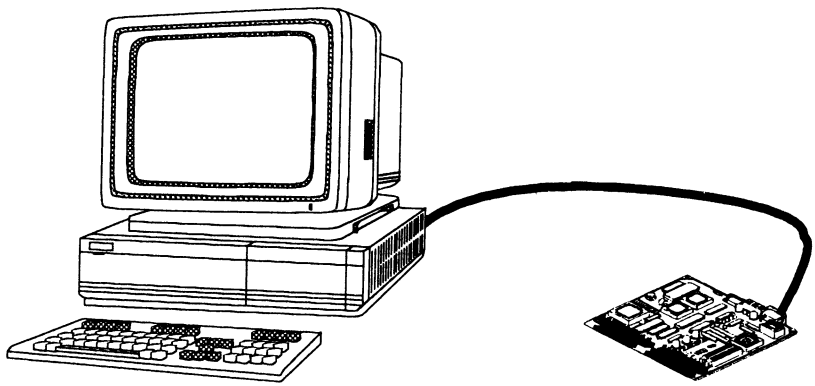
The iceMASTER Debug Module is a moderate cost development tool. It has the capability of in-circuit emulation for a specific COP8 microcontroller and in addition serves as a programming tool for COP8 OTP and EPROM product families. Summary of features is as follows:

- Real-time in-circuit emulation; full operating voltage range operation, full DC-10 MHz clock.
- All processor I/O pins can be cabled to an application development board with package compatible cable to socket and surface mount assembly.
- Full 32 kbyte of loadable programming space that overlays (replaces) the on-chip ROM or EPROM. On-chip RAM and I/O blocks are used directly or recreated as necessary.
- 100 frames of synchronous trace memory. The display can be HLL source (C source), assembly or mixed. The most recent history prior to a break is available in the trace memory.
- Configured break points; uses INTR instruction which is modestly intrusive.
- Software—only supported features are selectable.
- Tool set integrated interactive symbolic debugger—supports both assembler (COFF) and C Compiler (.COD) SDK linked object formats.
- Instruction by instruction memory/register changes displayed when in single step operation.
- Debugger software is processor customized, and reconfigured from a master model file.
- Processor specific symbolic display of registers and bit level assignments, configured from master model file.
- Halt/Idle mode notification.
- Programming menu supports full product line of programmable OTP and EPROM COP8 products. Program data is taken directly from the overlay RAM.
- Programming of 44 PLCC and 68 PLCC parts requires external programming adapters.
- Includes wallmount power supply.
- On-board VPP generator from 5V input or connection to external supply supported. Requires VPP level adjustment per the family programming specification (correct level is provided on an on-screen pop-down display).
- On-line HELP customized to specific processor using master model file.
- Includes a copy of COP8-DEV-IBMA assembler and linker SDK.

DM Order-Information

Debug Module Unit	
COP8-DM/888EB	
Cable Adapters	
DM-COP8/44P	44 PLCC
DM-COP8/68P	68 PLCC

Please contact local sales office for ordering information of programming adapter.



TL/DD/12837-53

FIGURE 47. COP8-DM Environment

Development Support (Continued)

COP8 ASSEMBLER/LINKER SOFTWARE DEVELOPMENT TOOL KIT

National Semiconductor offers a relocatable COP8 macro cross assembler, linker, librarian and utility software development tool kit. Features are summarized as follows:

- Basic and Feature Family instruction set by "device" type.
- Nested macro capability.
- Extensive set of assembler directives.
- Supported on PC/DOS platform.
- Generates National standard COFF output files.
- Integrated Linker and Librarian.
- Integrated utilities to generate ROM code file outputs.
- DUMPCOFF utility.

This product is integrated as a part of MetaLink tools as a development kit, fully supported by the MetaLink debugger. It may be ordered separately or it is bundled with the MetaLink products at no additional cost.

Order Information

Assembler SDK:	
COP8-DEV-IBMA	Assembler SDK on installable 3.5" PC/DOS Floppy Disk Drive format. Periodic upgrades and most recent version is available on National's BBS and Internet.

COP8 C COMPILER

A C Compiler is developed and marketed by Byte Craft Limited. The COP8C compiler is a fully integrated development tool specifically designed to support the compact embedded configuration of the COP8 family of products.

Features are summarized as follows:

- ANSI C with some restrictions and extensions that optimize development for the COP8 embedded application.
- BITS data type extension. Register declaration #pragma with direct bit level definitions.
- C language support for interrupt routines.
- Expert system, rule based code generation and optimization.
- Performs consistency checks against the architectural definitions of the target COP8 device.
- Generates program memory code.
- Supports linking of compiled object or COP8 assembled object formats.
- Global optiomization of linked code.
- Symbolic debug load format fully source level supported by the MetaLink debugger.

SINGLE CHIP OTP/EMULATOR SUPPORT

The COP8 family is supported by single chip OTP emulators. For detailed information refer to the emulator specific datasheet and the emulator selection table below:

OTP Emulator Ordering Information

Device Number	Clock Option	Package	Emulates
COP87L88EB-XE	Crystal	44 PLCC	COP888EB
COP87L89EB-XE	Crystal	68 PLCC	COP889EB

INDUSTRY WIDE OTP/EPROM PROGRAMMING SUPPORT

Programming support, in addition to the MetaLink development tools, is provided by a full range of independent approved vendors to meet the needs from the engineering laboratory to full production.

Approved List

Manufacturer	North America	Europe	Asia
BP Microsystems	(800) 225-2102 (713) 688-4600 Fax: (713) 688-0920	+49-8152-4183 +49-8856-932616	+852-234-16611 +852-2710-8121
Data I/O	(800) 426-1045 (206) 881-6444 Fax: (206) 882-1043	+44-0734-440011	Call North America
HI-LO	(510) 623-8860	Call Asia	+886-2-764-0215 Fax: +886-2-756-6403
ICE Technology	(800) 624-8040 (919) 430-7915	+44-1226-767404 Fax: 0-1226-370-434	
MetaLink	(800) 638-2423 (602) 926-0797 Fax: (602) 693-0681	+49-80 9156 96-0 Fax: +49-80 9123 86	+852-737-1800
Systems General	(408) 263-6667	+41-1-9450300	+886-2-917-3005 Fax: +886-2-911-1283
Needhams	(916) 924-8037 Fax: (916) 924-8065		

Development Support (Continued)

AVAILABLE LITERATURE

For more information, please see the COP8 Basic Family User's Manual, Literature Number 620895, COP8 Feature Family User's Manual, Literature Number 620897 and National's Family of 8-bit Microcontrollers COP8 Selection Guide, Literature Number 630009.

DIAL-A-HELPER SERVICE

Dial-A-Helper is a service provided by the Microcontroller Applications group. The Dial-A-Helper is an Electronic Information System that may be accessed as a Bulletin Board System (BBS) via data modem, as an FTP site on the Internet via standard FTP client application or as an FTP site on the Internet using a standard Internet browser such as Netscape or Mosaic.

The Dial-A-Helper system provides access to an automated information storage and retrieval system. The system capabilities include a MESSAGE SECTION (electronic mail, when accessed as a BBS) for communications to and from the Microcontroller Applications Group and a FILE SECTION which consists of several file areas where valuable application software and utilities could be found.

DIAL-A-HELPER BBS via a Standard Modem

Modem: CANADA/U.S.: (800) NSC-MICRO
(800) 672-6427
EUROPE: (+49) 0-814-135 13 32
Baud: 14.4k
Set-up: Length: 8-Bit
Parity: None
Stop Bit: 1
Operation: 24 Hrs., 7 Days

DIAL-A-HELPER via FTP

ftp nscmicro.nsc.com
user: anonymous
password: username@yourhost.site.domain

DIAL-A-HELPER via a WorldWide Web Browser

ftp://nscmicro.nsc.com

National Semiconductor on the WorldWide Web

See us on the WorldWide Web at: http://www.national.com

CUSTOMER RESPONSE CENTER

Complete product information and technical support is available from National's customer response centers.

CANADA/U.S.:	Tel:	(800) 272-9959
	email:	support @tevm2.nsc.com
EUROPE:	email:	europe.support@nsc.com
	Deutsch Tel:	+49 (0) 180-530 85 85
	English Tel:	+49 (0) 180-532 78 32
	Français Tel:	+49 (0) 180-532 93 58
	Italiano Tel:	+49 (0) 180-534 16 80
JAPAN:	Tel:	+81-043-299-2309
S.E. ASIA:	Beijing Tel:	(+86) 10-6856-8601
	Shanghai Tel:	(+86) 21-6415-4092
	Hong Kong Tel:	(+852) 2737-1600
	Korea Tel:	(+82) 2-3771-6909
	Malaysia Tel:	(+60-4) 644-9061
	Singapore Tel:	(+65) 255-2226
	Taiwan Tel:	+886-2-521-3288
AUSTRALIA:	Tel:	(+61) 3-9558-9999
INDIA:	Tel:	(+91) 80-559-9467

National Semiconductor

COP688CL/COP684CL, COP888CL/COP884CL, COP988CL/COP984CL 8-Bit Microcontroller

General Description

The COP888 family of microcontrollers uses an 8-bit single chip core architecture fabricated with National Semiconductor's M²CMOS™ process technology. The COP888CL is a member of this expandable 8-bit core processor family of microcontrollers. (Continued)

Key Features

- Two 16-bit timers, each with two 16-bit registers supporting:
  - — Processor Independent PWM mode
  - — External Event counter mode
  - — Input Capture mode
- 4 kbytes of on-chip ROM
- 128 bytes of on-chip RAM

Additional Peripheral Features

- Idle Timer
- Multi-input Wake Up (MIWU) with optional interrupts (8)
- WATCHDOG™ and Clock Monitor logic
- MICROWIRE/PLUS™ serial I/O

I/O Features

- Memory mapped I/O
- Software selectable I/O options (TRI-STATE® Output, Push-Pull Output, Weak Pull-Up Input, High Impedance Input)
- High current outputs
- Schmitt trigger inputs on port G
- Packages:
  - — 44 PLCC with 40 I/O pins
  - — 40 DIP with 36 I/O pins
  - — 28 DIP with 24 I/O pins
  - — 28 SO with 24 I/O pins

CPU/Instruction Set Feature

- 1 μs instruction cycle time
- Ten multi-source vectored interrupts servicing
  - — External Interrupt with selectable edge
  - — Idle Timer T0
  - — Timers (Each with 2 interrupts)
  - — MICROWIRE/PLUS
  - — Multi-Input Wake Up
  - — Software Trap
  - — Default VIS (default interrupt)
- Versatile and easy to use instruction set
- 8-bit Stack Pointer (SP)—stack in RAM
- Two 8-bit Register Indirect Data Memory Pointers (B, X)

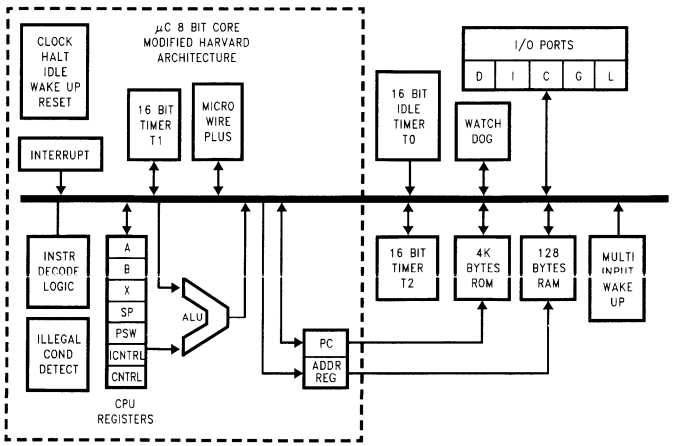
Fully Static CMOS

- Low current drain (typically < 1 μA)
- Single supply operation: 2.5V to 6.0V
- Temperature ranges: 0°C to +70°C, −40°C to +85°C, −55°C to +125°C

Development Support

- Emulation and OTP devices
- Real time emulation and full program debug offered by MetaLink Development System

Block Diagram



TL/DD/9766–1

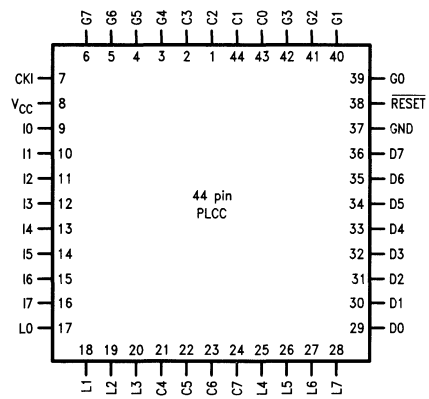
FIGURE 1. Block Diagram

General Description (Continued)

It is a fully static part, fabricated using double-metal silicon gate microCMOS technology. Features include an 8-bit memory mapped architecture, MICROWIRE/PLUS serial I/O, two 16-bit timer/counters supporting three modes (Processor Independent PWM generation, External Event counter, and Input Capture mode capabilities), and two power savings modes (HALT and IDLE), both with a multi-sourced wakeup/interrupt capability. This multi-sourced interrupt capability may also be used independent of the HALT or IDLE modes. Each I/O pin has software selectable configurations. The device operates over a voltage range of 2.5V to 6V. High throughput is achieved with an efficient, regular instruction set operating at a maximum of 1 μs per instruction rate.

Connection Diagrams

Plastic Chip Carrier

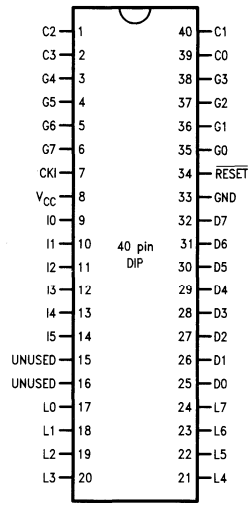


TL/DD/9766-2

Top View

Order Number COP688CL-XXX/V, COP888CL-XXX/V, COP988CL-XXX/V or COP988CLH-XXX/V
See NS Plastic Chip Package Number V44A

Dual-In-Line Package

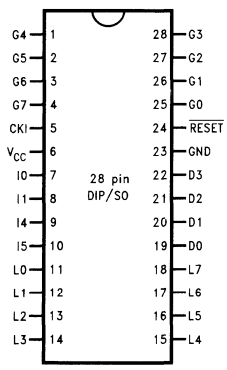


TL/DD/9766-4

Top View

Order Number COP688CL-XXX/N, COP888CL-XXX/N, COP988CL-XXX/N or COP988CLH-XXX/N
See NS Molded Package Number N40A

Dual-In-Line Package



TL/DD/9766-5

Top View

Order Number COP688CL-XXX/N, COP884CL-XXX/N, COP984CL-XXX/N or COP984CLH-XXX/N
See NS Molded Package Number N28B

Order Number COP684CL-XXX/WM, COP884CL-XXX/WM, COP984CL-XXX/WM, or COP984CLHXXX/WM
See NS Surface Mount Package Number M28B

FIGURE 2. Connection Diagrams

Connection Diagrams (Continued)

Pinouts for 28-, 40- and 44-Pin Packages

Port	Type	Alt. Fun	Alt. Fun	28-Pin Pack.	40-Pin Pack.	44-Pin Pack.
L0	I/O	MIWU		11	17	17
L1	I/O	MIWU		12	18	18
L2	I/O	MIWU		13	19	19
L3	I/O	MIWU		14	20	20
L4	I/O	MIWU	T2A	15	21	25
L5	I/O	MIWU	T2B	16	22	26
L6	I/O	MIWU		17	23	27
L7	I/O	MIWU		18	24	28
G0	I/O	INT		25	35	39
G1	WDOUT			26	36	40
G2	I/O	T1B		27	37	41
G3	I/O	T1A		28	38	42
G4	I/O	SO		1	3	3
G5	I/O	SK		2	4	4
G6	I	SI		3	5	5
G7	I/CKO	HALT RESTART		4	6	6
D0	O			19	25	29
D1	O			20	26	30
D2	O			21	27	31
D3	O			22	28	32
I0	I			7	9	9
I1	I			8	10	10
I2	I				11	11
I3	I				12	12
I4	I			9	13	13
I5	I			10	14	14
I6	I					15
I7	I					16
D4	O				29	33
D5	O				30	34
D6	O				31	35
D7	O				32	36
C0	I/O				39	43
C1	I/O				40	44
C2	I/O				1	1
C3	I/O				2	2
C4	I/O					21
C5	I/O					22
C6	I/O					23
C7	I/O					24
Unused*					16	
Unused*					15	
V_{CC}				6	8	8
GND				23	33	37
CKI				5	7	7
$\overline{\text{RESET}}$				24	34	38

* = On the 40-pin package Pins 15 and 16 must be connected to GND.

Absolute Maximum Ratings

If Military/Aerospace specified devices are required, please contact the National Semiconductor Sales Office/Distributors for availability and specifications.

Supply Voltage (V_{CC})	7V
Voltage at Any Pin	−0.3V to V_{CC} + 0.3V
Total Current into V_{CC} Pin (Source)	100 mA
Total Current out of GND Pin (Sink)	110 mA
Storage Temperature Range	−65°C to +140°C

Note: *Absolute maximum ratings indicate limits beyond which damage to the device may occur. DC and AC electrical specifications are not ensured when operating the device at absolute maximum ratings.*

DC Electrical Characteristics

COP98XCL: 0°C ≤ T_A ≤ +70°C unless otherwise specified

Parameter	Conditions	Min	Typ	Max	Units
Operating Voltage					
COP98XCL		2.5		4.0	V
COP98XCLH		4.0		6.0	V
Power Supply Ripple (Note 1)	Peak-to-Peak			0.1 V_{CC}	V
Supply Current (Note 2)					
CKI = 10 MHz	V_{CC} = 6V, t_c = 1 μs			12.5	mA
CKI = 4 MHz	V_{CC} = 4V, t_c = 2.5 μs			2.5	mA
HALT Current (Note 3)	V_{CC} = 6V, CKI = 0 MHz		<0.7	8	μA
	V_{CC} = 4V, CKI = 0 MHz		<0.4	5	μA
IDLE Current					
CKI = 10 MHz	V_{CC} = 6V, t_c = 1 μs			3.5	mA
Input Levels					
$\overline{\text{RESET}}$					
Logic High		0.8 V_{CC}			V
Logic Low				0.2 V_{CC}	V
CKI (External and Crystal Osc. Modes)					
Logic High		0.7 V_{CC}			V
Logic Low				0.2 V_{CC}	V
All Other Inputs					
Logic High		0.7 V_{CC}			V
Logic Low				0.2 V_{CC}	V
Hi-Z Input Leakage	V_{CC} = 6V	−1		+1	μA
Input Pullup Current	V_{CC} = 6V, V_{IN} = 0V	−40		−250	μA
G and L Port Input Hysteresis				0.35 V_{CC}	V
Output Current Levels					
D Outputs					
Source	V_{CC} = 4V, V_{OH} = 3.3V	−0.4			mA
	V_{CC} = 2.5V, V_{OH} = 1.8V	−0.2			mA
Sink	V_{CC} = 4V, V_{OL} = 1V	10			mA
	V_{CC} = 2.5V, V_{OL} = 0.4V	2.0			mA
All Others					
Source (Weak Pull-Up Mode)	V_{CC} = 4V, V_{OH} = 2.7V	−10		−100	μA
	V_{CC} = 2.5V, V_{OH} = 1.8V	−2.5		−33	μA
Source (Push-Pull Mode)	V_{CC} = 4V, V_{OH} = 3.3V	−0.4			mA
	V_{CC} = 2.5V, V_{OH} = 1.8V	−0.2			mA
Sink (Push-Pull Mode)	V_{CC} = 4V, V_{OL} = 0.4V	1.6			mA
	V_{CC} = 2.5V, V_{OL} = 0.4V	0.7			mA

Note 1: Rate of voltage change must be less then 0.5 V/ms.

Note 2: Supply current is measured after running 2000 cycles with a square wave CKI input, CKO open, inputs at rails and outputs open.

Note 3: The HALT mode will stop CKI from oscillating in the RC and the Crystal configurations. Test conditions: All inputs tied to V_{CC}, L and G0–G5 configured as outputs and set high. The D port set to zero. The clock monitor is disabled.

DC Electrical Characteristics $0°C \leq T_A \leq +70°C$ unless otherwise specified (Continued)

Parameter	Conditions	Min	Typ	Max	Units
TRI-STATE Leakage	$V_{CC} = 6.0V$	−1		+1	μA
Allowable Sink/Source Current per Pin					
D Outputs (Sink)				15	mA
All others				3	mA
Maximum Input Current without Latchup (Note 4)	$T_A = 25°C$			±100	mA
RAM Retention Voltage, V_r	500 ns Rise and Fall Time (Min)	2			V
Input Capacitance				7	pF
Load Capacitance on D2				1000	pF

AC Electrical Characteristics $0°C \leq T_A \leq +70°C$ unless otherwise specified

Parameter	Conditions	Min	Typ	Max	Units
Instruction Cycle Time (t_c)					
Crystal or Resonator	$4V \leq V_{CC} \leq 6V$	1		DC	μs
	$2.5V \leq V_{CC} < 4V$	2.5		DC	μs
R/C Oscillator	$4V \leq V_{CC} \leq 6V$	3		DC	μs
	$2.5V \leq V_{CC} < 4V$	7.5		DC	μs
Inputs					
t_{SETUP}	$4V \leq V_{CC} \leq 6V$	200			ns
	$2.5V \leq V_{CC} < 4V$	500			ns
t_{HOLD}	$4V \leq V_{CC} \leq 6V$	60			ns
	$2.5V \leq V_{CC} < 4V$	150			ns
Output Propagation Delay (Note 5)	$R_L = 2.2k$, $C_L = 100$ pF				
t_{PD1}, t_{PD0}					
SO, SK	$4V \leq V_{CC} \leq 6V$			0.7	μs
	$2.5V \leq V_{CC} < 4V$			1.75	μs
All Others	$4V \leq V_{CC} \leq 6V$			1	μs
	$2.5V \leq V_{CC} < 4V$			2.5	μs
MICROWIRE™ Setup Time (t_{UWS})		20			ns
MICROWIRE Hold Time (t_{UWH})		56			ns
MICROWIRE Output Propagation Delay (t_{UPD})				220	ns
Input Pulse Width					
Interrupt Input High Time		1			t_c
Interrupt Input Low Time		1			t_c
Timer Input High Time		1			t_c
Timer Input Low Time		1			t_c
Reset Pulse Width		1			μs

Note 4: Pins G6 and $\overline{\text{RESET}}$ are designed with a high voltage input network for factory testing. These pins allow input voltages greater than V_{CC} and the pins will have sink current to V_{CC} when biased at voltages greater than V_{CC} (the pins do not have source current when biased at a voltage below V_{CC}). The effective resistance to V_{CC} is 750Ω (typical). These two pins will not latch up. The voltage at the pins must be limited to less than 14V.

Note 5: The output propagation delay is referenced to the end of the instruction cycle where the output change occurs.

Absolute Maximum Ratings

If Military/Aerospace specified devices are required, please contact the National Semiconductor Sales Office/Distributors for availability and specifications.

Supply Voltage (V_{CC})	7V
Voltage at Any Pin	−0.3V to V_{CC} + 0.3V
Total Current into V_{CC} Pin (Source)	100 mA
Total Current out of GND Pin (Sink)	110 mA
Storage Temperature Range	−65°C to +140°C

Note: *Absolute maximum ratings indicate limits beyond which damage to the device may occur. DC and AC electrical specifications are not ensured when operating the device at absolute maximum ratings.*

DC Electrical Characteristics COP88XCL: $-40°C \le T_A \le +85°C$ unless otherwise specified

Parameter	Conditions	Min	Typ	Max	Units
Operating Voltage		2.5		6	V
Power Supply Ripple (Note 1)	Peak-to-Peak			0.1 V_{CC}	V
Supply Current (Note 2)					
CKI = 10 MHz	V_{CC} = 6V, t_c = 1 μs			12.5	mA
CKI = 4 MHz	V_{CC} = 4V, t_c = 2.5 μs			2.5	mA
HALT Current (Note 3)	V_{CC} = 6V, CKI = 0 MHz		<1	10	μA
IDLE Current					
CKI = 10 MHz	V_{CC} = 6V, t_c = 1 μs			3.5	mA
Input Levels					
$\overline{\text{RESET}}$					
Logic High		0.8 V_{CC}			V
Logic Low				0.2 V_{CC}	V
CKI (External and Crystal Osc. Modes)					
Logic High		0.7 V_{CC}			V
Logic Low				0.2 V_{CC}	V
All Other Inputs					
Logic High		0.7 V_{CC}			V
Logic Low				0.2 V_{CC}	V
Hi-Z Input Leakage	V_{CC} = 6V	−1		+1	μA
Input Pullup Current	V_{CC} = 6V, V_{IN} = 0V	−40		−250	μA
G and L Port Input Hysteresis				0.35 V_{CC}	V
Output Current Levels					
D Outputs					
Source	V_{CC} = 4V, V_{OH} = 3.3V	−0.4			mA
	V_{CC} = 2.5V, V_{OH} = 1.8V	−0.2			mA
Sink	V_{CC} = 4V, V_{OL} = 1V	10			mA
	V_{CC} = 2.5V, V_{OL} = 0.4V	2.0			mA
All Others					
Source (Weak Pull-Up Mode)	V_{CC} = 4V, V_{OH} = 2.7V	−10		−100	μA
	V_{CC} = 2.5V, V_{OH} = 1.8V	−2.5		−33	μA
Source (Push-Pull Mode)	V_{CC} = 4V, V_{OH} = 3.3V	−0.4			mA
	V_{CC} = 2.5V, V_{OH} = 1.8V	−0.2			mA
Sink (Push-Pull Mode)	V_{CC} = 4V, V_{OL} = 0.4V	1.6			mA
	V_{CC} = 2.5V, V_{OL} = 0.4V	0.7			mA
TRI-STATE Leakage	V_{CC} = 6.0V	−2		+2	μA

Note 1: Rate of voltage change must be less then 0.5 V/ms.

Note 2: Supply current is measured after running 2000 cycles with a square wave CKI input, CKO open, inputs at rails and outputs open.

Note 3: The HALT mode will stop CKI from oscillating in the RC and the Crystal configurations. Test conditions: All inputs tied to V_{CC}, L and G0–G5 configured as outputs and set high. The D port set to zero. The clock monitor is disabled.

DC Electrical Characteristics $-40°C \leq T_A \leq +85°C$ unless otherwise specified (Continued)

Parameter	Conditions	Min	Typ	Max	Units
Allowable Sink/Source Current per Pin					
D Outputs (Sink)				15	mA
All others				3	mA
Maximum Input Current without Latchup (Note 4)	$T_A = 25°C$			±100	mA
RAM Retention Voltage, V_r	500 ns Rise and Fall Time (Min)	2			V
Input Capacitance				7	pF
Load Capacitance on D2				1000	pF

AC Electrical Characteristics $-40°C \leq T_A \leq +85°C$ unless otherwise specified

Parameter	Conditions	Min	Typ	Max	Units
Instruction Cycle Time (t_c)					
Crystal or Resonator	$4V \leq V_{CC} \leq 6V$	1		DC	μs
	$2.5V \leq V_{CC} < 4V$	2.5		DC	μs
R/C Oscillator	$4V \leq V_{CC} \leq 6V$	3		DC	μs
	$2.5V \leq V_{CC} < 4V$	7.5		DC	μs
Inputs					
t_{SETUP}	$4V \leq V_{CC} \leq 6V$	200			ns
	$2.5V \leq V_{CC} < 4V$	500			ns
t_{HOLD}	$4V \leq V_{CC} \leq 6V$	60			ns
	$2.5V \leq V_{CC} < 4V$	150			ns
Output Propagation Delay (Note 5)	$R_L = 2.2k, C_L = 100$ pF				
t_{PD1}, t_{PD0}					
SO, SK	$4V \leq V_{CC} \leq 6V$			0.7	μs
	$2.5V \leq V_{CC} < 4V$			1.75	μs
All Others	$4V \leq V_{CC} \leq 6V$			1	μs
	$2.5V \leq V_{CC} < 4V$			2.5	μs
MICROWIRE Setup Time (t_{UWS})		20			ns
MICROWIRE Hold Time (t_{UWH})		56			ns
MICROWIRE Output Propagation Delay (t_{UPD})				220	ns
Input Pulse Width					
Interrupt Input High Time		1			t_c
Interrupt Input Low Time		1			t_c
Timer Input High Time		1			t_c
Timer Input Low Time		1			t_c
Reset Pulse Width		1			μs

Note 4: Pins G6 and $\overline{RESET}$ are designed with a high voltage input network for factory testing. These pins allow input voltages greater than V_{CC} and the pins will have sink current to V_{CC} when biased at voltages greater than V_{CC} (the pins do not have source current when biased at a voltage below V_{CC}). The effective resistance to V_{CC} is 750Ω (typical). These two pins will not latch up. The voltage at the pins must be limited to less than 14V.

Note 5: The output propagation delay is referenced to the end of the instruction cycle where the output change occurs.

Electrical Specifications

DC ELECTRICAL SPECIFICATIONS

COP688CL Absolute Specifications

Supply Voltage (V_{CC})	7V
Voltage at Any Pin	−0.3V to V_{CC} + 0.3V
Total Current into V_{CC} Pin (Source)	90 mA
Total Current out of GND Pin (Sink)	100 mA
Storage Temperature Range	−65°C to +150°C

Note: *Absolute maximum ratings indicate limits beyond which damage to the device may occur. DC and AC electrical specifications are not ensured when operating the device at absolute maximum ratings.*

DC Electrical Characteristics COP68XCL: −55°C ≤ T_A ≤ +125°C unless otherwise specified

Parameter	Conditions	Min	Typ	Max	Units
Operating Voltage		4.5		5.5	V
Power Supply Ripple (Note 1)	Peak-to-Peak			0.1 V_{CC}	V
Supply Current (Note 2)					
CKI = 10 MHz	V_{CC} = 5.5V, t_c = 1 μs			12.5	mA
CKI = 4 MHz	V_{CC} = 5.5V, t_c = 2.5 μs			5.5	mA
HALT Current (Note 3)	V_{CC} = 5.5V, CKI = 0 MHz		<10	30	μA
IDLE Current					
CKI = 10 MHz	V_{CC} = 5.5V, t_c = 1 μs			3.5	mA
CKI = 4 MHz	V_{CC} = 5.5V, t_c = 2.5 μs			2.5	mA
Input Levels					
$\overline{\text{RESET}}$					
Logic High		0.8 V_{CC}			V
Logic Low				0.2 V_{CC}	V
CKI (External and Crystal Osc. Modes)					
Logic High		0.7 V_{CC}			V
Logic Low				0.2 V_{CC}	V
All Other Inputs					
Logic High		0.7 V_{CC}			V
Logic Low				0.2 V_{CC}	V
Hi-Z Input Leakage	V_{CC} = 5.5V	−5		+5	μA
Input Pullup Current	V_{CC} = 5.5V, V_{IN} = 0V	−35		−400	μA
G and L Port Input Hysteresis				0.35 V_{CC}	V
Output Current Levels					
D Outputs					
Source	V_{CC} = 4.5V, V_{OH} = 3.8V	−0.4			mA
Sink	V_{CC} = 4.5V, V_{OL} = 1.0V	9			mA
All Others					
Source (Weak Pull-Up Mode)	V_{CC} = 4.5V, V_{OH} = 3.8V	−9.0		−140	μA
Source (Push-Pull Mode)	V_{CC} = 4.5V, V_{OH} = 3.8V	−0.4			mA
Sink (Push-Pull Mode)	V_{CC} = 4.5V, V_{OL} = 0.4V	1.4			mA
TRI-STATE Leakage	V_{CC} = 5.5V	−5.0		+5.0	μA

Note 1: Rate of voltage change must be less then 0.5 V/ms.

Note 2: Supply current is measured after running 2000 cycles with a square wave CKI input, CKO open, inputs at rails and outputs open.

Note 3: The HALT mode will stop CKI from oscillating in the RC and the Crystal configurations. Test conditions: All inputs tied to V_{CC}, L and G0–G5 configured as outputs and set high. The D port set to zero. The clock monitor is disabled.

DC Electrical Characteristics $-55°C \leq T_A \leq +25°C$ unless otherwise specified (Continued)

Parameter	Conditions	Min	Typ	Max	Units
Allowabie Sink/Source Current per Pin					
D Outputs (Sink)				12	mA
All others				2.5	mA
Maximum Input Current without Latchup (Note 4)				150	mA
RAM Retention Voltage, V_r	500 ns Rise and Fall Time (Min)	2.0			V
Input Capacitance				7	pF
Load Capacitance on D2				1000	pF

Note 1: Rate of voltage change must be less then 0.5 V/ms.

Note 2: Supply current is measured after running 2000 cycles with a square wave CKI input, CKO open, inputs at rails and outputs open.

Note 3: The HALT mode will stop CKI from oscillating in the RC and the Crystal configurations. Test conditions: All inputs tied to V_{CC}, L and G ports in the TRI-STATE mode and tied to ground, all outputs low and tied to ground. The Clock Monitor and the comparators are disabled.

AC Specifications for COP688CL

AC Electrical Characteristics $-55°C \leq T_A \leq +125°C$ unless otherwise specified

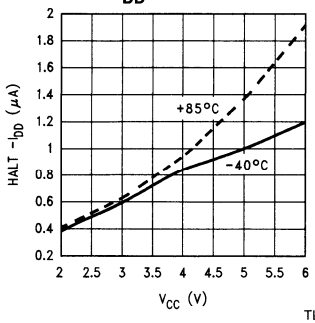
Parameter	Conditions	Min	Typ	Max	Units
Instruction Cycle Time (t_c)					
Crystal, Resonator, or External Oscillator	$V_{CC} \geq 4.5V$	1		DC	μs
R/C Oscillator (div-by 10)	$V_{CC} \geq 4.5V$	3		DC	μs
Inputs					
t_{SETUP}	$V_{CC} \geq 4.5V$	200			ns
t_{HOLD}	$V_{CC} \geq 4.5V$	60			ns
Output Propagation Delay (Note 5)	$R_L = 2.2k$, $C_L = 100$ pF				
t_{PD1}, t_{PD0}					
SO, SK	$V_{CC} \geq 4.5V$			0.7	μs
All Others	$V_{CC} \geq 4.5V$			1	μs
MICROWIRE Setup Time (t_{UWS})		20			ns
MICROWIRE Hold Time(t_{UWH})		56			ns
MICROWIRE Output Propagation Delay (t_{UPD})				220	ns
Input Pulse Width					
Interrupt Input High Time		1			t_c
Interrupt Input Low Time		1			t_c
Timer Input High Time		1			t_c
Timer Input Low Time		1			t_c
Reset Pulse Width		1			μs

Note 4: Pins G6 and $\overline{\text{RESET}}$ are designed with a high voltage input network for factory testing. These pins allow input voltages greater than V_{CC} and the pins will have sink current to V_{CC} when biased at voltages greater than V_{CC} (the pins do not have source current when biased at a voltage below V_{CC}). The effective resistance to V_{CC} is 750Ω (typical). These two pins will not latch up. The voltage at the pins must be limited to less than 14V.

Note 5: The output propagation delay is referenced to the end of the instruction cycle where the output change occurs.

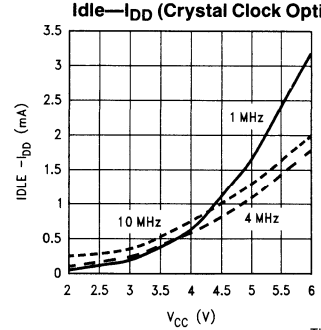
Typical Performance Characteristics ($-40°C \le T_A \le +85°C$)

Halt—I_{DD}



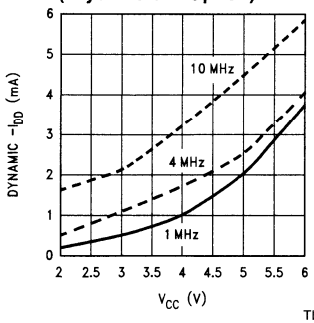
TL/DD/9766-27

Idle—I_{DD} (Crystal Clock Option)



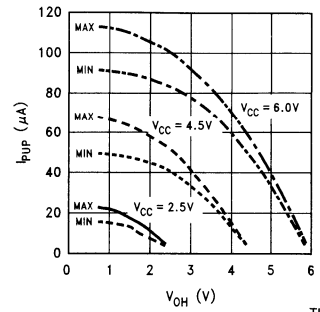
TL/DD/9766-28

Dynamic—I_{DD} vs V_{CC} (Crystal Clock Option)



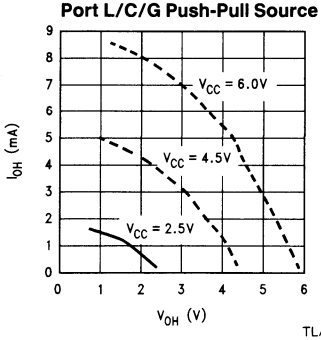
TL/DD/9766-29

Port L/C/G Weak Pull-Up Source Current



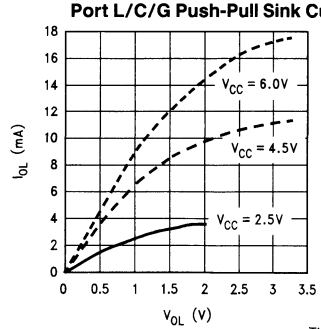
TL/DD/9766-30

Port L/C/G Push-Pull Source Current



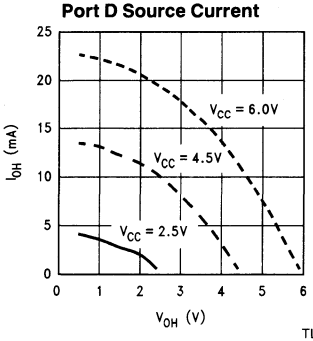
TL/DD/9766-31

Port L/C/G Push-Pull Sink Current



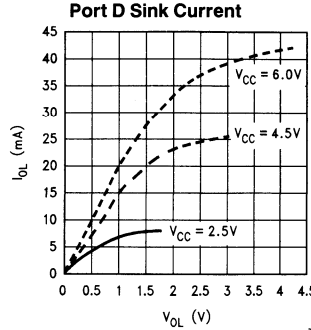
TL/DD/9766-32

Port D Source Current



TL/DD/9766-33

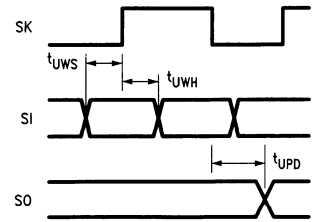
Port D Sink Current



TL/DD/9766-34

5

AC Electrical Characteristics (Continued)



TL/DD/9766-26

FIGURE 2. MICROWIRE/PLUS Timing

Pin Descriptions

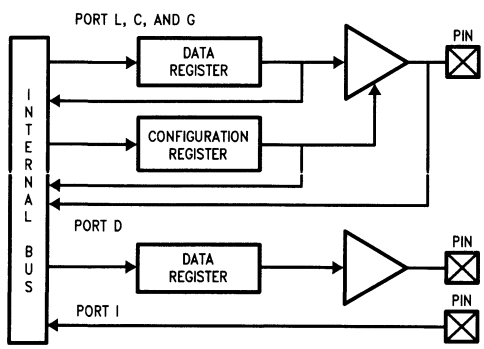
V_{CC} and GND are the power supply pins.

CKI is the clock input. This can come from an R/C generated oscillator, or a crystal oscillator (in conjunction with CKO). See Oscillator Description section.

$\overline{\text{RESET}}$ is the master reset input. See Reset Description section.

The device contains three bidirectional 8-bit I/O ports (C, G and L), where each individual bit may be independently configured as an input (Schmitt trigger inputs on ports G and L), output or TRI-STATE under program control. Three data memory address locations are allocated for each of these I/O ports. Each I/O port has two associated 8-bit memory mapped registers, the CONFIGURATION register and the output DATA register. A memory mapped address is also reserved for the input pins of each I/O port. (See the memory map for the various addresses associated with the I/O ports.) *Figure 3* shows the I/O port configurations. The DATA and CONFIGURATION registers allow for each port bit to be individually configured under software control as shown below:

CONFIGURATION Register	DATA Register	Port Set-Up
0	0	Hi-Z Input (TRI-STATE Output)
0	1	Input with Weak Pull-Up
1	0	Push-Pull Zero Output
1	1	Push-Pull One Output



TL/DD/9766-6

FIGURE 3. I/O Port Configurations

PORT L is an 8-bit I/O port. All L-pins have Schmitt triggers on the inputs.

Port L supports Multi-Input Wakeup (MIWU) on all eight pins. L4 and L5 are used for the timer input functions T2A and T2B.

Port L has the following alternate features:

- L0 MIWU
- L1 MIWU
- L2 MIWU
- L3 MIWU
- L4 MIWU or T2A
- L5 MIWU or T2B
- L6 MIWU
- L7 MIWU

Port G is an 8-bit port with 5 I/O pins (G0, G2–G5), an input pin (G6), and two dedicated output pins (G1 and G7). Pins G0 and G2–G6 all have Schmitt Triggers on their inputs. Pin G1 serves as the dedicated WDOUT WATCHDOG output, while pin G7 is either input or output depending on the oscillator mask option selected. With the crystal oscillator option selected, G7 serves as the dedicated output pin for the CKO clock output. With the single-pin R/C oscillator mask option selected, G7 serves as a general purpose input pin, but is also used to bring the device out of HALT mode with a low to high transition. There are two registers associated with the G Port, a data register and a configuration register. Therefore, each of the 5 I/O bits (G0, G2–G5) can be individually configured under software control.

Since G6 is an input only pin and G7 is the dedicated CKO clock output pin or general purpose input (R/C clock configuration), the associated bits in the data and configuration registers for G6 and G7 are used for special purpose functions as outlined below. Reading the G6 and G7 data bits will return zeros.

Note that the chip will be placed in the HALT mode by writing a "1" to bit 7 of the Port G Data Register. Similarly the chip will be placed in the IDLE mode by writing a "1" to bit 6 of the Port G Data Register.

Writing a "1" to bit 6 of the Port G Configuration Register enables the MICROWIRE/PLUS to operate with the alternate phase of the SK clock. The G7 configuration bit, if set high, enables the clock start up delay after HALT when the R/C clock configuration is used.

	Config Reg.	Data Reg.
G7	CLKDLY	HALT
G6	Alternate SK	IDLE

Port G has the following alternate features:

- G0 INTR (External Interrupt Input)
- G2 T1B (Timer T1 Capture Input)
- G3 T1A (Timer T1 I/O)
- G4 SO (MICROWIRE™ Serial Data Output)
- G5 SK (MICROWIRE Serial Clock)
- G6 SI (MICROWIRE Serial Data Input)

Pin Descriptions (Continued)

Port G has the following dedicated functions:

- G1 WDOUT WATCHDOG and/or Clock Monitor dedicated output
- G7 CKO Oscillator dedicated output or general purpose input

Port C is an 8-bit I/O port. The 40-pin device does not have a full complement of Port C pins. The unavailable pins are not terminated. A read operation for these unterminated pins will return unpredictable values.

Port I is an 8-bit Hi-Z input port. The 40-pin device does not have a full complement of Port I pins. Pins 15 and 16 on this package must be connected to GND.

The 28-pin device has four I pins (I0, I1, I4, I5). The user should pay attention when reading port I to the fact that I4 and I5 are in bit positions 4 and 5 rather than 2 and 3.

The unavailable pins (I4–I7) are not terminated i.e., they are floating. A read operation for these unterminated pins will return unpredictable values. The user must ensure that the software takes into account by either masking or restricting the accesses to bit operations. The unterminated port I pins will draw power only when addressed.

Port D is an 8-bit output port that is preset high when RESET goes low. The user can tie two or more D port outputs (except D2) together in order to get a higher drive.

Note: Care must be exercised with the D2 pin operation. At RESET, the external loads on this pin must ensure that the output voltages stay above 0.8 V_{CC} to prevent the chip from entering special modes. Also keep the external loading on D2 to less than 1000 pF.

Functional Description

The architecture of the device is modified Harvard architecture. With the Harvard architecture, the control store program memory (ROM) is separated from the data store memory (RAM). Both ROM and RAM have their own separate addressing space with separate address buses. The architecture, though based on Harvard architecture, permits transfer of data from ROM to RAM.

CPU REGISTERS

The CPU can do an 8-bit addition, subtraction, logical or shift operation in one instruction (t_C) cycle time.

There are five CPU registers:

A is the 8-bit Accumulator Register

PC is the 15-bit Program Counter Register

PU is the upper 7 bits of the program counter (PC)
PL is the lower 8 bits of the program counter (PC)

B is an 8-bit RAM address pointer, which can be optionally post auto incremented or decremented.

X is an 8-bit alternate RAM address pointer, which can be optionally post auto incremented or decremented.

SP is the 8-bit stack pointer, which points to the subroutine/interrupt stack (in RAM). The SP is initialized to RAM address 06F with reset.

All the CPU registers are memory mapped with the exception of the Accumulator (A) and the Program Counter (PC).

PROGRAM MEMORY

Program memory consists of 4096 bytes of ROM. These bytes may hold program instructions or constant data (data tables for the LAID instruction, jump vectors for the JID instruction, and interrupt vectors for the VIS instruction). The program memory is addressed by the 15-bit program counter (PC). All interrupts vector to program memory location 0FF Hex.

DATA MEMORY

The data memory address space includes the on-chip RAM and data registers, the I/O registers (Configuration, Data and Pin), the control registers, the MICROWIRE/PLUS SIO shift register, and the various registers, and counters associated with the timers (with the exception of the IDLE timer). Data memory is addressed directly by the instruction or indirectly by the B, X and SP pointers.

The device has 128 bytes of RAM. Sixteen bytes of RAM are mapped as "registers" at addresses 0F0 to 0FF Hex. These registers can be loaded immediately, and also decremented and tested with the DRSZ (decrement register and skip if zero) instruction. The memory pointer registers X, SP, and B are memory mapped into this space at address locations 0FC to 0FE Hex respectively, with the other registers (other than reserved register 0FF) being available for general usage.

The instruction set permits any bit in memory to be set, reset or tested. All I/O and registers (except A and PC) are memory mapped; therefore, I/O bits and register bits can be directly and individually set, reset and tested. The accumulator (A) bits can also be directly and individually tested.

Note: RAM contents are undefined upon power-up.

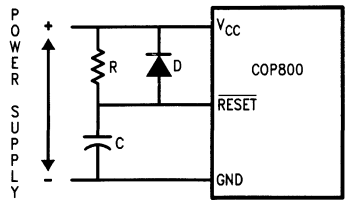
Reset

The $\overline{\text{RESET}}$ input when pulled low initializes the microcontroller. Initialization will occur whenever the $\overline{\text{RESET}}$ input is pulled low. Upon initialization, the data and configuration registers for Ports L, G, and C are cleared, resulting in these Ports being initialized to the TRI-STATE mode. Pin G1 of the G Port is an exception (as noted below) since pin G1 is dedicated as the WATCHDOG and/or Clock Monitor error output pin. Port D is initialized high with $\overline{\text{RESET}}$. The PC, PSW, CNTRL, ICNTRL, and T2CNTRL control registers are cleared. The Multi-Input Wakeup registers WKEN, WKEDG, and WKPND are cleared. The Stack Pointer, SP, is initialized to 06F Hex.

The device comes out of reset with both the WATCHDOG logic and the Clock Monitor detector armed, and with both the WATCHDOG service window bits set and the Clock Monitor bit set. The WATCHDOG and Clock Monitor detector circuits are inhibited during reset. The WATCHDOG service window bits are initialized to the maximum WATCHDOG service window of 64k t_C clock cycles. The Clock Monitor bit is initialized high, and will cause a Clock Monitor error following reset if the clock has not reached the minimum specified frequency at the termination of reset. A Clock Monitor error will cause an active low error output on pin G1. This error output will continue until 16–32 t_C clock cycles following the clock frequency reaching the minimum specified value, at which time the G1 output will enter the TRI-STATE mode.

The external RC network shown in *Figure 4* should be used to ensure that the $\overline{\text{RESET}}$ pin is held low until the power supply to the chip stabilizes.

Reset (Continued)



TL/DD/9766–7

RC > 5 × Power Supply Rise Time

FIGURE 4. Recommended Reset Circuit

Oscillator Circuits

The chip can be driven by a clock input on the CKI input pin which can be between DC and 10 MHz. The CKO output clock is on pin G7 (crystal configuration). The CKI input frequency is divided down by 10 to produce the instruction cycle clock ($1/t_c$).

Figure 5 shows the Crystal and R/C diagrams.

CRYSTAL OSCILLATOR

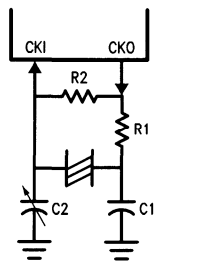
CKI and CKO can be connected to make a closed loop crystal (or resonator) controlled oscillator.

Table A shows the component values required for various standard crystal values.

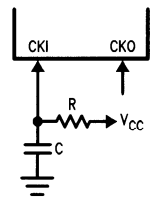
R/C OSCILLATOR

By selecting CKI as a single pin oscillator input, a single pin R/C oscillator circuit can be connected to it. CKO is available as a general purpose input, and/or HALT restart pin.

Table B shows the variation in the oscillator frequencies as functions of the component (R and C) values.



TL/DD/9766–8



TL/DD/9766–9

FIGURE 5. Crystal and R/C Oscillator Diagrams

TABLE A. Crystal Oscillator Configuration, T_A = 25°C

R1 (kΩ)	R2 (MΩ)	C1 (pF)	C2 (pF)	CKI Freq (MHz)	Conditions
0	1	30	30–36	10	V_{CC} = 5V
0	1	30	30–36	4	V_{CC} = 5.0V
0	1	200	100–150	0.455	V_{CC} = 5V

TABLE B. RC Oscillator Configuration, T_A = 25°C

R (kΩ)	C (pF)	CKI Freq (MHz)	Instr. Cycle (μs)	Conditions
3.3	82	2.2 to 2.7	3.7 to 4.6	V_{CC} = 5V
5.6	100	1.1 to 1.3	7.4 to 9.0	V_{CC} = 5V
6.8	100	0.9 to 1.1	8.8 to 10.8	V_{CC} = 5V

Note: 3k ≤ R ≤ 200k, 50 pF ≤ C ≤ 200 pF

Control Registers

CNTRL Register (Address X'00EE)

The Timer1 (T1) and MICROWIRE/PLUS control register contains the following bits:

SL1 & SL0 Select the MICROWIRE/PLUS clock divide by (00 = 2, 01 = 4, 1x = 8)

IEDG External interrupt edge polarity select (0 = Rising edge, 1 = Falling edge)

MSEL Selects G5 and G4 as MICROWIRE/PLUS signals SK and SO respectively

Control Registers (Continued)

T1C0 Timer T1 Start/Stop control in timer modes 1 and 2
Timer T1 Underflow Interrupt Pending Flag in timer mode 3

T1C1 Timer T1 mode control bit

T1C2 Timer T1 mode control bit

T1C3 Timer T1 mode control bit

T1C3	T1C2	T1C1	T1C0	MSEL	IEDG	SL1	SL0

Bit 7 Bit 0

PSW Register (Address X'00EF)

The PSW register contains the following select bits:

GIE Global interrupt enable (enables interrupts)

EXEN Enable external interrupt

BUSY MICROWIRE/PLUS busy shifting flag

EXPND External interrupt pending

T1ENA Timer T1 Interrupt Enable for Timer Underflow or T1A Input capture edge

T1PNDA Timer T1 Interrupt Pending Flag (Autoreload RA in mode 1, T1 Underflow in Mode 2, T1A capture edge in mode 3)

C Carry Flag

HC Half Carry Flag

HC	C	T1PNDA	T1ENA	EXPND	BUSY	EXEN	GIE

Bit 7 Bit 0

The Half-Carry bit is also affected by all the instructions that affect the Carry flag. The SC (Set Carry) and RC (Reset Carry) instructions will respectively set or clear both the carry flags. In addition to the SC and RC instructions, ADC, SUBC, RRC and RLC instructions affect the carry and Half Carry flags.

ICNTRL Register (Address X'00E8)

The ICNTRL register contains the following bits:

T1ENB Timer T1 Interrupt Enable for T1B Input capture edge

T1PNDB Timer T1 Interrupt Pending Flag for T1B capture edge

μWEN Enable MICROWIRE/PLUS interrupt

μWPND MICROWIRE/PLUS interrupt pending

T0EN Timer T0 Interrupt Enable (Bit 12 toggle)

T0PND Timer T0 Interrupt pending

LPEN L Port Interrupt Enable (Multi-Input Wakeup/Interrupt)
Bit 7 could be used as a flag

Unused	LPEN	T0PND	T0EN	μWPND	μWEN	T1PNDB	T1ENB

Bit 7 Bit 0

T2CNTRL Register (Address X'00C6)

The T2CNTRL register contains the following bits:

T2ENB Timer T2 Interrupt Enable for T2B Input capture edge

T2PNDB Timer T2 Interrupt Pending Flag for T2B capture edge

T2ENA Timer T2 Interrupt Enable for Timer Underflow or T2A Input capture edge

T2PNDA Timer T2 Interrupt Pending Flag (Autoreload RA in mode 1, T2 Underflow in mode 2, T2A capture edge in mode 3)

T2C0 Timer T2 Start/Stop control in timer modes 1 and 2 Timer T2 Underflow Interrupt Pending Flag in timer mode 3

T2C1 Timer T2 mode control bit

T2C2 Timer T2 mode control bit

T2C3 Timer T2 mode control bit

T2C3	T2C2	T2C1	T2C0	T2PNDA	T2ENA	T2PNDB	T2ENB

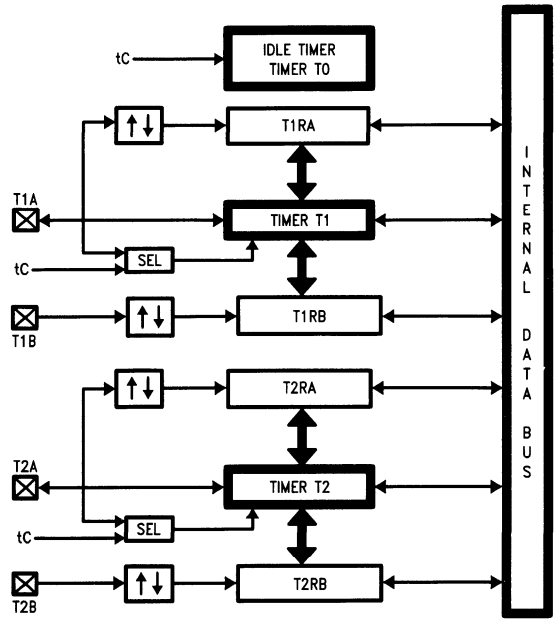
Bit 7 Bit 0

Timers

The device contains a very versatile set of timers (T0, T1, T2). All timers and associated autoreload/capture registers power up containing random data.

Figure 6 shows a block diagram for the timers.

Timers (Continued)



TL/DD/9766–11

FIGURE 6. Timers

TIMER T0 (IDLE TIMER)

The device supports applications that require maintaining real time and low power with the IDLE mode. This IDLE mode support is furnished by the IDLE timer T0, which is a 16-bit timer. The Timer T0 runs continuously at the fixed rate of the instruction cycle clock, t_C. The user cannot read or write to the IDLE Timer T0, which is a count down timer.

The Timer T0 supports the following functions:

Exit out of the Idle Mode (See Idle Mode description)
WATCHDOG logic (See WATCHDOG description)
Start up delay out of the HALT mode

The IDLE Timer T0 can generate an interrupt when the thirteenth bit toggles. This toggle is latched into the T0PND pending flag, and will occur every 4 ms at the maximum clock frequency (t_C = 1 μs). A control flag T0EN allows the interrupt from the thirteenth bit of Timer T0 to be enabled or disabled. Setting T0EN will enable the interrupt, while resetting it will disable the interrupt.

TIMER T1 AND TIMER T2

The device has a set of two powerful timer/counter blocks, T1 and T2. The associated features and functioning of a timer block are described by referring to the timer block Tx. Since the two timer blocks, T1 and T2, are identical, all comments are equally applicable to either timer block.

Each timer block consists of a 16-bit timer, Tx, and two supporting 16-bit autoreload/capture registers, RxA and RxB. Each timer block has two pins associated with it, TxA and TxB. The pin TxA supports I/O required by the timer block, while the pin TxB is an input to the timer block. The powerful and flexible timer block allows the device to easily perform all timer functions with minimal software overhead. The timer block has three operating modes: Processor Independent PWM mode, External Event Counter mode, and Input Capture mode.

The control bits TxC3, TxC2, and TxC1 allow selection of the different modes of operation.

Mode 1. Processor Independent PWM Mode

As the name suggests, this mode allows the device to generate a PWM signal with very minimal user intervention.

The user only has to define the parameters of the PWM signal (ON time and OFF time). Once begun, the timer block will continuously generate the PWM signal completely independent of the microcontroller. The user software services the timer block only when the PWM parameters require updating.

In this mode the timer Tx counts down at a fixed rate of t_C. Upon every underflow the timer is alternately reloaded with the contents of supporting registers, RxA and RxB. The very first underflow of the timer causes the timer to reload from the register RxA. Subsequent underflows cause the timer to be reloaded from the registers alternately beginning with the register RxB.

The Tx Timer control bits, TxC3, TxC2 and TxC1 set up the timer for PWM mode operation.

Figure 7 shows a block diagram of the timer in PWM mode.

Timers (Continued)

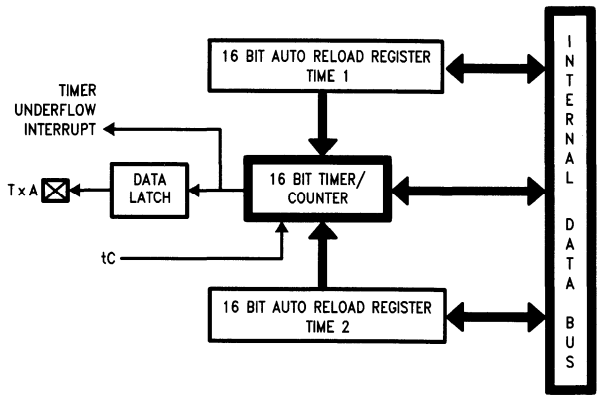


FIGURE 7. Timer in PWM Mode

The underflows can be programmed to toggle the TxA output pin. The underflows can also be programmed to generate interrupts.

Underflows from the timer are alternately latched into two pending flags, TxPNDA and TxPNDB. The user must reset these pending flags under software control. Two control enable flags, TxENA and TxENB, allow the interrupts from the timer underflow to be enabled or disabled. Setting the timer enable flag TxENA will cause an interrupt when a timer underflow causes the RxA register to be reloaded into the timer. Setting the timer enable flag TxENB will cause an interrupt when a timer underflow causes the RxB register to be reloaded into the timer. Resetting the timer enable flags will disable the associated interrupts.

Either or both of the timer underflow interrupts may be enabled. This gives the user the flexibility of interrupting once per PWM period on either the rising or falling edge of the PWM output. Alternatively, the user may choose to interrupt on both edges of the PWM output.

Mode 2. External Event Counter Mode

This mode is quite similar to the processor independent PWM mode described above. The main difference is that the timer, Tx, is clocked by the input signal from the TxA pin. The Tx timer control bits, TxC3, TxC2 and TxC1 allow the timer to be clocked either on a positive or negative edge from the TxA pin. Underflows from the timer are latched into the TxPNDA pending flag. Setting the TxENA control flag will cause an interrupt when the timer underflows.

In this mode the input pin TxB can be used as an independent positive edge sensitive interrupt input if the TxENB control flag is set. The occurrence of a positive edge on the TxB input pin is latched into the TxPNDB flag.

Figure 8 shows a block diagram of the timer in External Event Counter mode.

Note: The PWM output is not available in this mode since the TxA pin is being used as the counter input clock.

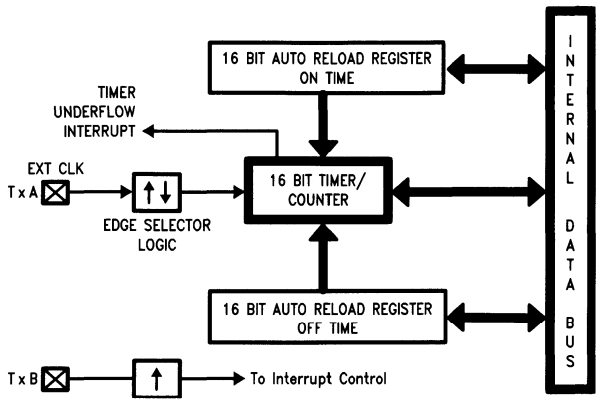


FIGURE 8. Timer in External Event Counter Mode

Timers (Continued)

Mode 3. Input Capture Mode

The device can precisely measure external frequencies or time external events by placing the timer block, Tx, in the input capture mode.

In this mode, the timer Tx is constantly running at the fixed t_C rate. The two registers, RxA and RxB, act as capture registers. Each register acts in conjunction with a pin. The register RxA acts in conjunction with the TxA pin and the register RxB acts in conjunction with the TxB pin.

The timer value gets copied over into the register when a trigger event occurs on its corresponding pin. Control bits, TxC3, TxC2 and TxC1, allow the trigger events to be specified either as a positive or a negative edge. The trigger condition for each input pin can be specified independently.

The trigger conditions can also be programmed to generate interrupts. The occurrence of the specified trigger condition on the TxA and TxB pins will be respectively latched into the pending flags, TxPNDA and TxPNDB. The control flag TxENA allows the interrupt on TxA to be either enabled or disabled. Setting the TxENA flag enables interrupts to be generated when the selected trigger condition occurs on the TxA pin. Similarly, the flag TxENB controls the interrupts from the TxB pin.

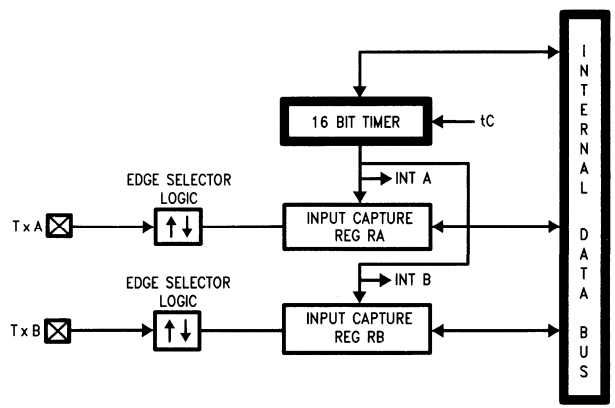
Underflows from the timer can also be programmed to generate interrupts. Underflows are latched into the timer TxC0 pending flag (the TxC0 control bit serves as the timer underflow interrupt pending flag in the Input Capture mode). Consequently, the TxC0 control bit should be reset when entering the Input Capture mode. The timer underflow interrupt is enabled with the TxENA control flag. When a TxA interrupt occurs in the Input Capture mode, the user must check both the TxPNDA and TxC0 pending flags in order to determine whether a TxA input capture or a timer underflow (or both) caused the interrupt.

Figure 9 shows a block diagram of the timer in Input Capture mode.

TIMER CONTROL FLAGS

The timers T1 and T2 have indentical control structures. The control bits and their functions are summarized below.

TxC0	Timer Start/Stop control in Modes 1 and 2 (Processor Independent PWM and External Event Counter), where 1 = Start, 0 = Stop Timer Underflow Interrupt Pending Flag in Mode 3 (Input Capture)
TxPNDA	Timer Interrupt Pending Flag
TxPNDB	Timer Interrupt Pending Flag
TxENA	Timer Interrupt Enable Flag
TxENB	Timer Interrupt Enable Flag 1 = Timer Interrupt Enabled 0 = Timer Interrupt Disabled
TxC3	Timer mode control
TxC2	Timer mode control
TxC1	Timer mode control



TL/DD/9766–15

FIGURE 9. Timer in Input Capture Mode

Timers (Continued)

The timer mode control bits (TxC3, TxC2 and TxC1) are detailed below:

TxC3	TxC2	TxC1	Timer Mode	Interrupt A Source	Interrupt B Source	Timer Counts On
0	0	0	MODE 2 (External Event Counter)	Timer Underflow	Pos. TxB Edge	TxA Pos. Edge
0	0	1	MODE 2 (External Event Counter)	Timer Underflow	Pos. TxB Edge	TxA Neg. Edge
1	0	1	MODE 1 (PWM) TxA Toggle	Autoreload RA	Autoreload RB	t_C
1	0	0	MODE 1 (PWM) No TxA Toggle	Autoreload RA	Autoreload RB	t_C
0	1	0	MODE 3 (Capture) Captures: TxA Pos. Edge TxB Pos. Edge	Pos. TxA Edge or Timer Underflow	Pos. TxB Edge	t_C
1	1	0	MODE 3 (Capture) Captures: TxA Pos. Edge TxB Neg. Edge	Pos. TxA Edge or Timer Underflow	Neg. TxB Edge	t_C
0	1	1	MODE 3 (Capture) Captures: TxA Neg. Edge TxB Pos. Edge	Neg. TxB Edge or Timer Underflow	Pos. TxB Edge	t_C
1	1	1	MODE 3 (Capture) Captures: TxA Neg. Edge TxB Neg. Edge	Neg. TxA Edge or Timer Underflow	Neg. TxB Edge	t_C

Power Save Modes

The device offers the user two power save modes of operation: HALT and IDLE. In the HALT mode, all microcontroller activities are stopped. In the IDLE mode, the on-board oscillator circuitry and timer T0 are active but all other microcontroller activities are stopped. In either mode, all on-board RAM, registers, I/O states, and timers (with the exception of T0) are unaltered.

HALT MODE

The device is placed in the HALT mode by writing a "1" to the HALT flag (G7 data bit). All microcontroller activities, including the clock, timers, are stopped. The WATCHDOG logic is disabled during the HALT mode. However, the clock monitor circuitry, if enabled, remains active and will cause the WATCHDOG output pin (WDOUT) to go low. If the HALT mode is used and the user does not want to activate the WDOUT pin, the Clock Monitor should be disabled after the device comes out of reset (resetting the Clock Monitor control bit with the first write to the WDSVR register). In the HALT mode, the power requirements of the device are minimal and the applied voltage (V_{CC}) may be decreased to V_r (V_r = 2.0V) without altering the state of the machine.

The device supports three different ways of exiting the HALT mode. The first method of exiting the HALT mode is with the Multi-Input Wakeup feature on the L port. The second method is with a low to high transition on the CKO (G7) pin. This method precludes the use of the crystal clock configuration (since CKO becomes a dedicated output), and so may be used with an RC clock configuration. The third method of exiting the HALT mode is by pulling the $\overline{\text{RESET}}$ pin low.

Since a crystal or ceramic resonator may be selected as the oscillator, the Wakeup signal is not allowed to start the chip running immediately since crystal oscillators and ceramic resonators have a delayed start up time to reach full amplitude and frequency stability. The IDLE timer is used to generate a fixed delay to ensure that the oscillator has indeed stabilized before allowing instruction execution. In this case, upon detecting a valid Wakeup signal, only the oscillator circuitry is enabled. The IDLE timer is loaded with a value of 256 and is clocked with the t_C instruction cycle clock. The t_C clock is derived by dividing the oscillator clock down by a factor of 10. The Schmitt trigger following the CKI inverter on the chip ensures that the IDLE timer is clocked only when the oscillator has a sufficiently large amplitude to meet the Schmitt trigger specifications. This Schmitt trigger is not part of the oscillator closed loop. The startup timeout from the IDLE timer enables the clock signals to be routed to the rest of the chip.

Power Save Modes (Continued)

If an RC clock option is being used, the fixed delay is introduced optionally. A control bit, CLKDLY, mapped as configuration bit G7, controls whether the delay is to be introduced or not. The delay is included if CLKDLY is set, and excluded if CLKDLY is reset. The CLKDLY bit is cleared on reset.

The device has two mask options associated with the HALT mode. The first mask option enables the HALT mode feature, while the second mask option disables the HALT mode. With the HALT mode enable mask option, the device will enter and exit the HALT mode as described above. With the HALT disable mask option, the device cannot be placed in the HALT mode (writing a "1" to the HALT flag will have no effect).

The WATCHDOG detector circuit is inhibited during the HALT mode. However, the clock monitor circuit, if enabled, remains active during HALT mode in order to ensure a clock monitor error if the device inadvertently enters the HALT mode as a result of a runaway program or power glitch.

IDLE MODE

The device is placed in the IDLE mode by writing a "1" to the IDLE flag (G6 data bit). In this mode, all activity, except the associated on-board oscillator circuitry, the WATCHDOG logic, the clock monitor and the IDLE Timer T0, is stopped.

As with the HALT mode, the device can be returned to normal operation with a reset, or with a Multi-Input Wake-up from the L Port. Alternately, the microcontroller resumes normal operation from the IDLE mode when the thirteenth bit (representing 4.096 ms at internal clock frequency of 1 MHz, t_c = 1 μs) of the IDLE Timer toggles.

This toggle condition of the thirteenth bit of the IDLE Timer T0 is latched into the T0PND pending flag.

The user has the option of being interrupted with a transition on the thirteenth bit of the IDLE Timer T0. The interrupt can be enabled or disabled via the T0EN control bit. Setting the T0EN flag enables the interrupt and vice versa.

The user can enter the IDLE mode with the Timer T0 interrupt enabled. In this case, when the T0PND bit gets set, the device will first execute the Timer T0 interrupt service routine and then return to the instruction following the "Enter Idle Mode" instruction.

Alternatively, the user can enter the IDLE mode with the IDLE Timer T0 interrupt disabled. In this case, the device will resume normal operation with the instruction immediately following the "Enter IDLE Mode" instruction.

Note: It is necessary to program two NOP instructions following both the set HALT mode and set IDLE mode instructions. These NOP instructions are necessary to allow clock resynchronization following the HALT or IDLE modes.

Multi-Input Wakeup

The Multi-Input Wakeup feature is used to return (wakeup) the device from either the HALT or IDLE modes. Alternately Multi-Input Wakeup/Interrupt feature may also be used to generate up to 8 edge selectable external interrupts.

Figure 10 shows the Multi-Input Wakeup logic.

The Multi-Input Wakeup feature utilizes the L Port. The user selects which particular L port bit (or combination of L Port bits) will cause the device to exit the HALT or IDLE modes. The selection is done through the Reg: WKEN. The Reg: WKEN is an 8-bit read/write register, which contains a control bit for every L port bit. Setting a particular WKEN bit enables a Wakeup from the associated L port pin.

The user can select whether the trigger condition on the selected L Port pin is going to be either a positive edge (low to high transition) or a negative edge (high to low transition). This selection is made via the Reg: WKEDG, which is an 8-bit control register with a bit assigned to each L Port pin. Setting the control bit will select the trigger condition to be a negative edge on that particular L Port pin. Resetting the bit selects the trigger condition to be a positive edge. Changing an edge select entails several steps in order to avoid a pseudo Wakeup condition as a result of the edge change. First, the associated WKEN bit should be reset, followed by the edge select change in WKEDG. Next, the associated WKPND bit should be cleared, followed by the associated WKEN bit being re-enabled.

An example may serve to clarify this procedure. Suppose we wish to change the edge select from positive (low going high) to negative (high going low) for L Port bit 5, where bit 5 has previously been enabled for an input interrupt. The program would be as follows:

```
RBIT  5, WKEN
SBIT  5, WKEDG
RBIT  5, WKPND
SBIT  5, WKEN
```

If the L port bits have been used as outputs and then changed to inputs with Multi-Input Wakeup/Interrupt, a safety procedure should also be followed to avoid inherited pseudo wakeup conditions. After the selected L port bits have been changed from output to input but before the associated WKEN bits are enabled, the associated edge select bits in WKEDG should be set or reset for the desired edge selects, followed by the associated WKPND bits being cleared.

This same procedure should be used following reset, since the L port inputs are left floating as a result of reset.

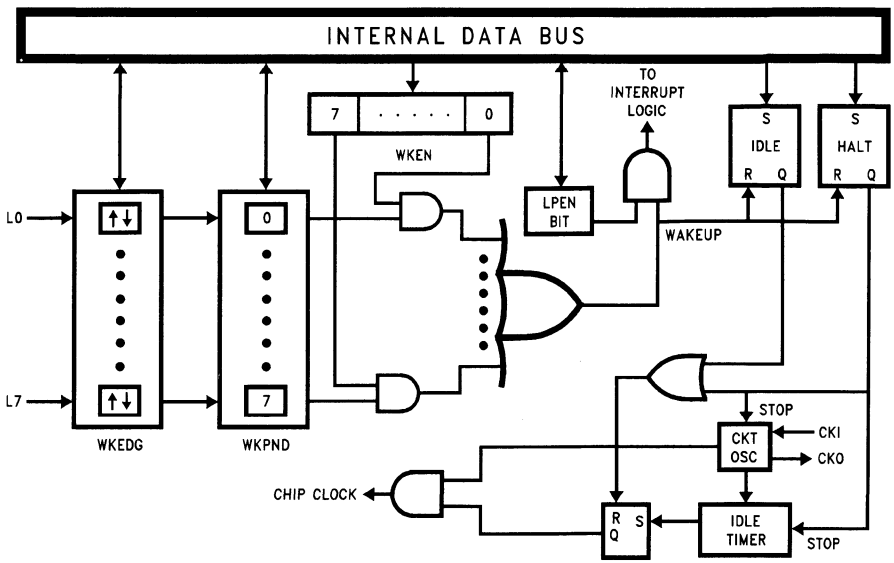
The occurrence of the selected trigger condition for Multi-Input Wakeup is latched into a pending register called WKPND. The respective bits of the WKPND register will be set on the occurrence of the selected trigger edge on the corresponding Port L pin. The user has the responsibility of clearing these pending flags. Since WKPND is a pending register for the occurrence of selected wakeup conditions, the device will not enter the HALT mode if any Wakeup bit is both enabled and pending. Consequently, the user has the responsibility of clearing the pending flags before attempting to enter the HALT mode.

The WKEN, WKPND and WKEDG are all read/write registers, and are cleared at reset.

PORT L INTERRUPTS

Port L provides the user with an additional eight fully selectable, edge sensitive interrupts which are all vectored into the same service subroutine.

The interrupt from Port L shares logic with the wake up circuitry. The register WKEN allows interrupts from Port L to



TL/DD/9766–16

FIGURE 10. Multi-Input Wake Up Logic

Multi-Input Wakeup (Continued)

be individually enabled or disabled. The register WKEDG specifies the trigger condition to be either a positive or a negative edge. Finally, the register WKPND latches in the pending trigger conditions.

The GIE (Global Interrupt Enable) bit enables the interrupt function.

A control flag, LPEN, functions as a global interrupt enable for Port L interrupts. Setting the LPEN flag will enable interrupts and vice versa. A separate global pending flag is not needed since the register WKPND is adequate.

Since Port L is also used for waking the device out of the HALT or IDLE modes, the user can elect to exit the HALT or IDLE modes either with or without the interrupt enabled. If he elects to disable the interrupt, then the device will restart execution from the instruction immediately following the instruction that placed the microcontroller in the HALT or IDLE modes. In the other case, the device will first execute the interrupt service routine and then revert to normal operation.

The Wakeup signal will not start the chip running immediately since crystal oscillators or ceramic resonators have a finite start up time. The IDLE Timer (T0) generates a fixed delay to ensure that the oscillator has indeed stabilized before allowing the device to execute instructions. In this case, upon detecting a valid Wakeup signal, only the oscillator circuitry and the IDLE Timer T0 are enabled. The IDLE Timer is loaded with a value of 256 and is clocked from the t_C instruction cycle clock. The t_C clock is derived by dividing down the oscillator clock by a factor of 10. A Schmitt trigger following the CKI on-chip inverter ensures that the IDLE timer is clocked only when the oscillator has a sufficiently large amplitude to meet the Schmitt trigger specifications. This Schmitt trigger is not part of the oscillator closed loop. The startup timeout from the IDLE timer enables the clock signals to be routed to the rest of the chip. If the RC clock option is used, the fixed delay is under software control. A control flag, CLKDLY, in the G7 configuration bit allows the clock start up delay to be optionally inserted. Setting CLKDLY flag high will cause clock start up delay to be inserted and resetting it will exclude the clock start up delay. The CLKDLY flag is cleared during reset, so the clock start up delay is not present following reset with the RC clock options.

Interrupts

The device supports a vectored interrupt scheme. It supports a total of ten interrupt sources. The following table lists all the possible interrupt sources, their arbitration ranking and the memory locations reserved for the interrupt vector for each source.

Two bytes of program memory space are reserved for each interrupt source. All interrupt sources except the software interrupt are maskable. Each of the maskable interrupts have an Enable bit and a Pending bit. A maskable interrupt is active if its associated enable and pending bits are set. If GIE = 1 and an interrupt is active, then the processor will be interrupted as soon as it is ready to start executing an instruction except if the above conditions happen during the Software Trap service routine. This exception is described in the Software Trap sub-section.

The interruption process is accomplished with the INTR instruction (opcode 00), which is jammed inside the Instruc-

Arbitration Ranking	Source	Description	Vector Address Hi-Low Byte
(1) Highest	Software	INTR Instruction	0yFE–0yFF
	Reserved	for Future Use	0yFC–0yFD
(2)	External	Pin G0 Edge	0yFA–0yFB
(3)	Timer T0	Underflow	0yF8–0yF9
(4)	Timer T1	T1A/Underflow	0yF6–0yF7
(5)	Timer T1	T1B	0yF4–0yF5
(6)	MICROWIRE/PLUS	BUSY Goes Low	0yF2–0yF3
	Reserved	for Future Use	0yF0–0yF1
	Reserved	for UART	0yEE–0yEF
	Reserved	for UART	0yEC–0yED
(7)	Timer T2	T2A/Underflow	0yEA–0yEB
(8)	Timer T2	T2B	0yE8–0yE9
	Reserved	for Future Use	0yE6–0yE7
	Reserved	for Future Use	0yE4–0yE5
(9)	Port L/Wakeup	Port L Edge	0yE2–0yE3
(10) Lowest	Default	VIS Instr. Execution without Any Interrupts	0yE0–0yE1

y is VIS page, y ≠ 0.

Interrupts (Continued)

tion Register and replaces the opcode about to be executed. The following steps are performed for every interrupt:

1. The GIE (Global Interrupt Enable) bit is reset.
2. The address of the instruction about to be executed is pushed into the stack.
3. The PC (Program Counter) branches to address 00FF. This procedure takes 7 t_C cycles to execute.

At this time, since GIE = 0, other maskable interrupts are disabled. The user is now free to do whatever context switching is required by saving the context of the machine in the stack with PUSH instructions. The user would then program a VIS (Vector Interrupt Select) instruction in order to branch to the interrupt service routine of the highest priority interrupt enabled and pending at the time of the VIS. Note that this is not necessarily the interrupt that caused the branch to address location 00FF Hex prior to the context switching.

Thus, if an interrupt with a higher rank than the one which caused the interruption becomes active before the decision of which interrupt to service is made by the VIS, then the interrupt with the higher rank will override any lower ones and will be acknowledged. The lower priority interrupt(s) are still pending, however, and will cause another interrupt immediately following the completion of the interrupt service routine associated with the higher priority interrupt just serviced. This lower priority interrupt will occur immediately following the RETI (Return from Interrupt) instruction at the end of the interrupt service routine just completed.

Inside the interrupt service routine, the associated pending bit has to be cleared by software. The RETI (Return from Interrupt) instruction at the end of the interrupt service routine will set the GIE (Global Interrupt Enable) bit, allowing the processor to be interrupted again if another interrupt is active and pending.

The VIS instruction looks at all the active interrupts at the time it is executed and performs an indirect jump to the beginning of the service routine of the one with the highest rank.

The addresses of the different interrupt service routines, called vectors, are chosen by the user and stored in ROM in a table starting at 01E0 (assuming that VIS is located between 00FF and 01DF). The vectors are 15-bit wide and therefore occupy 2 ROM locations.

VIS and the vector table must be located in the same 256-byte block (0y00 to 0yFF) except if VIS is located at the last address of a block. In this case, the table must be in the next block. The vector table cannot be inserted in the first 256-byte block.

The vector of the maskable interrupt with the lowest rank is located at 0yE0 (Hi-Order byte) and 0yE1 (Lo-Order byte) and so forth in increasing rank number. The vector of the maskable interrupt with the highest rank is located at 0yFA (Hi-Order byte) and 0yFB (Lo-Order byte).

The Software Trap has the highest rank and its vector is located at 0yFE and 0yFF.

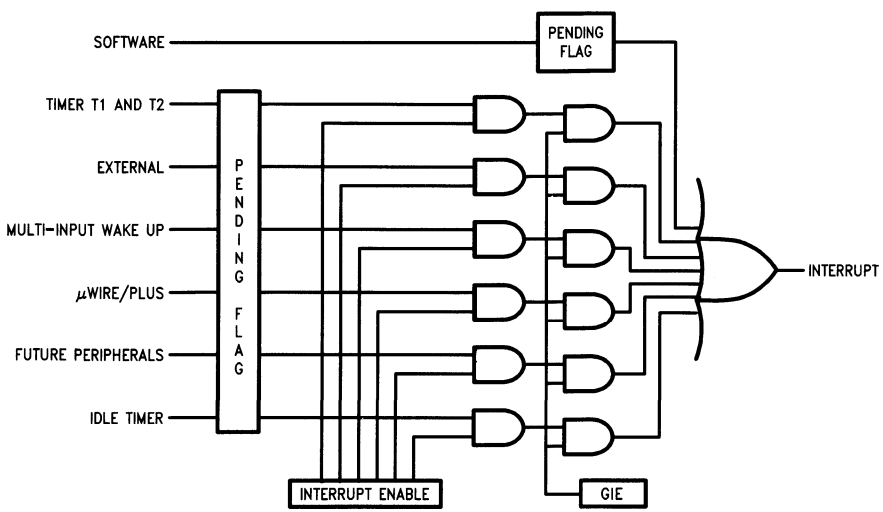
If, by accident, a VIS gets executed and no interrupt is active, then the PC (Program Counter) will branch to a vector located at 0yE0–0yE1.

WARNING

A Default VIS interrupt handle routine must be present. As a minimum, this handler should confirm that the GIE bit is cleared (this indicates that the interrupt sequence has been taken), take care of any required housekeeping, restore context and return. Some sort of Warm Restart procedure should be implemented. These events can occur without any error on the part of the system designer or programmer.

Note: There is always the possibility of an interrupt occurring during an instruction which is attempting to reset the GIE bit or any other interrupt enable bit. If this occurs when a single cycle instruction is being used to reset the interrupt enable bit, the interrupt enable bit will be reset but an interrupt may still occur. This is because interrupt processing is started at the same time as the interrupt bit is being reset. To avoid this scenario, the user should always use a two, three, or four cycle instruction to reset interrupt enable bits.

Figure 11 shows the Interrupt block diagram.



TL/DD/9766–18

FIGURE 11. Interrupt Block Diagram

Interrupts (Continued)

SOFTWARE TRAP

The Software Trap (ST) is a special kind of non-maskable interrupt which occurs when the INTR instruction (used to acknowledge interrupts) is fetched from ROM and placed inside the instruction register. This may happen when the PC is pointing beyond the available ROM address space or when the stack is over-popped.

When an ST occurs, the user can re-initialize the stack pointer and do a recovery procedure (similar to reset, but not necessarily containing all of the same initialization procedures) before restarting.

The occurrence of an ST is latched into the ST pending bit. The GIE bit is not affected and the ST pending bit **(not accessible by the user)** is used to inhibit other interrupts and to direct the program to the ST service routine with the VIS instruction. The RPND instruction is used to clear the software interrupt pending bit. This pending bit is also cleared on reset.

The ST has the highest rank among all interrupts.

Nothing (except another ST) can interrupt an ST being serviced.

WATCHDOG

The device contains a WATCHDOG and clock monitor. The WATCHDOG is designed to detect the user program getting stuck in infinite loops resulting in loss of program control or "runaway" programs. The Clock Monitor is used to detect the absence of a clock or a very slow clock below a specified rate on the CKI pin.

The WATCHDOG consists of two independent logic blocks: WD UPPER and WD LOWER. WD UPPER establishes the upper limit on the service window and WD LOWER defines the lower limit of the service window.

Servicing the WATCHDOG consists of writing a specific value to a WATCHDOG Service Register named WDSVR which is memory mapped in the RAM. This value is composed of three fields, consisting of a 2-bit Window Select, a 5-bit Key Data field, and the 1-bit Clock Monitor Select field. Table I shows the WDSVR register.

The lower limit of the service window is fixed at 2048 instruction cycles. Bits 7 and 6 of the WDSVR register allow the user to pick an upper limit of the service window.

Table II shows the four possible combinations of lower and upper limits for the WATCHDOG service window. This flexibility in choosing the WATCHDOG service window prevents any undue burden on the user software.

Bits 5, 4, 3, 2 and 1 of the WDSVR register represent the 5-bit Key Data field. The key data is fixed at 01100. Bit 0 of the WDSVR Register is the Clock Monitor Select bit.

TABLE I. WATCHDOG Service Register (WDSVR)

Window Select		Key Data					Clock Monitor
X	X	0	1	1	0	0	Y
7	6	5	4	3	2	1	0

TABLE II. WATCHDOG Service Window Select

$\overline{\text{WDSVR}}$ Bit 7	$\overline{\text{WDSVR}}$ Bit 6	Service Window (Lower-Upper Limits)
0	0	2k-8k t_C Cycles
0	1	2k-16k t_C Cycles
1	0	2k-32k t_C Cycles
1	1	2k-64k t_C Cycles

Clock Monitor

The Clock Monitor aboard the device can be selected or deselected under program control. The Clock Monitor is guaranteed not to reject the clock if the instruction cycle clock ($1/t_C$) is greater or equal to 10 kHz. This equates to a clock input rate on CKI of greater or equal to 100 kHz.

WATCHDOG Operation

The WATCHDOG and Clock Monitor are disabled during reset. The device comes out of reset with the WATCHDOG armed, the WATCHDOG Window Select bits (bits 6, 7 of the WDSVR Register) set, and the Clock Monitor bit (bit 0 of the WDSVR Register) enabled. Thus, a Clock Monitor error will occur after coming out of reset, if the instruction cycle clock frequency has not reached a minimum specified value, including the case where the oscillator fails to start.

The WDSVR register can be written to only once after reset and the key data (bits 5 through 1 of the WDSVR Register) must match to be a valid write. This write to the WDSVR register involves two irrevocable choices: (i) the selection of the WATCHDOG service window (ii) enabling or disabling of the Clock Monitor. Hence, the first write to WDSVR Register involves selecting or deselecting the Clock Monitor, select the WATCHDOG service window and match the WATCHDOG key data. Subsequent writes to the WDSVR register will compare the value being written by the user to the WATCHDOG service window value and the key data (bits 7 through 1) in the WDSVR Register. Table III shows the sequence of events that can occur.

The user must service the WATCHDOG at least once before the upper limit of the serivce window expires. The WATCHDOG may not be serviced more than once in every lower limit of the service window. The user may service the WATCHDOG as many times as wished in the time period between the lower and upper limits of the service window. The first write to the WDSVR Register is also counted as a WATCHDOG service.

The WATCHDOG has an output pin associated with it. This is the WDOUT pin, on pin 1 of the port G. WDOUT is active low. The WDOUT pin is in the high impedance state in the inactive state. Upon triggering the WATCHDOG, the logic will pull the WDOUT (G1) pin low for an additional 16 t_C–32 t_C cycles after the signal level on WDOUT pin goes below the lower Schmitt trigger threshold. After this delay, the device will stop forcing the WDOUT output low.

The WATCHDOG service window will restart when the WDOUT pin goes high It is recommended that the user tie the WDOUT pin back to V_{CC} through a resistor in order to pull WDOUT high.

A WATCHDOG service while the WDOUT signal is active will be ignored. The state of the WDOUT pin is not guaranteed on reset, but if it powers up low then the WATCHDOG will time out and WDOUT will enter high impedance state.

WATCHDOG Operation (Continued)

TABLE III. WATCHDOG Service Actions

Key Data	Window Data	Clock Monitor	Action
Match	Match	Match	Valid Service: Restart Service Window
Don't Care	Mismatch	Don't Care	Error: Generate WATCHDOG Output
Mismatch	Don't Care	Don't Care	Error: Generate WATCHDOG Output
Don't Care	Don't Care	Mismatch	Error: Generate WATCHDOG Output

TABLE IV. MICROWIRE/PLUS Master Mode Clock Select

SL1	SL0	SK
0	0	$2 \times t_C$
0	1	$4 \times t_C$
1	x	$8 \times t_C$

Where t_C is the instruction cycle clock

The Clock Monitor forces the G1 pin low upon detecting a clock frequency error. The Clock Monitor error will continue until the clock frequency has reached the minimum specified value, after which the G1 output will enter the high impedance TRI-STATE mode following 16 t_C–32 t_C clock cycles. The Clock Monitor generates a continual Clock Monitor error if the oscillator fails to start, or fails to reach the minimum specified frequency. The specification for the Clock Monitor is as follows:

$1/t_C > 10$ kHz—No clock rejection.

$1/t_C < 10$ Hz—Guaranteed clock rejection.

WATCHDOG AND CLOCK MONITOR SUMMARY

The following salient points regarding the WATCHDOG and Clock Monitor should be noted:

- Both WATCHDOG and Clock Monitor detector circuits are inhibited during reset.
- Following reset, the WATCHDOG and Clock Monitor are both enabled, with the WATCHDOG having the maximum service window selected.
- The WATCHDOG service window and Clock Monitor enable/disable option can only be changed once, during the initial WATCHDOG service following reset.
- The initial WATCHDOG service must match the key data value in the WATCHDOG Service register WDSVR in order to avoid a WATCHDOG error.
- Subsequent WATCHDOG services must match all three data fields in WDSVR in order to avoid WATCHDOG errors.
- The correct key data value cannot be read from the WATCHDOG Service register WDSVR. Any attempt to read this key data value of 01100 from WDSVR will read as key data value of all 0's.
- The WATCHDOG detector circuit is inhibited during both the HALT and IDLE modes.
- The Clock Monitor detector circuit is active during both the HALT and IDLE modes. Consequently, the device inadvertently entering the HALT mode will be detected as a Clock Monitor error (provided that the Clock Monitor enable option has been selected by the program).
- With the single-pin R/C oscillator mask option selected and the CLKDLY bit reset, the WATCHDOG service window will resume following HALT mode from where it left off before entering the HALT mode.
- With the crystal oscillator mask option selected, or with the single-pin R/C oscillator mask option selected and the CLKDLY bit set, the WATCHDOG service window will be set to its selected value from WDSVR following HALT. Consequently, the WATCHDOG should not be serviced for at least 2048 instruction cycles following HALT, but must be serviced within the selected window to avoid a WATCHDOG error.
- The IDLE timer T0 is not initialized with reset.
- The user can sync in to the IDLE counter cycle with an IDLE counter (T0) interrupt or by monitoring the T0PND flag. The T0PND flag is set whenever the thirteenth bit of the IDLE counter toggles (every 4096 instruction cycles). The user is responsible for resetting the T0PND flag.
- A hardware WATCHDOG service occurs just as the device exits the IDLE mode. Consequently, the Watchdog should not be serviced for at least 2048 instruction cycles following IDLE, but must be serviced within the selected window to avoid a WATCHDOG error.
- Following reset, the initial WATCHDOG service (where the service window and the Clock Monitor enable/disable must be selected) may be programmed anywhere within the maximum service window (65,536 instruction cycles) initialized by RESET. Note that this initial WATCHDOG service may be programmed within the initial 2048 instruction cycles without causing a WATCHDOG error.

Detection of Illegal Conditions

The device can detect various illegal conditions resulting from coding errors, transient noise, power supply voltage drops, runaway programs, etc.

Reading of undefined ROM gets zeros. The opcode for software interrupt is zero. If the program fetches instructions from undefined ROM, this will force a software interrupt, thus signaling that an illegal condition has occurred.

Detection of Illegal Conditions (Continued)

The subroutine stack grows down for each call (jump to subroutine), interrupt, or PUSH, and grows up for each return or POP. The stack pointer is initialized to RAM location 06F Hex during reset. Consequently, if there are more returns than calls, the stack pointer will point to addresses 070 and 071 Hex (which are undefined RAM). Undefined RAM from addresses 070 to 07F Hex is read as all 1's, which in turn will cause the program to return to address 7FFF Hex. This is an undefined ROM location and the instruction fetched (all 0's) from this location will generate a software interrupt signaling an illegal condition.

Thus, the chip can detect the following illegal conditions:

a. Executing from undefined ROM

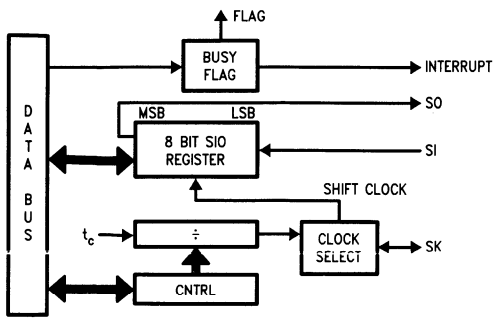
b. Over "POP"ing the stack by having more returns than calls.

When the software interrupt occurs, the user can re-initialize the stack pointer and do a recovery procedure before restarting (this recovery program is probably similar to that following reset, but might not contain the same program initialization procedures). The recovery program should reset the software interrupt pending bit using the RPND instruction.

MICROWIRE/PLUS

MICROWIRE/PLUS is a serial synchronous communications interface. The MICROWIRE/PLUS capability enables the device to interface with any of National Semiconductor's MICROWIRE peripherals (i.e. A/D converters, display drivers, E^{2}PROMs etc.) and with other microcontrollers which support the MICROWIRE interface. It consists of an 8-bit serial shift register (SIO) with serial data input (SI), serial data output (SO) and serial shift clock (SK). *Figure 12* shows a block diagram of the MICROWIRE logic.

The shift clock can be selected from either an internal source or an external source. Operating the MICROWIRE/PLUS arrangement with the internal clock source is called the Master mode of operation. Similarly, operating the MICROWIRE arrangement with an external shift clock is called the Slave mode of operation.



TL/DD/9766–20

FIGURE 12. MICROWIRE/PLUS Block Diagram

The CNTRL register is used to configure and control the MICROWIRE/PLUS mode. To use the MICROWIRE/PLUS, the MSEL bit in the CNTRL register is set to one. In the master mode, the SK clock rate is selected by the two bits, SL0 and SL1, in the CNTRL register. Table IV details the different clock rates that may be selected.

MICROWIRE/PLUS OPERATION

Setting the BUSY bit in the PSW register causes the MICROWIRE/PLUS to start shifting the data. It gets reset when eight data bits have been shifted. The user may reset the BUSY bit by software to allow less than 8 bits to shift. If enabled, an interrupt is generated when eight data bits have been shifted. The device may enter the MICROWIRE/PLUS mode either as a Master or as a Slave. *Figure 13* shows how two COP888CL microcontrollers and several peripherals may be interconnected using the MICROWIRE/PLUS arrangements.

Warning:

The SIO register should only be loaded when the SK clock is low. Loading the SIO register while the SK clock is high will result in undefined data in the SIO register. The SK clock is normally low when not shifting.

Setting the BUSY flag when the input SK clock is high in the MICROWIRE/PLUS slave mode may cause the current SK clock for the SIO shift register to be narrow. For safety, the BUSY flag should only be set when the input SK clock is low.

MICROWIRE/PLUS Master Mode Operation

In the MICROWIRE/PLUS Master mode of operation the shift clock (SK) is generated internally. The MICROWIRE Master always initiates all data exchanges. The MSEL bit in the CNTRL register must be set to enable the SO and SK functions onto the G Port. The SO and SK pins must also be selected as outputs by setting appropriate bits in the Port G configuration register. Table V summarizes the bit settings required for Master mode of operation.

MICROWIRE/PLUS Slave Mode Operation

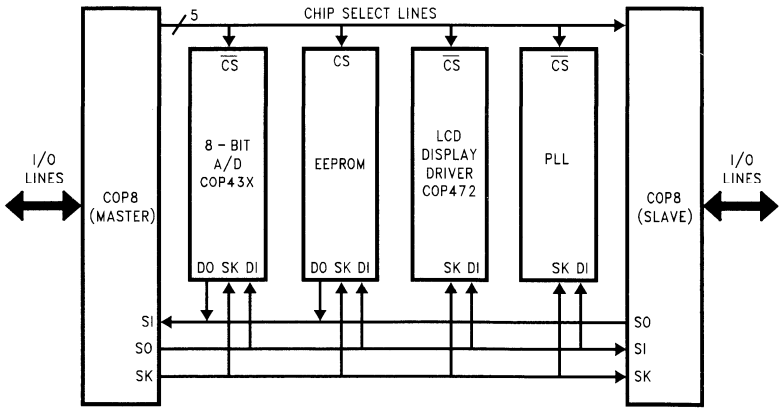
In the MICROWIRE/PLUS Slave mode of operation the SK clock is generated by an external source. Setting the MSEL bit in the CNTRL register enables the SO and SK functions onto the G Port. The SK pin must be selected as an input and the SO pin is selected as an output pin by setting and resetting the appropriate bit in the Port G configuration register. Table V summarizes the settings required to enter the Slave mode of operation.

The user must set the BUSY flag immediately upon entering the Slave mode. This will ensure that all data bits sent by the Master will be shifted properly. After eight clock pulses the BUSY flag will be cleared and the sequence may be repeated.

Alternate SK Phase Operation

The device allows either the normal SK clock or an alternate phase SK clock to shift data in and out of the SIO register. In both the modes the SK is normally low. In the normal mode data is shifted in on the rising edge of the SK clock and the data is shifted out on the falling edge of the SK clock. The SIO register is shifted on each falling edge of the SK clock. In the alternate SK phase operation, data is shifted in on the falling edge of the SK clock and shifted out on the rising edge of the SK clock.

MICROWIRE/PLUS (Continued)



TL/DD/9766-21

FIGURE 13. MICROWIRE/PLUS Application

A control flag, SKSEL, allows either the normal SK clock or the alternate SK clock to be selected. Resetting SKSEL causes the MICROWIRE/PLUS logic to be clocked from the normal SK signal. Setting the SKSEL flag selects the alternate SK clock. The SKSEL is mapped into the G6 configuration bit. The SKSEL flag will power up in the reset condition, selecting the normal SK signal.

TABLE V

This table assumes that the control flag MSEL is set.

G4 (SO) Config. Bit	G5 (SK) Config. Bit	G4 Fun.	G5 Fun.	Operation
1	1	SO	Int. SK	MICROWIRE/PLUS Master
0	1	TRI-STATE	Int. SK	MICROWIRE/PLUS Master
1	0	SO	Ext. SK	MICROWIRE/PLUS Slave
0	0	TRI-STATE	Ext. SK	MICROWIRE/PLUS Slave

Memory Map

All RAM, ports and registers (except A and PC) are mapped into data memory address space

Address	Contents
00 to 6F	On-Chip RAM bytes
70 to BF	Unused RAM Address Space
C0	Timer T2 Lower Byte
C1	Timer T2 Upper Byte
C2	Timer T2 Autoload Register T2RA Lower Byte
C3	Timer T2 Autoload Register T2RA Upper Byte
C4	Timer T2 Autoload Register T2RB Lower Byte
C5	Timer T2 Autoload Register T2RB Upper Byte
C6	Timer T2 Control Register
C7	WATCHDOG Service Register (Reg:WDSVR)
C8	MIWU Edge Select Register (Reg:WKEDG)
C9	MIWU Enable Register (Reg:WKEN)
CA	MIWU Pending Register (Reg:WKPND)
CB	Reserved
CC	Reserved
CD to CF	Reserved
D0	Port L Data Register
D1	Port L Configuration Register
D2	Port L Input Pins (Read Only)
D3	Reserved for Port L
D4	Port G Data Register
D5	Port G Configuration Register
D6	Port G Input Pins (Read Only)
D7	Port I Input Pins (Read Only)
D8	Port C Data Register
D9	Port C Configuration Register
DA	Port C Input Pins (Read Only)
DB	Reserved for Port C
DC	Port D Data Register
DD to DF	Reserved for Port D
E0 to E5	Reserved
E6	Timer T1 Autoload Register T1RB Lower Byte
E7	Timer T1 Autoload Register T1RB Upper Byte
E8	ICNTRL Register
E9	MICROWIRE Shift Register
EA	Timer T1 Lower Byte
EB	Timer T1 Upper Byte
EC	Timer T1 Autoload Register T1RA Lower Byte
ED	Timer T1 Autoload Register T1RA Upper Byte
EE	CNTRL Control Register
EF	PSW Register
F0 to FB	On-Chip RAM Mapped as Registers
FC	X Register
FD	SP Register
FE	B Register
FF	Reserved

Reading memory locations 70-7F Hex will return all ones. Reading other unused memory locations will return undefined data.

Addressing Modes

The device has ten addressing modes, six for operand addressing and four for transfer of control.

OPERAND ADDRESSING MODES

Register Indirect

This is the "normal" addressing mode. The operand is the data memory addressed by the B pointer or X pointer.

Register Indirect (with auto post increment or decrement of pointer)

This addressing mode is used with the LD and X instructions. The operand is the data memory addressed by the B pointer or X pointer. This is a register indirect mode that automatically post increments or decrements the B or X register after executing the instruction.

Direct

The instruction contains an 8-bit address field that directly points to the data memory for the operand.

Immediate

The instruction contains an 8-bit immediate field as the operand.

Short Immediate

This addressing mode is used with the Load B Immediate instruction. The instruction contains a 4-bit immediate field as the operand.

Indirect

This addressing mode is used with the LAID instruction. The contents of the accumulator are used as a partial address (lower 8 bits of PC) for accessing a data operand from the program memory.

TRANSFER OF CONTROL ADDRESSING MODES

Relative

This mode is used for the JP instruction, with the instruction field being added to the program counter to get the new program location. JP has a range from −31 to +32 to allow a 1-byte relative jump (JP + 1 is implemented by a NOP instruction). There are no "pages" when using JP, since all 15 bits of PC are used.

Absolute

This mode is used with the JMP and JSR instructions, with the instruction field of 12 bits replacing the lower 12 bits of the program counter (PC). This allows jumping to any location in the current 4k program memory segment.

Absolute Long

This mode is used with the JMPL and JSRL instructions, with the instruction field of 15 bits replacing the entire 15 bits of the program counter (PC). This allows jumping to any location in the current 4k program memory space.

Indirect

This mode is used with the JID instruction. The contents of the accumulator are used as a partial address (lower 8 bits of PC) for accessing a location in the program memory. The contents of this program memory location serve as a partial address (lower 8 bits of PC) for the jump to the next instruction.

Note: The VIS is a special case of the Indirect Transfer of Control addressing mode, where the double byte vector associated with the interrupt is transferred from adjacent addresses in the program memory into the program counter (PC) in order to jump to the associated interrupt service routine.

Instruction Set

Register and Symbol Definition

Registers	
A	8-Bit Accumulator Register
B	8-Bit Address Register
X	8-Bit Address Register
SP	8-Bit Stack Pointer Register
PC	15-Bit Program Counter Register
PU	Upper 7 Bits of PC
PL	Lower 8 Bits of PC
C	1 Bit of PSW Register for Carry
HC	1 Bit of PSW Register for Half Carry
GIE	1 Bit of PSW Register for Global Interrupt Enable
VU	Interrupt Vector Upper Byte
VL	Interrupt Vector Lower Byte

Symbols	
[B]	Memory Indirectly Addressed by B Register
[X]	Memory Indirectly Addressed by X Register
MD	Direct Addressed Memory
Mem	Direct Addressed Memory or [B]
Meml	Direct Addressed Memory or [B] or Immediate Data
Imm	8-Bit Immediate Data
Reg	Register Memory: Addresses F0 to FF (Includes B, X and SP)
Bit	Bit Number (0 to 7)
←	Loaded with
↔	Exchanged with

Instruction Set (Continued)

INSTRUCTION SET

ADD	A,Meml	ADD	A ← A + Meml
ADC	A,Meml	ADD with Carry	A ← A + Meml + C, C ← Carry HC ← Half Carry
SUBC	A,Meml	Subtract with Carry	A ← A − $\overline{\text{Meml}}$ + C, C ← Carry HC ← Half Carry
AND	A,Meml	Logical AND	A ← A and Meml
ANDSZ	A,Imm	Logical AND Immed., Skip if Zero	Skip next if (A and Imm) = 0
OR	A,Meml	Logical OR	A ← A or Meml
XOR	A,Meml	Logical EXclusive OR	A ← A xor Meml
IFEQ	MD,Imm	IF EQual	Compare MD and Imm, Do next if MD = Imm
IFEQ	A,Meml	IF EQual	Compare A and Meml, Do next if A = Meml
IFNE	A,Meml	IF Not Equal	Compare A and Meml, Do next if A ≠ Meml
IFGT	A,Meml	IF Greater Than	Compare A and Meml, Do next if A > Meml
IFBNE	#	If B Not Equal	Do next if lower 4 bits of B ≠ Imm
DRSZ	Reg	Decrement Reg., Skip if Zero	Reg ← Reg − 1, Skip if Reg = 0
SBIT	#,Mem	Set BIT	1 to bit, Mem (bit = 0 to 7 immediate)
RBIT	#,Mem	Reset BIT	0 to bit, Mem
IFBIT	#,Mem	IF BIT	If bit in A or Mem is true do next instruction
RPND		Reset PeNDing Flag	Reset Software Interrupt Pending Flag
X	A,Mem	EXchange A with Memory	A ⟷ Mem
X	A,[X]	EXchange A with Memory [X]	A ⟷ [X]
LD	A,Meml	LoaD A with Memory	A ← Meml
LD	A,[X]	LoaD A with Memory [X]	A ← [X]
LD	B,Imm	LoaD B with Immed.	B ← Imm
LD	Mem,Imm	LoaD Memory Immed	Mem ← Imm
LD	Reg,Imm	LoaD Register Memory Immed.	Reg ← Imm
X	A, [B ±]	EXchange A with Memory [B]	A ⟷ [B], (B ← B ± 1)
X	A, [X ±]	EXchange A with Memory [X]	A ⟷ [X], (X ← ± 1)
LD	A, [B ±]	LoaD A with Memory [B]	A ← [B], (B ← B ± 1)
LD	A, [X ±]	LoaD A with Memory [X]	A ← [X], (X ← X ± 1)
LD	[B ±],Imm	LoaD Memory [B] Immed.	[B] ← Imm, (B ← ± 1)
CLR	A	CLeaR A	A ← 0
INC	A	INCrement A	A ← A + 1
DEC	A	DECrementA	A ← A − 1
LAID		Load A InDirect from ROM	A ← ROM (PU,A)
DCOR	A	Decimal CORrect A	A ← BCD correction of A (follows ADC, SUBC)
RRC	A	Rotate A Right thru C	C ⟷ A7 ⟷ . . . ⟷ A0 ⟷ C
RLC	A	Rotate A Left thru C	C ← A7 ← . . . ← A0 ← C
SWAP	A	SWAP nibbles of A	A7 . . . A4 ⟷ A3 . . . A0
SC		Set C	C ← 1, HC ← 1
RC		Reset C	C ← 0, HC ← 0
IFC		IF C	IF C is true, do next instruction
IFNC		IF Not C	If C is not true, do next instruction
POP	A	POP the stack into A	SP ← SP + 1, A ← [SP]
PUSH	A	PUSH A onto the stack	[SP] ← A, SP ← SP − 1
VIS		Vector to Interrupt Service Routine	PU ← [VU], PL ← [VL]
JMPL	Addr.	Jump absolute Long	PC ← ii (ii = 15 bits, 0 to 32k)
JMP	Addr.	Jump absolute	PC9 . . . 0 ← i (i = 12 bits)
JP	Disp.	Jump relative short	PC ← PC + r (r is −31 to +32, except 1)
JSRL	Addr.	Jump SubRoutine Long	[SP] ← PL, [SP−1] ← PU,SP−2, PC ← ii
JSR	Addr	Jump SubRoutine	[SP] ← PL, [SP−1] ← PU,SP−2, PC9 . . . 0 ← i
JID		Jump InDirect	PL ← ROM (PU,A)
RET		RETurn from subroutine	SP+2, PL ← [SP], PU ← [SP−1]
RETSK		RETurn and SKip	SP+2, PL ← [SP],PU ← [SP−1]
RETI		RETurn from Interrupt	SP+2, PL ← [SP],PU ← [SP−1],GIE ← 1
INTR		Generate an Interrupt	[SP] ← PL, [SP−1] ← PU, SP−2, PC ← 0FF
NOP		No OPeration	PC ← PC+1

Instruction Execution Time

Most instructions are single byte (with immediate addressing mode instructions taking two bytes).

Most single byte instructions take one cycle time to execute.

Skipped instructions require x number of cycles to be skipped, where x equals the number of bytes in the skipped instruction opcode.

See the BYTES and CYCLES per INSTRUCTION table for details.

Bytes and Cycles per Instruction

The following table shows the number of bytes and cycles for each instruction in the format of byte/cycle.

Arithmetic and Logic Instructions

	[B]	Direct	Immed.
ADD	1/1	3/4	2/2
ADC	1/1	3/4	2/2
SUBC	1/1	3/4	2/2
AND	1/1	3/4	2/2
OR	1/1	3/4	2/2
XOR	1/1	3/4	2/2
IFEQ	1/1	3/4	2/2
IFNE	1/1	3/4	2/2
IFGT	1/1	3/4	2/2
IFBNE	1/1		
DRSZ		1/3	
SBIT	1/1	3/4	
RBIT	1/1	3/4	
IFBIT	1/1	3/4	

RPND	1/1

Instructions Using A & C

CLRA	1/1
INCA	1/1
DECA	1/1
LAID	1/3
DCOR	1/1
RRCA	1/1
RLCA	1/1
SWAPA	1/1
SC	1/1
RC	1/1
IFC	1/1
IFNC	1/1
PUSHA	1/3
POPA	1/3
ANDSZ	2/2

Transfer of Control Instructions

JMPL	3/4
JMP	2/3
JP	1/3
JSRL	3/5
JSR	2/5
JID	1/3
VIS	1/5
RET	1/5
RETSK	1/5
RETI	1/5
INTR	1/7
NOP	1/1

Memory Transfer Instructions

	Register Indirect		Direct	Immed.	Register Indirect Auto Incr. & Decr.		
	[B]	[X]			[B+, B−]	[X+, X−]	
X A,*	1/1	1/3	2/3		1/2	1/3	
LD A,*	1/1	1/3	2/3	2/2	1/2	1/3	
LD B, Imm				1/1			(IF B < 16)
LD B, Imm				2/2			(IF B > 15)
LD Mem, Imm	2/2		3/3		2/2		
LD Reg, Imm			2/3				
IFEQ MD, Imm			3/3				

* = > Memory location addressed by B or X or directly.

Opcode Table

Upper Nibble Along X-Axis

Lower Nibble Along Y-Axis

F	E	D	C	B	A	9	8	
JP −15	JP −31	LD 0F0, # i	DRSZ 0F0	RRCA	RC	ADC A,#i	ADC A,[B]	0
JP −14	JP −30	LD 0F1, # i	DRSZ 0F1	*	SC	SUBC A, #i	SUB A,[B]	1
JP −13	JP −29	LD 0F2, # i	DRSZ 0F2	X A, [X+]	X A,[B+]	IFEQ A,#i	IFEQ A,[B]	2
JP −12	JP −28	LD 0F3, # i	DRSZ 0F3	X A, [X−]	X A,[B−]	IFGT A,#i	IFGT A,[B]	3
JP −11	JP −27	LD 0F4, # i	DRSZ 0F4	VIS	LAID	ADD A,#i	ADD A,[B]	4
JP −10	JP −26	LD 0F5, # i	DRSZ 0F5	RPND	JID	AND A,#i	AND A,[B]	5
JP −9	JP −25	LD 0F6, # i	DRSZ 0F6	X A,[X]	X A,[B]	XOR A,#i	XOR A,[B]	6
JP −8	JP −24	LD 0F7, # i	DRSZ 0F7	*	*	OR A,#i	OR A,[B]	7
JP −7	JP −23	LD 0F8, # i	DRSZ 0F8	NOP	RLCA	LD A,#i	IFC	8
JP −6	JP −22	LD 0F9, # i	DRSZ 0F9	IFNE A,[B]	IFEQ Md,#i	IFNE A,#i	IFNC	9
JP −5	JP −21	LD 0FA, # i	DRSZ 0FA	LD A,[X+]	LD A,[B+]	LD [B+],#i	INCA	A
JP −4	JP −20	LD 0FB, # i	DRSZ 0FB	LD A,[X−]	LD A,[B−]	LD [B−],#i	DECA	B
JP −3	JP −19	LD 0FC, # i	DRSZ 0FC	LD Md,#i	JMPL	X A,Md	POPA	C
JP −2	JP −18	LD 0FD, # i	DRSZ 0FD	DIR	JSRL	LD A,Md	RETSK	D
JP −1	JP −17	LD 0FE, # i	DRSZ 0FE	LD A,[X]	LD A,[B]	LD [B],#i	RET	E
JP −0	JP −16	LD 0FF, # i	DRSZ 0FF	*	*	LD B,#i	RETI	F

Opcode Table (Continued)

Upper Nibble Along X-Axis

Lower Nibble Along Y-Axis

7	6	5	4	3	2	1	0	
IFBIT 0,[B]	ANDSZ A, #i	LD B,#0F	IFBNE 0	JSR x000–x0FF	JMP x000–x0FF	JP +17	INTR	0
IFBIT 1,[B]	*	LD B,#0E	IFBNE 1	JSR x100–x1FF	JMP x100–x1FF	JP +18	JP + 2	1
IFBIT 2,[B]	*	LD B,#0D	IFBNE 2	JSR x200–x2FF	JMP x200–x2FF	JP +19	JP + 3	2
IFBIT 3,[B]	*	LD B,#0C	IFBNE 3	JSR x300–x3FF	JMP x300–x3FF	JP +20	JP + 4	3
IFBIT 4,[B]	CLRA	LD B,#0B	IFBNE 4	JSR x400–x4FF	JMP x400–x4FF	JP +21	JP + 5	4
IFBIT 5,[B]	SWAPA	LD B,#0A	IFBNE 5	JSR x500–x5FF	JMP x500–x5FF	JP +22	JP + 6	5
IFBIT 6,[B]	DCORA	LD B,#09	IFBNE 6	JSR x600–x6FF	JMP x600–x6FF	JP +23	JP + 7	6
IFBIT 7,[B]	PUSHA	LD B,#08	IFBNE 7	JSR x700–x7FF	JMP x700–x7FF	JP +24	JP + 8	7
SBIT 0,[B]	RBIT 0,[B]	LD B,#07	IFBNE 8	JSR x800–x8FF	JMP x800–x8FF	JP +25	JP + 9	8
SBIT 1,[B]	RBIT 1,[B]	LD B,#06	IFBNE 9	JSR x900–x9FF	JMP x900–x9FF	JP +26	JP + 10	9
SBIT 2,[B]	RBIT 2,[B]	LD B,#05	IFBNE 0A	JSR xA00–xAFF	JMP xA00–xAFF	JP +27	JP + 11	A
SBIT 3,[B]	RBIT 3,[B]	LD B,#04	IFBNE 0B	JSR xB00–xBFF	JMP xB00–xBFF	JP +28	JP + 12	B
SBIT 4,[B]	RBIT 4,[B]	LD B,#03	IFBNE 0C	JSR xC00–xCFF	JMP xC00–xCFF	JP +29	JP + 13	C
SBIT 5,[B]	RBIT 5,[B]	LD B,#02	IFBNE 0D	JSR xD00–xDFF	JMP xD00–xDFF	JP +30	JP + 14	D
SBIT 6,[B]	RBIT 6,[B]	LD B,#01	IFBNE 0E	JSR xE00–xEFF	JMP xE00–xEFF	JP +31	JP + 15	E
SBIT 7,[B]	RBIT 7,[B]	LD B,#00	IFBNE 0F	JSR xF00–xFFF	JMP xF00–xFFF	JP +32	JP + 16	F

Where,

i is the immediate data

Md is a directly addressed memory location

* is an unused opcode

Note: The opcode 60 Hex is also the opcode for IFBIT #i,A

Mask Options

The mask programmable options are shown below. The options are programmed at the same time as the ROM pattern submission.

```
OPTION 1: CLOCK CONFIGURATION
  = 1     Crystal Oscillator (CKI/lO)
              G7 (CKO) is clock generator
              output to crystal/resonator
              CKI is the clock input
  = 2     Single-pin RC controlled
              oscillator (CKI/lO)
              G7 is available as a HALT
              restart and/or general purpose
              input

OPTION 2: HALT
  = 1     Enable HALT mode
  = 2     Disable HALT mode

OPTION 3: BONDING
  = 1     44-Pin PCC
  = 2     40-Pin DIP
  = 3     N.A.
  = 4     28-Pin DIP
  = 5     28-Pin SO
```

Development Support

SUMMARY

- iceMASTER™: IM-COP8/400—Full feature in-circuit emulation for all COP8 products. A full set of COP8 Basic and Feature Family device and package specific probes are available.
- COP8 Debug Module: Moderate cost in-circuit emulation and development programming unit.
- COP8 Evaluation and Programming Unit: EPU-COP888GG—low cost in-circuit simulation and development programming unit.
- Assembler: COP8-DEV-IBMA. A DOS installable cross development Assembler, Linker, Librarian and Utility Software Development Tool Kit.
- C Compiler: COP8C. A DOS installable cross development Software Tool kit.
- OPT/EPROM Programmer Support: Covering needs from engineering prototype, pilot production to full production environments.

Development Support (Continued)

IceMASTER (IM) IN-CIRCUIT EMULATION

The iceMASTER IM-COP8/400 is a full feature, PC based, in-circuit emulation tool developed and marketed by Meta-Link Corporation to support the whole COP8 family of products. National is a resale vendor for these products.

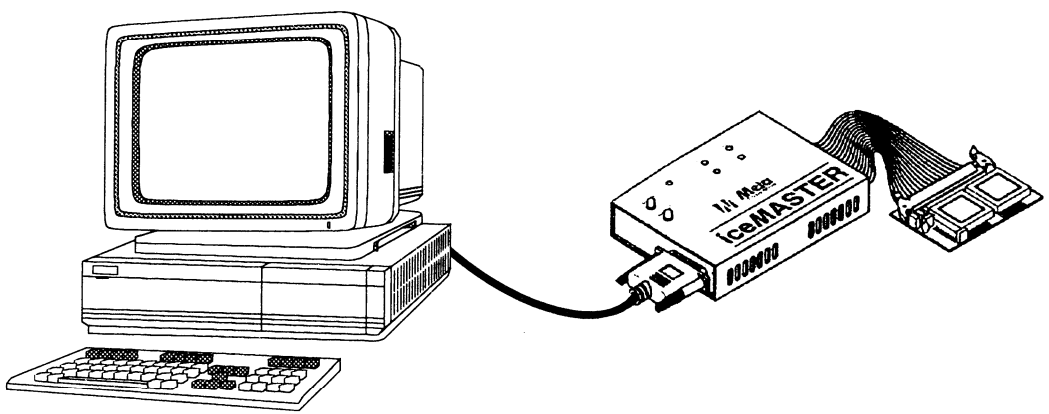
See *Figure 19* for configuration.

The iceMASTER IM-COP8/400 with its device specific COP8 Probe provides a rich feature set for developing, testing and maintaining product:

- Real-time in-circuit emulation; full 2.4V–5.5V operation range, full DC-10 MHz clock. Chip options are programmable or jumper selectable.
- Direct connection to application board by package compatible socket or surface mount assembly.
- Full 32 kbyte of loadable programming space that overlays (replaces) the on-chip ROM or EPROM. On-chip RAM and I/O blocks are used directly or recreated on the probe as necessary.
- Full 4k frame synchronous trace memory. Address, instruction, and 8 unspecified, circuit connectable trace lines. Display can be HLL source (e.g., C source), assembly or mixed.
- A full 64k hardware configurable break, trace on, trace off control, and pass count increment events.
- Tool set integrated interactive symbolic debugger—supports both assembler (COFF) and C Compiler (.COD) linked object formats.
- Real time performance profiling analysis; selectable bucket definition.
- Watch windows, content updated automatically at each execution break.
- Instruction by instruction memory/register changes displayed on source window when in single step operation.
- Single base unit and debugger software reconfigurable to support the entire COP8 family; only the probe personality needs to change. Debugger software is processor customized, and reconfigured from a master model file.
- Processor specific symbolic display of registers and bit level assignments, configured from master model file.
- Halt/Idle mode notification.
- On-line HELP customized to specific processor using master model file.
- Includes a copy of COP8-DEV-IBMA assembler and linker SDK.

IM Order Information

Base Unit	
IM-COP8/400-1	iceMASTER Base Unit, 110V Power Supply
IM-COP8/400-2	iceMASTER Base Unit, 220V Power Supply
iceMASTER Probe	
MHW-884CL28DWPC	28 DIP
MHW-888CL40DWPC	40 DIP
MHW-888CL44PWPC	44 PLCC
Adapter for SO package	
MHW-SO-SOIC28	28 SO



TL/DD/9766–35

FIGURE 19. COP8 iceMASTER Environment

5

Development Support (Continued)

IceMASTER DEBUG MODULE (DM)

The iceMASTER IM-COP8/400 is a PC based, combination in-circuit emulation tool and COP8 based OPT/EPROM programming tool developed and marketed by MetaLink Corporation to support the whole COP8 family of products. National is a resale vendor for these products.

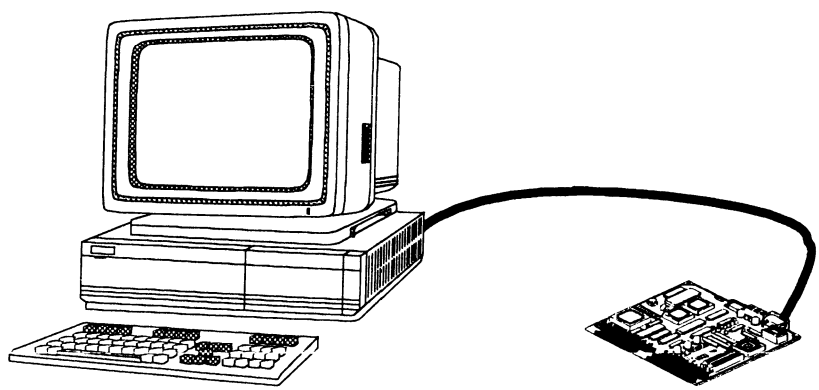
See *Figure 20* for configuration.

The iceMASTER Debug Module is a moderate cost development tool. It has the capability of in-circuit emulation for a specific COP8 microcontroller and in addition serves as a programming tool for COP8 OTP and EPROM product families. Summary of features is as follows:

- Real-time in-circuit emulation; full operating voltage range operation, full DC-10 MHz clock.
- All processor I/O pins can be cabled to an application development board with package compatible cable to socket and surface mount assembly.
- Full 32 kbyte of loadable programming space that overlays (replaces) the on-chip ROM or EPROM. On-chip RAM and I/O blocks are used directly or recreated on the probe as necessary.
- 100 frames of synchronous trace memory. The display can be HLL source (C source), assembly or mixed. The most recent history prior to a break is available in the trace memory.
- Configured break points; uses INTR instruction which is modestly intrusive.
- Software—only supported features are selectable.
- Tool set integrated interactive symbolic debugger - supports both assembler (COFF) and C Compiler (.COD) SDK linked object formats.
- Instruction by instruction memory/register changes displayed when in single step operation.
- Debugger software is processed customized, and reconfigured from a master model file.
- Processor specific symbolic display of registers and bit level assignments, configured from master model file.
- Halt/Idle mode notification.
- Programming menu supports full product line of programmable OTP and EPROM COP8 products. Program data is taken directly from the overlay RAM.
- Programming of 44PLCC and 68PLCC parts requires external programming adapters.
- Includes wallmount power supply.
- On-board V_{PP} generator from 5V input or connection to external supply supported. Requires V_{PP} level adjustment per the family programming specification (correct level is provided on an on-screen pop-down display).
- ON-Line HELP customized to specific processor using master model file.
- Includes a copy of COP8-DEV-IBMA assembler and linker SDK.

DM Order Information

Debug Module Unit	
COP8-DM/888CF	
Cable Adapters	
DM-COP8/28D	28 DIP
DM-COP8/40D	40 DIP
DM-COP8/44P	44 PLCC
Adapter for SO package	
MHW-SO -SOIC28	28 SO



TL/DD/9766–36

FIGURE 20. COP8-DM Environment

Development Support (Continued)

COP8 ASSEMBLER/LINKER SOFTWARE DEVELOPMENT TOOL KIT

National Semiconductor offers a relocateable COP8 macro cross assembler, linker, librarian and utility software development tool kit. Features are summarized as follows:

- Basic and Feature Family instruction set by "device" type
- Nested macro capability.
- Extensive set of assembler directives.
- Supported on PC/DOS platform.
- Generates National standard COFF output files.
- Integrated Linker and Librarian.
- Integrated utilities to generate ROM code file outputs.
- DUMPCOFF utility.

This product is integrated as a part of MetaLink tools as a development kit, fully supported by the MetaLink debugger. It may be ordered separately or it is bundled with the MetaLink products at no additional cost.

Order Information

Assembler SDK:	
COP8-DEV-IBMA	Assembler SDK on installable 3.5" PC/DOS Floppy Disk Drive format. Periodic upgrades and most recent version is available on National's BBS and Internet.

COP8 C COMPILER

A C Compiler is developed and marketed by Byte Craft Limited. The COP8C compiler is a fully integrated development tool specifically designed to support the compact embedded configuration of the COP8 family of products.

Features are summarized as follows:

- ANSI C with some restrictions and extensions that optimize development for the COP8 embedded application.
- BITS data type extension. Register declaration #pragma with direct bit level definitions.
- C language support for interrupt routines.
- Expert system, rule based code geration and optimization.
- Performs consistency checks against the architectural definitions of the target COP8 device.
- Generates program memory code.
- Supports linking of compiled object or COP8 assembled object formats.
- Global optimization of linked code.
- Symbolic debug load format fully source level supported by the MetaLink debugger.

Approved List

Manufacturer	North America	Europe	Asia
BP Microsystems	(800) 225-2102 (713) 688-4600 Fax: (713) 688-0920	+49-8152-4183 +49-8856-932616	+852-234-16611 +852-2710-8121
Data I/O	(800) 426-1045 (206) 881-6444 Fax: (206) 882-1043	+44-0734-440011	Call North America
HI-LO	(510) 623-8860	Call Asia	+886-2-764-0215 Fax: +886-2-756-6403
ICE Technology	(800) 624-8949 (919) 430-7915	+44-1226-767404 Fax: 0-1226-370-434	
MetaLink	(800) 638-2423 (602) 926-0797 Fax: (602) 693-0681	+49-80 9156 96-0 Fax: +49-80 9123 86	+852-737-1800
Systems General	(408) 263-6667	+41-1-9450300	+886-2-917-3005 Fax: +886-2-911-1283
Needhams	(916) 924-8037 Fax: (916) 924-8065		

Development Support (Continued)

SINGLE CHIP OTP/EMULATOR SUPPORT

The COP8 family is supported by single chip OTP emulators. For detailed information refer to the emulator specific datasheet and the emulator selection table below:

OTP Emulator Ordering Information

Device Number	Clock Option	Package	Emulates
COP87L84CLN-XE	Crystal	28 DIP	COP884CL
COP87L84CLM-XE	Crystal	28 SO	COP884CL
COP87L88CLN-XE	Crystal	40 DIP	COP888CL
COP87L88CLV-XE	Crystal	44 PLCC	COP888CL

INDUSTRY WIDE OTP/EPROM PROGRAMMING SUPPORT

Programming support, in addition to the MetaLink development tools, is provided by a full range of independent approved vendors to meet the needs from the engineering laboratory to full production.

AVAILABLE LITERATURE

For more information, please see the COP8 Basic Family User's Manual, Literature Number 620895, COP8 Feature Family User's Manual, Literature Number 620897 and National's Family of 8-bit Microcontrollers COP8 Selection Guide, Literature Number 630009.

DIAL-A-HELPER SERVICE

Dial-A-Helper is a service provided by the Microcontroller Applications group. The Dial-A-Helper is an Electronic Information System that may be accessed as a Bulletin Board System (BBS) via data modem, as an FTP site on the Internet via standard FTP client application or as an FTP site on the Internet using a standard Internet browser such as Netscape or Mosaic.

The Dial-A-Helper system provides access to an automated information storage and retrieval system. The system capabilities include a MESSAGE SECTION (electronic mail, when accessed as a BBS) for communications to and from the Microcontroller Applications Group and a FILE SECTION which consists of several file areas where valuable application software and utilities could be found.

DIAL-A-HELPER BBS via a Standard Modem

Modem: CANADA/U.S.: (800) NSC-MICRO
(800) 672-6427
EUROPE: (+49) 0-8141-351332
Baud: 14.4k
Set-Up: Length: 8-Bit
Parity: None
Stop Bit: 1
Operation: 24 Hours, 7 Days

DIAL-A-HELPER via FTP

ftp nscmicro.nsc.com
user: anonymous
password: username@yourhost.site.domain

DIAL-A-HELPER via a WorldWide Web Browser

ftp://nscmicro.nsc.com

National Semiconductor on the WorldWide Web

See us on the WorldWide Web at: http://www.national.com

CUSTOMER RESPONSE CENTER

Complete product information and technical support is available from National's customer response centers.

CANADA/U.S.:		Tel:	(800) 272-9959
	email:		support@tevm2.nsc.com
EUROPE	email:		europe.support@nsc.com
	Deutsch	Tel:	+49 (0) 180-530 85 85
	English	Tel:	+49 (0) 180-532 78 32
	Francais	Tel:	+49 (0) 180-532 93 58
	Italiano	Tel:	+49 (0) 180-534 16 80
JAPAN:		Tel:	+81-043-299-2309
S.E. ASIA:	Beijing	Tel:	(+86)10-6856-8601
	Shanghai	Tel:	(+86)21-6415-4092
	Hong Kong	Tel:	(+852) 2737-1600
	Korea	Tel:	(+82) 2-3771-6909
	Malaysia	Tel:	(+60-4) 644-9061
	Singapore	Tel:	(+65) 255-2226
	Taiwan	Tel:	+886-2-521-3288
AUSTRALIA:		Tel:	(+61) 3-9558-9999
INDIA:		Tel:	(+91) 80-559-9467

National Semiconductor

COP988CF/COP984CF/COP888CF/COP884CF 8-Bit CMOS Microcontroller with A/D Converter

General Description

The COP888 family of microcontrollers uses an 8-bit single chip core architecture fabricated with National Semiconductor's M²CMOS™ process technology. The COP888CF is a member of this expandable 8-bit core processor family of microcontrollers. (Continued)

Key Features

- A/D converter (8-bit, 8-channel, with prescaler and both differential and single ended modes)
- Two 16-bit timers, each with two 16-bit registers supporting:
  - — Processor Independent PWM mode
  - — External Event counter mode
  - — Input Capture mode
- 4 kbytes of on-chip ROM
- 128 bytes of on-chip RAM

Additional Peripheral Features

- Idle Timer
- Multi-Input Wake Up (MIWU) with optional interrupts (8)
- WATCHDOG™ and Clock Monitor logic
- MICROWIRE/PLUS™ serial I/O

I/O Features

- Memory mapped I/O
- Software selectable I/O options (TRI-STATE® Output, Push-Pull Output, Weak Pull-Up Input, High Impedance Input)
- High current outputs
- Schmitt trigger inputs on Port G
- Packages:
  - — 44 PLCC with 38 I/O pins
  - — 40 DIP with 34 I/O pins
  - — 28 DIP with 24 I/O pins
  - — 28 SO with 21 I/O pins

CPU/Instruction Set Feature

- 1 μs instruction cycle time
- Ten multi-source vectored interrupts servicing
  - — External interrupt with selectable edge
  - — Idle Timer T0
  - — Two Timers (Each with 2 interrupts)
  - — MICROWIRE/PLUS
  - — Multi-Input Wake Up
  - — Software Trap
  - — Default VIS (default interrupt)
- Versatile and easy to use instruction set
- 8-bit Stack Pointer (SP)—stack in RAM
- Two 8-bit Register Indirect Data Memory Pointers (B, X)

Fully Static CMOS

- Low current drain (typically < 1 μA)
- Single supply operation: 2.5V to 6.0V
- Temperature ranges: 0°C to +70°C, and −40°C to +85°C

Development Support

- Emulation and OTP devices
- Real time emulation and full program debug offered by MetaLink Development System

Block Diagram

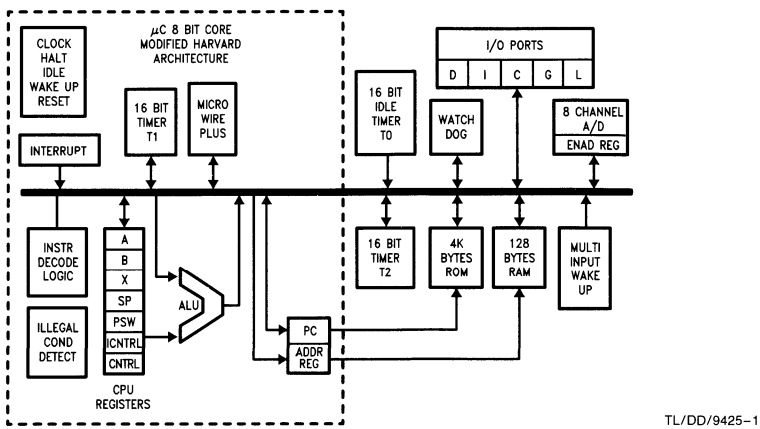


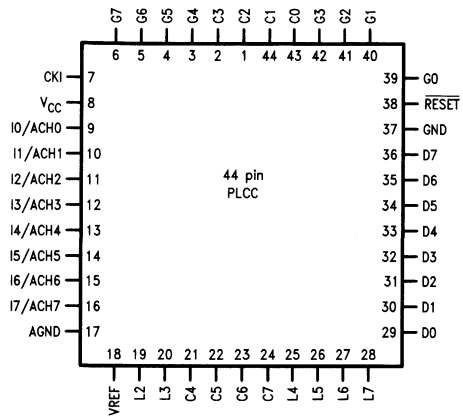
FIGURE 1. Block Diagram

General Description (Continued)

It is a fully static part, fabricated using double-metal silicon gate microCMOS technology. Features include an 8-bit memory mapped architecture, MICROWIRE/PLUS serial I/O, two 16-bit timer/counters supporting three modes (Processor Independent PWM generation, External Event counter, and Input Capture mode capabilities), an 8-channel, 8-bit A/D converter with both differential and single ended modes, and two power savings modes (HALT and IDLE), both with a multi-sourced wakeup/interrupt capability. This multi-sourced interrupt capability may also be used independent of the HALT or IDLE modes. Each I/O pin has software selectable configurations. The device operates over a voltage range of 2.5V to 6V. High throughput is achieved with an efficient, regular instruction set operating at a maximum of 1 μs per instruction rate.

Connection Diagrams

Plastic Chip Carrier

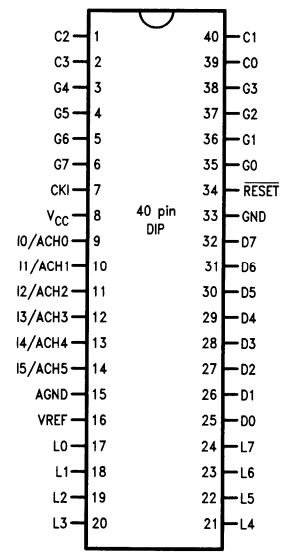


TL/DD/9425-2

Top View

Order Number COP888CF-XXX/V
COP988CF-XXX/V or COP988CFH-XXX/V
See NS Plastic Chip Package Number V44A

Dual-In-Line Package

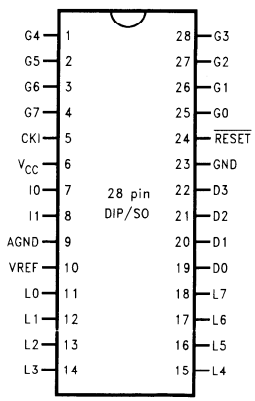


TL/DD/9425-4

Top View

Order Number COP888CF-XXX/N,
COP988CF-XXX/N or COP988CFH-XXX/N
See NS Molded Package Number N40A

Dual-In-Line Package



TL/DD/9425-37

Top View

Order Number COP884CF-XXX/N,
COP884CF-XXX/WM, COP984CF-XXX/N,
COP984CFH-XXX/N, COP984CFH-XXX/WM
or COP984CFH-XXX/WM
See NS Package Number N28B or M28B

FIGURE 2. Connection Diagrams

Connection Diagrams (Continued)

Pinouts for 28-, 40- and 44-Pin Packages

Port	Type	Alt. Fun	Alt. Fun	28-Pin Pack.	40-Pin Pack.	44-Pin Pack.
L0	I/O	MIWU		11	17	—
L1	I/O	MIWU		12	18	—
L2	I/O	MIWU		13	19	19
L3	I/O	MIWU		14	20	20
L4	I/O	MIWU	T2A	15	21	25
L5	I/O	MIWU	T2B	16	22	26
L6	I/O	MIWU		17	23	27
L7	I/O	MIWU		18	24	28
G0	I/O	INT		25	35	39
G1	WDOUT			26	36	40
G2	I/O	T1B		27	37	41
G3	I/O	T1A		28	38	42
G4	I/O	SO		1	3	3
G5	I/O	SK		2	4	4
G6	I	SI		3	5	5
G7	I/CKO	HALT Restart		4	6	6
D0	O			19	25	29
D1	O			20	26	30
D2	O			21	27	31
D3	O			22	28	32
I0	I	ACH0		7	9	9
I1	I	ACH1		8	10	10
I2	I	ACH2			11	11
I3	I	ACH3			12	12
I4	I	ACH4			13	13
I5	I	ACH5			14	14
I6	I	ACH6				15
I7	I	ACH7				16
D4	O				29	33
D5	O				30	34
D6	O				31	35
D7	O				32	36
C0	I/O				39	43
C1	I/O				40	44
C2	I/O				1	1
C3	I/O				2	2
C4	I/O					21
C5	I/O					22
C6	I/O					23
C7	I/O					24
V_{REF}	$+V_{REF}$			10	16	18
AGND	AGND			9	15	17
V_{CC}				6	8	8
GND				23	33	37
CKI				5	7	7
$\overline{\text{RESET}}$				24	34	38

Absolute Maximum Ratings

If Military/Aerospace specified devices are required, please contact the National Semiconductor Sales Office/Distributors for availability and specifications.

Supply Voltage (V_{CC})	7V
Voltage at Any Pin	−0.3V to V_{CC} + 0.3V
Total Current into V_{CC} Pin (Source)	100 mA
Total Current out of GND Pin (Sink)	110 mA
Storage Temperature Range	−65°C to +140°C

Note: *Absolute maximum ratings indicate limits beyond which damage to the device may occur. DC and AC electrical specifications are not ensured when operating the device at absolute maximum ratings.*

DC Electrical Characteristics 988CF: 0°C ≤ T_A ≤ +70°C unless otherwise specified

Parameter	Conditions	Min	Typ	Max	Units
Operating Voltage					
988CF		2.5		4.0	V
998CFH		4.0		6.0	V
Power Supply Ripple (Note 1)	Peak-to-Peak			0.1 V_{CC}	V
Supply Current (Note 2)					
CKI = 10 MHz	V_{CC} = 6V, t_c = 1 μs			12.5	mA
CKI = 4 MHz	V_{CC} = 6V, t_c = 2.5 μs			5.5	mA
CKI = 4 MHz	V_{CC} = 4V, t_c = 2.5 μs			2.5	mA
CKI = 1 MHz	V_{CC} = 4V, t_c = 10 μs			1.4	mA
HALT Current (Note 3)	V_{CC} = 6V, CKI = 0 MHz		<0.7	8	μA
	V_{CC} = 4.0V, CKI = 0 MHz		<0.3	4	μA
IDLE Current					
CKI = 10 MHz	V_{CC} = 6V, t_c = 1 μs			3.5	mA
CKI = 4 MHz	V_{CC} = 6V, t_c = 2.5 μs			2.5	mA
CKI = 1 MHz	V_{CC} = 4.0V, t_c = 10 μs			0.7	mA
Input Levels					
$\overline{\text{RESET}}$					
Logic High		0.8 V_{CC}			V
Logic Low				0.2 V_{CC}	V
CKI (External and Crystal Osc. Modes)					
Logic High		0.7 V_{CC}			V
Logic Low				0.2 V_{CC}	V
All Other Inputs					
Logic High		0.7 V_{CC}			V
Logic Low				0.2 V_{CC}	V
Hi-Z Input Leakage	V_{CC} = 6V	−1		+1	μA
Input Pullup Current	V_{CC} = 6V, V_{IN} = 0V	−40		−250	μA
G and L Port Input Hysteresis				0.35 V_{CC}	V
Output Current Levels					
D Outputs					
Source	V_{CC} = 4V, V_{OH} = 3.3V	−0.4			mA
	V_{CC} = 2.5V, V_{OH} = 1.8V	−0.2			mA
Sink	V_{CC} = 4V, V_{OL} = 1V	10			mA
	V_{CC} = 2.5V, V_{OL} = 0.4V	2.0			mA
All Others					
Source (Weak Pull-Up Mode)	V_{CC} = 4V, V_{OH} = 2.7V	−10		−100	μA
	V_{CC} = 2.5V, V_{OH} = 1.8V	−2.5		−33	μA
Source (Push-Pull Mode)	V_{CC} = 4V, V_{OH} = 3.3V	−0.4			mA
	V_{CC} = 2.5V, V_{OH} = 1.8V	−0.2			mA
Sink (Push-Pull Mode)	V_{CC} = 4V, V_{OL} = 0.4V	1.6			mA
	V_{CC} = 2.5V, V_{OL} = 0.4V	0.7			mA

Note 1: Rate of voltage change must be less then 0.5 V/ms.

Note 2: Supply current is measured after running 2000 cycles with a square wave CKI input, CKO open, inputs at rails and outputs open.

Note 3: The HALT mode will stop CKI from oscillating in the RC and the Crystal configurations. Test conditions: All inputs tied to V_{CC}, L and G0–G5 configured as outputs and set high. The D port set to zero. The A/D is disabled. V_{REF} is tied to AGND (effectively shorting the Reference resistor). The clock monitor is disabled.

DC Electrical Characteristics $0°C \leq T_A \leq +70°C$ unless otherwise specified (Continued)

Parameter	Conditions	Min	Typ	Max	Units
TRI-STATE Leakage	$V_{CC} = 6.0V$	−1		+1	μA
Allowable Sink/Source Current per Pin D Outputs (Sink) All others				 15 3	 mA mA
Maximum Input Current without Latchup (Note 6)	$T_A = 25°C$			±100	mA
RAM Retention Voltage, V_r	500 ns Rise and Fall Time (Min)	2			V
Input Capacitance				7	pF
Load Capacitance on D2				1000	pF

A/D Converter Specifications $V_{CC} = 5V \pm 10\%$ $(V_{SS} - 0.050V) \leq$ Any Input $\leq (V_{CC} + 0.050V)$

Parameter	Conditions	Min	Typ	Max	Units
Resolution				8	Bits
Reference Voltage Input	AGND = 0V	3		V_{CC}	V
Absolute Accuracy	$V_{REF} = V_{CC}$			±1	LSB
Non-Linearity	$V_{REF} = V_{CC}$ Deviation from the Best Straight Line			±½	LSB
Differential Non-Linearity	$V_{REF} = V_{CC}$			±½	LSB
Input Reference Resistance		1.6		4.8	kΩ
Common Mode Input Range (Note 7)		AGND		V_{REF}	V
DC Common Mode Error				±¼	LSB
Off Channel Leakage Current			1		μA
On Channel Leakage Current			1		μA
A/D Clock Frequency (Note 5)		0.1		1.67	MHz
Conversion Time (Note 4)			12		A/D Clock Cycles

Note 4: Conversion Time includes sample and hold time.

Note 5: See Prescaler description.

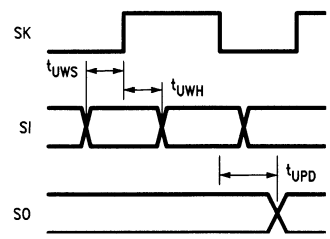
Note 6: Pins G6 and $\overline{\text{RESET}}$ are designed with a high voltage input network for factory testing. These pins allow input voltages greater than V_{CC} and the pins will have sink current to V_{CC} when biased at voltages greater than V_{CC} (the pins do not have source current when biased at a voltage below V_{CC}). The effective resistance to V_{CC} is 750Ω (typical). These two pins will not latch up. The voltage at the pins must be limited to less than 14V.

Note 7: For $V_{IN}(-) \geq V_{IN}(+)$ the digital output code will be 0000 0000. Two on-chip diodes are tied to each analog input. The diodes will forward conduct for analog input voltages below ground or above the V_{CC} supply. Be careful, during testing at low V_{CC} levels (4.5V), as high level analog inputs (5V) can cause this input diode to conduct—especially at elevated temperatures, and cause errors for analog inputs near full-scale. The spec allows 50 mV forward bias of either diode. This means that as long as the analog V_{IN} does not exceed the supply voltage by more than 50 mV, the output code will be correct. To achieve an absolute 0 V_{DC} to 5 V_{DC} input voltage range will therefore require a minimum supply voltage of 4.950 V_{DC} over temperature variations, initial tolerance and loading. The voltage at any analog input should be $-0.3V$ to $V_{CC} + 0.3V$.

AC Electrical Characteristics $0°C \leq T_A \leq +70°C$ unless otherwise specified

Parameter	Conditions	Min	Typ	Max	Units
Instruction Cycle Time (t_c)					
Crystal, Resonator	$4V \leq V_{CC} \leq 6V$	1		DC	μs
	$2.5V \leq V_{CC} < 4V$	2.5		DC	μs
R/C Oscillator	$4V \leq V_{CC} \leq 6V$	3		DC	μs
	$2.5V \leq V_{CC} < 4V$	7.5		DC	μs
Inputs					
t_{SETUP}	$4V \leq V_{CC} \leq 6V$	200			ns
	$2.5V \leq V_{CC} < 4V$	500			ns
t_{HOLD}	$4V \leq V_{CC} \leq 6V$	60			ns
	$2.5V \leq V_{CC} < 4V$	150			ns
Output Propagation Delay (Note 8)	$R_L = 2.2k, C_L = 100$ pF				
t_{PD1}, t_{PD0}					
SO, SK	$4V \leq V_{CC} \leq 6V$			0.7	μs
	$2.5V \leq V_{CC} < 4V$			1.75	μs
All Others	$4V \leq V_{CC} \leq 6V$			1	μs
	$2.5V \leq V_{CC} < 4V$			2.5	μs
MICROWIRE™ Setup Time (t_{UWS})		20			ns
MICROWIRE Hold Time (t_{UWH})		56			ns
MICROWIRE Output Propagation Delay (t_{UPD})				220	ns
Input Pulse Width					
Interrupt Input High Time		1			t_c
Interrupt Input Low Time		1			t_c
Timer Input High Time		1			t_c
Timer Input Low Time		1			t_c
Reset Pulse Width		1			μs

Note 8: The output propagation delay is referenced to the end of the instruction cycle where the output change occurs.



TL/DD/9425–26

FIGURE 3. MICROWIRE/PLUS Timing

Absolute Maximum Ratings

If Military/Aerospace specified devices are required, please contact the National Semiconductor Sales Office/Distributors for availability and specifications.

Supply Voltage (V_{CC})	7V
Voltage at Any Pin	−0.3V to V_{CC} + 0.3V
Total Current into V_{CC} Pin (Source)	100 mA
Total Current out of GND Pin (Sink)	110 mA
Storage Temperature Range	−65°C to +140°C

Note: *Absolute maximum ratings indicate limits beyond which damage to the device may occur. DC and AC electrical specifications are not ensured when operating the device at absolute maximum ratings.*

DC Electrical Characteristics 888CF: $-40°C \le T_A \le +85°C$ unless otherwise specified

Parameter	Conditions	Min	Typ	Max	Units
Operating Voltage		2.5		6	V
Power Supply Ripple (Note 1)	Peak-to-Peak			0.1 V_{CC}	V
Supply Current (Note 2)					
CKI = 10 MHz	V_{CC} = 6V, t_c = 1 μs			12.5	mA
CKI = 4 MHz	V_{CC} = 4V, t_c = 2.5 μs			2.5	mA
HALT Current (Note 3)	V_{CC} = 6V, CKI = 0 MHz		<1	10	μA
IDLE Current					
CKI = 10 MHz	V_{CC} = 6V, t_c = 1 μs			3.5	mA
CKI = 1 MHz	V_{CC} = 4V, t_c = 10 μs			0.7	mA
Input Levels					
$\overline{\text{RESET}}$					
Logic High		0.8 V_{CC}			V
Logic Low				0.2 V_{CC}	V
CKI (External and Crystal Osc. Modes)					
Logic High		0.7 V_{CC}			V
Logic Low				0.2 V_{CC}	V
All Other Inputs					
Logic High		0.7 V_{CC}			V
Logic Low				0.2 V_{CC}	V
Hi-Z Input Leakage	V_{CC} = 6V	−2		+2	μA
Input Pullup Current	V_{CC} = 6V, V_{IN} = 0V	−40		−250	μA
G and L Port Input Hysteresis				0.35 V_{CC}	V
Output Current Levels					
D Outputs					
Source	V_{CC} = 4V, V_{OH} = 3.3V	−0.4			mA
	V_{CC} = 2.5V, V_{OH} = 1.8V	−0.2			mA
Sink	V_{CC} = 4V, V_{OL} = 1V	10			mA
	V_{CC} = 2.5V, V_{OL} = 0.4V	2.0			mA
All Others					
Source (Weak Pull-Up Mode)	V_{CC} = 4V, V_{OH} = 2.7V	−10		−100	μA
	V_{CC} = 2.5V, V_{OH} = 1.8V	−2.5		−33	μA
Source (Push-Pull Mode)	V_{CC} = 4V, V_{OH} = 3.3V	−0.4			mA
	V_{CC} = 2.5V, V_{OH} = 1.8V	−0.2			mA
Sink (Push-Pull Mode)	V_{CC} = 4V, V_{OL} = 0.4V	1.6			mA
	V_{CC} = 2.5V, V_{OL} = 0.4V	0.7			mA
TRI-STATE Leakage	V_{CC} = 6.0V	−2		+2	μA

Note 1: Rate of voltage change must be less then 0.5 V/ms.

Note 2: Supply current is measured after running 2000 cycles with a square wave CKI input, CKO open, inputs at rails and outputs open.

Note 3: The HALT mode will stop CKI from oscillating in the RC and the Crystal configurations. Test conditions: All inputs tied to V_{CC}, L and G0–G5 configured as outputs and set high. The D port set to zero. The A/D is disabled. V_{REF} is tied to AGND (effectively shorting the Reference resistor). The clock monitor is disabled.

DC Electrical Characteristics 888CF: $-40°C \leq T_A \leq +85°C$ unless otherwise specified (Continued)

Parameter	Conditions	Min	Typ	Max	Units
Allowable Sink/Source Current per Pin D Outputs (Sink) All others				 15 3	 mA mA
Maximum Input Current without Latchup (Note 6)	$T_A = 25°C$			±100	mA
RAM Retention Voltage, V_r	500 ns Rise and Fall Time (Min)	2			V
Input Capacitance				7	pF
Load Capacitance on D2				1000	pF

A/D Converter Specifications 888CF:

$V_{CC} = 5V \pm 10\%$ $(V_{SS} - 0.050V) \leq$ Any Input $\leq (V_{CC} + 0.050V)$

Parameter	Conditions	Min	Typ	Max	Units
Resolution				8	Bits
Reference Voltage Input	AGND = 0V	3		V_{CC}	V
Absolute Accuracy	$V_{REF} = V_{CC}$			±1	LSB
Non-Linearity	$V_{REF} = V_{CC}$ Deviation from the Best Straight Line			±½	LSB
Differential Non-Linearity	$V_{REF} = V_{CC}$			±½	LSB
Input Reference Resistance		1.6		4.8	kΩ
Common Mode Input Range (Note 7)		AGND		V_{REF}	V
DC Common Mode Error				±¼	LSB
Off Channel Leakage Current			1		μA
On Channel Leakage Current			1		μA
A/D Clock Frequency (Note 5)		0.1		1.67	MHz
Conversion Time (Note 4)			12		A/D Clock Cycles

Note 4: Conversion Time includes sample and hold time.

Note 5: See Prescaler description.

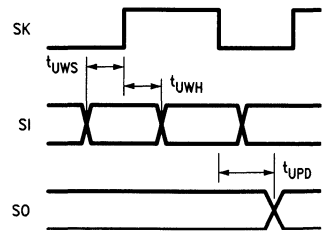
Note 6: Pins G6 and $\overline{\text{RESET}}$ are designed with a high voltage input network for factory testing. These pins allow input voltages greater than V_{CC} and the pins will have sink current to V_{CC} when biased at voltages greater than V_{CC} (the pins do not have source current when biased at a voltage below V_{CC}). The effective resistance to V_{CC} is 750Ω (typical). These two pins will not latch up. The voltage at the pins must be limited to less than 14V.

Note 7: For $V_{IN}(-) \geq V_{IN}(+)$ the digital output code will be 0000 0000. Two on-chip diodes are tied to each analog input. The diodes will forward conduct for analog input voltages below ground or above the V_{CC} supply. Be careful, during testing at low V_{CC} levels (4.5V), as high level analog inputs (5V) can cause this input diode to conduct—especially at elevated temperatures, and cause errors for analog inputs near full-scale. The spec allows 50 mV forward bias of either diode. This means that as long as the analog V_{IN} does not exceed the supply voltage by more than 50 mV, the output code will be correct. To achieve an absolute 0 V_{DC} to 5 V_{DC} input voltage range will therefore require a minimum supply voltage of 4.950 V_{DC} over temperature variations, initial tolerance and loading. The voltage on any analog input should be $-0.3V$ to $V_{CC} + 0.3V$.

AC Electrical Characteristics 888CF: $-40°C \le T_A \le +85°C$ unless otherwise specified

Parameter	Conditions	Min	Typ	Max	Units
Instruction Cycle Time (t_c)					
Crystal, Resonator	$4V \le V_{CC} \le 6V$	1		DC	μs
	$2.5V \le V_{CC} < 4V$	2.5		DC	μs
R/C Oscillator	$4V \le V_{CC} \le 6V$	3		DC	μs
	$2.5V \le V_{CC} < 4V$	7.5		DC	μs
Inputs					
t_{SETUP}	$4V \le V_{CC} \le 6V$	200			ns
	$2.5V \le V_{CC} < 4V$	500			ns
t_{HOLD}	$4V \le V_{CC} \le 6V$	60			ns
	$2.5V \le V_{CC} < 4V$	150			ns
Output Propagation Delay (Note 8)	$R_L = 2.2k, C_L = 100$ pF				
t_{PD1}, t_{PD0}					
SO, SK	$4V \le V_{CC} \le 6V$			0.7	μs
	$2.5V \le V_{CC} < 4V$			1.75	μs
All Others	$4V \le V_{CC} \le 6V$			1	μs
	$2.5V \le V_{CC} < 4V$			2.5	μs
MICROWIRE™ Setup Time (t_{UWS})		20			ns
MICROWIRE Hold Time (t_{UWH})		56			ns
MICROWIRE Output Propagation Delay (t_{UPD})				220	ns
Input Pulse Width					
Interrupt Input High Time		1			t_c
Interrupt Input Low Time		1			t_c
Timer Input High Time		1			t_c
Timer Input Low Time		1			t_c
Reset Pulse Width		1			μs

Note 8: The output propagation delay is referenced to end of the instruction cycle where the output change occurs.

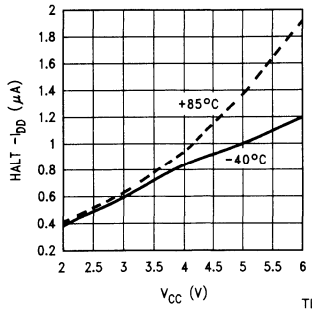


TL/DD/9425–26

FIGURE 3. MICROWIRE/PLUS Timing

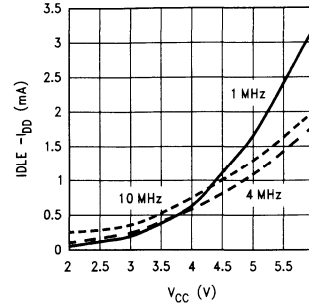
Typical Performance Characteristics (−40°C to +85°C)

Halt—I_{DD} vs V_{CC}



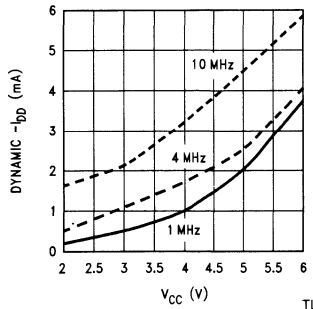
TL/DD/9425-29

Idle—I_{DD} (Crystal Clock Option)



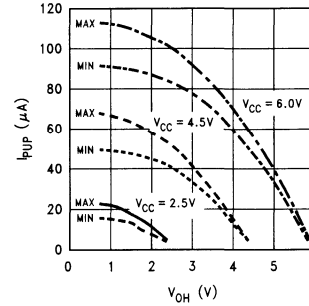
TL/DD/9425-30

Dynamic—I_{DD} (Crystal Clock Option)



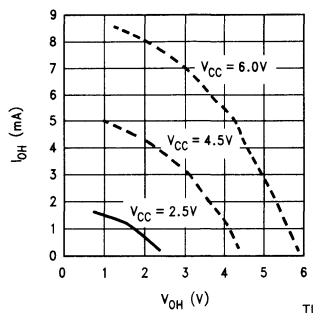
TL/DD/9425-31

Port L/C/G Weak Pull-Up Source Current



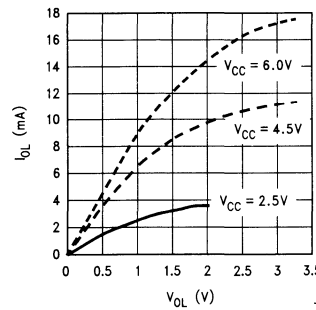
TL/DD/9425-32

Port L/C/G Push-Pull Source Current



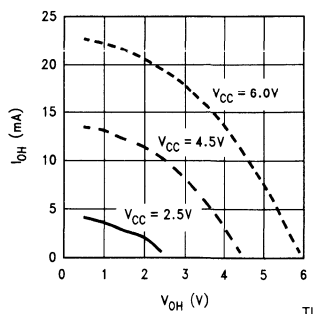
TL/DD/9425-33

Port L/C/G Push-Pull Sink Current



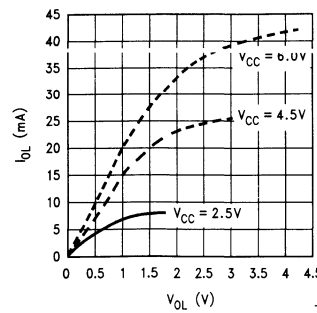
TL/DD/9425-34

Port D Source Current



TL/DD/9425-35

Port D Sink Current



TL/DD/9425-36

Pin Descriptions

V_{CC} and GND are the power supply pins.

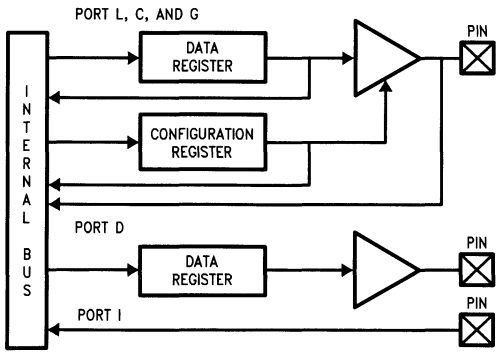
V_{REF} and AGND are the reference voltage pins for the on-board A/D converter.

CKI is the clock input. This can come from an R/C generated oscillator, or a crystal oscillator (in conjunction with CKO). See Oscillator Description section.

$\overline{\text{RESET}}$ is the master reset input. See Reset Description section.

The device contains three bidirectional 8-bit I/O ports (C, G and L), where each individual bit may be independently configured as an input (Schmitt trigger inputs on ports G and L), output or TRI-STATE under program control. Three data memory address locations are allocated for each of these I/O ports. Each I/O port has two associated 8-bit memory mapped registers, the CONFIGURATION register and the output DATA register. A memory mapped address is also reserved for the input pins of each I/O port. (See the memory map for the various addresses associated with the I/O ports.) *Figure 4* shows the I/O port configurations. The DATA and CONFIGURATION registers allow for each port bit to be individually configured under software control as shown below:

CONFIGURATION Register	DATA Register	Port Set-Up
0	0	Hi-Z Input (TRI-STATE Output)
0	1	Input with Weak Pull-Up
1	0	Push-Pull Zero Output
1	1	Push-Pull One Output



TL/DD/9425–6

FIGURE 4. I/O Port Configurations

PORT L is an 8-bit I/O port. All L-pins have Schmitt triggers on the inputs.

Port L supports Multi-Input Wakeup (MIWU) on all eight pins. L4 and L5 are used for the timer input functions T2A and T2B. L0 and L1 are not available on the 44-pin version of the device, since they are replaced by V_{REF} and AGND. L0 and L1 are not terminated on the 44-pin version. Consequently, reading L0 or L1 as inputs will return unreliable data with the 44-pin package, so this data should be masked out with user software when the L port is read for input data. It is recommended that the pins be configured as outputs.

Port L has the following alternate features:

- L0 MIWU
- L1 MIWU
- L2 MIWU
- L3 MIWU
- L4 MIWU or T2A
- L5 MIWU or T2B
- L6 MIWU
- L7 MIWU

Port G is an 8-bit port with 5 I/O pins (G0, G2–G5), an input pin (G6), and two dedicated output pins (G1 and G7). Pins G0 and G2–G6 all have Schmitt Triggers on their inputs. Pin G1 serves as the dedicated WDOUT WatchDog output, while pin G7 is either input or output depending on the oscillator mask option selected. With the crystal oscillator option selected, G7 serves as the dedicated output pin for the CKO clock output. With the single-pin R/C oscillator mask option selected, G7 serves as a general purpose input pin, but is also used to bring the device out of HALT mode with a low to high transition on G7. There are two registers associated with the G Port, a data register and a configuration register. Therefore, each of the 5 I/O bits (G0, G2–G5) can be individually configured under software control.

Since G6 is an input only pin and G7 is the dedicated CKO clock output pin or general purpose input (R/C clock configuration), the associated bits in the data and configuration registers for G6 and G7 are used for special purpose functions as outlined below. Reading the G6 and G7 data bits will return zeros.

Note that the chip will be placed in the HALT mode by writing a "1" to bit 7 of the Port G Data Register. Similarly the chip will be placed in the IDLE mode by writing a "1" to bit 6 of the Port G Data Register.

Writing a "1" to bit 6 of the Port G Configuration Register enables the MICROWIRE/PLUS to operate with the alternate phase of the SK clock. The G7 configuration bit, if set high, enables the clock start up delay after HALT when the R/C clock configuration is used.

	Config Reg.	Data Reg.
G7	CLKDLY	HALT
G6	Alternate SK	IDLE

Port G has the following alternate features:

- G0 INTR (External Interrupt Input)
- G2 T1B (Timer T1 Capture Input)
- G3 T1A (Timer T1 I/O)
- G4 SO (MICROWIRE Serial Data Output)
- G5 SK (MICROWIRE Serial Clock)
- G6 SI (MICROWIRE Serial Data Input)

Port G has the following dedicated functions:

- G1 WDOUT WatchDog and/or Clock Monitor dedicated output
- G7 CKO Oscillator dedicated output or general purpose input

Port C is an 8-bit I/O port. The 40-pin device does not have a full complement of Port C pins. The unavailable pins are not terminated. A read operation for these unterminated pins will return unpredictable values.

Pin Descriptions (Continued)

Port I is an 8-bit Hi-Z input port, and also provides the analog inputs to the A/D converter. The 28-pin device does not have a full complement of Port I pins. The unavailable pins are not terminated (i.e. they are floating). A read operation from these unterminated pins will return unpredictable values. The user should ensure that the software takes this into account by either masking out these inputs, or else restricting the accesses to bit operations only. If unterminated, Port I pins will draw power only when addressed.

Port D is an 8-bit output port that is preset high when RESET goes low. The user can tie two or more D port outputs (except D2) together in order to get a higher drive.

Note: Care must be exercised with the D2 pin operation. At RESET, the external loads on this pin must ensure that the output voltages stay above 0.8 V_{CC} to prevent the chip from entering special modes. Also keep the external loading on D2 to less than 1000 pF.

Functional Description

The architecture of the device is modified Harvard architecture. With the Harvard architecture, the control store program memory (ROM) is separated from the data store memory (RAM). Both ROM and RAM have their own separate addressing space with separate address buses. The architecture, though based on Harvard architecture, permits transfer of data from ROM to RAM.

CPU REGISTERS

The CPU can do an 8-bit addition, subtraction, logical or shift operation in one instruction (t_C) cycle time.

There are five CPU registers:

A is the 8-bit Accumulator Register

PC is the 15-bit Program Counter Register

PU is the upper 7 bits of the program counter (PC)
PL is the lower 8 bits of the program counter (PC)

B is an 8-bit RAM address pointer, which can be optionally post auto incremented or decremented.

X is an 8-bit alternate RAM address pointer, which can be optionally post auto incremented or decremented.

SP is the 8-bit stack pointer, which points to the subroutine/interrupt stack (in RAM). The SP is initialized to RAM address 06F with reset.

All the CPU registers are memory mapped with the exception of the Accumulator (A) and the Program Counter (PC).

PROGRAM MEMORY

Program memory consists of 4096 bytes of ROM. These bytes may hold program instructions or constant data (data tables for the LAID instruction, jump vectors for the JID instruction, and interrupt vectors for the VIS instruction). The program memory is addressed by the 15-bit program counter (PC). All interrupts vector to program memory location 0FF Hex.

DATA MEMORY

The data memory address space includes the on-chip RAM and data registers, the I/O registers (Configuration, Data and Pin), the control registers, the MICROWIRE/PLUS SIO shift register, and the various registers, and counters associated with the timers (with the exception of the IDLE timer). Data memory is addressed directly by the instruction or indirectly by the B, X and SP pointers.

The device has 128 bytes of RAM. Sixteen bytes of RAM are mapped as "registers" at addresses 0F0 to 0FF Hex. These registers can be loaded immediately, and also decremented and tested with the DRSZ (decrement register and skip if zero) instruction. The memory pointer registers X, SP, and B are memory mapped into this space at address locations 0FC to 0FE Hex respectively, with the other registers (other than reserved register 0FF) being available for general usage.

The instruction set permits any bit in memory to be set, reset or tested. All I/O and registers (except A and PC) are memory mapped; therefore, I/O bits and register bits can be directly and individually set, reset and tested. The accumulator (A) bits can also be directly and individually tested.

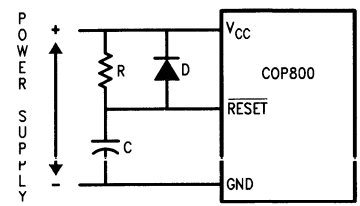
Note: RAM contents are undefined upon power-up.

Reset

The $\overline{\text{RESET}}$ input when pulled low initializes the microcontroller. Initialization will occur whenever the $\overline{\text{RESET}}$ input is pulled low. Upon initialization, the data and configuration registers for Ports L, G, and C are cleared, resulting in these Ports being initialized to the TRI-STATE mode. Pin G1 of the G Port is an exception (as noted below) since pin G1 is dedicated as the WatchDog and/or Clock Monitor error output pin. Port D is initialized high with $\overline{\text{RESET}}$. The PC, PSW, CNTRL, ICNTRL, and T2CNTRL control registers are cleared. The Multi-Input Wakeup registers WKEN, WKEDG, and WKPND are cleared. The A/D control register ENAD is cleared, resulting in the ADC being powered down initially. The Stack Pointer, SP, is initialized to 06F Hex.

The device comes out of reset with both the WatchDog logic and the Clock Monitor detector armed, and with both the WatchDog service window bits set and the Clock Monitor bit set. The WatchDog and Clock Monitor detector circuits are inhibited during reset. The WatchDog service window bits are initialized to the maximum WatchDog service window of 64k t_C clock cycles. The Clock Monitor bit is initialized high, and will cause a Clock Monitor error following reset if the clock has not reached the minimum specified frequency at the termination of reset. A Clock Monitor error will cause an active low error output on pin G1. This error output will continue until 16–32 t_C clock cycles following the clock frequency reaching the minimum specified value, at which time the G1 output will enter the TRI-STATE mode.

The external RC network shown in *Figure 5* should be used to ensure that the $\overline{\text{RESET}}$ pin is held low until the power supply to the chip stabilizes.



TL/DD/9425–7

RC > 5 × Power Supply Rise Time

FIGURE 5. Recommended Reset Circuit

Oscillator Circuits

The chip can be driven by a clock input on the CKI input pin which can be between DC and 10 MHz. The CKO output clock is on pin G7 (crystal configuration). The CKI input frequency is divided down by 10 to produce the instruction cycle clock ($1/t_c$).

Figure 6 shows the Crystal and R/C diagrams.

CRYSTAL OSCILLATOR

CKI and CKO can be connected to make a closed loop crystal (or resonator) controlled oscillator.

Table A shows the component values required for various standard crystal values.

R/C OSCILLATOR

By selecting CKI as a single pin oscillator input, a single pin R/C oscillator circuit can be connected to it. CKO is available as a general purpose input, and/or HALT restart pin.

Table B shows the variation in the oscillator frequencies as functions of the component (R and C) values.

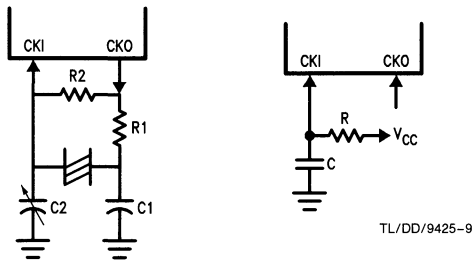


FIGURE 6. Crystal and R/C Oscillator Diagrams

TABLE A. Crystal Oscillator Configuration, T_A = 25°C

R1 (kΩ)	R2 (MΩ)	C1 (pF)	C2 (pF)	CKI Freq (MHz)	Conditions
0	1	30	30–36	10	V_{CC} = 5V
0	1	30	30–36	4	V_{CC} = 5V
0	1	200	100–150	0.455	V_{CC} = 5V

TABLE B. R/C Oscillator Configuration, T_A = 25°C

R (kΩ)	C (pF)	CKI Freq (MHz)	Instr. Cycle (μs)	Conditions
3.3	82	2.2 to 2.7	3.7 to 4.6	V_{CC} = 5V
5.6	100	1.1 to 1.3	7.4 to 9.0	V_{CC} = 5V
6.8	100	0.9 to 1.1	8.8 to 10.8	V_{CC} = 5V

Note: 3k ≤ R ≤ 200k
50 pF ≤ C ≤ 200 pF

Control Registers

CNTRL Register (Address X'00EE)

The Timer1 (T1) and MICROWIRE/PLUS control register contains the following bits:

SL1 & SL0	Select the MICROWIRE/PLUS clock divide by (00 = 2, 01 = 4, 1x = 8)
IEDG	External interrupt edge polarity select (0 = Rising edge, 1 = Falling edge)
MSEL	Selects G5 and G4 as MICROWIRE/PLUS signals SK and SO respectively
T1C0	Timer T1 Start/Stop control in timer modes 1 and 2
	Timer T1 Underflow Interrupt Pending Flag in timer mode 3
T1C1	Timer T1 mode control bit
T1C2	Timer T1 mode control bit
T1C3	Timer T1 mode control bit

T1C3	T1C2	T1C1	T1C0	MSEL	IEDG	SL1	SL0

Bit 7 Bit 0

PSW Register (Address X'00EF)

The PSW register contains the following select bits:

GIE	Global interrupt enable (enables interrupts)
EXEN	Enable external interrupt
BUSY	MICROWIRE/PLUS busy shifting flag
EXPND	External interrupt pending
T1ENA	Timer T1 Interrupt Enable for Timer Underflow or T1A Input capture edge
T1PNDA	Timer T1 Interrupt Pending Flag (Autoreload RA in mode 1, T1 Underflow in Mode 2, T1A capture edge in mode 3)
C	Carry Flag
HC	Half Carry Flag

HC	C	T1PNDA	T1ENA	EXPND	BUSY	EXEN	GIE

Bit 7 Bit 0

5

Control Registers (Continued)

The Half-Carry bit is also affected by all the instructions that affect the Carry flag. The SC (Set Carry) and RC (Reset Carry) instructions will respectively set or clear both the carry flags. In addition to the SC and RC instructions, ADC, SUBC, RRC and RLC instructions affect the carry and Half Carry flags.

ICNTRL Register (Address X'00E8)

The ICNTRL register contains the following bits:

- T1ENB Timer T1 Interrupt Enable for T1B Input capture edge
- T1PNDB Timer T1 Interrupt Pending Flag for T1B capture edge
- μWEN Enable MICROWIRE/PLUS interrupt
- μWPND MICROWIRE/PLUS interrupt pending
- T0EN Timer T0 Interrupt Enable (Bit 12 toggle)
- T0PND Timer T0 Interrupt pending
- LPEN L Port Interrupt Enable (Multi-Input Wakeup/Interrupt)

  Bit 7 could be used as a flag

Unused	LPEN	T0PND	T0EN	μWPND	μWEN	T1PNDB	T1ENB
Bit 7							Bit 0

T2CNTRL Register (Address X'00C6)

The T2CNTRL register contains the following bits:

- T2ENB Timer T2 Interrupt Enable for T2B Input capture edge
- T2PNDB Timer T2 Interrupt Pending Flag for T2B capture edge
- T2ENA Timer T2 Interrupt Enable for Timer Underflow or T2A Input capture edge
- T2PNDA Timer T2 Interrupt Pending Flag (Autoreload RA in mode 1, T2 Underflow in mode 2, T2A capture edge in mode 3)
- T2C0 Timer T2 Start/Stop control in timer modes 1 and 2 Timer T2 Underflow Interrupt Pending Flag in timer mode 3
- T2C1 Timer T2 mode control bit
- T2C2 Timer T2 mode control bit
- T2C3 Timer T2 mode control bit

T2C3	T2C2	T2C1	T2C0	T2PNDA	T2ENA	T2PNDB	T2ENB
Bit 7							Bit 0

Timers

The device contains a very versatile set of timers (T0, T1, T2). All timers and associated autoreload/capture registers power up containing random data.

Figure 7 shows a block diagram for the timers.

TIMER T0 (IDLE TIMER)

The device supports applications that require maintaining real time and low power with the IDLE mode. This IDLE mode support is furnished by the IDLE timer T0, which is a 16-bit timer. The Timer T0 runs continuously at the fixed rate of the instruction cycle clock, t_C. The user cannot read or write to the IDLE Timer T0, which is a count down timer.

The Timer T0 supports the following functions:

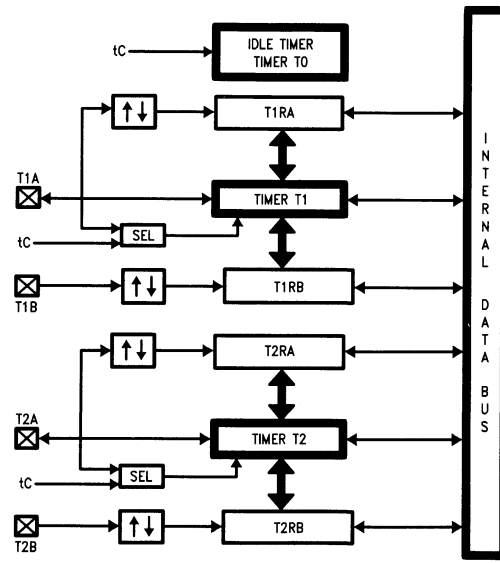
- Exit out of the Idle Mode (See Idle Mode description)
- WatchDog logic (See WatchDog description)
- Start up delay out of the HALT mode

The IDLE Timer T0 can generate an interrupt when the thirteenth bit toggles. This toggle is latched into the T0PND pending flag, and will occur every 4 ms at the maximum clock frequency (t_C = 1 μs). A control flag T0EN allows the interrupt from the thirteenth bit of Timer T0 to be enabled or disabled. Setting T0EN will enable the interrupt, while resetting it will disable the interrupt.

TIMER T1 AND TIMER T2

The device has a set of two powerful timer/counter blocks, T1 and T2. The associated features and functioning of a timer block are described by referring to the timer block Tx. Since the two timer blocks, T1 and T2, are identical, all comments are equally applicable to either timer block.

Each timer block consists of a 16-bit timer, Tx, and two supporting 16-bit autoreload/capture registers, RxA and RxB. Each timer block has two pins associated with it, TxA and TxB. The pin TxA supports I/O required by the timer block, while the pin TxB is an input to the timer block. The powerful and flexible timer block allows the device to



TL/DD/9425–11

FIGURE 7. Timers

easily perform all timer functions with minimal software overhead. The timer block has three operating modes: Processor Independent PWM mode, External Event Counter mode, and Input Capture mode.

The control bits TxC3, TxC2, and TxC1 allow selection of the different modes of operation.

Timers (Continued)

Mode 1. Processor Independent PWM Mode

As the name suggests, this mode allows the COP888CF to generate a PWM signal with very minimal user intervention. The user only has to define the parameters of the PWM signal (ON time and OFF time). Once begun, the timer block will continuously generate the PWM signal completely independent of the microcontroller. The user software services the timer block only when the PWM parameters require updating.

In this mode the timer Tx counts down at a fixed rate of t_C. Upon every underflow the timer is alternately reloaded with the contents of supporting registers, RxA and RxB. The very first underflow of the timer causes the timer to reload from the register RxA. Subsequent underflows cause the timer to be reloaded from the registers alternately beginning with the register RxB.

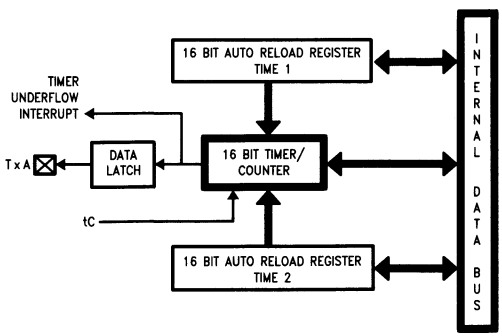
The Tx Timer control bits, TxC3, TxC2 and TxC1 set up the timer for PWM mode operation.

Figure 8 shows a block diagram of the timer in PWM mode.

The underflows can be programmed to toggle the TxA output pin. The underflows can also be programmed to generate interrupts.

Underflows from the timer are alternately latched into two pending flags, TxPNDA and TxPNDB. The user must reset these pending flags under software control. Two control enable flags, TxENA and TxENB, allow the interrupts from the timer underflow to be enabled or disabled. Setting the timer enable flag TxENA will cause an interrupt when a timer underflow causes the RxA register to be reloaded into the timer. Setting the timer enable flag TxENB will cause an interrupt when a timer underflow causes the RxB register to be reloaded into the timer. Resetting the timer enable flags will disable the associated interrupts.

Either or both of the timer underflow interrupts may be enabled. This gives the user the flexibility of interrupting once per PWM period on either the rising or falling edge of the PWM output. Alternatively, the user may choose to interrupt on both edges of the PWM output.



TL/DD/9425–13

FIGURE 8. Timer in PWM Mode

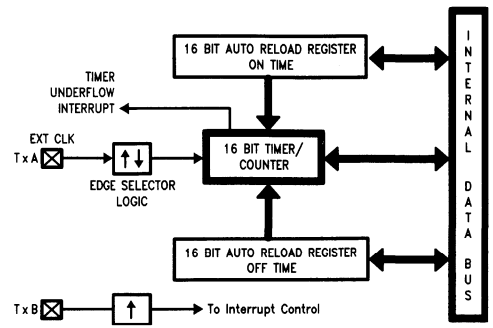
Mode 2. External Event Counter Mode

This mode is quite similar to the processor independent PWM mode described above. The main difference is that the timer, Tx, is clocked by the input signal from the TxA pin. The Tx timer control bits, TxC3, TxC2 and TxC1 allow the timer to be clocked either on a positive or negative edge from the TxA pin. Underflows from the timer are latched into the TxPNDA pending flag. Setting the TxENA control flag will cause an interrupt when the timer underflows.

In this mode the input pin TxB can be used as an independent positive edge sensitive interrupt input if the TxENB control flag is set. The occurrence of a positive edge on the TxB input pin is latched into the TxPNDB flag.

Figure 9 shows a block diagram of the timer in External Event Counter mode.

Note: The PWM output is not available in this mode since the TxA pin is being used as the counter input clock.



TL/DD/9425–14

FIGURE 9. Timer in External Event Counter Mode

Mode 3. Input Capture Mode

The device can precisely measure external frequencies or time external events by placing the timer block, Tx, in the input capture mode.

In this mode, the timer Tx is constantly running at the fixed t_C rate. The two registers, RxA and RxB, act as capture registers. Each register acts in conjunction with a pin. The register RxA acts in conjunction with the TxA pin and the register RxB acts in conjunction with the TxB pin.

The timer value gets copied over into the register when a trigger event occurs on its corresponding pin. Control bits, TxC3, TxC2 and TxC1, allow the trigger events to be specified either as a positive or a negative edge. The trigger condition for each input pin can be specified independently.

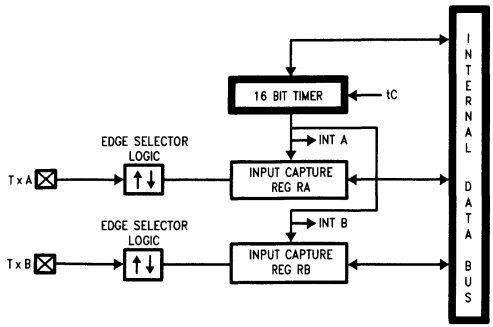
The trigger conditions can also be programmed to generate interrupts. The occurrence of the specified trigger condition on the TxA and TxB pins will be respectively latched into the pending flags, TxPNDA and TxPNDB. The control flag TxENA allows the interrupt on TxA to be either enabled or disabled. Setting the TxENA flag enables interrupts to be generated when the selected trigger condition occurs on the TxA pin. Similarly, the flag TxENB controls the interrupts from the TxB pin.

Underflows from the timer can also be programmed to generate interrupts. Underflows are latched into the timer TxC0 pending flag (the TxC0 control bit serves as the timer underflow interrupt pending flag in the Input Capture mode). Consequently, the TxC0 control bit should be reset when entering the Input Capture mode. The timer underflow interrupt is enabled with the TxENA control flag. When a TxA interrupt occurs in the Input Capture mode, the user must check both

Timers (Continued)

whether a TxA input capture or a timer underflow (or both) caused the interrupt.

Figure 10 shows a block diagram of the timer in Input Capture mode.



TL/DD/9425–15

FIGURE 10. Timer in Input Capture Mode

TIMER CONTROL FLAGS

The timers T1 and T2 have indentical control structures. The control bits and their functions are summarized below.

TxC0	Timer Start/Stop control in Modes 1 and 2 (Processor Independent PWM and External Event Counter), where 1 = Start, 0 = Stop Timer Underflow Interrupt Pending Flag in Mode 3 (Input Capture)
TxPNDA	Timer Interrupt Pending Flag
TxPNDB	Timer Interrupt Pending Flag
TxENA	Timer Interrupt Enable Flag
TxENB	Timer Interrupt Enable Flag 1 = Timer Interrupt Enabled 0 = Timer Interrupt Disabled
TxC3	Timer mode control
TxC2	Timer mode control
TxC1	Timer mode control

The timer mode control bits (TxC3, TxC2 and TxC1) are detailed below:

TxC3	TxC2	TxC1	Timer Mode	Interrupt A Source	Interrupt B Source	Timer Counts On
0	0	0	MODE 2 (External Event Counter)	Timer Underflow	Pos. TxB Edge	TxA Pos. Edge
0	0	1	MODE 2 (External Event Counter)	Timer Underflow	Pos. TxB Edge	TxA Neg. Edge
1	0	1	MODE 1 (PWM) TxA Toggle	Autoreload RA	Autoreload RB	t_C
1	0	0	MODE 1 (PWM) No TxA Toggle	Autoreload RA	Autoreload RB	t_C
0	1	0	MODE 3 (Capture) Captures: TxA Pos. Edge TxB Pos. Edge	Pos. TxA Edge or Timer Underflow	Pos. TxB Edge	t_C
1	1	0	MODE 3 (Capture) Captures: TxA Pos. Edge TxB Neg. Edge	Pos. TxA Edge or Timer Underflow	Neg. TxB Edge	t_C
0	1	1	MODE 3 (Capture) Captures: TxA Neg. Edge TxB Pos. Edge	Neg. TxB Edge or Timer Underflow	Pos. TxB Edge	t_C
1	1	1	MODE 3 (Capture) Captures: TxA Neg. Edge TxB Neg. Edge	Neg. TxA Edge or Timer Underflow	Neg. TxB Edge	t_C

Power Save Modes

The device offers the user two power save modes of operation: HALT and IDLE. In the HALT mode, all microcontroller activities are stopped. In the IDLE mode, the on-board oscillator circuitry and timer T0 are active but all other microcontroller activities are stopped. In either mode, all on-board RAM, registers, I/O states, and timers (with the exception of T0) are unaltered.

HALT MODE

The device is placed in the HALT mode by writing a "1" to the HALT flag (G7 data bit). All microcontroller activities, including the clock, timers, and A/D converter, are stopped. The WatchDog logic is disabled during the HALT mode. However, the clock monitor circuitry if enabled remains active and will cause the WatchDog output pin (WDOUT) to go low. If the HALT mode is used and the user does not want to activate the WDOUT pin, the Clock Monitor should be disabled after the device comes out of reset (resetting the Clock Monitor control bit with the first write to the WDSVR register). In the HALT mode, the power requirements of the device are minimal and the applied voltage (V_{CC}) may be decreased to V_r (V_r = 2.0V) without altering the state of the machine.

The device supports three different ways of exiting the HALT mode. The first method of exiting the HALT mode is with the Multi-Input Wakeup feature on the L port. The second method is with a low to high transition on the CKO (G7) pin. This method precludes the use of the crystal clock configuration (since CKO becomes a dedicated output), and so may be used with an RC clock configuration. The third method of exiting the HALT mode is by pulling the $\overline{\text{RESET}}$ pin low.

Since a crystal or ceramic resonator may be selected as the oscillator, the Wakeup signal is not allowed to start the chip running immediately since crystal oscillators and ceramic resonators have a delayed start up time to reach full amplitude and frequency stability. The IDLE timer is used to generate a fixed delay to ensure that the oscillator has indeed stabilized before allowing instruction execution. In this case, upon detecting a valid Wakeup signal, only the oscillator circuitry is enabled. The IDLE timer is loaded with a value of 256 and is clocked with the t_c instruction cycle clock. The t_c clock is derived by dividing the oscillator clock down by a factor of 10. The Schmitt trigger following the CKI inverter on the chip ensures that the IDLE timer is clocked only when the oscillator has a sufficiently large amplitude to meet the Schmitt trigger specifications. This Schmitt trigger is not part of the oscillator closed loop. The startup timeout from the IDLE timer enables the clock signals to be routed to the rest of the chip.

If an RC clock option is being used, the fixed delay is introduced optionally. A control bit, CLKDLY, mapped as configuration bit G7, controls whether the delay is to be introduced or not. The delay is included if CLKDLY is set, and excluded if CLKDLY is reset. The CLKDLY bit is cleared on reset.

The device has two mask options associated with the HALT mode. The first mask option enables the HALT mode feature, while the second mask option disables the HALT mode. With the HALT mode enable mask option, the device will enter and exit the HALT mode as described above. With the HALT disable mask option, the device cannot be placed in the HALT mode (writing a "1" to the HALT flag will have no effect).

The WatchDog detector circuit is inhibited during the HALT mode. However, the clock monitor circuit if enabled remains active during HALT mode in order to ensure a clock monitor error if the device inadvertently enters the HALT mode as a result of a runaway program or power glitch.

IDLE MODE

The device is placed in the IDLE mode by writing a "1" to the IDLE flag (G6 data bit). In this mode, all activity, except the associated on-board oscillator circuitry, the WatchDog logic, the clock monitor and the IDLE Timer T0, is stopped.

As with the HALT mode, the device can be returned to normal operation with a reset, or with a Multi-Input Wakeup from the L Port. Alternately, the microcontroller resumes normal operation from the IDLE mode when the thirteenth bit (representing 4.096 ms at internal clock frequency of 1 MHz, t_c = 1 μs) of the IDLE Timer toggles.

This toggle condition of the thirteenth bit of the IDLE Timer T0 is latched into the T0PND pending flag.

The user has the option of being interrupted with a transition on the thirteenth bit of the IDLE Timer T0. The interrupt can be enabled or disabled via the T0EN control bit. Setting the T0EN flag enables the interrupt and vice versa.

The user can enter the IDLE mode with the Timer T0 interrupt enabled. In this case, when the T0PND bit gets set, the device will first execute the Timer T0 interrupt service routine and then return to the instruction following the "Enter Idle Mode" instruction.

Alternatively, the user can enter the IDLE mode with the IDLE Timer T0 interrupt disabled. In this case, the device will resume normal operation with the instruction immediately following the "Enter IDLE Mode" instruction.

Note: It is necessary to program two NOP instructions following both the set HALT mode and set IDLE mode instructions. These NOP instructions are necessary to allow clock resynchronization following the HALT or IDLE modes.

Multi-Input Wakeup

The Multi-Input Wakeup feature is used to return (wakeup) the device from either the HALT or IDLE modes. Alternately Multi-Input Wakeup/Interrupt feature may also be used to generate up to 8 edge selectable external interrupts.

Figure 11 shows the Multi-Input Wakeup logic.

The Multi-Input Wakeup feature utilizes the L Port. The user selects which particular L port bit (or combination of L Port bits) will cause the device to exit the HALT or IDLE modes. The selection is done through the Reg: WKEN. The Reg: WKEN is an 8-bit read/write register, which contains a control bit for every L port bit. Setting a particular WKEN bit enables a Wakeup from the associated L port pin.

The user can select whether the trigger condition on the selected L Port pin is going to be either a positive edge (low to high transition) or a negative edge (high to low transition). This selection is made via the Reg: WKEDG, which is an 8-bit control register with a bit assigned to each L Port pin. Setting the control bit will select the trigger condition to be a negative edge on that particular L Port pin. Resetting the bit selects the trigger condition to be a positive edge. Changing an edge select entails several steps in order to avoid a pseudo Wakeup condition as a result of the edge change. First, the associated WKEN bit should be reset, followed by the edge select change in WKEDG. Next, the associated WKPND bit should be cleared, followed by the associated WKEN bit being re-enabled.

An example may serve to clarify this procedure. Suppose we wish to change the edge select from positive (low going high) to negative (high going low) for L Port bit 5, where bit 5 has previously been enabled for an input interrupt. The program would be as follows:

```
RBIT  5, WKEN
SBIT  5, WKEDG
RBIT  5, WKPND
SBIT  5, WKEN
```

If the L port bits have been used as outputs and then changed to inputs with Multi-Input Wakeup/Interrupt, a safety procedure should also be followed to avoid inherited pseudo wakeup conditions. After the selected L port bits have been changed from output to input but before the associated WKEN bits are enabled, the associated edge select bits in WKEDG should be set or reset for the desired edge selects, followed by the associated WKPND bits being cleared.

This same procedure should be used following reset, since the L port inputs are left floating as a result of reset.

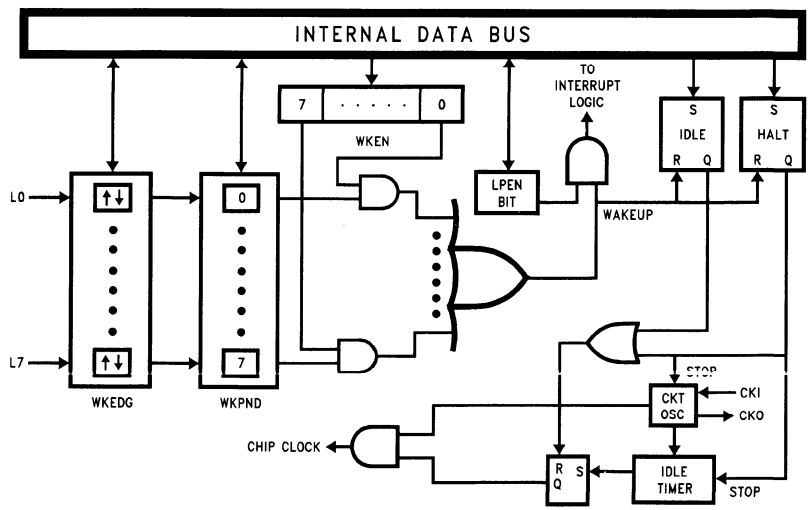
The occurrence of the selected trigger condition for Multi-Input Wakeup is latched into a pending register called WKPND. The respective bits of the WKPND register will be set on the occurrence of the selected trigger edge on the corresponding Port L pin. The user has the responsibility of clearing these pending flags. Since WKPND is a pending register for the occurrence of selected wakeup conditions, the device will not enter the HALT mode if any Wakeup bit is both enabled and pending. Consequently, the user has the responsibility of clearing the pending flags before attempting to enter the HALT mode.

The WKEN, WKPND and WKEDG are all read/write registers, and are cleared at reset.

PORT L INTERRUPTS

Port L provides the user with an additional eight fully selectable, edge sensitive interrupts which are all vectored into the same service subroutine.

The interrupt from Port L shares logic with the wake up circuitry. The register WKEN allows interrupts from Port L to be individually enabled or disabled. The register WKEDG specifies the trigger condition to be either a positive or a negative edge. Finally, the register WKPND latches in the pending trigger conditions.



TL/DD/9425–16

FIGURE 11. Multi-Input Wake Up Logic

Multi-Input Wakeup (Continued)

The GIE (global interrupt enable) bit enables the interrupt function. A control flag, LPEN, functions as a global interrupt enable for Port L interrupts. Setting the LPEN flag will enable interrupts and vice versa. A separate global pending flag is not needed since the register WKPND is adequate.

Since Port L is also used for waking the device out of the HALT or IDLE modes, the user can elect to exit the HALT or IDLE modes either with or without the interrupt enabled. If he elects to disable the interrupt, then the device will restart execution from the instruction immediately following the instruction that placed the microcontroller in the HALT or IDLE modes. In the other case, the device will first execute the interrupt service routine and then revert to normal operation.

The Wakeup signal will not start the chip running immediately since crystal oscillators or ceramic resonators have a finite start up time. The IDLE Timer (T0) generates a fixed delay to ensure that the oscillator has indeed stabilized before allowing the device to execute instructions. In this case, upon detecting a valid Wakeup signal, only the oscillator circuitry and the IDLE Timer T0 are enabled. The IDLE Timer is loaded with a value of 256 and is clocked from the t_C instruction cycle clock. The t_C clock is derived by dividing down the oscillator clock by a factor of 10. A Schmitt trigger following the CKI on-chip inverter ensures that the IDLE timer is clocked only when the oscillator has a sufficiently large amplitude to meet the Schmitt trigger specifications. This Schmitt trigger is not part of the oscillator closed loop. The startup timeout from the IDLE timer enables the clock signals to be routed to the rest of the chip.

If the RC clock option is used, the fixed delay is under software control. A control flag, CLKDLY, in the G7 configuration bit allows the clock start up delay to be optionally inserted. Setting CLKDLY flag high will cause clock start up delay to be inserted and resetting it will exclude the clock start up delay. The CLKDLY flag is cleared during reset, so the clock start up delay is not present following reset with the RC clock options.

A/D Converter

The device contains an 8-channel, multiplexed input, successive approximation, A/D converter. Two dedicated pins, V_{REF} and AGND are provided for voltage reference.

OPERATING MODES

The A/D converter supports ratiometric measurements. It supports both Single Ended and Differential modes of operation.

Four specific analog channel selection modes are supported. These are as follows:

Allow any specific channel to be selected at one time. The A/D converter performs the specific conversion requested and stops.

Allow any specific channel to be scanned continuously. In other words, the user will specify the channel and the A/D converter will keep on scanning it continuously. The user can come in at any arbitrary time and immediately read the result of the last conversion. The user does not have to wait for the current conversion to be completed.

Allow any differential channel pair to be selected at one time. The A/D converter performs the specific differential conversion requested and stops.

Allow any differential channel pair to be scanned continuously. In other words, the user will specify the differential channel pair and the A/D converter will keep on scanning it continuously. The user can come in at any arbitrary time and immediately read the result of the last differential conversion. The user does not have to wait for the current conversion to be completed.

The A/D converter is supported by two memory mapped registers, the result register and the mode control register. When the device is reset, the control register is cleared and the A/D is powered down. The A/D result register has unknown data following reset.

A/D Control Register

A control register, Reg: ENAD, contains 3 bits for channel selection, 3 bits for prescaler selection, and 2 bits for mode selection. An A/D conversion is initiated by writing to the ENAD control register. The result of the conversion is available to the user from the A/D result register, Reg: ADRSLT.

Reg: ENAD

CHANNEL SELECT	MODE SELECT	PRESCALER SELECT
Bits 7, 6, 5	Bits 4,3	Bits 2, 1, 0

CHANNEL SELECT

This 3-bit field selects one of eight channels to be the $V_{IN}+$. The mode selection determines the $V_{IN}-$ input.

Single Ended mode:

Bit 7	Bit 6	Bit 5	Channel No.
0	0	0	0
0	0	1	1
0	1	0	2
0	1	1	3
1	0	0	4
1	0	1	5
1	1	0	6
1	1	1	7

Differential mode:

Bit 7	Bit 6	Bit 5	Channel Pairs (+ . −)
0	0	0	0, 1
0	0	1	1, 0
0	1	0	2, 3
0	1	1	3, 2
1	0	0	4, 5
1	0	1	5, 4
1	1	0	6, 7
1	1	1	7, 6

MODE SELECT

This 2-bit field is used to select the mode of operation (single conversion, continuous conversions, differential, single ended) as shown in the following table.

Bit 4	Bit 3	Mode
0	0	Single Ended mode, single conversion
0	1	Single Ended mode, continuous scan of a single channel into the result register
1	0	Differential mode, single conversion
1	1	Differential mode, continuous scan of a channel pair into the result register

A/D Converter (Continued)

PRESCALER SELECT

This 3-bit field is used to select one of the seven prescaler clocks for the A/D converter. The prescaler also allows the A/D clock inhibit power saving mode to be selected. The following table shows the various prescaler options.

Bit 2	Bit 1	Bit 0	Clock Select
0	0	0	Inhibit A/D clock
0	0	1	Divide by 1
0	1	0	Divide by 2
0	1	1	Divide by 4
1	0	0	Divide by 6
1	0	1	Divide by 12
1	1	0	Divide by 8
1	1	1	Divide by 16

ADC Operation

The A/D converter interface works as follows. Writing to the A/D control register ENAD initiates an A/D conversion unless the prescaler value is set to 0, in which case the ADC clock is stopped and the ADC is powered down. The conversion sequence starts at the beginning of the write to ENAD operation powering up the ADC. At the first falling edge of the converter clock following the write operation (not counting the falling edge if it occurs at the same time as the write operation ends), the sample signal turns on for two clock cycles. The ADC is selected in the middle of the sample period. If the ADC is in single conversion mode, the conversion complete signal from the ADC will generate a power down for the A/D converter. If the ADC is in continuous mode, the conversion complete signal will restart the conversion sequence by deselecting the ADC for one converter clock cycle before starting the next sample. The ADC 8-bit result is loaded into the A/D result register (ADRSLT) except during LOAD clock high, which prevents transient data (resulting from the ADC writing a new result over an old one) being read from ADRSLT.

Inadvertant changes to the ENAD register during conversion are prevented by the control logic of the A/D. Any attempt to write any bit of the ENAD Register except ADBSY, while ADBSY is a one, is ignored. ADBSY must be cleared either by completion of an A/D conversion or by the user before the prescaler, conversion mode or channel select values can be changed. After stopping the current conversion, the user can load different values for the prescaler, conversion mode or channel select and start a new conversion in one instruction.

It is important for the user to realize that, when used in differential mode, only the positive input to the A/D converter is sampled and held. The negative input is constantly connected and should be held stable for the duration of the conversion. Failure to maintain a stable negative input will result in incorrect conversion.

PRESCALER

The A/D Converter (ADC) contains a prescaler option which allows seven different clock selections. The A/D clock frequency is equal to CKI divided by the prescaler value. Note that the prescaler value must be chosen such that the A/D clock falls within the specified range. The maximum A/D frequency is 1.67 MHz. This equates to a 600 ns ADC clock cycle.

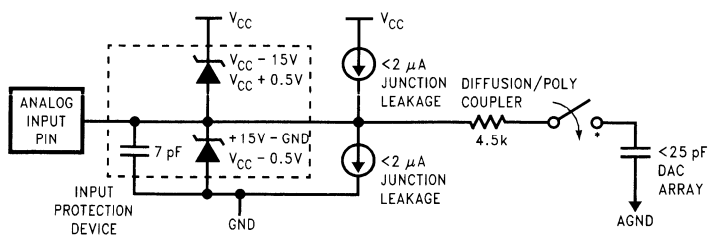
The A/D converter takes 12 ADC clock cycles to complete a conversion. Thus the minimum ADC conversion time for the device is 7.2 μs when a prescaler of 6 has been selected. These 12 ADC clock cycles necessary for a conversion consist of 1 cycle at the beginning for reset, 2 cycles for sampling, 8 cycles for converting, and 1 cycle for loading the result into the A/D result register (ADRSLT). This A/D result register is a read-only register. The device cannot write into ADRSLT.

The prescaler also allows an A/D clock inhibit option, which saves power by powering down the A/D when it is not in use.

Note: The A/D converter is also powered down when the device is in either the HALT or IDLE modes. If the ADC is running when the device enters the HALT or IDLE modes, the ADC will power down during the HALT or IDLE, and then will reinitialize the conversion when the device comes out of the HALT or IDLE modes.

Analog Input and Source Resistance Considerations

Figure 12 shows the A/D pin model in single ended mode. The differential mode has similiar A/D pin model. The leads to the analog inputs should be kept as short as possible. Both noise and digital clock coupling to an A/D input can cause conversion errors. The clock lead should be kept away from the analog input line to reduce coupling. The A/D channel input pins do not have any internal output driver circuitry connected to them because this circuitry would load the analog input signals due to output buffer leakage current.



TL/DD/9425–28

*The analog switch is closed only during the sample time.

FIGURE 12. A/D Pin Model (Single Ended Mode)

A/D Converter (Continued)

Source impedances greater than 1 kΩ on the analog input lines will adversely affect internal RC charging time during input sampling. As shown in *Figure 12*, the analog switch to the DAC array is closed only during the 2 A/D cycle sample time. Large source impedances on the analog inputs may result in the DAC array not being charged to the correct voltage levels, causing scale errors.

If large source resistance is necessary, the recommended solution is to slow down the A/D clock speed in proportion to the source resistance. The A/D converter may be operated at the maximum speed for R_S less than 1 kΩ. For R_S greater than 1 kΩ, A/D clock speed needs to be reduced. For example, with R_S = 2 kΩ, the A/D converter may be operated at half the maximum speed. A/D converter clock speed may be slowed down by either increasing the A/D prescaler divide-by or decreasing the CKI clock frequency. The A/D clock speed may be reduced to its minimum frequency of 100 kHz.

Interrupts

The device supports a vectored interrupt scheme. It supports a total of ten interrupt sources. The following table lists all the possible interrupt sources, their arbitration ranking and the memory locations reserved for the interrupt vector for each source.

Two bytes of program memory space are reserved for each interrupt source. All interrupt sources except the software interrupt are maskable. Each of the maskable interrupts have an Enable bit and a Pending bit. A maskable interrupt is active if its associated enable and pending bits are set. If GIE = 1 and an interrupt is active, then the processor will be interrupted as soon as it is ready to start executing an instruction except if the above conditions happen during the Software Trap service routine. This exception is described in the Software Trap sub-section.

The interruption process is accomplished with the INTR instruction (opcode 00), which is jammed inside the Instruction Register and replaces the opcode about to be executed. The following steps are performed for every interrupt:

1. The GIE (Global Interrupt Enable) bit is reset.
2. The address of the instruction about to be executed is pushed into the stack.
3. The PC (Program Counter) branches to address 00FF. This procedure takes 7 t_C cycles to execute.

At this time, since GIE = 0, other maskable interrupts are disabled. The user is now free to do whatever context switching is required by saving the context of the machine in the stack with PUSH instructions. The user would then program a VIS (Vector Interrupt Select) instruction in order to branch to the interrupt service routine of the highest priority interrupt enabled and pending at the time of the VIS. Note that this is not necessarily the interrupt that caused the branch to address location 00FF Hex prior to the context switching.

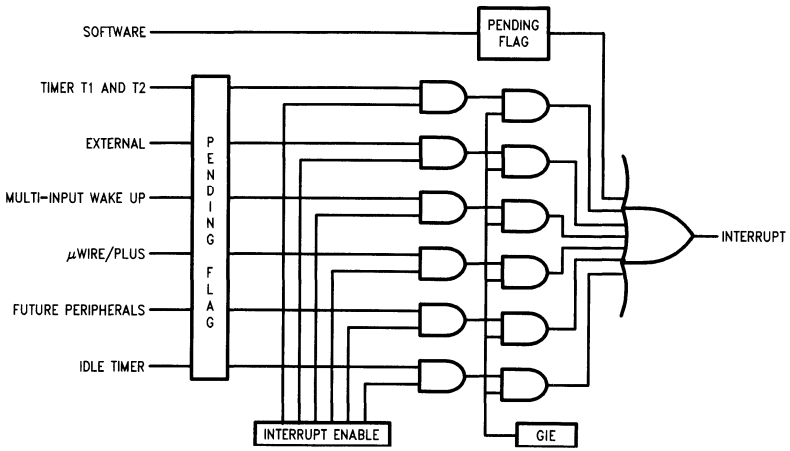
Thus, if an interrupt with a higher rank than the one which caused the interruption becomes active before the decision of which interrupt to service is made by the VIS, then the interrupt with the higher rank will override any lower ones and will be acknowledged. The lower priority interrupt(s) are still pending, however, and will cause another interrupt immediately following the completion of the interrupt service routine associated with the higher priority interrupt just serviced. This lower priority interrupt will occur immediately following the RETI (Return from Interrupt) instruction at the end of the interrupt service routine just completed.

Inside the interrupt service routine, the associated pending bit has to be cleared by software. The RETI (Return from Interrupt) instruction at the end of the interrupt service routine will set the GIE (Global Interrupt Enable) bit, allowing the processor to be interrupted again if another interrupt is active and pending.

Arbitration Ranking	Source	Description	Vector Address Hi-Low Byte
(1) Highest	Software	INTR Instruction	0yFE–0yFF
	Reserved	for Future Use	0yFC–0yFD
(2)	External	Pin G0 Edge	0yFA–0yFB
(3)	Timer T0	Underflow	0yF8–0yF9
(4)	Timer T1	T1A/Underflow	0yF6–0yF7
(5)	Timer T1	T1B	0yF4–0yF5
(6)	MICROWIRE/PLUS	BUSY Goes Low	0yF2–0yF3
	Reserved	for Future Use	0yF0–0yF1
	Reserved	for UART	0yEE–0yEF
	Reserved	for UART	0yEC–0yED
(7)	Timer T2	T2A/Underflow	0yEA–0yEB
(8)	Timer T2	T2B	0yE8–0yE9
	Reserved	for Future Use	0yE6–0yE7
	Reserved	for Future Use	0yE4–0yE5
(9)	Port L/Wakeup	Port L Edge	0yE2–0yE3
(10) Lowest	Default	VIS Instr. Execution without Any Interrupts	0yE0–0yE1

y is VIS page, y ≠ 0

Interrupts (Continued)



TL/DD/9425–18

FIGURE 13. Interrupt Block Diagram

The VIS instruction looks at all the active interrupts at the time it is executed and performs an indirect jump to the beginning of the service routine of the one with the highest rank.

The addresses of the different interrupt service routines, called vectors, are chosen by the user and stored in ROM in a table starting at 01E0 (assuming that VIS is located between 00FF and 01DF). The vectors are 15-bit wide and therefore occupy 2 ROM locations.

VIS and the vector table must be located in the same 256-byte block (0y00 to 0yFF) except if VIS is located at the last address of a block. In this case, the table must be in the next block. The vector table cannot be inserted in the first 256-byte block.

The vector of the maskable interrupt with the lowest rank is located at 0yE0 (Hi-Order byte) and 0yE1 (Lo-Order byte) and so forth in increasing rank number. The vector of the maskable interrupt with the highest rank is located at 0yFA (Hi-Order byte) and 0yFB (Lo-Order byte).

The Software Trap has the highest rank and its vector is located at 0yFE and 0yFF.

If, by accident, a VIS gets executed and no interrupt is active, then the PC (Program Counter) will branch to a vector located at 0yE0–0yE1.

WARNING:

A Default VIS interrupt handler routine must be present. As a minimum, this handler should confirm that the GIE bit is cleared (this indicates that the interrupt sequence has been taken), take care of any required housekeeping, restore context and return. Some sort of Warm Restart procedure should be implemented. These events can occur without any error on the part of the system designer or programmer.

Note: There is always the possibility of an interrupt occurring during an instruction which is attempting to reset the GIE bit or any other interrupt enable bit. If this occurs when a single cycle instruction is being used to reset the interrupt enable bit, the interrupt enable bit will be reset but an interrupt may still occur. This is because interrupt processing is started at the same time as the interrupt bit is being reset. To avoid this scenario, the user should always use a two, three, or four cycle instruction to reset interrupt enable bits.

Figure 13 shows the Interrupt block diagram.

SOFTWARE TRAP

The Software Trap (ST) is a special kind of non-maskable interrupt which occurs when the INTR instruction (used to acknowledge interrupts) is fetched from ROM and placed inside the instruction register. This may happen when the PC is pointing beyond the available ROM address space or when the stack is over-popped.

When an ST occurs, the user can re-initialize the stack pointer and do a recovery procedure (similar to RESET, but not necessarily containing all of the same initialization procedures) before restarting.

The occurrence of an ST is latched into the ST pending bit. The GIE bit is not affected and the ST pending bit **(not accessible by the user)** is used to inhibit other interrupts and to direct the program to the ST service routine with the VIS instruction. The RPND instruction is used to clear the software interrupt pending bit. This bit is also cleared on reset.

The ST has the highest rank among all interrupts.

Nothing (except another ST) can interrupt an ST being serviced.

WATCHDOG

The device contains a WATCHDOG and clock monitor. The WATCHDOG is designed to detect the user program getting stuck in infinite loops resulting in loss of program control or "runaway" programs. The Clock Monitor is used to detect the absence of a clock or a very slow clock below a specified rate on the CKI pin.

The WATCHDOG consists of two independent logic blocks: WD UPPER and WD LOWER. WD UPPER establishes the upper limit on the service window and WD LOWER defines the lower limit of the service window.

Servicing the WATCHDOG consists of writing a specific value to a WATCHDOG Service Register named WDSVR which is memory mapped in the RAM. This value is composed of three fields, consisting of a 2-bit Window Select, a 5-bit Key Data field, and the 1-bit Clock Monitor Select field. Table I shows the WDSVR register.

The lower limit of the service window is fixed at 2048 instruction cycles. Bits 7 and 6 of the WDSVR register allow the user to pick an upper limit of the service window.

Table II shows the four possible combinations of lower and upper limits for the WATCHDOG service window. This flexibility in choosing the WATCHDOG service window prevents any undue burden on the user software.

Bits 5, 4, 3, 2 and 1 of the WDSVR register represent the 5-bit Key Data field. The key data is fixed at 01100. Bit 0 of the WDSVR Register is the Clock Monitor Select bit.

TABLE I. WATCHDOG Service Register

Window Select		Key Data					Clock Monitor
X	X	0	1	1	0	0	Y
7	6	5	4	3	2	1	0

TABLE II. WATCHDOG Service Window Select

WDSVR Bit 7	WDSVR Bit 6	Service Window (Lower-Upper Limits)
0	0	2k-8k t_C Cycles
0	1	2k-16k t_C Cycles
1	0	2k-32k t_C Cycles
1	1	2k-64k t_C Cycles

Clock Monitor

The Clock Monitor aboard the device can be selected or deselected under program control. The Clock Monitor is guaranteed not to reject the clock if the instruction cycle clock ($1/t_C$) is greater or equal to 10 kHz. This equates to a clock input rate on CKI of greater or equal to 100 kHz.

WATCHDOG Operation

The WATCHDOG and Clock Monitor are disabled during reset. The device comes out of reset with the WATCHDOG armed, the WATCHDOG Window Select (bits 6, 7 of the WDSVR Register) set, and the Clock Monitor bit (bit 0 of the WDSVR Register) enabled. Thus, a Clock Monitor error will occur after coming out of reset, if the instruction cycle clock frequency has not reached a minimum specified value, including the case where the oscillator fails to start.

The WDSVR register can be written to only once after reset and the key data (bits 5 through 1 of the WDSVR Register) must match to be a valid write. This write to the WDSVR register involves two irrevocable choices: (i) the selection of the WATCHDOG service window (ii) enabling or disabling of the Clock Monitor. Hence, the first write to WDSVR Register involves selecting or deselecting the Clock Monitor, select the WATCHDOG service window and match the WATCHDOG key data. Subsequent writes to the WDSVR register will compare the value being written by the user to the WATCHDOG service window value and the key data (bits 7 through 1) in the WDSVR Register. Table III shows the sequence of events that can occur.

The user must service the WATCHDOG at least once before the upper limit of the service window expires. The WATCHDOG may not be serviced more than once in every lower limit of the service window. The user may service the WATCHDOG as many times as wished in the time period between the lower and upper limits of the service window. The first write to the WDSVR Register is also counted as a WATCHDOG service.

The WATCHDOG has an output pin associated with it. This is the WDOUT pin, on pin 1 of the port G. WDOUT is active low. The WDOUT pin is in the high impedance state in the inactive state. Upon triggering the WATCHDOG, the logic will pull the WDOUT (G1) pin low for an additional 16 t_C–32 t_C cycles after the signal level on WDOUT pin goes below the lower Schmitt trigger threshold. After this delay, the device will stop forcing the WDOUT output low.

The WATCHDOG service window will restart when the WDOUT pin goes high. It is recommended that the user tie the WDOUT pin back to V_{CC} through a resistor in order to pull WDOUT high.

A WATCHDOG service while the WDOUT signal is active will be ignored. The state of the WDOUT pin is not guaranteed on reset, but if it powers up low then the WATCHDOG will time out and WDOUT will enter high impedance state.

The Clock Monitor forces the G1 pin low upon detecting a clock frequency error. The Clock Monitor error will continue until the clock frequency has reached the minimum specified value, after which the G1 output will enter the high impedance TRI-STATE mode following 16 t_C–32 t_C clock cycles. The Clock Monitor generates a continual Clock Monitor error if the oscillator fails to start, or fails to reach the minimum specified frequency. The specification for the Clock Monitor is as follows:

$1/t_C > 10$ kHz—No clock rejection.

$1/t_C < 10$ Hz—Guaranteed clock rejection.

TABLE III. WATCHDOG Service Actions

Key Data	Window Data	Clock Monitor	Action
Match	Match	Match	Valid Service: Restart Service Window
Don't Care	Mismatch	Don't Care	Error: Generate WATCHDOG Output
Mismatch	Don't Care	Don't Care	Error: Generate WATCHDOG Output
Don't Care	Don't Care	Mismatch	Error: Generate WATCHDOG Output

WATCHDOG Operation (Continued)

WATCHDOG AND CLOCK MONITOR SUMMARY

The following salient points regarding the WATCHDOG and CLOCK MONITOR should be noted:

- Both the WATCHDOG and Clock Monitor detector circuits are inhibited during RESET.
- Following RESET, the WATCHDOG and CLOCK MONITOR are both enabled, with the WATCHDOG having the maximum service window selected.
- The WATCHDOG service window and Clock Monitor enable/disable option can only be changed once, during the initial WATCHDOG service following RESET.
- The initial WATCHDOG service must match the key data value in the WATCHDOG Service register WDSVR in order to avoid a WATCHDOG error.
- Subsequent WATCHDOG services must match all three data fields in WDSVR in order to avoid WATCHDOG errors.
- The correct key data value cannot be read from the WATCHDOG Service register WDSVR. Any attempt to read this key data value of 01100 from WDSVR will read as key data value of all 0's.
- The WATCHDOG detector circuit is inhibited during both the HALT and IDLE modes.
- The Clock Monitor detector circuit is active during both the HALT and IDLE modes. Consequently, the device inadvertently entering the HALT mode will be detected as a Clock Monitor error (provided that the Clock Monitor enable option has been selected by the program).
- With the single-pin R/C oscillator mask option selected and the CLKDLY bit reset, the WATCHDOG service window will resume following HALT mode from where it left off before entering the HALT mode.
- With the crystal oscillator mask option selected, or with the single-pin R/C oscillator mask option selected and the CLKDLY bit set, the WATCHDOG service window will be set to its selected value from WDSVR following HALT. Consequently, the WATCHDOG should not be serviced for at least 2048 instruction cycles following HALT, but must be serviced within the selected window to avoid a WATCHDOG error.
- The IDLE timer T0 is not initialized with RESET.
- The user can sync in to the IDLE counter cycle with an IDLE counter (T0) interrupt or by monitoring the T0PND flag. The T0PND flag is set whenever the thirteenth bit of the IDLE counter toggles (every 4096 instruction cycles). The user is responsible for resetting the T0PND flag.
- A hardware WATCHDOG service occurs just as the device exits the IDLE mode. Consequently, the WATCHDOG should not be serviced for at least 2048 instruction cycles following IDLE, but must be serviced within the selected window to avoid a WATCHDOG error.
- Following RESET, the initial WATCHDOG service (where the service window and the CLOCK MONITOR enable/disable must be selected) may be programmed anywhere within the maximum service window (65,536 instruction cycles) initialized by RESET. Note that this initial WATCHDOG service may be programmed within the initial 2048 instruction cycles without causing a WATCHDOG error.

Detection of Illegal Conditions

The device can detect various illegal conditions resulting from coding errors, transient noise, power supply voltage drops, runaway programs, etc.

Reading of undefined ROM gets zeros. The opcode for software interrupt is zero. If the program fetches instructions from undefined ROM, this will force a software interrupt, thus signaling that an illegal condition has occurred.

The subroutine stack grows down for each call (jump to subroutine), interrupt, or PUSH, and grows up for each return or POP. The stack pointer is initialized to RAM location 06F Hex during reset. Consequently, if there are more returns than calls, the stack pointer will point to addresses 070 and 071 Hex (which are undefined RAM). Undefined RAM from addresses 070 to 07F Hex is read as all 1's, which in turn will cause the program to return to address 7FFF Hex. This is an undefined ROM location and the instruction fetched (all 0's) from this location will generate a software interrupt signaling an illegal condition.

Thus, the chip can detect the following illegal conditions:

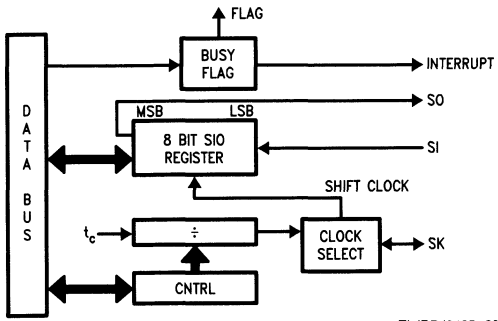
a. Executing from undefined ROM

b. Over "POP"ing the stack by having more returns than calls.

When the software interrupt occurs, the user can re-initialize the stack pointer and do a recovery procedure before restarting (this recovery program is probably similar to that following reset, but might not contain the same program initialization procedures).

MICROWIRE/PLUS

MICROWIRE/PLUS is a serial synchronous communications interface. The MICROWIRE/PLUS capability enables the device to interface with any of National Semiconductor's MICROWIRE peripherals (i.e. A/D converters, display drivers, E²PROMs etc.) and with other microcontrollers which support the MICROWIRE interface. It consists of an 8-bit serial shift register (SIO) with serial data input (SI), serial data output (SO) and serial shift clock (SK). *Figure 14* shows a block diagram of the MICROWIRE/PLUS logic.



TL/DD/9425–20

FIGURE 14. MICROWIRE/PLUS Block Diagram

The shift clock can be selected from either an internal source or an external source. Operating the MICROWIRE/PLUS arrangement with the internal clock source is called the Master mode of operation. Similarly, operating the MICROWIRE/PLUS arrangement with an external shift clock is called the Slave mode of operation.

The CNTRL register is used to configure and control the MICROWIRE/PLUS mode. To use the MICROWIRE/PLUS, the MSEL bit in the CNTRL register is set to one. In the master mode the SK clock rate is selected by the two bits, SL0 and SL1, in the CNTRL register. TABLE IV details the different clock rates that may be selected.

TABLE IV. MICROWIRE/PLUS Master Mode Clock Selection

SL1	SL0	SK
0	0	$2 \times t_c$
0	1	$4 \times t_c$
1	x	$8 \times t_c$

Where t_c is the instruction cycle clock

MICROWIRE/PLUS OPERATION

Setting the BUSY bit in the PSW register causes the MICROWIRE/PLUS to start shifting the data. It gets reset when eight data bits have been shifted. The user may reset the BUSY bit by software to allow less than 8 bits to shift. If enabled, an interrupt is generated when eight data bits have been shifted. The device may enter the MICROWIRE/PLUS mode either as a Master or as a Slave. *Figure 15* shows how two COP888CF microcontrollers and several peripherals may be interconnected using the MICROWIRE/PLUS arrangements.

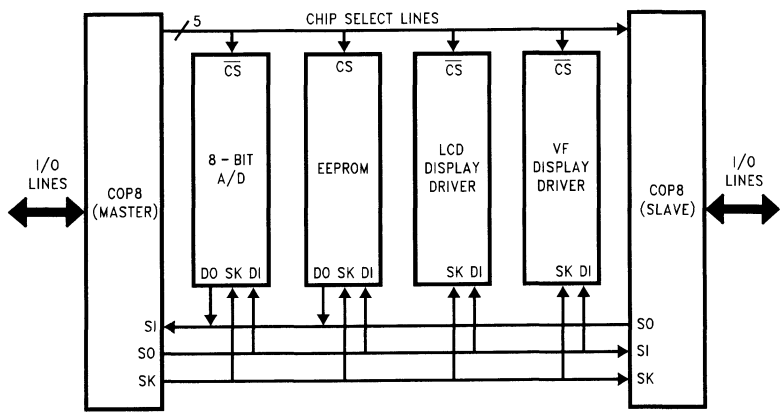
Warning:

The SIO register should only be loaded when the SK clock is low. Loading the SIO register while the SK clock is high will result in undefined data in the SIO register. SK clock is normally low when not shifting.

Setting the BUSY flag when the input SK clock is high in the MICROWIRE/PLUS slave mode may cause the current SK clock for the SIO shift register to be narrow. For safety, the BUSY flag should only be set when the input SK clock is low.

MICROWIRE/PLUS Master Mode Operation

In the MICROWIRE/PLUS Master mode of operation the shift clock (SK) is generated internally. The MICROWIRE Master always initiates all data exchanges. The MSEL bit in the CNTRL register must be set to enable the SO and SK functions onto the G Port. The SO and SK pins must also be selected as outputs by setting appropriate bits in the Port G configuration register. Table V summarizes the bit settings required for Master mode of operation.



TL/DD/9425–21

FIGURE 15. MICROWIRE/PLUS Application

MICROWIRE/PLUS (Continued)

MICROWIRE/PLUS Slave Mode Operation

In the MICROWIRE/PLUS Slave mode of operation the SK clock is generated by an external source. Setting the MSEL bit in the CNTRL register enables the SO and SK functions onto the G Port. The SK pin must be selected as an input and the SO pin is selected as an output pin by setting and resetting the appropriate bit in the Port G configuration register. Table V summarizes the settings required to enter the Slave mode of operation.

The user must set the BUSY flag immediately upon entering the Slave mode. This will ensure that all data bits sent by the Master will be shifted properly. After eight clock pulses the BUSY flag will be cleared and the sequence may be repeated.

Alternate SK Phase Operation

The device allows either the normal SK clock or an alternate phase SK clock to shift data in and out of the SIO register. In both the modes the SK is normally low. In the normal mode data is shifted in on the rising edge of the SK clock and the data is shifted out on the falling edge of the SK clock. The SIO register is shifted on each falling edge of the SK clock in the normal mode. In the alternate SK phase mode the SIO register is shifted on the rising edge of the SK clock.

A control flag, SKSEL, allows either the normal SK clock or the alternate SK clock to be selected. Resetting SKSEL causes the MICROWIRE/PLUS logic to be clocked from the normal SK signal. Setting the SKSEL flag selects the alternate SK clock. The SKSEL is mapped into the G6 configuration bit. The SKSEL flag will power up in the reset condition, selecting the normal SK signal.

TABLE V

This table assumes that the control flag MSEL is set.

G4 (SO) Config. Bit	G5 (SK) Config. Bit	G4 Fun.	G5 Fun.	Operation
1	1	SO	Int. SK	MICROWIRE/PLUS Master
0	1	TRI-STATE	Int. SK	MICROWIRE/PLUS Master
1	0	SO	Ext. SK	MICROWIRE/PLUS Slave
0	0	TRI-STATE	Ext. SK	MICROWIRE/PLUS Slave

Memory Map

All RAM, ports and registers (except A and PC) are mapped into data memory address space

Address	Contents
00 to 6F	On-Chip RAM bytes
70 to BF	Unused RAM Address Space
C0	Timer T2 Lower Byte
C1	Timer T2 Upper Byte
C2	Timer T2 Autoload Register T2RA Lower Byte
C3	Timer T2 Autoload Register T2RA Upper Byte
C4	Timer T2 Autoload Register T2RB Lower Byte
C5	Timer T2 Autoload Register T2RB Upper Byte
C6	Timer T2 Control Register
C7	WATCHDOG Service Register (Reg:WDSVR)
C8	MIWU Edge Select Register (Reg:WKEDG)
C9	MIWU Enable Register (Reg:WKEN)
CA	MIWU Pending Register (Reg:WKPND)
CB	A/D Converter Control Register (Reg:ENAD)
CC	A/D Converter Result Register (Reg: ADRSLT)
CD to CF	Reserved
D0	Port L Data Register
D1	Port L Configuration Register
D2	Port L Input Pins (Read Only)
D3	Reserved for Port L
D4	Port G Data Register
D5	Port G Configuration Register
D6	Port G Input Pins (Read Only)
D7	Port I Input Pins (Read Only)
D8	Port C Data Register
D9	Port C Configuration Register
DA	Port C Input Pins (Read Only)
DB	Reserved for Port C
DC	Port D Data Register
DD to DF	Reserved for Port D
E0 to E5	Reserved
E6	Timer T1 Autoload Register T1RB Lower Byte
E7	Timer T1 Autoload Register T1RB Upper Byte
E8	ICNTRL Register
E9	MICROWIRE Shift Register
EA	Timer T1 Lower Byte
EB	Timer T1 Upper Byte
EC	Timer T1 Autoload Register T1RA Lower Byte
ED	Timer T1 Autoload Register T1RA Upper Byte
EE	CNTRL Control Register
EF	PSW Register
F0 to FB	On-Chip RAM Mapped as Registers
FC	X Register
FD	SP Register
FE	B Register
FF	Reserved

Reading memory locations 70-7F Hex will return all ones. Reading other unused memory locations will return undefined data.

Addressing Modes

The device has ten addressing modes, six for operand addressing and four for transfer of control.

OPERAND ADDRESSING MODES

Register Indirect

This is the "normal" addressing mode. The operand is the data memory addressed by the B pointer or X pointer.

Register Indirect (with auto post increment or decrement of pointer)

This addressing mode is used with the LD and X instructions. The operand is the data memory addressed by the B pointer or X pointer. This is a register indirect mode that automatically post increments or decrements the B or X register after executing the instruction.

Direct

The instruction contains an 8-bit address field that directly points to the data memory for the operand.

Immediate

The instruction contains an 8-bit immediate field as the operand.

Short Immediate

This addressing mode is used with the Load B Immediate instruction. The instruction contains a 4-bit immediate field as the operand.

Indirect

This addressing mode is used with the LAID instruction. The contents of the accumulator are used as a partial address (lower 8 bits of PC) for accessing a data operand from the program memory.

TRANSFER OF CONTROL ADDRESSING MODES

Relative

This mode is used for the JP instruction, with the instruction field being added to the program counter to get the new program location. JP has a range from −31 to +32 to allow a 1-byte relative jump (JP + 1 is implemented by a NOP instruction). There are no "pages" when using JP, since all 15 bits of PC are used.

Absolute

This mode is used with the JMP and JSR instructions, with the instruction field of 12 bits replacing the lower 12 bits of the program counter (PC). This allows jumping to any location in the current 4k program memory segment.

Absolute Long

This mode is used with the JMPL and JSRL instructions, with the instruction field of 15 bits replacing the entire 15 bits of the program counter (PC). This allows jumping to any location in the current 4k program memory space.

Indirect

This mode is used with the JID instruction. The contents of the accumulator are used as a partial address (lower 8 bits of PC) for accessing a location in the program memory. The contents of this program memory location serve as a partial address (lower 8 bits of PC) for the jump to the next instruction.

Note: The VIS is a special case of the Indirect Transfer of Control addressing mode, where the double byte vector associated with the interrupt is transferred from adjacent addresses in the program memory into the program counter (PC) in order to jump to the associated interrupt service routine.

Instruction Set

Register and Symbol Definition

Registers	
A	8-Bit Accumulator Register
B	8-Bit Address Register
X	8-Bit Address Register
SP	8-Bit Stack Pointer Register
PC	15-Bit Program Counter Register
PU	Upper 7 Bits of PC
PL	Lower 8 Bits of PC
C	1 Bit of PSW Register for Carry
HC	1 Bit of PSW Register for Half Carry
GIE	1 Bit of PSW Register for Global Interrupt Enable
VU	Interrupt Vector Upper Byte
VL	Interrupt Vector Lower Byte

Symbols	
[B]	Memory Indirectly Addressed by B Register
[X]	Memory Indirectly Addressed by X Register
MD	Direct Addressed Memory
Mem	Direct Addressed Memory or [B]
Meml	Direct Addressed Memory or [B] or Immediate Data
Imm	8-Bit Immediate Data
Reg	Register Memory: Addresses F0 to FF (Includes B, X and SP)
Bit	Bit Number (0 to 7)
←	Loaded with
↔	Exchanged with

5

Instruction Set (Continued)

INSTRUCTION SET

ADD	A,Meml	ADD	A ← A + Meml
ADC	A,Meml	ADD with Carry	A ← A + Meml + C, C ← Carry HC ← Half Carry
SUBC	A,Meml	Subtract with Carry	A ← A $\overline{\text{Meml}}$ + C, C ← Carry HC ← Half Carry
AND	A,Meml	Logical AND	A ← A and Meml
ANDSZ	A,Imm	Logical AND Immed., Skip if Zero	Skip next if (A and Imm) = 0
OR	A,Meml	Logical OR	A ← A or Meml
XOR	A,Meml	Logical EXclusive OR	A ← A xor Meml
IFEQ	MD,Imm	IF EQual	Compare MD and Imm, Do next if MD = Imm
IFEQ	A,Meml	IF EQual	Compare A and Meml, Do next if A = Meml
IFNE	A,Meml	IF Not Equal	Compare A and Meml, Do next if A ≠ Meml
IFGT	A,Meml	IF Greater Than	Compare A and Meml, Do next if A > Meml
IFBNE	#	If B Not Equal	Do next if lower 4 bits of B ≠ Imm
DRSZ	Reg	Decrement Reg., Skip if Zero	Reg ← Reg − 1, Skip if Reg = 0
SBIT	#,Mem	Set BIT	1 to bit, Mem (bit = 0 to 7 immediate)
RBIT	#,Mem	Reset BIT	0 to bit, Mem
IFBIT	#,Mem	IF BIT	If bit in A or Mem is true do next instruction
RPND		Reset PeNDing Flag	Reset Software Interrupt Pending Flag
X	A,Mem	EXchange A with Memory	A ⟷ Mem
X	A,[X]	EXchange A with Memory [X]	A ⟷ [X]
LD	A,Meml	LoaD A with Memory	A ← Meml
LD	A,[X]	LoaD A with Memory [X]	A ← [X]
LD	B,Imm	LoaD B with Immed.	B ← Imm
LD	Mem,Imm	LoaD Memory Immed	Mem ← Imm
LD	Reg,Imm	LoaD Register Memory Immed.	Reg ← Imm
X	A, [B ±]	EXchange A with Memory [B]	A ⟷ [B], (B ← B ±1)
X	A, [X ±]	EXchange A with Memory [X]	A ⟷ [X], (X ← ±1)
LD	A, [B±]	LoaD A with Memory [B]	A ← [B], (B ← B ±1)
LD	A, [X±]	LoaD A with Memory [X]	A ← [X], (X ← X±1)
LD	[B±],Imm	LoaD Memory [B] Immed.	[B] ← Imm, (B ← B±1)
CLR	A	CLeaR A	A ← 0
INC	A	INCrement A	A ← A + 1
DEC	A	DECrementA	A ← A − 1
LAID		Load A InDirect from ROM	A ← ROM (PU,A)
DCOR	A	Decimal CORrect A	A ← BCD correction of A (follows ADC, SUBC)
RRC	A	Rotate A Right thru C	C → A7 → . . . → A0 → C
RLC	A	Rotate A Left thru C	C ← A7 ← . . . ← A0 ← C
SWAP	A	SWAP nibbles of A	A7 . . . A4 ⟷ A3 . . . A0
SC		Set C	C ← 1, HC ← 1
RC		Reset C	C ← 0, HC ← 0
IFC		IF C	IF C is true, do next instruction
IFNC		IF Not C	If C is not true, do next instruction
POP	A	POP the stack into A	SP ← SP + 1, A ← [SP]
PUSH	A	PUSH A onto the stack	[SP] ← A, SP ← SP − 1
VIS		Vector to Interrupt Service Routine	PU ← [VU], PL ← [VL]
JMPL	Addr.	Jump absolute Long	PC ← ii (ii = 15 bits, 0 to 32k)
JMP	Addr.	Jump absolute	PC9 . . . 0 ← i (i = 12 bits)
JP	Disp.	Jump relative short	PC ← PC + r (r is −31 to +32, except 1)
JSRL	Addr.	Jump SubRoutine Long	[SP] ← PL, [SP−1] ← PU,SP−2, PC ← ii
JSR	Addr	Jump SubRoutine	[SP] ← PL, [SP−1] ← PU,SP−2, PC9 . . . 0 ← i
JID		Jump InDirect	PL ← ROM (PU,A)
RET		RETurn from subroutine	SP + 2, PL ← [SP], PU ← [SP−1]
RETSK		RETurn and SKip	SP + 2, PL ← [SP],PU ← [SP−1]
RETI		RETurn from Interrupt	SP + 2, PL ← [SP],PU ← [SP−1],GIE ← 1
INTR		Generate an Interrupt	[SP] ← PL, [SP−1] ← PU, SP−2, PC ← 0FF
NOP		No OPeration	PC ← PC + 1

Instruction Execution Time

Most instructions are single byte (with immediate addressing mode instructions taking two bytes).

Most single byte instructions take one cycle time to execute.

Skipped instructions require x number of cycles to be skipped, where x equals the number of bytes in the skipped instruction opcode.

See the BYTES and CYCLES per INSTRUCTION table for details.

Bytes and Cycles per Instruction

The following table shows the number of bytes and cycles for each instruction in the format of byte/cycle.

Arithmetic and Logic Instructions

	[B]	Direct	Immed.
ADD	1/1	3/4	2/2
ADC	1/1	3/4	2/2
SUBC	1/1	3/4	2/2
AND	1/1	3/4	2/2
OR	1/1	3/4	2/2
XOR	1/1	3/4	2/2
IFEQ	1/1	3/4	2/2
IFNE	1/1	3/4	2/2
IFGT	1/1	3/4	2/2
IFBNE	1/1		
DRSZ		1/3	
SBIT	1/1	3/4	
RBIT	1/1	3/4	
IFBIT	1/1	3/4	

RPND	1/1

Instructions Using A & C

CLRA	1/1
INCA	1/1
DECA	1/1
LAID	1/3
DCOR	1/1
RRCA	1/1
RLCA	1/1
SWAPA	1/1
SC	1/1
RC	1/1
IFC	1/1
IFNC	1/1
PUSHA	1/3
POPA	1/3
ANDSZ	2/2

Transfer of Control Instructions

JMPL	3/4
JMP	2/3
JP	1/3
JSRL	3/5
JSR	2/5
JID	1/3
VIS	1/5
RET	1/5
RETSK	1/5
RETI	1/5
INTR	1/7
NOP	1/1

Memory Transfer Instructions

	Register Indirect		Direct	Immed.	Register Indirect Auto Incr. & Decr.		
	[B]	[X]			[B+, B−]	[X+, X−]	
X A,*	1/1	1/3	2/3		1/2	1/3	
LD A,*	1/1	1/3	2/3	2/2	1/2	1/3	
LD B, Imm				1/1			(IF B < 16)
LD B, Imm				2/2			(IF B > 15)
LD Mem, Imm	2/2		3/3		2/2		
LD Reg, Imm			2/3				
IFEQ MD, Imm			3/3				

* = > Memory location addressed by B or X or directly.

5

Opcode Table

Upper Nibble Along X-Axis

Lower Nibble Along Y-Axis

F	E	D	C	B	A	9	8	
JP −15	JP −31	LD 0F0, # i	DRSZ 0F0	RRCA	RC	ADC A, #i	ADC A,[B]	0
JP −14	JP −30	LD 0F1, # i	DRSZ 0F1	*	SC	SUBC A, #i	SUB A,[B]	1
JP −13	JP −29	LD 0F2, # i	DRSZ 0F2	X A, [X+]	X A,[B+]	IFEQ A,#i	IFEQ A,[B]	2
JP −12	JP −28	LD 0F3, # i	DRSZ 0F3	X A, [X−]	X A,[B−]	IFGT A,#i	IFGT A,[B]	3
JP −11	JP −27	LD 0F4, # i	DRSZ 0F4	VIS	LAID	ADD A,#i	ADD A,[B]	4
JP −10	JP −26	LD 0F5, # i	DRSZ 0F5	RPND	JID	AND A,#i	AND A,[B]	5
JP −9	JP −25	LD 0F6, # i	DRSZ 0F6	X A,[X]	X A,[B]	XOR A,#i	XOR A,[B]	6
JP −8	JP −24	LD 0F7, # i	DRSZ 0F7	*	*	OR A,#i	OR A,[B]	7
JP −7	JP −23	LD 0F8, # i	DRSZ 0F8	NOP	RLCA	LD A,#i	IFC	8
JP −6	JP −22	LD 0F9, # i	DRSZ 0F9	IFNE A,[B]	IFEQ Md,#i	IFNE A,#i	IFNC	9
JP −5	JP −21	LD 0FA, # i	DRSZ 0FA	LD A,[X+]	LD A,[B+]	LD [B+],#i	INCA	A
JP −4	JP −20	LD 0FB, # i	DRSZ 0FB	LD A,[X−]	LD A,[B−]	LD [B−],#i	DECA	B
JP −3	JP −19	LD 0FC, # i	DRSZ 0FC	LD Md,#i	JMPL	X A,Md	POPA	C
JP −2	JP −18	LD 0FD, # i	DRSZ 0FD	DIR	JSRL	LD A,Md	RETSK	D
JP −1	JP −17	LD 0FE, # i	DRSZ 0FE	LD A,[X]	LD A,[B]	LD [B],#i	RET	E
JP −0	JP −16	LD 0FF, # i	DRSZ 0FF	*	*	LD B,#i	RETI	F

Opcode Table (Continued)

Upper Nibble Along X-Axis

Lower Nibble Along Y-Axis

7	6	5	4	3	2	1	0	
IFBIT 0,[B]	ANDSZ A, #i	LD B,#0F	IFBNE 0	JSR x000–x0FF	JMP x000–x0FF	JP +17	INTR	0
IFBIT 1,[B]	*	LD B,#0E	IFBNE 1	JSR x100–x1FF	JMP x100–x1FF	JP +18	JP + 2	1
IFBIT 2,[B]	*	LD B,#0D	IFBNE 2	JSR x200–x2FF	JMP x200–x2FF	JP +19	JP + 3	2
IFBIT 3,[B]	*	LD B,#0C	IFBNE 3	JSR x300–x3FF	JMP x300–x3FF	JP +20	JP + 4	3
IFBIT 4,[B]	CLRA	LD B,#0B	IFBNE 4	JSR x400–x4FF	JMP x400–x4FF	JP +21	JP + 5	4
IFBIT 5,[B]	SWAPA	LD B,#0A	IFBNE 5	JSR x500–x5FF	JMP x500–x5FF	JP +22	JP + 6	5
IFBIT 6,[B]	DCORA	LD B,#09	IFBNE 6	JSR x600–x6FF	JMP x600–x6FF	JP +23	JP + 7	6
IFBIT 7,[B]	PUSHA	LD B,#08	IFBNE 7	JSR x700–x7FF	JMP x700–x7FF	JP +24	JP + 8	7
SBIT 0,[B]	RBIT 0,[B]	LD B,#07	IFBNE 8	JSR x800–x8FF	JMP x800–x8FF	JP +25	JP + 9	8
SBIT 1,[B]	RBIT 1,[B]	LD B,#06	IFBNE 9	JSR x900–x9FF	JMP x900–x9FF	JP +26	JP + 10	9
SBIT 2,[B]	RBIT 2,[B]	LD B,#05	IFBNE 0A	JSR xA00–xAFF	JMP xA00–xAFF	JP +27	JP + 11	A
SBIT 3,[B]	RBIT 3,[B]	LD B,#04	IFBNE 0B	JSR xB00–xBFF	JMP xB00–xBFF	JP +28	JP + 12	B
SBIT 4,[B]	RBIT 4,[B]	LD B,#03	IFBNE 0C	JSR xC00–xCFF	JMP xC00–xCFF	JP +29	JP + 13	C
SBIT 5,[B]	RBIT 5,[B]	LD B,#02	IFBNE 0D	JSR xD00–xDFF	JMP xD00–xDFF	JP +30	JP + 14	D
SBIT 6,[B]	RBIT 6,[B]	LD B,#01	IFBNE 0E	JSR xE00–xEFF	JMP xE00–xEFF	JP +31	JP + 15	E
SBIT 7,[B]	RBIT 7,[B]	LD B,#00	IFBNE 0F	JSR xF00–xFFF	JMP xF00–xFFF	JP +32	JP + 16	F

Where,

i is the immediate data

Md is a directly addressed memory location

* is an unused opcode

Note: The opcode 60 Hex is also the opcode for IFBIT #i,A

Mask Options

The mask programmable options are shown below. The options are programmed at the same time as the ROM pattern submission.

```
OPTION 1: CLOCK CONFIGURATION
  = 1     Crystal Oscillator (CKI/lO)
              G7 (CKO) is clock generator
              output to crystal/resonator
              CKI is the clock input
  = 2     Single-pin RC controlled
              oscillator (CKI/lO)
              G7 is available as a HALT
              restart and/or general purpose
              input

OPTION 2: HALT
  = 1     Enable HALT mode
  = 2     Disable HALT mode

OPTION 3: BONDING
  = 1     44-Pin PLCC
  = 2     40-Pin DIP
  = 3     N/A
  = 4     28-Pin DIP
  = 5     28-Pin SO
```

Development Support

SUMMARY

- iceMASTER™: IM-COP8/400—Full feature in-circuit emulation for all COP8 products. A full set of COP8 Basic and Feature Family device and package specific probes are available.
- COP8 Debug Module: Moderate cost in-circuit emulation and development programming unit.
- COP8 Evaluation and Programming Unit: EPU-COP888GG—low cost In-circuit simulation and development programming unit.
- Assembler: COP8-DEV-IBMA. A DOS installable cross development Assembler, Linker, Librarian and Utility Software Development Tool Kit.
- C Compiler: COP8C. A DOS installable cross development Software Tool Kit.
- OTP/EPROM Programmer Support: Covering needs from engineering prototype, pilot production to full production environments.

iceMASTER (IM) IN-CIRCUIT EMULATION

The iceMASTER IM-COP8/400 is a full feature, PC based, in-circuit emulation tool developed and marketed by MetaLink Corporation to support the whole COP8 family of products. National is a resale vendor for these products.

See *Figure 19* for configuration.

The iceMASTER IM-COP8/400 with its device specific COP8 Probe provides a rich feature set for developing, testing and maintaining product:

- Real-time in-circuit emulation; full 2.4V–5.5V operation range, full DC-10 MHz clock. Chip options are programmable or jumper selectable.
- Direct connection to application board by package compatible socket or surface mount assembly.
- Full 32 kbyte of loadable programming space that overlays (replaces) the on-chip ROM or EPROM. On-chip RAM and I/O blocks are used directly or recreated on the probe as necessary.
- Full 4k frame synchronous trace memory. Address, instruction, and 8 unspecified, circuit connectable trace lines. Display can be HLL source (e.g., C source), assembly or mixed.
- A full 64k hardware configurable break, trace on, trace off control, and pass count increment events.
- Tool set integrated interactive symbolic debugger—supports both assembler (COFF) and C Compiler (.COD) linked object formats.
- Real time performance profiling analysis; selectable bucket definition.
- Watch windows, content updated automatically at each execution break.
- Instruction by instruction memory/register changes displayed on source window when in single step operation.
- Single base unit & debugger software reconfigurable to support the entire COP8 family; only the probe personality needs to change. Debugger software is processor customized, and reconfigured from a master model file.
- Processor specific symbolic display of registers and bit level assignments, configured from master model file.
- Halt/Idle mode notification.
- On-line HELP customized to specific processor using master model file.
- Includes a copy of COP8-DEV-IBMA assembler and linker SDK.

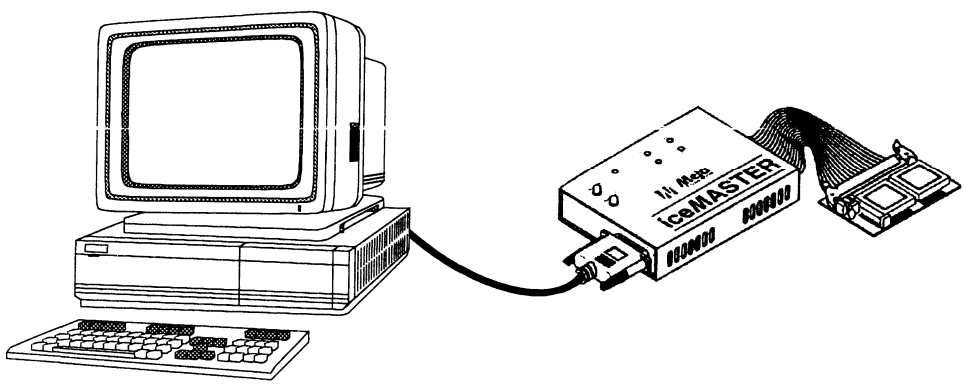


FIGURE 19. COP8 iceMASTER Environment

TL/DD/9425–38

Development Support (Continued)

IM Order Information

Base Unit	
IM-COP8/400-1	iceMASTER base unit, 110V power supply
IM-COP8/400-2	iceMASTER base unit, 220V power supply
iceMASTER Probe	
MHW-884CF28DWPC	28 DIP
MHW-888CF40DWPC	40 DIP
MHW-888CF44PWPC	44 PLCC
Adapter for SO Package	
MHW-SOIC 28	28 SO

iceMASTER DEBUG MODULE (DM)

The iceMASTER Debug Module is a PC based, combination in-circuit emulation tool and COP8 based OTP/EPROM programming tool developed and marketed by MetaLink Corporation to support the whole COP8 family of products. National is a resale vendor for these products.

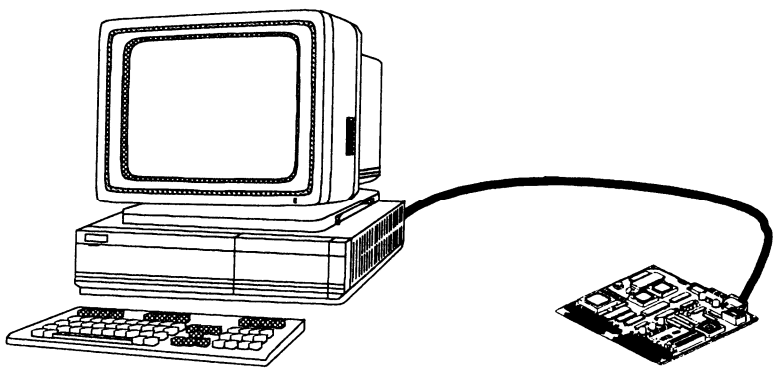
See *Figure 20* for configuration.

The iceMASTER Debug Module is a moderate cost development tool. It has the capability of in-circuit emulation for a specific COP8 microcontroller and in addition serves as a programming tool for COP8 OTP and EPROM product families. Summary of features is as follows:

- Real-time in-circuit emulation; full operating voltage range operation, full DC-10 MHz clock.
- All processor I/O pins can be cabled to an application development board with package compatible cable to socket and surface mount assembly.
- Full 32 kbyte of loadable programming space that overlays (replaces) the on-chip ROM or EPROM. On-chip RAM and I/O blocks are used directly or recreated as necessary.
- 100 frames of synchronous trace memory. The display can be HLL source (C source), assembly or mixed. The most recent history prior to a break is available in the trace memory.
- Configured break points; uses INTR instruction which is modestly intrusive.
- Software—only supported features are selectable.
- Tool set integrated interactive symbolic debugger—supports both assembler (COFF) and C Compiler (.COD) SDK linked object formats.
- Instruction by instruction memory/register changes displayed when in single step operation.
- Debugger software is processor customized, and reconfigured from a master model file.
- Processor specific symbolic display of registers and bit level assignments, configured from master model file.
- Halt/Idle mode notification.
- Programming menu supports full product line of programmable OTP and EPROM COP8 products. Program data is taken directly from the overlay RAM.
- Programming of 44 PLCC and 68 PLCC parts requires external programming adapters.
- Includes wallmount power supply.
- On-board VPP generator from 5V input or connection to external supply supported. Requires VPP level adjustment per the family programming specification (correct level is provided on an on-screen pop-down display).
- On-line HELP customized to specific processor using master model file.
- Includes a copy of COP8-DEV-IBMA assembler and linker SDK.

DM Order Information

Debug Model Unit	
COP8-DM/888CF	
Cable Adapters	
DM-COP8/28D	28 DIP
DM-COP8/40D	40 DIP
DM-COP8/44P	44 PLCC
Adapter for SO Package	
MHW-SOIC 28	28 SO



TL/DD/9425-39

FIGURE 20. COP8-DM Environment

Development Support (Continued)

COP8 ASSEMBLER/LINKER SOFTWARE DEVELOPMENT TOOL KIT

National Semiconductor offers a relocateable COP8 macro cross assembler, linker, librarian and utility software development tool kit. Features are summarized as follows:

- Basic and Feature Family instruction set by "device" type.
- Nested macro capability.
- Extensive set of assembler directives.
- Supported on PC/DOS platform.
- Generates National standard COFF output files.
- Integrated Linker and Librarian.
- Integrated utilities to generate ROM code file outputs.
- DUMPCOFF utility.

This product is integrated as a part of MetaLink tools as a development kit, fully supported by the MetaLink debugger. It may be ordered separately or it is bundled with the MetaLink products at no additional cost.

Order Information

Assembler SDK:	
COP8-DEV-IBMA	Assembler SDK on installable 3.5" PC/DOS Floppy Disk Drive format. Periodic upgrades and most recent version is available on National's BBS and Internet.

COP8 C COMPILER

A C Compiler is developed and marketed by Byte Craft Limited. The COP8C compiler is a fully integrated development tool specifically designed to support the compact embedded configuration of the COP8 family of products.

Features are summarized as follows:

- ANSI C with some restrictions and extensions that optimize development for the COP8 embedded application.
- BITS data type extension. Register declaration #pragma with direct bit level definitions.
- C language support for interrupt routines.
- Expert system, rule based code geration and optimization.
- Performs consistency checks against the architectural definitions of the target COP8 device.
- Generates program memory code.
- Supports linking of compiled object or COP8 assembled object formats.
- Global optimization of linked code.
- Symbolic debug load format fully sourced level supported by the MetaLink debugger.

SINGLE CHIP OTP/EMULATOR SUPPORT

The COP8 family is supported by single chip OTP emulators. For detailed information refer to the emulator specific datasheet and the emulator selection table below:

Approved List

Manufacturer	North America	Europe	Asia
BP Microsystems	(800) 225-2102 (713) 688-4600 Fax: (713) 688-0920	+49-8152-4183 +49-8856-932616	+852-234-16611 +852-2710-8121
Data I/O	(800) 426-1045 (206) 881-6444 Fax: (206) 882-1043	+44-0734-440011	Call North America
HI-LO	(510) 623-8860	Call Asia	+886-2-764-0215 Fax: +886-2-756-6403
ICE Technology	(800) 624-8949 (919) 430-7915	+44-1226-767404 Fax:0-1226-370-434	
MetaLink	(800) 638-2423 (602) 926-0797 Fax: (602) 693-0681	+49-80 9156 96-0 Fax: +49-80 9123 86	+852-737-1800
Systems General	(408) 263-6667	+41-1-9450300	+886-2-917-3005 Fax: +886-2-911-1283
Needhams	(916) 924-8037 Fax: (916) 924-8065		

Development Support (Continued)

OTP Emulator Ordering Information

Device Number	Clock Option	Package	Emulates
COP87L84CFM-XE	Crystal	28 SO	COP884CF
COP87L84CFN-XE	Crystal	28 DIP	COP884CF
COP87L88CFN-XE	Crystal	40 DIP	COP888CF

INDUSTRY WIDE OTP/EPROM PROGRAMMING SUPPORT

Programming support, in addition to the MetaLink development tools, is provided by a full range of independent approved vendors to meet the needs from the engineering laboratory to full production.

AVAILABLE LITERATURE

For more information, please see the COP8 Basic Family User's Manual, Literature Number 620895, COP8 Feature Family User's Manual, Literature Number 620897 and National's Family of 8-bit Microcontrollers COP8 Selection Guide, Literature Number 630009.

DIAL-A-HELPER SERVICE

Dial-A-Helper is a service provided by the Microcontroller Applications group. The Dial-A-Helper is an Electronic Information System that may be accessed as a Bulletin Board System (BBS) via data modem, as an FTP site on the Internet via standard FTP client application or as an FTP site on the Internet using a standard Internet browser such as Netscape or Mosaic.

The Dial-A-Helper system provides access to an automated information storage and retrieval system . The system capabilities include a MESSAGE SECTION (electronic mail, when accessed as a BBS) for communications to and from the Microcontroller Applications Group and a FILE SECTION which consists of several file areas where valuable application software and utilities could be found.

DIAL-A-HELPER BBS via a Standard Modem

Modem:	CANADA/U.S.:	(800) NSC-MICRO (800) 672-6427
	EUROPE:	(+49) 0-8141-351332
	Baud:	14.4k
	Set-Up:	Length: 8-Bit Parity: None Stop Bit: 1
	Operation:	24 Hours, 7 Days

DIAL-A-HELPER via FTP

ftp nscmicro.nsc.com
user: anonymous
password: username@yourhost.site.domain

DIAL-A-HELPER via WorldWide Web Browser

ftp://nscmicro.nsc.com

National Semiconductor on the WorldWide Web

See us on the WorldWide Web at: http://www.national.com

CUSTOMER RESPONSE CENTER

Complete product information and technical support is available from National's customer response centers.

CANADA/U.S.:	Tel:	(800)272-9959
	email:	support@tevm2.nsc.com
EUROPE:	email:	europe.support@nsc.com
	Deutsch Tel:	+49 (0) 180-530 85 85
	English Tel:	+49 (0) 180-532 78 32
	Français Tel:	+49 (0) 180-532 93 58
	Italiano Tel:	+49 (0) 180-534 16 80
JAPAN:	Tel:	+81-043-299-2309
S.E. ASIA:	Beijing Tel:	(+86) 10-6856-8601
	Shanghai Tel:	(+86) 21-6415-4092
	Hong Kong Tel:	(+852) 2737-1600
	Korea Tel:	(+82) 2-3771-6909
	Malaysia Tel:	(+60-4) 644-9061
	Singapore Tel:	(+65) 255-2226
	Taiwan Tel:	+886-2-521-3288
AUSTRALIA:	Tel:	(+61) 3-9558-9999
INDIA:	Tel:	(+91) 80-559-9467

National Semiconductor

ADVANCE INFORMATION

COP8ACC5
8-Bit Microcontroller with High Resolution A/D Conversion

General Description

The COP8ACC5 is a member of the COP8™ 8-bit Microcontroller family. It is a fully static Microcontroller, fabricated using double-metal silicon gate microCMOS technology.

(Continued)

Key Features

- Analog Function Block for high resolution A/D including
  - — Analog comparator with seven input muxes
  - — Constant Current Source and $V_{CC/2}$ Reference
  - — 16-bit capture timer (upcounter) clocked from CKI with auto reset on timer startup
- Quiet design (reduced radiated emissions)
- 4096 bytes on-board ROM
- 128 bytes on-board RAM

Additional Peripheral Features

- Idle Timer
- One 16-bit timer with two 16-bit registers supporting:
  - — Processor Independent PWM mode
  - — External Event counter mode
  - — Input Capture mode
- Multi-Input Wake-Up (MIWU) with optional interrupts (4)
- WATCHDOG™ and clock monitor logic
- MICROWIRE/PLUS™ serial I/O with programmable shift clock-polarity

I/O Features

- Memory mapped I/O
- Software selectable I/O options (TRI-STATE® Output, Push-Pull Output, Weak Pull-Up Input, High Impedance Input)
- High current outputs
- Schmitt Trigger inputs on ports G and L
- Packages: 28 DIP/SO with 24 I/O pins
  20 SO with 16 I/O pins

CPU/Instruction Set Features

- 1 μs instruction cycle time
- Eight multi-source vectored interrupt servicing
  - — External Interrupt
  - — Idle Timer T0
  - — Timer T1 associated Interrupts
  - — MICROWIRE/PLUS
  - — Multi-Input Wake Up
  - — Software Trap
  - — Default VIS
  - — A/D (Capture Timer)
- Versatile and easy to use instruction set
- 8-bit Stack Pointer (SP)—stack in RAM
- Two 8-bit Registers Indirect Data Memory Pointers (B and X)

Fully Static CMOS

- Low current drain (typically < 5 μA)
- Two power saving modes: HALT and IDLE
- Single supply operation: 2.5V to 5.5V
- Temperature ranges: 0°C to +70°C, −40°C to +85°C, −55°C to +125°C

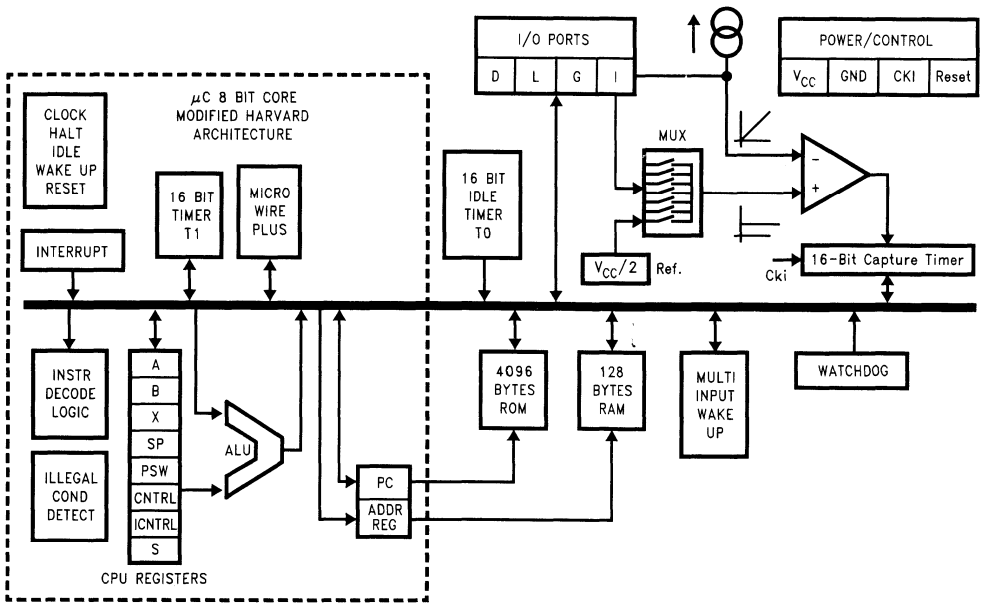
Development System

- Emulation and OTP devices
- Real time emulation and full program debug offered by MetaLink development system

Applications

- Battery Chargers
- Appliances
- Data Acquisition systems

Block Diagram



TL/DD/12865–1

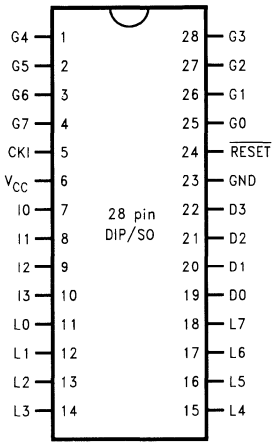
FIGURE 1. Block Diagram

General Description (Continued)

The device provides up to 6 channels of A/D performing a measurement with 12 bits of resolution in less than 0.5 ms at a clock-rate of 10 MHz. There is only an external capacitor required to complete the measurement setup and establishing low cost, high-resolution (up to 16 bits) and accurate A/D.

This device is a complete microcomputer containing all system timing, interrupt logic, ROM, RAM, and I/O necessary to implement dedicated control functions in a variety of applications. Features include an 8-bit memory mapped architecture, MICROWIRE/PLUS serial I/O, one 16-bit PWM-timer with two autoreload registers, multi-sourced interrupts an Analog Function Block and an idle timer WATCHDOG. Each I/O pin has software selectable options to adapt the device to the specific application. The device operates over a voltage range of 2.5V to 5.5V. High throughput is achieved with an efficient, regular instruction set operating at a minimum of 2 ms per instruction cycle. Low radiated emissions are achieved by gradual turn-on output drivers and internal I_{CC} smoothing filters on the chip logic and crystal oscillator.

Connection Diagrams

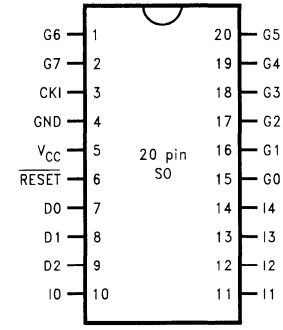


TL/DD/12865–2

Top View

Order Number COP8ACC528N9, COP8ACC528N8 or COP8ACC528N6
See NS Molded Package Number N28A

Order Number COP8ACC528M9, COP8ACC528M8 or COP8ACC528M6
See NS Molded Package Number M28B



TL/DD/12865–3

Top View

Order Number COP8ACC520M9, COP8ACC520N8 or COP8ACC520M6
See NS Molded Package Number M20B

FIGURE 2

Connection Diagrams (Continued)

Pinouts for 28-Pin, 20-Pin Packages

Port	Type	Alt. Fun	Alt. Fun	28-Pin DIP/SO	20-Pin SO
L4	I/O	MIWU	Ext. Int.	4	
L5	I/O	MIWU	Ext. Int.	5	
L6	I/O	MIWU	Ext. Int.	6	
L7	I/O	MIWU	Ext. Int.	7	
G0	I/O	INT		23	15
G1	WDOUT			24	16
G2	I/O	T1B		25	17
G3	I/O	T1A		26	18
G4	I/O	SO		27	19
G5	I/O	SK		28	20
G6	I	SI		1	1
G7	I/CKO	HALT Restart		2	2
D0	O			11	7
D1	O			12	8
D2	O			13	9
D3	O			14	
I0	I	Analog CH1		15	10
I1	I	I_{SRC}		16	11
I2	I	Analog CH2		17	12
I3	I	Analog CH3		18	13
I4	I	Analog CH4		19	14
I5	I	Analog CH5		20	
I6	I	Analog CH6		21	
I7	I	C_{OUT}		22	
V_{CC}				9	5
GND				8	4
CKI				3	3
$\overline{\text{RESET}}$				10	6

Absolute Maximum Ratings

If Military/Aerospace specified devices are required, please contact the National Semiconductor Sales Office/Distributors for availability and specifications.

Supply Voltage (V_{CC})	7V
Voltage at Any Pin	−0.3V to V_{CC} +0.3V
Total Current into V_{CC} Pin (Source)	100 mA
Total Current out of GND Pin (Sink)	110 mA
Storage Temperature Range	−65°C to +140°C

Note: *Absolute maximum ratings indicate limits beyond which damage to the device may occur. DC and AC electrical specifications are not ensured when operating the device at absolute maximum ratings.*

DC Electrical Characteristics 0°C ≤ T_A ≤ +70°C unless otherwise specified

Parameter	Conditions	Min	Typ	Max	Units
Operating Voltage	Peak-to-Peak	2.5		5.5	V
Power Supply Ripple (Note 1)				0.1 V_{CC}	V
Supply Current (Note 2)					
CKI = 4 MHz	V_{CC} = 5.5V, t_C = 2.5 μs			5.5	mA
CKI = 4 MHz	V_{CC} = 4V, t_C = 2.5 μs			2.5	mA
CKI = 1 MHz	V_{CC} = 4V, t_C = 10 μs			1.4	mA
HALT Current (Note 3)	V_{CC} = 5.5V, CKI = 0 MHz		< 5	8	μA
	V_{CC} = 4V, CKI = 0 MHz		< 3	4	μA
IDLE Current					
CKI = 4 MHz	V_{CC} = 5.5V, t_C = 2.5 μs			1.5	mA
CKI = 1 MHz	V_{CC} = 4V, t_C = 10 μs			0.5	mA
Input Levels (V_{IH}, V_{IL})					
$\overline{\text{RESET}}$					
Logic High		0.8 V_{CC}			V
Logic Low				0.2 V_{CC}	V
CKI, All Other Inputs					
Logic High		0.7 V_{CC}			V
Logic Low				0.2 V_{CC}	V
Hi-Z Input Leakage	V_{CC} = 5.5V	1		1	μA
Input Pullup Current	V_{CC} = 5.5V, V_{IN} = 0V	−40		−250	μA
G and L Port Input Hysteresis	(Note 5)			0.35 V_{CC}	V
Output Current Levels					
D Outputs					
Source	V_{CC} = 4V, V_{OH} = 3.3V	−0.4			mA
	V_{CC} = 2.5V, V_{OH} = 1.8V	−0.2			mA
Sink	V_{CC} = 4V, V_{OL} = 1V	10			mA
	V_{CC} = 2.5V, V_{OL} = 0.4V	2.0			mA
All Others					
Source (Weak Pull-Up Mode)	V_{CC} = 4V, V_{OH} = 2.7V	−10		−110	μA
	V_{CC} = 2.5V, V_{OH} = 1.8V	−2.5		−33	μA
Source (Push-Pull Mode)	V_{CC} = 4V, V_{OH} = 3.3V	−0.4			mA
	V_{CC} = 2.5V, V_{OH} = 1.8V	−0.2			mA
Sink (Push-Pull Mode)	V_{CC} = 4V, V_{OL} = 0.4V	1.6			mA
	V_{CC} = 2.5V, V_{OL} = 0.4V	0.7			mA
TRI-STATE Leakage	V_{CC} = 5.5V	1		1	μA
Allowable Sink/Source Current per Pin					
D Outputs (Sink)				15	mA
All others				3	mA
Maximum Input Current without Latchup (Note 4)	Room Temp			±200	mA
RAM Retention Voltage, V_r	500 ns Rise and Fall Time (min)	2			V
Input Capacitance	(Note 5)			7	pF
Load Capacitance on D2	(Note 5)			1000	pF

AC Electrical Characteristics $0°C \le T_A \le +70°C$ unless otherwise specified

Parameter	Conditions	Min	Typ	Max	Units
Instruction Cycle Time (t_C)					
Crystal, Resonator	$2.5V \le V_{CC} \le 4V$	2.5		DC	μs
	$4V \le V_{CC} \le 5.5V$	1.0		DC	μs
R/C Oscillator	$2.5V \le V_{CC} \le 4V$	7.5		DC	μs
	$4V \le V_{CC} \le 5.5V$	3.0		DC	μs
Inputs					
t_{SETUP}	$4V \le V_{CC} \le 5.5V$	200			ns
	$2.5V \le V_{CC} \le 4V$	500			ns
t_{HOLD}	$4V \le V_{CC} \le 5.5V$	60			ns
	$2.5V \le V_{CC} \le 4V$	150			ns
Output Propagation Delay (Note 5)	$R_L = 2.2k$, $C_L = 100$ pF				
t_{PD1}, t_{PD0}					
SO, SK	$4V \le V_{CC} \le 5.5V$			0.7	μs
	$2.5V \le V_{CC} \le 4V$			1.75	μs
All Others	$4V \le V_{CC} \le 5.5V$				1 μs
	$2.5V \le V_{CC} \le 4V$			2.5	μs
MICROWIRE Setup Time (t_{UWS}) (Note 5)	$V_{CC} \ge 4V$	20			ns
MICROWIRE Hold Time (t_{UWH}) (Note 5)	$V_{CC} \ge 4V$	56			ns
MICROWIRE Output Propagation Delay (t_{UPD})	$V_{CC} \ge 4V$			220	ns
Input Pulse Width (Note 6)					
Interrupt Input High Time		1			t_C
Interrupt Input Low Time		1			t_C
Timer 1, 2, 3 Input High Time		1			t_C
Timer 1, 2, 3 Input Low Time		1			t_C
Reset Pulse Width		1			μs

Note 1: Maximum rate of voltage change must be < 0.5V/ms.

Note 2: Supply current is measured after running 2000 cycles with a square wave CKI input, CKO open, inputs at rails and outputs open.

Note 3: The HALT mode will stop CKI from oscillating in the RC and the Crystal configurations. Measurement of I_{DD} HALT is done with device neither sourcing or sinking current; with L, C, and G0–G5 programmed as low outputs and not driving a load; all outputs programmed low and not driving a load; all inputs tied to V_{CC}; clock monitor and comparator disabled. Parameter refers to HALT mode entered via setting bit 7 of the G Port data register. Part will pull up CKI during HALT in crystal clock mode.

Note 4: Pins G6 and $\overline{\text{RESET}}$ are designed with a high voltage input network. These pins allow input voltages > V_{CC} and the pins will have sink current to V_{CC} when biased at voltages > V_{CC} (the pins do not have source current when biased at a voltage below V_{CC}). The effective resistance to V_{CC} is 750Ω (typical). These two pins will not latch up. The voltage at the pins must be limited to less than 14V. **WARNING: Voltages in excess of 14V will cause damage to the pins. This warning excludes ESD transients.**

Note 5: The output propagation delay is referenced to the end of the instruction cycle where the output change occurs.

Note 6: Parameter characterized but not tested.

Note 7: t_C = Instruction Cycle Time.

Absolute Maximum Ratings

If Military/Aerospace specified devices are required, please contact the National Semiconductor Sales Office/Distributors for availability and specifications.

Supply Voltage (V_{CC})	7V
Voltage at Any Pin	−0.3V to V_{CC} +0.3V
Total Current into V_{CC} Pin (Source)	100 mA
Total Current out of GND Pin (Sink)	110 mA
Storage Temperature Range	−65°C to +140°C

Note: *Absolute maximum ratings indicate limits beyond which damage to the device may occur. DC and AC electrical specifications are not ensured when operating the device at absolute maximum ratings.*

DC Electrical Characteristics $-40°C \le T_A \le +85°C$ unless otherwise specified

Parameter	Conditions	Min	Typ	Max	Units
Operating Voltage		2.5		5.5	V
Power Supply Ripple (Note 1)	Peak-to-Peak			0.1 V_{CC}	V
Supply Current (Note 2)					
CKI = 4 MHz	V_{CC} = 5.5V, t_C = 2.5 μs			5.5	mA
CKI = 4 MHz	V_{CC} = 4V, t_C = 2.5 μs			2.5	mA
CKI = 1 MHz	V_{CC} = 4V, t_C = 10 μs			1.4	mA
HALT Current (Note 3)	V_{CC} = 5.5V, CKI = 0 MHz		< 5	10	μA
	V_{CC} = 4V, CKI = 0 MHz		< 3	6	μA
IDLE Current					
CKI = 4 MHz	V_{CC} = 5.5V, t_C = 2.5 μs			1.5	mA
CKI = 1 MHz	V_{CC} = 4V, t_C = 10 μs			0.5	mA
Input Levels (V_{IH}, V_{IL})					
$\overline{\text{RESET}}$					
Logic High		0.8 V_{CC}			V
Logic Low				0.2 V_{CC}	V
CKI, All Other Inputs					
Logic High		0.7 V_{CC}			V
Logic Low				0.2 V_{CC}	V
Hi-Z Input Leakage	V_{CC} = 5.5V	−2		+2	μA
Input Pullup Current	V_{CC} = 5.5V, V_{IN} = 0V	−40		−250	μA
G and L Port Input Hysteresis	(Note 5)			0.35 V_{CC}	V
Output Current Levels					
D Outputs					
Source	V_{CC} = 4V, V_{OH} = 3.3V	−0.4			mA
	V_{CC} = 2.5V, V_{OH} = 1.8V	−0.2			mA
Sink	V_{CC} = 4V, V_{OL} = 1V	10			mA
	V_{CC} = 2.5V, V_{OL} = 0.4V	2.0			mA
All Others					
Source (Weak Pull-Up Mode)	V_{CC} = 4V, V_{OH} = 2.7V	−10		−110	μA
	V_{CC} = 2.5V, V_{OH} = 1.8V	−2.5		−33	μA
Source (Push-Pull Mode)	V_{CC} = 4V, V_{OH} = 3.3V	−0.4			mA
	V_{CC} = 2.5V, V_{OH} = 1.8V	−0.2			mA
Sink (Push-Pull Mode)	V_{CC} = 4V, V_{OL} = 0.4V	1.6			mA
	V_{CC} = 2.5V, V_{OL} = 0.4V	0.7			mA
TRI-STATE Leakage	V_{CC} = 5.5V	−2		+2	μA
Allowable Sink/Source Current per Pin					
D Outputs (Sink)				15	mA
All others				3	mA
Maximum Input Current without Latchup (Note 4)	Room Temp			±200	mA
RAM Retention Voltage, V_r	500 ns Rise and Fall Time (min)	2			V
Input Capacitance	(Note 5)			7	pF
Load Capacitance on D2	(Note 5)			1000	pF

AC Electrical Characteristics $-40°C \le T_A \le +85°C$ unless otherwise specified

Parameter	Conditions	Min	Typ	Max	Units
Instruction Cycle Time (t_C)					
Crystal, Resonator	$2.5V \le V_{CC} < 4V$	2.5		DC	μs
	$4V \le V_{CC} \le 5.5V$	1.0		DC	μs
R/C Oscillator	$2.5V \le V_{CC} < 4V$	7.5		DC	μs
	$4V \le V_{CC} < 5.5V$	3.0		DC	μs
Inputs					
t_{SETUP}	$4V \le V_{CC} \le 5.5V$	200			ns
	$2.5V \le V_{CC} < 4V$	500			ns
t_{HOLD}	$4V \le V_{CC} \le 5.5V$	60			ns
	$2.5V \le V_{CC} < 4V$	150			ns
Output Propagation Delay (Note 5)	$R_L = 2.2k$, $C_L = 100$ pF				
t_{PD1}, t_{PD0}					
SO, SK	$4V \le V_{CC} \le 5.5V$			0.7	μs
	$2.5V \le V_{CC} < 4V$			1.75	μs
All Others	$4V \le V_{CC} \le 5.5V$			1	μs
	$2.5V \le V_{CC} < 4V$			2.5	μs
MICROWIRE Setup Time (t_{UWS}) (Note 5)	$V_{CC} \ge 4V$	20			ns
MICROWIRE Hold Time (t_{UWH}) (Note 5)	$V_{CC} \ge 4V$	56			ns
MICROWIRE Output Propagation Delay (t_{UPD})	$V_{CC} \ge 4V$			220	ns
Input Pulse Width (Note 6)					
Interrupt Input High Time		1			t_C
Interrupt Input Low Time		1			t_C
Timer 1, 2, 3 Input High Time		1			t_C
Timer 1, 2, 3 Input Low Time		1			t_C
Reset Pulse Width		1			μs

Note 1: Maximum rate of voltage change must be < 0.5 V/ms.

Note 2: Supply current is measured after running 2000 cycles with a square wave CKI input, CKO open, inputs at rails and outputs open.

Note 3: The HALT mode will stop CKI from oscillating in the RC and the Crystal configurations. Measurement of I_{DD} HALT is done with device neither sourcing or sinking current; with L, C, and G0–G5 programmed as low outputs and not driving a load; all outputs programmed low and not driving a load; all inputs tied to V_{CC}; clock monitor and comparator disabled. Parameter refers to HALT mode entered via setting bit 7 of the G Port data register. Part will pull up CKI during HALT in crystal clock mode.

Note 4: Pins G6 and $\overline{\text{RESET}}$ are designed with a high voltage input network. These pins allow input voltages > V_{CC} and the pins will have sink current to V_{CC} when biased at voltages greater than V_{CC} (the pins do not have source current when biased at a voltage below V_{CC}). The effective resistance to V_{CC} is 750Ω (typical). These two pins will not latch up. The voltage at the pins must be limited to less than 14V. **WARNING: Voltages in excess of 14V will cause damage to the pins. This warning excludes ESD transients.**

Note 5: The output propagation delay is referenced to the end of the instruction cycle where the output change occurs.

Note 6: Parameter characterized but not tested.

Note 7: t_C = Instruction Cycle Time.

Absolute Maximum Ratings

If Military/Aerospace specified devices are required, please contact the National Semiconductor Sales Office/Distributors for availability and specifications.

Supply Voltage (V_{CC})	7V
Voltage at Any Pin	−0.3V to V_{CC} +0.3V
Total Current into V_{CC} Pin (Source)	100 mA
Total Current out of GND Pin (Sink)	110 mA
Storage Temperature Range	−65°C to +140°C

Note: *Absolute maximum ratings indicate limits beyond which damage to the device may occur. DC and AC electrical specifications are not ensured when operating the device at absolute maximum ratings.*

DC Electrical Characteristics $-55°C \le T_A \le +125°C$ unless otherwise specified

Parameter	Conditions	Min	Typ	Max	Units
Operating Voltage		4.5		5.5	V
Power Supply Ripple (Note 1)	Peak-to-Peak			0.1 V_{CC}	V
Supply Current (Note 2)					
CKI = 4 MHz	V_{CC} = 5.5V, t_C = 2.5 μs			5.5	mA
HALT Current (Note 3)	V_{CC} = 5.5V, CKI = 0 MHz		< 5	30	μA
IDLE Current					
CKI = 4 MHz	V_{CC} = 5.5V, t_C = 2.5 μs			1.5	mA
Input Levels (V_{IH}, V_{IL})					
$\overline{\text{RESET}}$					
Logic High		0.8 V_{CC}			V
Logic Low				0.2 V_{CC}	V
CKI, All Other Inputs					
Logic High		0.7 V_{CC}			V
Logic Low				0.2 V_{CC}	V
Hi-Z Input Leakage	V_{CC} = 5.5V	5		5	μA
Input Pullup Current	V_{CC} = 5.5V, VIN = 0V	35		400	μA
G and L Port Input Hysteresis	(Note 5)			0.35 V_{CC}	V
Output Current Levels					
D Outputs					
Source	V_{CC} = 4.5V, V_{OH} = 3.3V	−0.4			mA
Sink	V_{CC} = 4.5V, V_{OL} = 1V	10			mA
All Others					
Source (Weak Pull-Up Mode)	V_{CC} = 4.5V, V_{OH} = 2.7V	−10		−110	μA
Source (Push-Pull Mode)	V_{CC} = 4.5V, V_{OH} = 3.3V	−0.4			mA
Sink (Push-Pull Mode)	V_{CC} = 4.5V, V_{OL} = 0.4V	1.6			mA
TRI-STATE Leakage	V_{CC} = 5.5V	−2		+2	mA
Allowable Sink/Source Current per Pin					
D Outputs (Sink)				15	mA
All others				3	mA
Maximum Input Current without Latchup (Note 4)	Room Temp			±200	mA
RAM Retention Voltage, V_r	500 ns Rise and Fall Time (min)	2			V
Input Capacitance	(Note 5)			7	pF
Load Capacitance on D2	(Note 5)			1000	pF

AC Electrical Characteristics $-55°C \leq T_A \leq +125°C$ unless otherwise specified

Parameter	Conditions	Min	Typ	Max	Units
Instruction Cycle Time (t_C)					
Crystal, Resonator	$4.5V \leq V_{CC} \leq 5.5V$	1.0		DC	μs
R/C Oscillator	$4.5V \leq V_{CC} \leq 5.5V$	3.0		DC	μs
Inputs					
t_{SETUP}	$4.5 \leq V_{CC} \leq 5.5V$	200			ns
t_{HOLD}	$4.5V \leq V_{CC} \leq 5.5V$	60			ns
Output Propagation Delay (Note 5)	$R_L = 2.2k$, $C_L = 100$ pF				
tPD1, tPD0					
SO, SK	$4.5V \leq V_{CC} \leq 5.5V$			0.7	μs
All Others	$4.5V \leq V_{CC} \leq 5.5V$			1	μs
MICROWIRE Setup Time (t_{UWS}) (Note 5)	$V_{CC} \geq 4.5V$	20			ns
MICROWIRE Hold Time (t_{UWH}) (Note 5)	$V_{CC} \geq 4.5V$	56			ns
MICROWIRE Output Propagation Delay (t_{UPD})	$V_{CC} \geq 4.5V$			220	ns
Input Pulse Width (Note 6)					
Interrupt Input High Time		1			t_C
Interrupt Input Low Time		1			t_C
Timer 1,2, 3 Input High Time		1			t_C
Timer 1,2, 3 Input Low Time		1			t_C
Reset Pulse Width		1			μs

Note 1: Maximum rate of voltage change must be < 0.5V/ms.

Note 2: Supply current is measured after running 2000 cycles with a square wave CKI input, CKO open, inputs at rails and outputs open.

Note 3: The HALT mode will stop CKI from oscillating in the RC and the Crystal configurations. Measurement of I_{DD} HALT is done with device neither sourcing or sinking current; with L, C, and G0–G5 programmed as low outputs and not driving a load; all outputs programmed low and not driving a load; all inputs tied to V_{CC}; clock monitor and comparator disabled. Parameter refers to HALT mode entered via setting bit 7 of the G Port data register. Part will pull up CKI during HALT in crystal clock mode.

Note 4: Pins G6 and $\overline{\text{RESET}}$ are designed with a high voltage input network. These pins allow input voltages > V_{CC} and the pins will have sink current to V_{CC} when biased at voltages > V_{CC} (the pins do not have source current when biased at a voltage below V_{CC}). The effective resistance to V_{CC} is 750Ω (typical). These two pins will not latch up. The voltage at the pins must be limited to less than 14V. **WARNING: Voltages in excess of 14V will cause damage to the pins. This warning excludes ESD transients.**

Note 5: The output propagation delay is referenced to the end of the instruction cycle where the output change occurs.

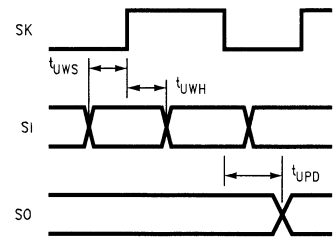
Note 6: Parameter characterized but not tested.

Note 7: t_C = Instruction Cycle Time.

Comparator AC and DC Characteristics V_{CC} = 5V, $-40°C \leq T_A \leq +85°C$

Parameter	Conditions	Min	Typ	Max	Units
Input Offset Voltage	$0.4V < V_{IN} < V_{CC} - 1.5V$		10	25	mV
Input Common Mode Voltage Range (Note 8)		0.4		$V_{CC} - 1.5$	V
Voltage Gain			300k		V/V
$V_{CC}/2$ Reference	$4.0V < V_{CC} < 5.5V$	$0.5\ V_{CC} - 0.04$	$0.5V_{CC}$	$0.5V_{CC} + 0.04$	V
DC Supply Current For Comparator (when enabled)	V_{CC} = 5.5V			250	μA
DC Supply Current For $V_{CC}/2$ reference (when enabled)	V_{CC} 5.5V		50	80	μA
DC Supply Current For Constant Current Source (when enabled)	V_{CC} = 5.5V			200	μA
Constant Current Source	$4.0V < V_{CC} < 5.5V$	10	20	40	μA
Current Source Variation	$4.0V < V_{CC} < 5.5V$ Temp = Constant			2	μA
Current Source Enable Time			1.5	2	μs
Comparator Response Time	10 mV overdrive, 100 pF load			1	μs

Note 8: The device is capable of operating over a common mode voltage range of 0 to V_{CC} − 1.5V, however increased offset voltage will be observed between 0V and 0.4V.



TL/DD/12865–4

FIGURE 2. MICROWIRE/PLUS Timing

Pin Descriptions

V_{CC} and GND are the power supply pins. All V_{CC} and GND pins must be connected.

CKI is the clock input. This can come from an R/C generated oscillator, or a crystal oscillator (in conjunction with CKO). See Oscillator Description section.

$\overline{\text{RESET}}$ is the master reset input. See Reset description section.

The device contains two bidirectional (one 8-bit, one 4-bit) I/O ports (G and L), where each individual bit may be independently configured as an input (Schmitt trigger inputs on ports L and G), output or TRI-STATE under program control. Three data memory address locations are allocated for each of these I/O ports. Each I/O port has two associated 8-bit memory mapped registers, the CONFIGURATION register and the output DATA register. A memory mapped address is also reserved for the input pins of each I/O port. (See the memory map for the various addresses associated with the I/O ports.) *Figure 3* shows the I/O port configurations. The DATA and CONFIGURATION registers allow for each port bit to be individually configured under software control as shown below:

PORT L is a 4-bit I/O port. All L-pins have Schmitt triggers on the inputs.

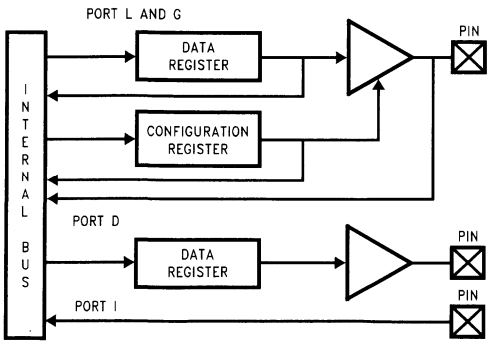
The Port L supports Multi-Input Wake Up on all four pins. The Port L has the following alternate features:

L4 MIWU or external interrupt

L5 MIWU or external interrupt

L6 MIWU or external interrupt

L7 MIWU or external interrupt



TL/DD/12865–5

FIGURE 3. I/P Port Configurations

Configuration Register	Data Register	Port Set-Up
0	0	Hi-Z Input (TRI-STATE Output)
0	1	Input with Weak Pull-Up
1	0	Push-Pull Zero Output
1	1	Push-Pull One Output

Please note: The lower 4 L-bits read all ones (L0:L3). This is independant from the states of the associated bits in the L-port Data- and Configuration register. The lower 4 bits in the L-port Data- and Configuration register can be used as general purpose status indicators (flags).

Port G is an 8-bit port with 5 I/O pins (G0, G2–G5), an input pin (G6), and a dedicated output pin (G7). Pins G0 and G2–G6 all have Schmitt Triggers on their inputs. Pin G1 serves as the dedicated WDOUT WATCHDOG output, while pin G7 is either input or output depending on the oscillator mask option selected. With the crystal oscillator option selected, G7 serves as the dedicated output pin for the CKO clock output. With the single-pin R/C oscillator mask option selected, G7 serves as a general purpose input pin but is also used to bring the device out of HALT mode with a low to high transition on G7. There are two registers associated with the G Port, a data register and a configuration register. Therefore, each of the 5 I/O bits (G0, G2–G5) can be individually configured under software control.

Since G6 is an input only pin and G7 is the dedicated CKO clock output pin (crystal clock option) or general purpose input (R/C clock option), the associated bits in the data and configuration registers for G6 and G7 are used for special purpose functions as outlined below. Reading the G6 and G7 data bits will return zeros.

Note that the chip will be placed in the HALT mode by writing a "1" to bit 7 of the Port G Data Register. Similarly the chip will be placed in the IDLE mode by writing a "1" to bit 6 of the Port G Data Register.

Writing a "1" to bit 6 of the Port G Configuration Register enables the MICROWIRE/PLUS to operate with the alternate phase of the SK clock. The G7 configuration bit, if set high, enables the clock start up delay after HALT when the R/C clock configuration is used.

	Config Reg.	Data Reg.
G7	CLKDLY	HALT
G6	Alternate SK	IDLE

Port G has the following alternate features:

G0 INTR (External Interrupt Input)

G2 T1B (Timer T1 Capture Input)

G3 T1A (Timer T1 I/O)

G4 SO (MICROWIRE Serial Data Output)

G5 SK (MICROWIRE Serial Clock)

G6 SI (MICROWIRE Serial Data Input)

Port G has the following dedicated functions:

G1 WDOUT WATCHDOG and/or Clock Monitor dedicated output.

Pin Descriptions (Continued)

G7 CKO Oscillator dedicated output or general purpose input

Port I is an eight-bit Hi-Z input port.

Port I0–I7 are used for the analog function block.

The Port I has the following alternate features:

I0 COMPIN1+ (Comparator Positive Input 1)

I1 COMPIN− (Comparator Negative Input/Current Source Out)

I2 COMPIN0+ (Comparator Positive Input 0)

I3 COMPOUT/COMPIN2+ (Comparator Output/Comparator Positive Input 2)

I4 COMPIN3+ (Comparator Positive Input 3)

I5 COMPIN4+ (Comparator Positive Input 4)

I6 COMPIN5+ (Comparator Positive Input 5)

I7 COMPOUT (Comparator Output)

Port D is a 4-bit output port that is preset high when $\overline{\text{RESET}}$ goes low. The user can tie two or more D port outputs (except D2) together in order to get a higher drive.

Functional Description

The architecture of the device is a modified Harvard architecture. With the Harvard architecture, the control store program memory (ROM) is separated from the data store memory (RAM). Both ROM and RAM have their own separate addressing space with separate address buses. The architecture, though based on the Harvard architecture, permits transfer of data from ROM to RAM.

CPU REGISTERS

The CPU can do an 8-bit addition, subtraction, logical or shift operation in one instruction (t_C) cycle time.

There are six CPU registers:

A is the 8-bit Accumulator Register

PC is the 15-bit Program Counter Register

PU is the upper 7 bits of the program counter (PC)

PL is the lower 8 bits of the program counter (PC)

B is an 8-bit RAM address pointer, which can be optionally post auto incremented or decremented.

X is an 8-bit alternate RAM address pointer, which can be optionally post auto incremented or decremented.

SP is the 8-bit stack pointer, which points to the subroutine/interrupt stack (in RAM). The SP is initialized to RAM address 06F with reset.

All the CPU registers are memory mapped with the exception of the Accumulator (A) and the Program Counter (PC).

PROGRAM MEMORY

The program memory consists of 4096 bytes of ROM. These bytes may hold program instructions or constant data (data tables for the LAID instruction, jump vectors for the JID instruction, and interrupt vectors for the VIS instruction). The program memory is addressed by the 15-bit program counter (PC). All interrupts in the device vector to program memory location 0FF Hex.

DATA MEMORY

The data memory address space includes the on-chip RAM and data registers, the I/O registers (Configuration, Data and Pin), the control registers, the MICROWIRE/PLUS SIO shift register, and the various registers, and counters associated with the timers (with the exception of the IDLE timer). Data memory is addressed directly by the instruction or indirectly by the B, X, SP pointers and S register.

The data memory consists of 128 bytes of RAM. Sixteen bytes of RAM are mapped as "registers" at addresses 0F0 to 0FF Hex. These registers can be loaded immediately, and also decremented and tested with the DRSZ (decrement register and skip if zero) instruction. The memory pointer registers X, B and SP are memory mapped into this space at address locations 0FC to 0FF Hex respectively, with the other registers being available for general usage.

The instruction set permits any bit in memory to be set, reset or tested. All I/O and registers (except A and PC) are memory mapped; therefore, I/O bits and register bits can be directly and individually set, reset and tested. The accumulator (A) bits can also be directly and individually tested.

Note: RAM contents are undefined upon power-up.

Reset

The $\overline{\text{RESET}}$ input when pulled low initializes the microcontroller. Initialization will occur whenever the $\overline{\text{RESET}}$ input is pulled low. Upon initialization, the data and configuration registers for ports L and G are cleared, resulting in these Ports being initialized to the TRI-STATE mode. Pin G1 of the G Port is an exception (as noted below) since pin G1 is dedicated as the WATCHDOG and/or Clock Monitor error output pin. Port D is set high. The PC, PSW, ICNTRL and CNTRL-control registers are cleared. The Comparator Select Register is cleared. The S register is initialized to zero. The Multi-Input Wakeup registers WKEN and WKEDG are cleared. Wakeup register WKPND is unknown. The stack pointer, SP, is initialized to 6F Hex.

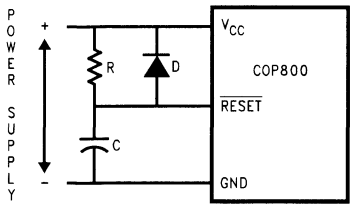
The device comes out of reset with both the WATCHDOG logic and the Clock Monitor detector armed, with the WATCHDOG service window bits set and the Clock Monitor bit set. The WATCHDOG and Clock Monitor circuits are inhibited during reset. The WATCHDOG service window bits being initialized high default to the maximum WATCHDOG service window of 64k t_C clock cycles. The Clock Monitor bit being initialized high will cause a Clock Monitor error following reset if the clock has not reached the minimum specified frequency at the termination of reset. A Clock Monitor error will cause an active low error output on pin G1. This error output will continue until 16 t_C-32 t_C clock cycles following the clock frequency reaching the minimum specified value, at which time the G1 output will enter the TRI-STATE mode.

The external RC network shown in *Figure 4* should be used to ensure that the RESET pin is held low until the power supply to the chip stabilizes.

Oscillator Circuits

The chip can be driven by a clock input on the CKI input pin which can be between DC and 10 MHz. The CKO output clock is on pin G7 (crystal configuration). The CKI input frequency is divided down by 10 to produce the instruction cycle clock (t_C).

Oscillator Circuits (Continued)



TL/DD/12865–6

RC > 5 x POWER SUPPLY RISE TIME

FIGURE 4. Recommended Reset Circuit

Figure 5 shows the Crystal and R/C Oscillator diagrams.

CRYSTAL OSCILLATOR

CKI and CKO can be connected to make a closed loop crystal (or resonator) controlled oscillator.

Table I shows the component values required for various standard crystal values.

TABLE I. Crystal Oscillator Configuration, T_A = 25°C

R1 (kΩ)	R2 (MΩ)	C1 (pF)	C2 (pF)	CKI Freq (MHz)	Conditions
0	1	30	30–36	10	V_{CC} = 5V
0	1	30	30–36	4	V_{CC} = 5V
0	1	200	100–150	0.455	V_{CC} = 5V

R/C OSCILLATOR

By selecting CKI as a single pin oscillator input, a single pin R/C oscillator circuit can be connected to it. CKO is available as a general purpose input, and/or HALT restart input.

Note: Use of the R/C oscillator option will result in higher electromagnetic emissions.

Table II shows the variation in the oscillator frequencies as functions of the component (R and C) values.

TABLE II. RC Oscillator Configuration, T_A = 25°C

R (kΩ)	C (pF)	CKI Freq (MHz)	Instr. Cycle (μs)	Conditions
3.3	82	2.2 to 2.7	3.7 to 4.6	V_{CC} = 5V
5.6	100	1.1 to 1.3	7.4 to 9.0	V_{CC} = 5V
6.8	100	0.9 to 1.1	8.8 to 10.8	V_{CC} = 5V

Note: 3k ≤ R ≤ 200k
50 pF ≤ C ≤ 200 pF

Control Registers

CNTRL Register (Address X'00EE)

The Timer1 (T1) and MICROWIRE/PLUS control register contains the following bits:

SL1 and SL0 Select the MICROWIRE/PLUS clock divide by (00 = 2, 01 = 4, 1x = 8)

IEDG External interrupt edge polarity select (0 = Rising edge, 1 = Falling edge)

MSEL Selects G5 and G4 as MICROWIRE/PLUS signals SK and SO respectively

T1C0 Timer T1 Start/Stop control in timer modes 1 and 2. T1 Underflow Interrupt Pending Flag in timer mode 3

T1C1 Timer T1 mode control bit

T1C2 Timer T1 mode control bit

T1C3 Timer T1 mode control bit

T1C3	T1C2	T1C1	T1C0	MSEL	IEDG	SL1	SL0

Bit 7 Bit 0

PSW Register (Address X'00EF)

The PSW register contains the following select bits:

GIE Global interrupt enable (enables interrupts)

EXEN Enable external interrupt

BUSY MICROWIRE/PLUS busy shifting flag

EXPND External interrupt pending

T1ENA Timer T1 Interrupt Enable for Timer Underflow or T1A Input capture edge

T1PNDA Timer T1 Interrupt Pending Flag (Autoreload RA in mode 1, T1 Underflow in Mode 2, T1A capture edge in mode 3)

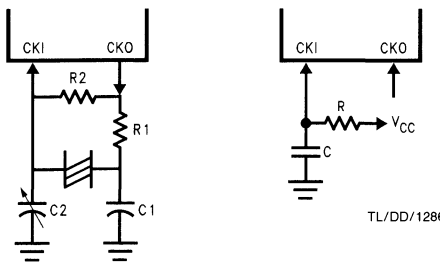
C Carry Flag

HC Half Carry Flag

HC	C	T1PNDA	T1ENA	EXPND	BUSY	EXEN	GIE

Bit 7 Bit 0

The Half-Carry flag is also affected by all the instructions that affect the Carry flag. The SC (Set Carry) and RC (Reset Carry) instructions will respectively set or clear both the carry flags. In addition to the SC and RC instructions, ADC, SUBC, RRC and RLC instructions affect the Carry and Half Carry flags.



TL/DD/12865–19

TL/DD/12865–7

FIGURE 5. Crystal and R/C Oscillator Diagrams

Control Registers (Continued)

ICNTRL Register (Address X'00E8)

The ICNTRL register contains the following bits:

T1ENB Timer T1 Interrupt Enable for T1B Input capture edge

T1PNDB Timer T1 Interrupt Pending Flag for T1B capture edge

μWEN Enable MICROWIRE/PLUS interrupt

μWPND MICROWIRE/PLUS interrupt pending

T0EN Timer T0 Interrupt Enable (Bit 12 toggle)

T0PND Timer T0 Interrupt pending

LPEN L Port Interrupt Enable (Multi-Input Wakeup/Interrupt)

Bit 7 could be used as a flag

Unused	LPEN	T0PND	T0EN	WPND	WEN	T1PNDB	T1ENB

Bit 7 Bit 0

CAPCNTL Register (Address X'00)

The CAPCNTL register contains the following bits:

CAPIEN Capture Interrupts enable

CAPPND Capture pending

CAPOVL Capture Timer overflow

CAPRUN Capture Timer Run

CAPMOD Reset Timer

Unused	CAPMOD	CAPRUN	CAPOVL	CAPPND	CAPIEN

Bit 7 Bit 0

Timers

The device contains a very versatile set of timers (T0 and T1). All timers and associated autoreload/capture registers power up containing random data.

TIMER T0 (IDLE TIMER)

The device supports applications that require maintaining real time and low power with the IDLE mode. This IDLE mode support is furnished by the IDLE timer T0, which is a 16-bit timer. The Timer T0 runs continuously at the fixed rate of the instruction cycle clock, t_C. The user cannot read or write to the IDLE Timer T0, which is a count down timer.

The Timer T0 supports the following functions:

- Exit out of the Idle Mode (See Idle Mode description)
- WATCHDOG logic (See WATCHDOG description)
- Start up delay out of the HALT mode

Figure 6 is a functional block diagram showing the structure of the IDLE Timer and its associated interrupt logic.

Bits 11 through 15 of the ITMR register can be selected for triggering the IDLE Timer interrupt. Each time the selected bit underflows (every 4k, 8k, 16k, 32k or 64k instruction cycles), the IDLE Timer interrupt pending bit T0PND is set, thus generating an interrupt (if enabled), and bit 6 of the Port G data register is reset, thus causing an exit from the IDLE mode if the device is in that mode.

In order for an interrupt to be generated, the IDLE Timer interrupt enable bit T0EN must be set, and the GIE (Global Interrupt Enable) bit must also be set. The T0PND flag and T0EN bit are bits 5 and 4 of the ICNTRL register, respectively. The interrupt can be used for any purpose. Typically, it is used to perform a task upon exit from the IDLE mode. For more information on the IDLE mode, refer to the Power Save Modes section.

The Idle Timer period is selected by bits 0–2 of the ITMR register Bits 3–7 of the ITMR Register are reserved and should not be used as software flags.

TABLE III. Idle Timer Window Length

ITSEL2	ITSEL1	ITSEL0	Idle Timer Period (Instruction Cycles)
0	0	0	4,096
0	0	1	8,192
0	1	0	16,384
0	1	1	32,768
1	X	X	65,536

The ITMR register is cleared on Reset and the Idle Timer period is reset to 4,096 instruction cycles.

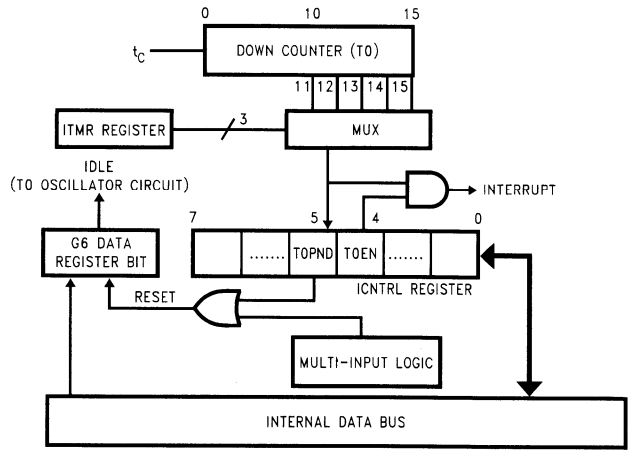
ITMR Register (Address X'0xCF)

Reserved	ITSEL2	ITSEL1	ITSEL0

Bit 7 Bit 0

Any time the IDLE Timer period is changed there is the possibility of generating a spurious IDLE Timer interrupt by setting the T0PND bit. The user is advised to disable IDLE Timer interrupts prior to changing the value of the ITSEL bits of the ITMR Register and then clear the T0PND bit before attempting to synchronize operation to the IDLE Timer.

Timers (Continued)



TL/DD/12865–8

FIGURE 6. Functional Block Diagram for Idle Timer T0

TIMER T1

The device has a powerful timer/counter block. The timer consists of a 16-bit timer, T1, and two supporting 16-bit autoreload/capture registers, R1A and R1B. The timer block has two pins associated with it, T1A and T1B. The pin T1A supports I/O required by the timer block, while the pin T1B is an input to the timer block. The powerful and flexible timer block allows the device to easily perform all timer functions with minimal software overhead. The timer block has three operating modes: Processor Independent PWM mode, External Event Counter mode, and Input Capture mode.

The control bits T1C3, T1C2, and T1C1 allow selection of the different modes of operation.

Mode 1. Processor Independent PWM Mode

As the name suggests, this mode allows the device to generate a PWM signal with very minimal user intervention. The user only has to define the parameters of the PWM signal (ON time and OFF time). Once begun, the timer block will continuously generate the PWM signal completely independent of the microcontroller. The user software services the timer block only when the PWM parameters require updating.

In this mode the timer T1 counts down at a fixed rate of t_C. Upon every underflow the timer is alternately reloaded with the contents of supporting registers, R1A and R1B. The very first underflow of the timer causes the timer to reload from the register R1A. Subsequent underflows cause the timer to be reloaded from the registers alternately beginning with the register R1B.

The T1 Timer control bits, T1C3, T1C2 and T1C1 set up the timer for PWM mode operation.

Figure 7 shows a block diagram of the timer in PWM mode.

The underflows can be programmed to toggle the T1A output pin. The underflows can also be programmed to generate interrupts.

Underflows from the timer are alternately latched into two pending flags, T1PNDA and T1PNDB. The user must reset these pending flags under software control. Two control enable flags, T1ENA and T1ENB, allow the interrupts from the timer underflow to be enabled or disabled. Setting the timer enable flag T1ENA will cause an interrupt when a timer underflow causes the R1A register to be reloaded into the timer. Setting the timer enable flag T1ENB will cause an interrupt when a timer underflow causes the R1B register to be reloaded into the timer. Resetting the timer enable flags will disable the associated interrupts.

Either or both of the timer underflow interrupts may be enabled. This gives the user the flexibility of interrupting once per PWM period on either the rising or falling edge of the PWM output. Alternatively, the user may choose to interrupt on both edges of the PWM output.

Timers (Continued)

Mode 2. External Event Counter Mode

This mode is quite similar to the processor independent PWM mode previously described. The main difference is that the timer, T1, is clocked by the input signal from the T1A pin. The T1 timer control bits, T1C3, T1C2 and T1C1 allow the timer to be clocked either on a positive or negative edge from the T1A pin. Underflows from the timer are latched into the T1PNDA pending flag. Setting the T1ENA control flag will cause an interrupt when the timer underflows.

In this mode the input pin T1B can be used as an independent positive edge sensitive interrupt input if the T1ENB control flag is set. The occurrence of a positive edge on the T1B input pin is latched into the T1PNDB flag.

Figure 8 shows a block diagram of the timer in External Event Counter mode.

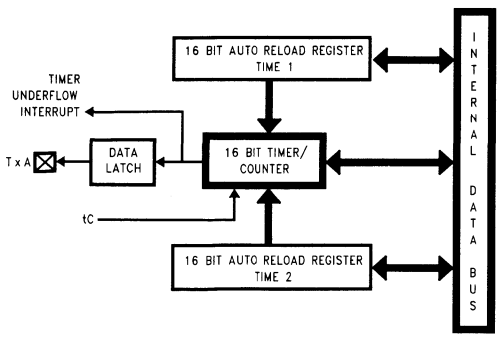
Note: The PWM output is not available in this mode since the T1A pin is being used as the counter input clock.

Mode 3. Input Capture Mode

The device can precisely measure external frequencies or time external events by placing the timer block, T1, in the input capture mode.

In this mode, the timer T1 is constantly running at the fixed t_C rate. The two registers, R1A and R1B, act as capture registers. Each register acts in conjunction with a pin. The register R1A acts in conjunction with the T1A pin and the register R1B acts in conjunction with the T1B pin.

The timer value gets copied over into the register when a trigger event occurs on its corresponding pin. Control bits, T1C3, T1C2 and T1C1, allow the trigger events to be specified either as a positive or a negative edge. The trigger condition for each input pin can be specified independently.

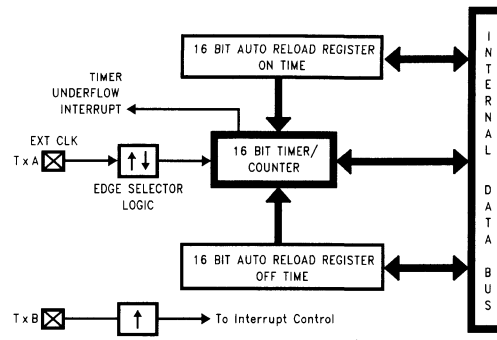


TL/DD/12865–9

FIGURE 7. Timer in PWM Mode

The trigger conditions can also be programmed to generate interrupts. The occurrence of the specified trigger condition on the T1A and T1B pins will be respectively latched into the pending flags, T1PNDA and T1PNDB. The control flag T1ENA allows the interrupt on T1A to be either enabled or disabled. Setting the T1ENA flag enables interrupts to be generated when the selected trigger condition occurs on the T1A pin. Similarly, the flag T1ENB controls the interrupts from the T1B pin.

Underflows from the timer can also be programmed to generate interrupts. Underflows are latched into the timer T1C0 pending flag (the T1C0 control bit serves as the timer underflow interrupt pending flag in the Input Capture mode). Consequently, the T1C0 control bit should be reset when entering the Input Capture mode. The timer underflow interrupt is enabled with the T1ENA control flag. When a T1A interrupt occurs in the Input Capture mode, the user must check both the T1PNDA and T1C0 pending flags in order to determine whether a T1A input capture or a timer underflow (or both) caused the interrupt.



TL/DD/12865–10

FIGURE 8. Timer in External Event Counter Mode

Timers (Continued)

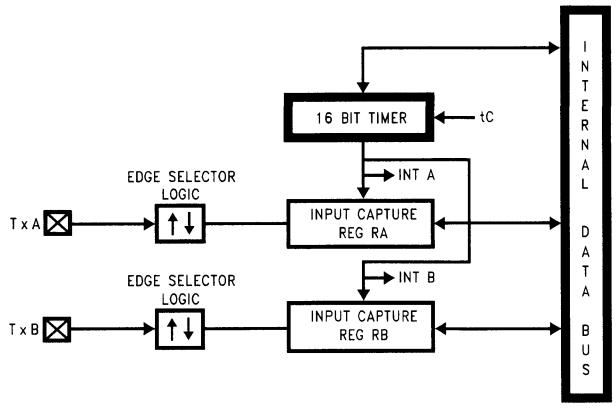


FIGURE 9. Timer in Input Capture Mode

Figure 9 shows a block diagram of the timer in Input Capture mode.

TIMER CONTROL FLAGS

The Timer T1 control bits and their functions are summarized below.

T1C0 Timer Start/Stop control in Modes 1 and 2 (Processor Independent PWM and External Event Counter), where 1 = Start, 0 = Stop Timer Underflow Interrupt Pending Flag in Mode 3 (Input Capture)

T1PNDA Timer Interrupt Pending Flag

T1PNDB Timer Interrupt Pending Flag

T1ENA Timer Interrupt Enable Flag

T1ENB Timer Interrupt Enable Flag

1 = Timer Interrupt Enabled

0 = Timer Interrupt Disabled

T1C3 Timer mode control

T1C2 Timer mode control

T1C1 Timer mode control

HIGH SPEED CAPTURE TIMER

The device provides a 16-bit high-speed capture timer. The timer consists of a 16-bit up-counter that is clocked with the device clock input frequency (CKI) and an 8-bit control register. The 16-bit counter is mapped as two read/write 8-bit registers. This timer is specifically designed to be used in conjunction with the Analog Function Block (comparator, analog multiplexer, constant current source) to implement a low-cost, high-resolution, single-slope A/D.

The timer is automatically stopped in the event of a capture to allow the software to read the timer value. Coming out of reset the counter is disabled (stopped) and reads all "0".

Setting the Capture Timer Run bit CAPRUN bit in the Capture Control Register (CAPCNTL) will start the counter. The counter will count up until a capture event (negative edge) is received. Upon a capture the counter will be stopped, the Capture Pending bit (CAPPND) is set, and the CAPRUN bit is automatically reset. If capture interrupts are enabled (CAPIEN=1), the capture event will generate an interrupt. Setting the CAPRUN bit again by software will start a new counting cycle. If the Capture Mode bit is reset (CAPMOD=0) the capture timer will be automatically initialized to all "0" with each setting of the CAPRUN bit. If CAPMOD=1 the timer will not be cleared when setting the CAPRUN bit, thus allowing the user's software to pre-load the timer registers with any desired value. This mode can be used in conjunction with the timer's overflow to implement for example a programmable delay counter.

Timers (Continued)

The timer mode control bits (T1C3, T1C2 and T1C1) are detailed below:

TABLE IV. Timer Mode Control

T1C3	T1C2	T1C1	Timer Mode	Interrupt A Source	Interrupt B Source	Timer Counts On
0	0	0	MODE 2 (External Event Counter)	Timer Underflow	Pos. T1B Edge	T1A Pos. Edge
0	0	1	MODE 2 (External Event Counter)	Timer Underflow	Pos. T1B Edge	T1A Neg. Edge
1	0	1	MODE 1 (PWM) T1A Toggle	Autoreload RA	Autoreload RB	t_C
1	0	0	MODE 1 (PWM) No T1A Toggle	Autoreload RA	Autoreload RB	t_C
0	1	0	MODE 3 (Capture) Captures: T1A Pos. Edge T1B Pos. Edge	Pos. T1A Edge Timer Underflow	Pos. T1B Edge	t_C
1	1	0	MODE 3 (Capture) Captures: T1A Pos. Edge T1B Neg. Edge	Pos. T1A Edge or Timer Underflow	Neg. T1B Edge	t_C
0	1	1	MODE 3 (Capture) Captures: T1A Neg. Edge T1B Pos. Edge	Neg. T1A Edge or Timer Underflow	Pos. T1B Edge	t_C
1	1	1	MODE 3 (Capture) Captures: T1A Neg. Edge T1B Neg. Edge	Neg. T1A Edge or Timer Underflow	Neg. T1B Edge	t_C

"CAPTURE MODE" is only active when the CAPRUN bit is set, i.e. any capture events received while the timer is stopped (CAPRUN = 0) will be ignored and will not cause the CAPPND bit to be set. The capture counter can also be stopped (frozen) by the user's software resetting the CAPRUN bit.

If the user program tries to set the CAPRUN bit at the same time that the hardware gets a capture event and tries to reset the CAPRUN bit, the hardware will have precedence.

Should the counter overflow before a capture condition occurs, the Capture Overflow bit (CAPOVL) bit in the CAPCNTL register will be set. If Capture interrupts are enabled (CAPIEN = 1) an overflow will generate an interrupt. The user software should reset this bit before the next overflow occurs, otherwise subsequent overflow conditions cannot be detected.

Capture Overflow interrupt and Capture Pending interrupt share the same interrupt vector.

Power Save Modes

The device offers the user two power save modes of operation: HALT and IDLE. In the HALT mode, all microcontroller activities are stopped. In the IDLE mode, the on-board oscillator circuitry and timer T0 are active but all other microcontroller activities are stopped. In either mode, all on-board RAM, registers, I/O states, and timers (with the exception of T0) are unaltered.

HALT MODE

The device can be placed in the HALT mode by writing a "1" to the HALT flag (G7 data bit). All microcontroller activities, including the clock and timers, are stopped. The WATCHDOG logic on the device is disabled during the HALT mode. However, the clock monitor circuitry, if enabled, remains active and will cause the WATCHDOG output pin (WDOUT) to go low. If the HALT mode is used and the user does not want to activate the WDOUT pin, the Clock Monitor should be disabled after the device comes out of reset (resetting the Clock Monitor control bit with the first write to the WDSVR register). In the HALT mode, the power requirements of the device are minimal and the applied voltage (V_{CC}) may be decreased to V_r (V_r = 2.0V) without altering the state of the machine.

The device supports three different ways of exiting the HALT mode. The first method of exiting the HALT mode is with the Multi-Input Wakeup feature on the Port L.

The second method is with a low to high transition on the CKO (G7) pin. This method precludes the use of the crystal clock configuration (since CKO becomes a dedicated output), and so may only be used with an RC clock configuration. The third method of exiting the HALT mode is by pulling the $\overline{\text{RESET}}$ pin low.

Since a crystal or ceramic resonator may be selected as the oscillator, the Wakeup signal is not allowed to start the chip running immediately since crystal oscillators and ceramic resonators have a delayed start up time to reach full ampli-

Power Save Modes (Continued)

tude and frequency stability. The IDLE timer is used to generate a fixed delay to ensure that the oscillator has indeed stabilized before allowing instruction execution. In this case, upon detecting a valid Wakeup signal, only the oscillator circuitry is enabled. The IDLE timer is loaded with a value of 256 and is clocked with the t_C instruction cycle clock. The t_C clock is derived by dividing the oscillator clock down by a factor of 10. The Schmitt trigger following the CKI inverter on the chip ensures that the IDLE timer is clocked only when the oscillator has a sufficiently large amplitude to meet the Schmitt trigger specifications. This Schmitt trigger is not part of the oscillator closed loop. The startup timeout from the IDLE timer enables the clock signals to be routed to the rest of the chip.

If an RC clock option is being used, the fixed delay is introduced optionally. A control bit, CLKDLY, mapped as configuration bit G7, controls whether the delay is to be introduced or not. The delay is included if CLKDLY is set, and excluded if CLKDLY is reset. The CLKDLY bit is cleared on reset.

The device has two mask options associated with the HALT mode. The first mask option enables the HALT mode feature, while the second mask option disables the HALT mode. With the HALT mode enable mask option, the device will enter and exit the HALT mode as described above. With the HALT disable mask option, the device cannot be placed in the HALT mode (writing a "1" to the HALT flag will have no effect, the HALT flag will remain "0").

IDLE MODE

The device is placed in the IDLE mode by writing a "1" to the IDLE flag (G6 data bit). In this mode, all activities, except the associated on-board oscillator circuitry and the IDLE Timer T0, are stopped.

As with the HALT mode, the device can be returned to normal operation with a reset, or with a Multi-Input Wakeup from the L Port. Alternately, the microcontroller resumes normal operation from the IDLE mode when the thirteenth bit (representing 4.096 ms at internal clock frequency of 10 MHz, t_C = 1 μs) of the IDLE Timer toggles.

This toggle condition of the thirteenth bit of the IDLE Timer T0 is latched into the T0PND pending flag.

The user has the option of being interrupted with a transition on the thirteenth bit of the IDLE Timer T0. The interrupt can be enabled or disabled via the T0EN control bit. Setting the T0EN flag enables the interrupt and vice versa.

The user can enter the IDLE mode with the Timer T0 interrupt enabled. In this case, when the T0PND bit gets set, the device will first execute the Timer T0 interrupt service routine and then return to the instruction following the "Enter Idle Mode" instruction.

Alternatively, the user can enter the IDLE mode with the IDLE Timer T0 interrupt disabled. In this case, the device will resume normal operation with the instruction immediately following the "Enter IDLE Mode" instruction.

Note: It is necessary to program two NOP instructions following both the set HALT mode and set IDLE mode instructions. These NOP instructions are necessary to allow clock resynchronization following the HALT or IDLE modes.

Multi-Input Wakeup

The Multi-Input Wakeup feature is used to return (wakeup) the device from either the HALT or IDLE modes. Alternately Multi-Input Wakeup/Interrupt feature may also be used to generate up to 4 edge selectable external interrupts.

Figure 10 shows the Multi-Input Wakeup logic.

The Multi-Input Wakeup feature utilizes the L Port. The user selects which particular L port bit (or combination of L Port bits) will cause the device to exit the HALT or IDLE modes. The selection is done through the register WKEN. The register WKEN is an 8-bit read/write register, which contains a control bit for every L port bit. Setting a particular WKEN bit enables a Wakeup from the associated L port pin.

The user can select whether the trigger condition on the selected L Port pin is going to be either a positive edge (low to high transition) or a negative edge (high to low transition). This selection is made via the register WKEDG, which is an 8-bit control register with a bit assigned to each L Port pin. Setting the control bit will select the trigger condition to be a negative edge on that particular L Port pin. Resetting the bit selects the trigger condition to be a positive edge. Changing an edge select entails several steps in order to avoid a Wakeup condition as a result of the edge change. First, the associated WKEN bit should be reset, followed by the edge select change in WKEDG. Next, the associated WKPND bit should be cleared, followed by the associated WKEN bit being re-enabled.

An example may serve to clarify this procedure. Suppose we wish to change the edge select from positive (low going high) to negative (high going low) for L Port bit 5, where bit 5 has previously been enabled for an input interrupt. The program would be as follows:

```
RBIT 5, WKEN
SBIT 5, WKEDG
RBIT 5, WKPND
SBIT 5, WKEN
```

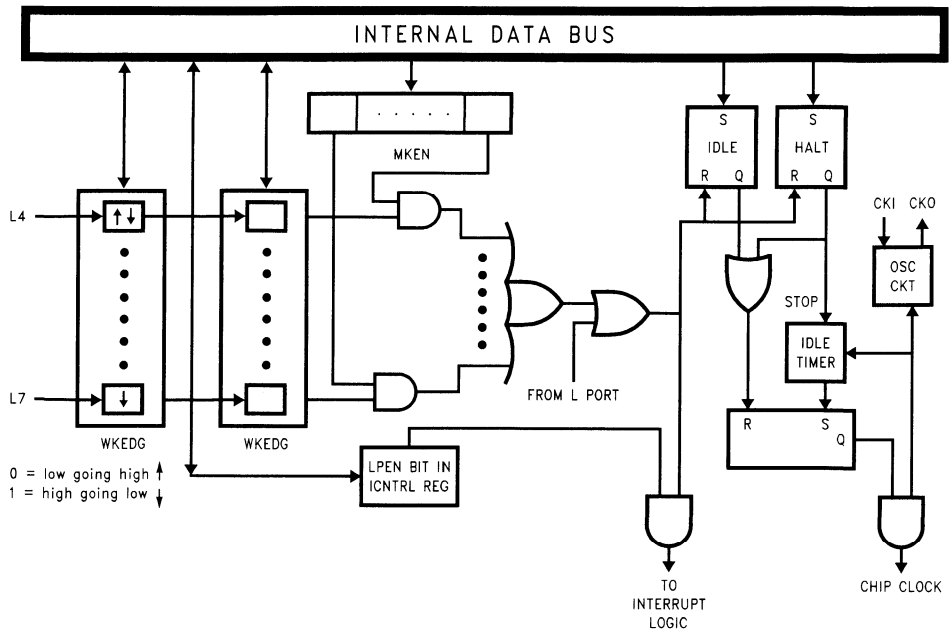
If the L port bits have been used as outputs and then changed to inputs with Multi-Input Wakeup/Interrupt, a safety procedure should also be followed to avoid wakeup conditions. After the selected L port bits have been changed from output to input but before the associated WKEN bits are enabled, the associated edge select bits in WKEDG should be set or reset for the desired edge selects, followed by the associated WKPND bits being cleared.

This same procedure should be used following reset, since the L port inputs are left floating as a result of reset.

The occurrence of the selected trigger condition for Multi-Input Wakeup is latched into a pending register called WKPND. The respective bits of the WKPND register will be set on the occurrence of the selected trigger edge on the corresponding Port L pin. The user has the responsibility of clearing these pending flags. Since WKPND is a pending register for the occurrence of selected wakeup conditions, the device will not enter the HALT mode if any Wakeup bit is both enabled and pending. Consequently, the user must clear the pending flags before attempting to enter the HALT mode.

WKEN, WKPND and WKEDG are all read/write registers, and are cleared at reset.

Multi-Input Wakeup (Continued)



TL/DD/12865–12

FIGURE 10. Multi-Input Wake Up Logic

PORT L INTERRUPTS

Port L provides the user with an additional eight fully selectable, edge sensitive interrupts which are all vectored into the same service subroutine.

The interrupt from Port L shares logic with the wake up circuitry. The register WKEN allows interrupts from Port L to be individually enabled or disabled. The register WKEDG specifies the trigger condition to be either a positive or a negative edge. Finally, the register WKPND latches in the pending trigger conditions.

The GIE (Global Interrupt Enable) bit enables the interrupt function.

A control flag, LPEN, functions as a global interrupt enable for Port L interrupts. Setting the LPEN flag will enable interrupts and vice versa. A separate global pending flag is not needed since the register WKPND is adequate.

Since Port L is also used for waking the device out of the HALT or IDLE modes, the user can elect to exit the HALT or IDLE modes either with or without the interrupt enabled. If he elects to disable the interrupt, then the device will restart execution from the instruction immediately following the instruction that placed the microcontroller in the HALT or IDLE modes. In the other case, the device will first execute the interrupt service routine and then revert to normal operation. (See HALT MODE for clock option wakeup information.)

Analog Function Block

This device contains an analog function block with the intent to provide a function which allows for single slope, low cost, A/D conversion of up to 6 channels.

CMPSL REGISTER (ADDRESS X'00B7)

The CMPSL register contains the following bits:

CMPNEG — Will drive I1 to a low level. This bit can be used to discharge an external capacitor. This bit is disabled if the comparator is not enabled (CMPEN = 0).

CMPEN — Enable the comparator ("1" = enable)

CSEN — Enables the internal constant current source. This current source provides a nominal 20 μA constant current at the I1 pin. This current can be used to ensure a linear charging rate on an external capacitor. This bit has no affect and the current source is disabled if the comparator is not enabled (CMPEN = 0).

CMPOE — Enables the comparator output to either pin I3 or pin I7 ("1" = enable) depending on the value of CMPISEL0/1/2.

CMPISEL0/1/2 — Will select one of seven possible sources (I0/I2/I3/I4/I5/I6/internal reference) as a positive input to the comparator (see Table V for more information.)

CMPT2B — Selects the timer T2B input to be driven directly by the comparator output. If the comparator is disabled (CMPEN = 0), this function is disabled, i.e. the T2B input is connected to Port L5.

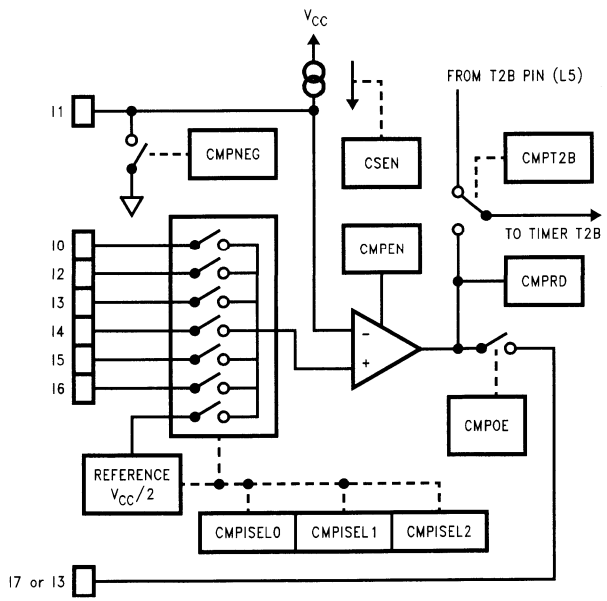
CMPT2B	CMPISEL2	CMPISEL1	CMPISEL0	CMPOE	CSEN	CMPEN	CMPNEG

Bit 7 Bit 0

The Comparator Select Register is cleared on RESET (the comparator is disabled). To save power the program should also disable the comparator before the μC enters the HALT/IDLE modes. Disabling the comparator will turn off the constant current source and the $V_{CC}/2$ reference, disconnect the comparator output from the T2B input and pin I3 or I7 and remove the low on I1 caused by CMPNEG.

It is often useful for the user's program to read the result of a comparator operation. Since I1 is always selected to be COMPIN—when the comparator is enabled (CMPEN = 1), the comparator output can be read internally by reading bit 1 (CMPRD) of register PORTI (RAM address 0xD7).

The following table lists the comparator inputs and outputs versus the value of the CMPISEL0/1/2 bits. The output will only be driven if the CMPOE bit is set to 1.



TL/DD/12865-13

FIGURE 11. COP884CT Analog Function Block

Analog Function Block (Continued)

TABLE V. Comparator Input Selection

Control Bit			Comparator Input Source		Comparator Output
CMPISEL2	CMPISEL1	CMPISEL0	Neg. Input	Pos. Input	
0	0	0	I1	I2	I3
0	0	1	I1	I2	I7
0	1	0	I1	I3	I7
0	1	1	I1	I0	I7
1	0	0	I1	I4	I7
1	0	1	I1	I5	I7
1	1	0	I1	I6	I7
1	1	1	I1	$V_{CC}/2$ Ref.	I7

Reset

The state of the Comparator Block immediately after RESET is as follows:

1. The CMPSL Register is set to all zeros
2. The Comparator is disabled
3. The Constant Current Source is disabled
4. CMPNEG is turned off
5. The Port I inputs are electrically isolated from the comparator
6. The T2B input is as normally selected by the T2CNTRL Register
7. CMPISEL0–CMPISEL2 are set to zero
8. All Port I inputs are selected to the default digital input mode

The comparator outputs have the same specification as Ports L and G except that the rise and fall times are symmetrical.

Interrupts

The device supports a vectored interrupt scheme. It supports a total of twelve interrupt sources. The following table lists all the possible device interrupt sources, their arbitration rankings and the memory locations reserved for the interrupt vector for each source.

Two bytes of program memory space are reserved for each interrupt source. All interrupt sources except the software interrupt are maskable. Each of the maskable interrupts have an Enable bit and one or more Pending bits. A maskable interrupt is active if its associated enable and pending bits are set. If GIE = 1 and an interrupt is active, then the processor will be interrupted as soon as it is ready to start executing an instruction except if the above conditions happen during the Software Trap service routine. This exception is described in the Software Trap sub-section.

The interruption process is accomplished with the INTR instruction (opcode 00), which is jammed inside the Instruction Register and replaces the opcode about to be executed. The following steps are performed for every interrupt:

1. The GIE (Global Interrupt Enable) bit is reset.
2. The address of the instruction about to be executed is pushed into the stack.
3. The PC (Program Counter) branches to address 00FF.

This procedure takes 7 t_C cycles to execute.

TABLE VI. Interrupt Vector Table

ARBITRATION RANKING	SOURCE DESCRIPTION		VECTOR* ADDRESS (Hi-Low Byte)
(1) Highest	Software	INTR Instruction	0yFE–0yFF
(2)	Reserved		0yFC–0yFD
(3)	External	G0	0yFA–0yFB
(4)	Timer T0	Idle Timer	0yF8–0yF9
(5)	Timer T1	T1A/Underflow	0yF6–0yF7
(6)	Timer T1	T1B	0yF4–0yF5
(7)	Microwire/Plus	Busy Low	0yF2–0yF3
(8)	Reserved		0yF0–0yF1
(9)	Reserved		0yEE–0yEF
(10)	Reserved		0yEC–0yED
(11)	High Speed Capture Timer	Capture Overflow/ Capture Pending	0yEA–0yEB
(12)	Reserved		0yE8–0yE9
(13)	Reserved		0yE6–0yE7
(14)	Reserved		0yE4–0yE5
(15)	Port L/Wakeup	Port L Edge	0yE2–0yE3
(16) Lowest	Default VIS	Reserved	0yE0–0yE1

*y is a variable which represents the VIS block. VIS and the vector table must be located in the same 256-byte block except if VIS is located at the last address of a block. In this case, the table must be in the next block.

Interrupts (Continued)

At this time, since GIE = 0, other maskable interrupts are disabled. The user is now free to do whatever context switching is required by saving the context of the machine in the stack with PUSH instructions. The user would then program a VIS (Vector Interrupt Select) instruction in order to branch to the interrupt service routine of the highest priority interrupt enabled and pending at the time of the VIS. Note that this is not necessarily the interrupt that caused the branch to address location 00FF Hex prior to the context switching.

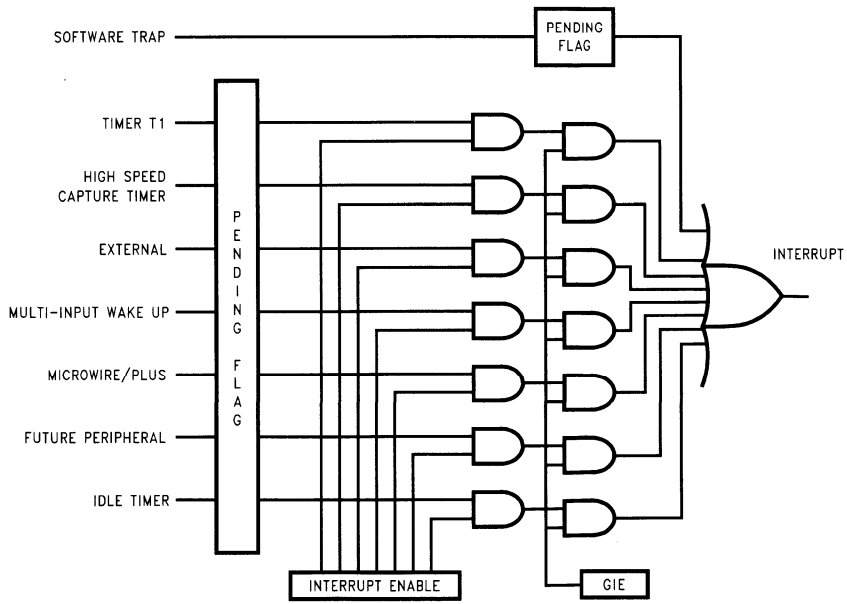
Thus, if an interrupt with a higher rank than the one which caused the interruption becomes active before the decision of which interrupt to service is made by the VIS, then the interrupt with the higher rank will override any lower ones and will be acknowledged. The lower priority interrupt(s) are still pending, however, and will cause another interrupt immediately following the completion of the interrupt service routine associated with the higher priority interrupt just serviced. This lower priority interrupt will occur immediately following the RETI (Return from Interrupt) instruction at the end of the interrupt service routine just completed.

Inside the interrupt service routine, the associated pending bit has to be cleared by software. The RETI (Return from Interrupt) instruction at the end of the interrupt service routine will set the GIE (Global Interrupt Enable) bit, allowing the processor to be interrupted again if another interrupt is active and pending.

The VIS instruction looks at all the active interrupts at the time it is executed and performs an indirect jump to the beginning of the service routine of the one with the highest rank.

The addresses of the different interrupt service routines, called vectors, are chosen by the user and stored in ROM in a table starting at 01E0 (assuming that VIS is located between 00FF and 01DF). The vectors are 15-bit wide and therefore occupy 2 ROM locations.

VIS and the vector table must be located in the same 256-byte block (0y00 to 0yFF) except if VIS is located at the last address of a block. In this case, the table must be in the next block. The vector table cannot be inserted in the first 256-byte block (y ≠ 0).



TL/DD/12865–14

FIGURE 12. Interrupt Block Diagram

Interrupts (Continued)

The vector of the maskable interrupt with the lowest rank is located at 0yE0 (Hi-Order byte) and 0yE1 (Lo-Order byte) and so forth in increasing rank number. The vector of the maskable interrupt with the highest rank is located at 0yFA (Hi-Order byte) and 0yFB (Lo-Order byte).

The Software Trap has the highest rank and its vector is located at 0yFE and 0yFF.

If, by accident, a VIS gets executed and no interrupt is active, then the PC (Program Counter) will branch to a vector located at 0yE0-0yE1.

WARNING

A Default VIS interrupt handler routine must be present. As a minimum, this handler should confirm that the GIE bit is cleared (this indicates that the interrupt sequence has been taken), take care of any required housekeeping, restore context and return. Some sort of Warm Restart procedure should be implemented. These events can occur without any error on the part of the system designer or programmer.

Note: There is always the possibility of an interrupt occurring during an instruction which is attempting to reset the GIE bit or any other interrupt enable bit. If this occurs when a single cycle instruction is being used to reset the interrupt enable bit, the interrupt enable bit will be reset but an interrupt may still occur. This is because interrupt processing is started at the same time as the interrupt bit is being reset. To avoid this scenario, the user should always use a two, three, or four cycle instruction to reset interrupt enable bits.

Figure 12 shows the Interrupt block diagram.

SOFTWARE TRAP

The Software Trap (ST) is a special kind of non-maskable interrupt which occurs when the INTR instruction (used to acknowledge interrupts) is fetched from ROM and placed inside the instruction register. This may happen when the PC is pointing beyond the available ROM address space or when the stack is over-popped.

When an ST occurs, the user can re-initialize the stack pointer and do a recovery procedure (similar to reset, but not necessarily containing all of the same initialization procedures) before restarting.

The occurrence of an ST is latched into the ST pending bit. The GIE bit is not affected and the ST pending bit (**not accessible by the user**) is used to inhibit other interrupts and to direct the program to the ST service routine with the VIS instruction. The RPND instruction is used to clear the software interrupt pending bit. This pending bit is also cleared on reset.

The ST has the highest rank among all interrupts.

Nothing (except another ST) can interrupt an ST being serviced.

WATCHDOG

The devices contain a WATCHDOG and clock monitor. The WATCHDOG is designed to detect the user program getting stuck in infinite loops resulting in loss of program control or "runaway" programs. The Clock Monitor is used to detect the absence of a clock or a very slow clock below a specified rate on the CKI pin.

The WATCHDOG consists of two independent logic blocks: WD UPPER and WD LOWER. WD UPPER establishes the upper limit on the service window and WD LOWER defines the lower limit of the service window.

Servicing the WATCHDOG consists of writing a specific value to a WATCHDOG Service Register named WDSVR which is memory mapped in the RAM. This value is composed of three fields, consisting of a 2-bit Window Select, a 5-bit Key Data field, and the 1-bit Clock Monitor Select field. Table VII shows the WDSVR register.

TABLE VII. WATCHDOG Service Register (WDSVR)

Window Select		Key Data					Clock Monitor
X	X	0	1	1	0	0	Y
7	6	5	4	3	2	1	0

The lower limit of the service window is fixed at 2048 instruction cycles. Bits 7 and 6 of the WDSVR register allow the user to pick an upper limit of the service window.

Table VIII shows the four possible combinations of lower and upper limits for the WATCHDOG service window. This flexibility in choosing the WATCHDOG service window prevents any undue burden on the user software.

Bits 5, 4, 3, 2 and 1 of the WDSVR register represent the 5-bit Key Data field. The key data is fixed at 01100. Bit 0 of the WDSVR Register is the Clock Monitor Select bit.

TABLE VIII. WATCHDOG Service Window Select

WDSVR Bit 7	WDSVR Bit 6	Service Window (Lower-Upper Limits)
0	0	2k–8k t_C Cycles
0	1	2k–16k t_C Cycles
1	0	2k–32k t_C Cycles
1	1	2k–64k t_C Cycles

Clock Monitor

The Clock Monitor aboard the device can be selected or deselected under program control. The Clock Monitor is guaranteed not to reject the clock if the instruction cycle clock ($1/t_C$) is greater or equal to 10 kHz. This equates to a clock input rate on CKI of greater or equal to 100 kHz.

WATCHDOG Operation

The WATCHDOG and Clock Monitor are disabled during reset. The device comes out of reset with the WATCHDOG armed, the WATCHDOG Window Select bits (bits 6, 7 of the WDSVR Register) set, and the Clock Monitor bit (bit 0 of the WDSVR Register) enabled. Thus, a Clock Monitor error will occur after coming out of reset, if the instruction cycle clock frequency has not reached a minimum specified value, including the case where the oscillator fails to start.

The WDSVR register can be written to only once after reset and the key data (bits 5 through 1 of the WDSVR Register) must match to be a valid write. This write to the WDSVR register involves two irrevocable choices: (i) the selection of

WATCHDOG Operation (Continued)

the WATCHDOG service window (ii) enabling or disabling of the Clock Monitor. Hence, the first write to WDSVR Register involves selecting or deselecting the Clock Monitor, select the WATCHDOG service window and match the WATCHDOG key data. Subsequent writes to the WDSVR register will compare the value being written by the user to the WATCHDOG service window value and the key data (bits 7 through 1) in the WDSVR Register. Table IX shows the sequence of events that can occur.

The user must service the WATCHDOG at least once before the upper limit of the service window expires. The WATCHDOG may not be serviced more than once in every lower limit of the service window. The user may service the WATCHDOG as many times as wished in the time period between the lower and upper limits of the service window. The first write to the WDSVR Register is also counted as a WATCHDOG service.

The WATCHDOG has an output pin associated with it. This is the WDOUT pin, on pin 1 of the port G. WDOUT is active low. The WDOUT pin is in the high impedance state in the inactive state. Upon triggering the WATCHDOG, the logic will pull the WDOUT (G1) pin low for an additional 16 t_C–32 t_C cycles after the signal level on WDOUT pin goes below the lower Schmitt trigger threshold. After this delay, the device will stop forcing the WDOUT output low. The WATCHDOG service window will restart when the WDOUT pin goes high. It is recommended that the user tie the WDOUT pin back to V_{CC} through a resistor in order to pull WDOUT high.

A WATCHDOG service while the WDOUT signal is active will be ignored. The state of the WDOUT pin is not guaranteed on reset, but if it powers up low then the WATCHDOG will time out and WDOUT will enter high impedance state.

The Clock Monitor forces the G1 pin low upon detecting a clock frequency error. The Clock Monitor error will continue until the clock frequency has reached the minimum specified value, after which the G1 output will enter the high impedance TRI-STATE mode following 16 t_C–32 t_C clock cycles. The Clock Monitor generates a continual Clock Monitor error if the oscillator fails to start, or fails to reach the minimum specified frequency. The specification for the Clock Monitor is as follows:

$1/t_C > 10$ kHz—No clock rejection.
$1/t_C < 10$ Hz—Guaranteed clock rejection.

WATCHDOG AND CLOCK MONITOR SUMMARY

The following salient points regarding the WATCHDOG and CLOCK MONITOR should be noted:

- Both the WATCHDOG and CLOCK MONITOR detector circuits are inhibited during RESET.
- Following RESET, the WATCHDOG and CLOCK MONITOR are both enabled, with the WATCHDOG having the maximum service window selected.
- The WATCHDOG service window and CLOCK MONITOR enable/disable option can only be changed once, during the initial WATCHDOG service following RESET.
- The initial WATCHDOG service must match the key data value in the WATCHDOG Service register WDSVR in order to avoid a WATCHDOG error.
- Subsequent WATCHDOG services must match all three data fields in WDSVR in order to avoid WATCHDOG errors.
- The correct key data value cannot be read from the WATCHDOG Service register WDSVR. Any attempt to read this key data value of 01100 from WDSVR will read as key data value of all 0's.
- The WATCHDOG detector circuit is inhibited during both the HALT and IDLE modes.
- The CLOCK MONITOR detector circuit is active during both the HALT and IDLE modes. Consequently, the device inadvertently entering the HALT mode will be detected as a CLOCK MONITOR error (provided that the CLOCK MONITOR enable option has been selected by the program).
- With the single-pin R/C oscillator mask option selected and the CLKDLY bit reset, the WATCHDOG service window will resume following HALT mode from where it left off before entering the HALT mode.
- With the crystal oscillator mask option selected, or with the single-pin R/C oscillator mask option selected and the CLKDLY bit set, the WATCHDOG service window will be set to its selected value from WDSVR following HALT. Consequently, the WATCHDOG should not be serviced for at least 2048 instruction cycles following HALT, but must be serviced within the selected window to avoid a WATCHDOG error.
- The IDLE timer T0 is not initialized with RESET.
- The user can sync in to the IDLE counter cycle with an IDLE counter (T0) interrupt or by monitoring the T0PND flag. The T0PND flag is set whenever the thirteenth bit of the IDLE counter toggles (every 4096 instruction cycles). The user is responsible for resetting the T0PND flag.
- A hardware WATCHDOG service occurs just as the device exits the IDLE mode. Consequently, the WATCHDOG should not be serviced for at least 2048 instruction cycles following IDLE, but must be serviced within the selected window to avoid a WATCHDOG error.
- Following RESET, the initial WATCHDOG service (where the service window and the CLOCK MONITOR enable/disable must be selected) may be programmed anywhere within the maximum service window (65,536 instruction cycles) initialized by RESET. Note that this initial WATCHDOG service may be programmed within the initial 2048 instruction cycles without causing a WATCHDOG error.

TABLE IX. WATCHDOG Service Actions

Key Data	Window Data	Clock Monitor	Action
Match	Match	Match	Valid Service: Restart Service Window
Don't Care	Mismatch	Don't Care	Error: Generate WATCHDOG Output
Mismatch	Don't Care	Don't Care	Error: Generate WATCHDOG Output
Don't Care	Don't Care	Mismatch	Error: Generate WATCHDOG Output

Detection of Illegal Conditions

The device can detect various illegal conditions resulting from coding errors, transient noise, power supply voltage drops, runaway programs, etc.

Reading of undefined ROM gets zeros. The opcode for software interrupt is 00. If the program fetches instructions from undefined ROM, this will force a software interrupt, thus signaling that an illegal condition has occurred.

The subroutine stack grows down for each call (jump to subroutine), interrupt, or PUSH, and grows up for each return or POP. The stack pointer is initialized to RAM location 06F Hex during reset. Consequently, if there are more returns than calls, the stack pointer will point to addresses 070 and 071 Hex (which are undefined RAM). Undefined RAM from addresses 070 to 07F (Segment 0), and all other segments (i.e., Segments 4... etc.) is read as all 1's, which in turn will cause the program to return to address 7FFF Hex. This is an undefined ROM location and the instruction fetched (all 0's) from this location will generate a software interrupt signaling an illegal condition.

Thus, the chip can detect the following illegal conditions:

1. Executing from undefined ROM
2. Over "POP"ing the stack by having more returns than calls.

When the software interrupt occurs, the user can re-initialize the stack pointer and do a recovery procedure before restarting (this recovery program is probably similar to that following reset, but might not contain the same program initialization procedures). The recovery program should reset the software interrupt pending bit using the RPND instruction.

MICROWIRE/PLUS

MICROWIRE/PLUS is a serial synchronous communications interface. The MICROWIRE/PLUS capability enables the device to interface with any of National Semiconductor's MICROWIRE peripherals (i.e. A/D converters, display drivers, E2PROMs etc.) and with other microcontrollers which support the MICROWIRE interface. It consists of an 8 bit serial shift register (SIO) with serial data input (SI), serial data output (SO) and serial shift clock (SK). *Figure 13* shows a block diagram of the MICROWIRE/PLUS logic.

The shift clock can be selected from either an internal source or an external source. Operating the MICROWIRE/PLUS arrangement with the internal clock source is called the Master mode of operation. Similarly, operating the MICROWIRE/PLUS arrangement with an external shift clock is called the Slave mode of operation.

The CNTRL register is used to configure and control the MICROWIRE/PLUS mode. To use the MICROWIRE/PLUS, the MSEL bit in the CNTRL register is set to one. In the master mode, the SK clock rate is selected by the two bits, SL0 and SL1, in the CNTRL register. Table X details the different clock rates that may be selected.

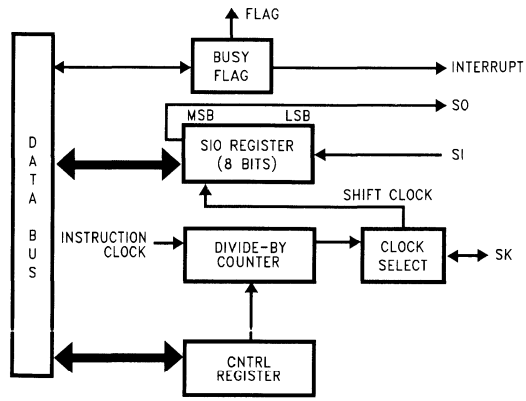
TABLE X. MICROWIRE/PLUS Master Mode Clock Select

SL1	SL0	SK period
0	0	2 X t_C
0	1	4 X t_C
1	x	8 X t_C

Where t_C is the instruction cycle clock

MICROWIRE/PLUS OPERATION

Setting the BUSY bit in the PSW register causes the MICROWIRE/PLUS to start shifting the data. It gets reset when eight data bits have been shifted. The user may reset the BUSY bit by software to allow less than 8 bits to shift. If enabled, an interrupt is generated when eight data bits have been shifted. The device may enter the MICROWIRE/PLUS mode either as a Master or as a Slave. *Figure 14* shows how two devices, microcontrollers and several peripherals may be interconnected using the MICROWIRE/PLUS arrangements.



TL/DD/12865–15

FIGURE 13. MICROWIRE/PLUS Block Diagram

MICROWIRE/PLUS (Continued)

WARNING

The SIO register should only be loaded when the SK clock is low. Loading the SIO register while the SK clock is high will result in undefined data in the SIO register. SK clock is normally low when not shifting.

Setting the BUSY flag when the input SK clock is high in the MICROWIRE/PLUS slave mode may cause the current SK clock for the SIO shift register to be narrow. For safety, the BUSY flag should only be set when the input SK clock is low.

MICROWIRE/PLUS Master Mode Operation

In the MICROWIRE/PLUS Master mode of operation the shift clock (SK) is generated internally. The MICROWIRE Master always initiates all data exchanges. The MSEL bit in the CNTRL register must be set to enable the SO and SK functions onto the G Port. The SO and SK pins must also be selected as outputs by setting appropriate bits in the Port G configuration register. Table XI summarizes the bit settings required for Master mode of operation.

MICROWIRE/PLUS Slave Mode Operation

In the MICROWIRE/PLUS Slave mode of operation the SK clock is generated by an external source. Setting the MSEL bit in the CNTRL register enables the SO and SK functions onto the G Port. The SK pin must be selected as an input and the SO pin is selected as an output pin by setting and resetting the appropriate bits in the Port G configuration register. Table XI summarizes the settings required to enter the Slave mode of operation.

The user must set the BUSY flag immediately upon entering the Slave mode. This will ensure that all data bits sent by the Master will be shifted properly. After eight clock pulses the BUSY flag will be cleared and the sequence may be repeated.

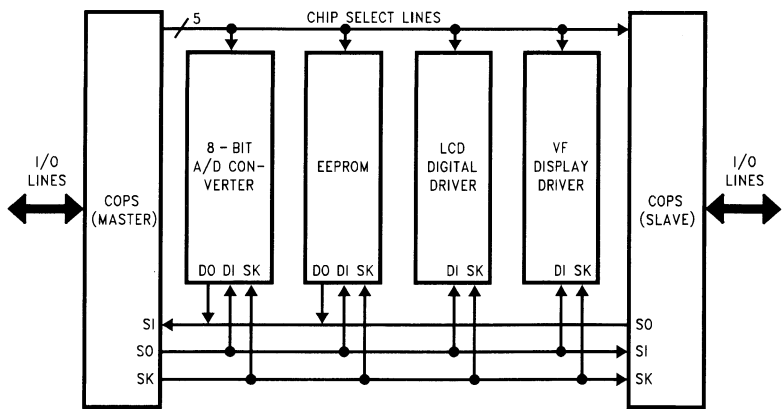
TABLE XI. MICROWIRE Mode Settings

G4 (SO) Config. Bit	G5 (SK) Config. Bit	G4 Fun.	G5 Fun.4	Operation
1	1	SO	Int. SK	MICROWIRE/PLUS Master
0	1	TRI-STATE	Int. SK	MICROWIRE/PLUS Master
1	0	SO	Ext. SK	MICROWIRE/PLUS Slave
0	0	TRI-STATE	Ext. SK	MICROWIRE/PLUS Slave
This table assumes that the control flag MSEL is set.				

Alternate SK Phase Operation

The device allows either the normal SK clock or an alternate phase SK clock to shift data in and out of the SIO register. In both the modes the SK is normally low. In the normal mode data is shifted in on the rising edge of the SK clock and the data is shifted out on the falling edge of the SK clock. The SIO register is shifted on each falling edge of the SK clock. In the alternate SK phase operation, data is shifted in on the falling edge of the SK clock and shifted out on the rising edge of the SK clock.

A control flag, SKSEL, allows either the normal SK clock or the alternate SK clock to be selected. Resetting SKSEL causes the MICROWIRE/PLUS logic to be clocked from the normal SK signal. Setting the SKSEL flag selects the alternate SK clock. The SKSEL is mapped into the G6 configuration bit. The SKSEL flag will power up in the reset condition, selecting the normal SK signal.



TL/DD/12865-16

FIGURE 14. MICROWIRE/PLUS Application

Memory Map

All RAM, ports and registers (except A and PC) are mapped into data memory address space.

Address S/ADD REG	Contents
0000 to 006F	On-Chip RAM bytes (112 bytes)
0070 to 007F	Unused RAM Address Space (Reads As All Ones)
xx80 to xxAF	Unused RAM Address Space (Reads Undefined Data)
xxB0	Reserved
XXB1	Reserved
xxB2	Reserved
xxB3	Reserved
xxB4	Reserved
xxB5	Reserved
xxB6	Reserved
xxB7	Comparator Select Register (CMPSL)
xxB8 to xxBF	Reserved
xxC0	Reserved
xxC1	Reserved
xxC2	Reserved
xxC3	Reserved
xxC4	Reserved
xxC5	Reserved
xxC6	Reserved
xxC7	WATCHDOG Service Register (Reg:WDSVR)
xxC8	MIWU Edge Select Register (Reg:WKEDG)
xxC9	MIWU Enable Register (Reg:WKEN)
xxCA	MIWU Pending Register (Reg:WKPND)
xxCB	Reserved
xxCC	CAPTLO (Capture Timer Low-Byte)
xxCD	CAPTHI (Capture Timer High-Byte)
xxCE	CAPCNTL (Capture Timer Control Register)
xxCF	Idle Timer Control Register
xxD0	Port L Data Register
xxD1	Port L Configuration Register
xxD2	Port L Input Pins (Read Only)
xxD3	Reserved
xxD4	Port G Data Register
xxD5	Port G Configuration Register
xxD6	Port G Input Pins (Read Only)
xxD7	Port I Input Pins (Read Only)
xxD8	Reserved
xxD9	Reserved
xxDA	Reserved
xxDB	Reserved
xxDC	Port D
xxDD to DF	Reserved
xxE0 to xxE5	Reserved
xxE6	Timer T1 Autoload Register T1RB Lower Byte
xxE7	Timer T1 Autoload Register T1RB Upper Byte
xxE8	ICNTRL Register
xxE9	MICROWIRE/PLUS Shift Register
xxEA	Timer T1 Lower Byte
xxEB	Timer T1 Upper Byte
xxEC	Timer T1 Autoload Register T1RA Lower Byte
xxED	Timer T1 Autoload Register T1RA Upper Byte
xxEE	CNTRL Control Register
xxEF	PSW Register
xxF0 to FB	On-Chip RAM Mapped as Registers
xxFC	X Register
xxFD	SP Register
xxFE	B Register
xxFF	Reserved
0100-017F	Reserved

Reading memory locations 0070H-007FH (Segment 0) will return all ones. Reading unused memory locations 0080H-00AFH (Segment 0) will return undefined data. Reading memory locations from other Segments (i.e., Segment 2, Segment 3, ... etc.) will return all ones.

Addressing Modes

There are ten addressing modes, six for operand addressing and four for transfer of control.

OPERAND ADDRESSING MODES

Register Indirect

This is the "normal" addressing mode. The operand is the data memory addressed by the B pointer or X pointer.

Register Indirect (with auto post increment or decrement of pointer)

This addressing mode is used with the LD and X instructions. The operand is the data memory addressed by the B pointer or X pointer. This is a register indirect mode that automatically post increments or decrements the B or X register after executing the instruction.

Direct

The instruction contains an 8-bit address field that directly points to the data memory for the operand.

Immediate

The instruction contains an 8-bit immediate field as the operand.

Short Immediate

This addressing mode is used with the Load B Immediate instruction. The instruction contains a 4-bit immediate field as the operand.

Indirect

This addressing mode is used with the LAID instruction. The contents of the accumulator are used as a partial address (lower 8 bits of PC) for accessing a data operand from the program memory.

TRANSFER OF CONTROL ADDRESSING MODES

Relative

This mode is used for the JP instruction, with the instruction field being added to the program counter to get the new program location. JP has a range from −31 to +32 to allow a 1-byte relative jump (JP + 1 is implemented by a NOP instruction). There are no "pages" when using JP, since all 15 bits of PC are used.

Absolute

This mode is used with the JMP and JSR instructions, with the instruction field of 12 bits replacing the lower 12 bits of the program counter (PC). This allows jumping to any location in the current 4k program memory segment.

Absolute Long

This mode is used with the JMPL and JSRL instructions, with the instruction field of 15 bits replacing the entire 15 bits of the program counter (PC). This allows jumping to any location up to 32k in the program memory space.

Indirect

This mode is used with the JID instruction. The contents of the accumulator are used as a partial address (lower 8 bits of PC) for accessing a location in the program memory. The contents of this program memory location serve as a partial address (lower 8 bits of PC) for the jump to the next instruction.

Note: The VIS is a special case of the Indirect Transfer of Control addressing mode, where the double byte vector associated with the interrupt is transferred from adjacent addresses in the program memory into the program counter (PC) in order to jump to the associated interrupt service routine.

Instruction Set

Register and Symbol Definition

	Registers
A	8-Bit Accumulator Register
B	8-Bit Address Register
X	8-Bit Address Register
SP	8-Bit Stack Pointer Register
PC	15-Bit Program Counter Register
PU	Upper 7 Bits of PC
PL	Lower 8 Bits of PC
C	1-Bit of PSW Register for Carry
HC	1-Bit of PSW Register for Half Carry
GIE	1-Bit of PSW Register for Global Interrupt Enable
VU	Interrupt Vector Upper Byte
VL	Interrupt Vector Lower Byte
	Symbols
[B]	Memory Indirectly Addressed by B Register
[X]	Memory Indirectly Addressed by X Register
MD	Direct Addressed Memory
Mem	Direct Addressed Memory or [B]
Meml	Direct Addressed Memory or [B] or Immediate Data
Imm	8-Bit Immediate Data
Reg	Register Memory: Addresses F0 to FF (Includes B, X and SP)
Bit	Bit Number (0 to 7)
←	Loaded with
↔	Exchanged with

Instruction Set (Continued)

ADD	A,Meml	ADD	A ← A + Meml
ADC	A,Meml	ADD with Carry	A ← A + Meml + C, C ← Carry, HC ← Half Carry
SUBC	A,Meml	Subtract with Carry	A ← A − Meml + C, C ← Carry, HC ← Half Carry
AND	A,Meml	Logical AND	A ← A and $\overline{\text{Meml}}$
ANDSZ	A,Imm	Logical AND Immed., Skip if Zero	Skip next if (A and Imm) = 0
OR	A,Meml	Logical OR	A ← A or Meml
XOR	A,Meml	Logical EXclusive OR	A ← A xor Meml
IFEQ	MD,Imm	IF EQual	Compare MD and Imm, Do next if MD = Imm
IFEQ	A,Meml	IF EQual	Compare A and Meml, Do next if A = Meml
IFNE	A,Meml	IF Not Equal	Compare A and Meml, Do next if A ≠ Meml
IFGT	A,Meml	IF Greater Than	Compare A and Meml, Do next if A > Meml
IFBNE	#	If B Not Equal	Do next if lower 4 bits of B ≠ Imm
DRSZ	Reg	Decrement Reg., Skip if Zero	Reg ← Reg − 1, Skip if Reg = 0
SBIT	#,Mem	Set BIT	1 to bit, Mem (bit = 0 to 7 immediate)
RBIT	#,Mem	Reset BIT	0 to bit, Mem
IFBIT	#,Mem	IF BIT	If bit #,A or Mem is true do next instruction
RPND		Reset PeNDing Flag	Reset Software Interrupt Pending Flag
X	A,Mem	EXchange A with Memory	A ⟷ Mem
X	A,[X]	EXchange A with Memory [X]	A ⟷ [X]
LD	A,Meml	LoaD A with Memory	A ← Meml
LD	A,[X]	LoaD A with Memory [X]	A ← [X]
LD	B,Imm	LoaD B with Immed.	B ← Imm
LD	Mem,Imm	LoaD Memory Immed	Mem ← Imm
LD	Reg,Imm	LoaD Register Memory Immed.	Reg ← Imm
X	A, [B±]	EXchange A with Memory [B]	A ⟷ [B], (B ← B ±1)
X	A, [X±]	EXchange A with Memory [X]	A ⟷ [X], (X ← X±1)
LD	A, [B±]	LoaD A with Memory [B]	A ← [B], (B ← B±1)
LD	A, [X±]	LoaD A with Memory [X]	A ← [X], (X ← X±1)
LD	[B±],Imm	LoaD Memory [B] Immed.	[B] ← Imm, (B ← B±1)
CLR	A	CLeaR A	A ← 0
INC	A	INCrement A	A ← A + 1
DEC	A	DECrement A	A ← A − 1
LAID		Load A InDirect from ROM	A ← ROM (PU,A)
DCOR	A	Decimal CORrect A	A ← BCD correction of A (follows ADC, SUBC)
RRC	A	Rotate A Right thru C	C → A7 → ... → A0 → C
RLC	A	Rotate A Left thru C	C ← A7 ← ... ← A0 ← C
SWAP	A	SWAP nibbles of A	A7...A4 ⟷ A3...A0
SC		Set C	C ← 1, HC ← 1
RC		Reset C	C ← 0, HC ← 0
IFC		IF C	IF C is true, do next instruction
IFNC		IF Not C	If C is not true, do next instruction
POP	A	POP the stack into A	SP ← SP + 1, A ← [SP]
PUSH	A	PUSH A onto the stack	[SP] ← A, SP ← SP − 1
VIS		Vector to Interrupt Service Routine	PU ← [VU], PL ← [VL]
JMPL	Addr.	Jump absolute Long	PC ← ii (ii = 15 bits, 0 to 32k)
JMP	Addr.	Jump absolute	PC9...0 ← i (i = 12 bits)
JP	Disp.	Jump relative short	PC ← PC + r (r is −31 to +32, except 1)
JSRL	Addr.	Jump SubRoutine Long	[SP] ← PL, [SP-1] ← PU,SP-2, PC ← ii
JSR	Addr	Jump SubRoutine	[SP] ← PL, [SP-1] ← PU,SP-2, PC9...0 ← i
JID		Jump InDirect	PL ← ROM (PU,A)
RET		RETurn from subroutine	SP + 2, PL ← [SP], PU ← [SP-1]
RETSK		RETurn and SKip	SP + 2, PL ← [SP],PU ← [SP-1], skip next instruction
RETI		RETurn from Interrupt	SP + 2, PL ← [SP],PU ← [SP-1],GIE ← 1
INTR		Generate an Interrupt	[SP] ← PL, [SP-1] ← PU, SP-2, PC ← 0FF
NOP		No OPeration	PC ← PC + 1

Instruction Execution Time

Most instructions are single byte (with immediate addressing mode instructions taking two bytes).

Most single byte instructions take one cycle time to execute.

Skipped instructions require x number of cycles to be skipped, where x equals the number of bytes in the skipped instruction opcode.

See the BYTES and CYCLES per INSTRUCTION table for details.

Bytes and Cycles per Instruction

The following table shows the number of bytes and cycles for each instruction in the format of byte/cycle.

Arithmetic and Logic Instructions

	[B]	Direct	Immed
ADD	1/1	3/4	2/2
ADC	1/1	3/4	2/2
SUBC	1/1	3/4	2/2
AND	1/1	3/4	2/2
OR	1/1	3/4	2/2
XOR	1/1	3/4	2/2
IFEQ	1/1	3/4	2/2
IFGT	1/1	3/4	2/2
IFBNE	1/1		
DRSZ	1/1	1/3	
SBIT	1/1	3/4	
RBIT	1/1	3/4	
IFBIT	1/1	3/4	

RPND	1/1

Instructions Using A and C

CLRA	1/1
INCA	1/1
DECA	1/1
LAID	1/3
DCORA	1/1
RRCA	1/1
RLCA	1/1
SWAPA	1/1
SC	1/1
RC	1/1
IFC	1/1
IFNC	1/1
PUSHA	1/3
POPA	1/3
ANDSZ	2/2

Transfer of Control Instructions

JMPL	3/4
JMP	2/3
JP	1/3
JSRL	3/5
JSR	2/5
JID	1/3
VIS	1/5
RET	1/5
RETSK	1/5
RETI	1/5
INTR	1/7
NOP	1/1

Memory Transfer Instructions

	Register Indirect		Direct	Immed.	Register Indirect Auto Incr and Decr		
	[B]	[X]			[B+, B−]	[X+, X−]	
X A,*	1/1	1/3	2/3		1/2	1/3	
LD A,*	1/1	1/3	2/3	2/2	1/2	1/3	
LD B,Imm				1/1			(If B < 16)
LD B,Imm				2/2			(If B > 15)
LD Mem,Imm	2/2		3/3		2/2		
LD Reg,Imm			2/3				
IFEQ MD,Imm			3/3				

*Memory location addressed by B or X or directly.

Opcode Table

UPPER NIBBLE															
F	E	D	C	B	A	9	8	7	6	5	4	3	2	1	0
JP −15	JP −31	LD 0F0, #i	DRSZ 0F0	RRCA	RC	ADC A,#i	ADC A,[B]	IFBIT 0,[B]	ANDSZ A, #i	LD B,#0F	IFBNE 0	JSR x000−x0FF	JMP x000−x0FF	JP + 17	INTR
JP −14	JP −30	LD 0F1, #i	DRSZ 0F1	*	SC	SUBC A,#i	SUBC A,[B]	IFBIT 1,[B]	*	LD B,#0E	IFBNE 1	JSR x100−x1FF	JMP x100−x1FF	JP + 18	JP +2
JP −13	JP −29	LD 0F2, #i	DRSZ 0F2	X A, [X+]	X A, [B+]	IFEQ A, #i	IFEQ A,[B]	IFBIT 2,[B]	*	LD B,#0D	IFBNE 2	JSR x200−x2FF	JMP x200−x2FF	JP + 19	JP +3
JP −12	JP −28	LD 0F3, #i	DRSZ 0F3	X A, [X−]	X A, [B−]	IFGT A, #i	IFGT A,[B]	IFBIT 3,[B]	*	LD B,#0C	IFBNE 3	JSR x300−x3FF	JMP x300−x3FF	JP + 20	JP +4
JP −11	JP −27	LD 0F4, #i	DRSZ 0F4	VIS	LAID	ADD A, #i	ADD A,[B]	IFBIT 4,[B]	CLRA	LD B,#0B	IFBNE 4	JSR x400−x4FF	JMP x400−x4FF	JP + 21	JP +5
JP −10	JP −26	LD 0F5, #i	DRSZ 0F5	RPND	JID	AND A, #i	AND A,[B]	IFBIT 5,[B]	SWAPA	LD B,#0A	IFBNE 5	JSR x500−x5FF	JMP x500−x5FF	JP + 22	JP +6
JP −9	JP −25	LD 0F6, #i	DRSZ 0F6	X A,[X]	X A,[B]	XOR A, #i	XOR A,[B]	IFBIT 6,[B]	DCORA	LD B,#09	IFBNE 6	JSR x600−x6FF	JMP x600−x6FF	JP + 23	JP +7
JP −8	JP −24	LD 0F7, #i	DRSZ 0F7	*	*	OR A,#i	OR A,[B]	IFBIT 7,[B]	PUSHA	LD B,#08	IFBNE 7	JSR x700−x7FF	JMP x700−x7FF	JP + 24	JP +8
JP −7	JP −23	LD 0F8, #i	DRSZ 0F8	NOP	RLCA	LD A,#i	IFC	SBIT 0,[B]	RBIT 0,[B]	LD B,#07	IFBNE 8	JSR x800−x8FF	JMP x800−x8FF	JP + 25	JP +9
JP −6	JP −22	LD 0F9, #i	DRSZ 0F9	IFNE A,[B]	IFEQ Md,#i	IFNE A,#i	IFNC	SBIT 1,[B]	RBIT 1,[B]	LD B,#06	IFBNE 9	JSR x900−x9FF	JMP x900−x9FF	JP + 26	JP +10
JP −5	JP −21	LD 0FA, #i	DRSZ 0FA	LD A, [X+]	LD A, [B+]	LD [B+], #i	INCA	SBIT 2,[B]	RBIT 2,[B]	LD B,#05	IFBNE 0A	JSR xA00−xAFF	JMP xA00−xAFF	JP + 27	JP +11
JP −4	JP −20	LD 0FB, #i	DRSZ 0FB	LD A, [X−]	LD A, [B−]	LD [B−], #i	DECA	SBIT 3,[B]	RBIT 3,[B]	LD B,#04	IFBNE 0B	JSR xB00−xBFF	JMP xB00−xBFF	JP + 28	JP +12
JP −3	JP −19	LD 0FC, #i	DRSZ 0FC	LD Md,#i	JMPL	X A,Md	POPA	SBIT 4,[B]	RBIT 4,[B]	LD B,#03	IFBNE 0C	JSR xC00−xCFF	JMP xC00−xCFF	JP + 29	JP +13
JP −2	JP −18	LD 0FD, #i	DRSZ 0FD	DIR	JSRL	LD A,Md	RETSK	SBIT 5,[B]	RBIT 5,[B]	LD B,#02	IFBNE 0D	JSR xD00−xDFF	JMP xD00−xDFF	JP + 30	JP +14
JP −1	JP −17	LD 0FE, #i	DRSZ 0FE	LD A,[X]	LD A,[B]	LD [B],#i	RET	SBIT 6,[B]	RBIT 6,[B]	LD B,#01	IFBNE 0E	JSR xE00−xEFF	JMP xE00−xEFF	JP + 31	JP +15
JP −0	JP −16	LD 0FF, #i	DRSZ 0FF	*	*	LD B,#i	RETI	SBIT 7,[B]	RBIT 7,[B]	LD B,#00	IFBNE 0F	JSR xF00−xFFF	JMP xF00−xFFF	JP + 32	JP +16

LOWER NIBBLE

where,

i is the immediate data

Md is a directly addressed memory location

* is an unused opcode

Note: The opcode 60 Hex is also the opcode for IFBIT #i,A

Mask Options

The mask programmable options are shown below. The options are programmed at the same time as the ROM pattern submission.

```
OPTION 1: CLOCK CONFIGURATION
 = 1   Crystal Oscillator (CKI/lO)
       G7 (CKO) is clock generator output
       to crystal/resonator CKI is the
       clock input
 = 2   Single-pin RC controlled oscillator
       (CKI/lO)
       G7 is available as a HALT restart
       and/or general purpose input
OPTION 2: HALT
 = 1   Enable HALT mode
 = 2   Disable HALT mode
OPTION 3: BONDING OPTIONS
 = 1   28-Pin DIP
 = 2   28-Pin SO
 = 3   N/A
 = 4   20-Pin SO
```

Development Support

Summary

- iceMASTER: IM-COP8/400—Full feature in-circuit emulation for all COP8 products. A full set of COP8 Basic and Feature Family device and package specific probes are available.
- COP8 Debug Module: Moderate cost in-circuit emulation and development programming unit.
- Assembler: COP8-DEV-IBMA. A DOS installable cross development Assembler, Linker, Librarian and Utility Software Development Tool Kit.
- C Compiler: COP8C. A DOS installable cross development Software Tool Kit.
- OTP/EPROM Programmer Support: Covering needs from engineering prototype, pilot production to full production environments.

Development Support (Continued)

IceMASTER (IM) IN-CIRCUIT EMULATION

The iceMASTER IM-COP8/400 is a full feature, PC based, in-circuit emulation tool developed and marketed by MetaLink Corporation to support the whole COP8 family of products. National is a resale vendor for these products.

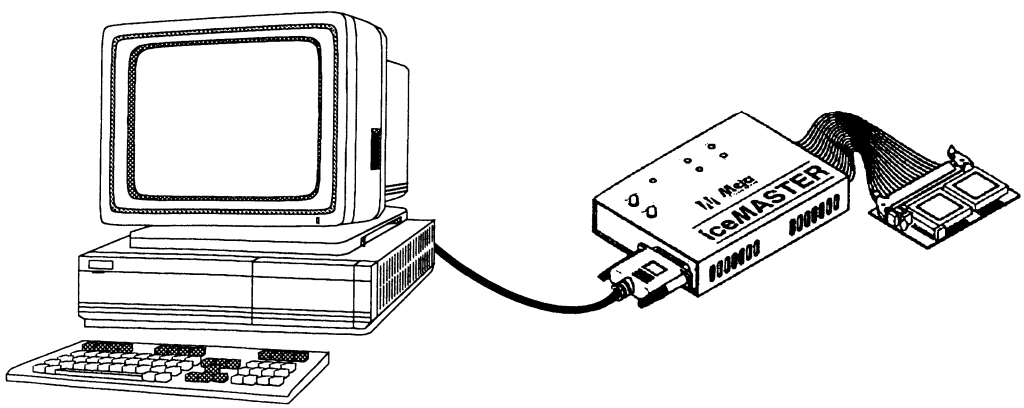
See *Figure 15* for configuration.

The iceMASTER IM-COP8/400 with its device specific COP8 Probe provides a rich feature set for developing, testing and maintaining product:

- Real-time in-circuit emulation; full 2.4V–5.5V operation range, full DC-10 MHz clock. Chip options are programmable or jumper selectable.
- Direct connection to application board by package compatible socket or surface mount assembly.
- Full 32 KByte of loadable programming space that overlays (replaces) the on-chip ROM or EPROM. On-chip RAM and I/O blocks are used directly or recreated on the probe as necessary.
- Full 4k frame synchronous trace memory. Address, instruction, and eight unspecified, circuit connectable trace lines. Display can be HLL source (e.g., C source), assembly or mixed.
- A full 64k hardware configurable break, trace on, trace off control, and pass count increment events.
- Tool set integrated interactive symbolic debugger—supports both assembler (COFF) and C Compiler (.COD) linked object formats.
- Real time performance profiling analysis; selectable bucket definition.
- Watch windows, content updated automatically at each execution break.
- Instruction by instruction memory/register changes displayed on source window when in single step operation.
- Single base unit and debugger software reconfigurable to support the entire COP8 family; only the probe personality needs to change. Debugger software is processor customized, and reconfigured from a master model file.
- Processor specific symbolic display of registers and bit level assignments, configured from master model file.
- Halt/Idle mode notification.
- On-line HELP customized to specific processor using master model file.
- Includes a copy of COP8-DEV-IBMA assembler and linker SDK.

IM Order Information

Base Unit	
IM-COP8/400-1	iceMASTER Base Unit, 110V Power Supply
IM-COP8/400-2	iceMASTER Base Unit, 220V Power Supply
iceMASTER Probe	
COP8AC-IM28N	28 DIP
COP8AC-IM20N	20 DIP
Surface Mount Adapter	
MHW-SOIC28	28 SO
MHW-SOIC20	20 SO



TL/DD/12865-17

FIGURE 15. COP8 iceMASTER Environment

Development Support (Continued)

IceMASTER DEBUG MODULE (DM)

The iceMASTER Debug Module is a PC based, combination in-circuit emulation tool and COP8 based OTP/EPROM programming tool developed and marketed by MetaLink Corporation to support the whole COP8 family of products. National is a resale vendor for these products.

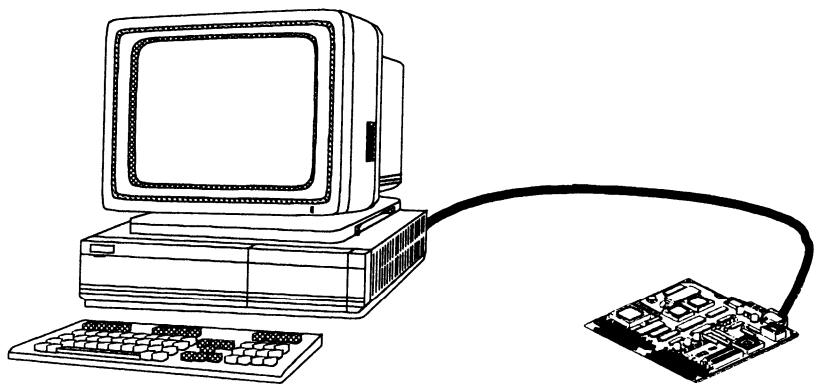
See *Figure 16* for configuration.

The iceMASTER Debug Module is a moderate cost development tool. It has the capability of in-circuit emulation for a specific COP8 microcontroller and in addition serves as a programming tool for COP8 OTP and EPROM product families. Summary of features is as follows:

- Real-time in-circuit emulation; full operating voltage range operation, full DC-10 MHz clock.
- All processor I/O pins can be cabled to an application development board with package compatible cable to socket and surface mount assembly.
- Full 32 kbyte of loadable programming space that overlays (replaces) the on-chip ROM or EPROM. On-chip RAM and I/O blocks are used directly or recreated as necessary.
- 100 frames of synchronous trace memory. The display can be HLL source (C source), assembly or mixed. The most recent history prior to a break is available in the trace memory.
- Configured break points; uses INTR instruction which is modestly intrusive.
- Software—only supported features are selectable.
- Tool set integrated interactive symbolic debugger—supports both assembler (COFF) and C Compiler (.COD) SDK linked object formats.
- Instruction by instruction memory/register changes displayed when in single step operation.
- Debugger software is processor customized, and reconfigured from a master model file.
- Processor specific symbolic display of registers and bit level assignments, configured from master model file.
- Halt/Idle mode notification.
- Programming menu supports full product line of programmable OTP and EPROM COP8 products. Program data is taken directly from the overlay RAM.
- Programming of 44PLCC and 68PLCC parts requires external programming adapters.
- Includes wallmount power supply.
- On-board V_{PP} generator from 5V input or connection to external supply supported. Requires V_{PP} level adjustment per the family programming specification (correct level is provided on an on-screen pop-down display).
- On-line HELP customized to specific processor using master model file.
- Includes a copy of COP8-DEV-IBMA assembler and linker SDK.

DM Order Information

Debug Module Unit	
COP8AC-DM	
Cable Adapters	
DM-COP8/28D	28 DIP
DM-COP8/20D	20 DIP
Surface Mount Adapters	
DM-COP8/28D-SO	28 DIP to SO
DM-COP8/20D-SO	20 DIP to SO



TL/DD/12865–18

FIGURE 16. COP-DM Environment

Development Support (Continued)

COP8 ASSEMBLER/LINKER SOFTWARE DEVELOPMENT TOOL KIT

National Semiconductor offers a relocatable COP8 macro cross assembler, linker, librarian and utility software development tool kit. Features are summarized as follows:

- Basic and Feature Family instruction set by "device" type.
- Nested macro capability.
- Extensive set of assembler directives.
- Supported on PC/DOS platform.
- Generates National standard COFF output files.
- Integrated Linker and Librarian.
- Integrated utilities to generate ROM code file outputs.
- DUMPCOFF utility.

This product is integrated as a part of MetaLink tools as a development kit, fully supported by the MetaLink debugger. It may be ordered separately or it is bundled with the MetaLink products at no additional cost.

Order Information

Assembler SDK:	
COP8-DEV-IBMA	Assembler SDK on installable 3.5" PC/DOS Floppy Disk Drive format. Periodic upgrades and most recent version is available on National's BBS and Internet.

COP8 C COMPILER

A C Compiler is developed and marketed by Byte Craft Limited. The COP8C compiler is a fully integrated development tool specifically designed to support the compact embedded configuration of the COP8 family of products.

Features are summarized as follows:

- ANSI C with some restrictions and extensions that optimize development for the COP8 embedded application.
- BITS data type extension. Register declaration #pragma with direct bit level definitions.
- C language support for interrupt routines.
- Expert system, rule based code generation and optimization.
- Performs consistency checks against the architectural definitions of the target COP8 device.
- Generates program memory code.
- Supports linking of compiled object or COP8 assembled object formats.
- Global optimization of linked code.
- Symbolic debug load format fully source level supported by the MetaLink debugger.

SINGLE CHIP OTP/EMULATOR SUPPORT

The COP8 family is supported by single chip OTP emulators. For detailed information refer to the emulator specific datasheet and the emulator selection table below:

OTP Emulator Ordering Information

Device Number	Clock Option	Package	Emulates
COP8ACC720Mx	Crystal	20 SO	COP8ACC520Mx
COP8ACC728Nx	Crystal	20 DIP	COP8ACC528Nx
COP8ACC728Mx	Crystal	20 SO	COP8ACC528Mx

x = temp. range (6, 7, 8)

INDUSTRY WIDE OTP/EPROM PROGRAMMING SUPPORT

Programming support, in addition to the MetaLink development tools, is provided by a full range of independent approved vendors to meet the needs from the engineering laboratory to full production.

Approved List:

Manufacturer	North America	Europe	Asia
BP Microsystems	(800) 225-2102 (713) 688-4600 Fax: (713) 688-0920	+49-8152-4183 +49-8856-932616	+852-234-16611 +852-2710-8121
Data I/O	(800) 426-1045 (206) 881-6444 Fax: (206) 882-1043	+44-0734-440011	Call North America
HI-LO	(510) 623-8860	Call Asia	+886-2-764-0215 Fax: +886-2-756-6403
ICE Technology	(800) 624-8949 (919) 430-7915	+44-1226-767404 Fax: 0-1226-370-434	
MetaLink	(800) 638-2423 (602) 926-0797 Fax: (602) 693-0681	+49-80 9156 96-0 Fax: +49-80 9123 86	+852-737-1800
Systems General	(408) 263-6667	+41-1-9450300	+886-2-917-3005 Fax: +886-2-911-1283
Needhams	(916) 924-8037 Fax: (916) 924-8065		

Development Support (Continued)

AVAILABLE LITERATURE

For more information, please see the COP8 Basic Family User's Manual, Literature Number 620895, COP8 Feature Family User's Manual, Literature Number 620897 and National's Family of 8-bit Microcontrollers COP8 Selection Guide, Literature Number 630009.

DIAL-A-HELPER SERVICE

Dial-A-Helper is a service provided by the Microcontroller Applications group. The Dial-A-Helper is an Electronic Information System that may be accessed as a Bulletin Board System (BBS) via data modem, as an FTP site on the Internet via standard FTP client application or as an FTP site on the Internet using a standard Internet browser such as Netscape or Mosaic.

The Dial-A-Helper system provides access to an automated information storage and retrieval system. The system capabilities include a MESSAGE SECTION (electronic mail, when accessed as a BBS) for communications to and from the Microcontroller Applications Group and a FILE SECTION which consists of several file areas where valuable application software and utilities could be found.

DIAL-A-HELPER BBS via a Standard Modem

Modem: CANADA/U.S.: (800) NSC-MICRO
(800) 672-6427
EUROPE: (+49) 0-814-135 13 32
Baud: 14.4k
Set-Up: Length: 8-Bit
Parity: None
Stop Bit: 1
Operation: 24 Hours, 7 Days

DIAL-A-HELPER via FTP

ftp nscmicro.nsc.com
user: anonymous
password: username@yourhost.site.domain

DIAL-A-HELPER via a WorldWide Web Browser

ftp://nscmicro.nsc.com

National Semiconductor on the WorldWide Web

See us on the WorldWide Web at: http://www.national.com

CUSTOMER RESPONSE CENTER

Complete product information and technical support is available from National's customer response centers.

CANADA/U.S.:	Tel:	(800) 272-9959
	email:	support @tevm2.nsc.com
EUROPE:	email:	europe.support@nsc.com
	Deutsch Tel:	+49 (0) 180-530 85 85
	English Tel:	+49 (0) 180-532 78 32
	Français Tel:	+49 (0) 180-532 93 58
	Italiano Tel:	+49 (0) 180-534 16 80
JAPAN:	Tel:	+81-043-299-2309
S.E. ASIA:	Beijing Tel:	(+86) 10-6856-8601
	Shanghai Tel:	(+86) 21-6415-4092
	Hong Kong Tel:	(+852) 2737-1600
	Korea Tel:	(+82) 2-3771-6909
	Malaysia Tel:	(+60-4) 644-9061
	Singapore Tel:	(+65) 255-2226
	Taiwan Tel:	+886-2-521-3288
AUSTRALIA:	Tel:	(+61) 3-9558-9999
INDIA:	Tel:	(+91) 80-559-9467

National Semiconductor

COP688EK/COP684EK/COP888EK/COP884EK/ COP988EK/COP984EK 8-Bit Microcontroller with Analog Function Block

General Description

The COP888 family of microcontrollers uses an 8-bit single chip core architecture fabricated with National Semiconductor's M²CMOS™ process technology. The COP888EK is a member of this expandable 8-bit core processor family of microcontrollers. (Continued)

Key Features

- Analog function block with
  - — Analog comparator with seven input multiplexor
  - — Constant current source and $V_{CC}/2$ reference
- Three 16-bit timers, each with two 16-bit registers supporting:
  - — Processor Independent PWM mode
  - — External Event counter mode
  - — Input Capture mode
- 8 kbytes of on-chip ROM
- 256 bytes of on-chip RAM

Additional Peripheral Features

- Idle Timer
- Multi-Input Wake Up (MIWU) with optional interrupts (8)
- WATCHDOG™ and Clock Monitor logic
- MICROWIRE/PLUS™ serial I/O

I/O Features

- Memory mapped I/O
- Software selectable I/O options (TRI-STATE® Output, Push-Pull Output, Weak Pull-Up Input, High Impedance Input)
- High current outputs
- Schmitt trigger inputs on Port G
- Packages:
  - — 44 PLCC with 40 I/O pins
  - — 40 DIP with 36 I/O pins
  - — 28 DIP with 24 I/O pins
  - — 28 SO with 24 I/O pins

CPU/Instruction Set Feature

- 1 μs instruction cycle time
- Twelve multi-source vectored interrupts servicing
  - — External Interrupt with selectable edge
  - — Idle Timer T0
  - — Three Timers (Each with 2 interrupts)
  - — MICROWIRE/PLUS
  - — Multi-Input Wake Up
  - — Software Trap
  - — Default VIS (default interrupt)
- Versatile and easy to use instruction set
- 8-bit Stack Pointer (SP)—stack in RAM
- Two 8-bit Register Indirect Data Memory Pointers (B, X)

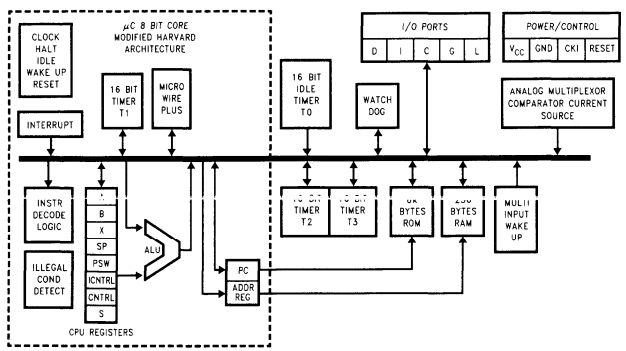
Fully Static CMOS

- Low current drain (typically < 1 μA)
- Single supply operation: 2.5V to 6.0V
- Temperature ranges: 0°C to +70°C, and −40°C to +85°C, and −55°C to +125°C

Development Support

- Emulation and OTP devices
- Real time emulation and full program debug offered by MetaLink Development System

Block Diagram



TL/DD/12094–1

FIGURE 1. Block Diagram

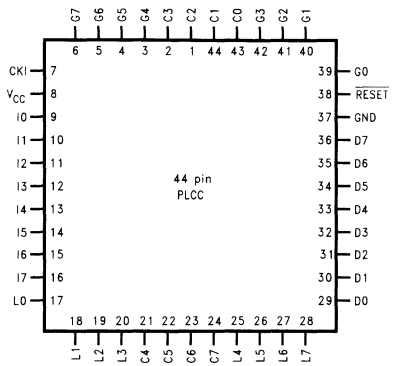
General Description (Continued)

It is a fully static part, fabricated using double-metal silicon gate microCMOS technology. Features include an 8-bit memory mapped architecture, MICROWIRE/PLUS serial I/O, three 16-bit timer/counters supporting three modes (Processor Independent PWM generation, External Event counter, and Input Capture mode capabilities), one analog comparator with seven input multiplexor, and two power saving modes (HALT and IDLE), both with a multi-sourced wakeup/interrupt capability. This multi-sourced interrupt capability may also be used independent of the HALT or IDLE modes. Each I/O pin has software selectable configurations. The devices operate over a voltage range of 2.5V to 6V. High throughput is achieved with an efficient, regular instruction set operating at a maximum rate of 1 μs per instruction.

Low radiated emissions are achieved by gradual turn-on output drivers and internal I_{CC} filters on the chip logic and crystal oscillator.

Connection Diagrams

Plastic Chip Carrier

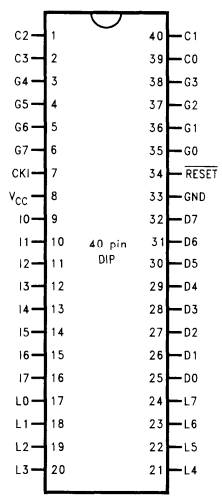


TL/DD/12094–2

Top View

Order Number COP688EK-XXX/V, COP888EK-XXX/V, COP988EK-XXX/V or COP988EKH-XXX/V
See NS Plastic Chip Package Number V44A

Dual-In-Line Package

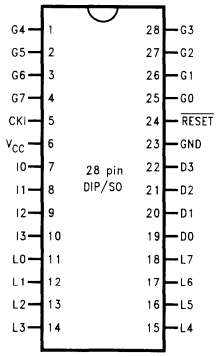


TL/DD/12094–3

Top View

Order Number COP688EK-XXX/N, COP888EK-XXX/N, COP988EK-XXX/N or COP988EKH-XXX/N
See NS Molded Package Number N40A

Dual-In-Line Package



TL/DD/12094–4

Top View

Order Number COP684EK-XXX/N, COP884EK-XXX/N, COP984EK-XXX/N or COP984EKH-XXX/N
See NS Molded Package Number N28B
Order Number COP684EK-XXX/WM, COP884EK-XXX/WM, COP984EK-XXX/WM or COP984EKH-XXX/WM
See NS Molded Package Number M28B

FIGURE 2. Connection Diagrams

Connection Diagrams (Continued)

Pinouts for 28-, 40- and 44-Pin Packages

Port	Type	Alt. Fun	Alt. Fun	28-Pin Pack.	40-Pin Pack.	44-Pin Pack.
L0	I/O	MIWU		11	17	17
L1	I/O	MIWU		12	18	18
L2	I/O	MIWU		13	19	19
L3	I/O	MIWU		14	20	20
L4	I/O	MIWU	T2A	15	21	25
L5	I/O	MIWU	T2B	16	22	26
L6	I/O	MIWU	T3A	17	23	27
L7	I/O	MIWU	T3B	18	24	28
G0	I/O	INT		25	35	39
G1	WDOUT			26	36	40
G2	I/O	T1B		27	37	41
G3	I/O	T1A		28	38	42
G4	I/O	SO		1	3	3
G5	I/O	SK		2	4	4
G6	I	SI		3	5	5
G7	I/CKO	HALT Restart		4	6	6
D0	O			19	25	29
D1	O			20	26	30
D2	O			21	27	31
D3	O			22	28	32
I0	I	COMPIN1+		7	9	9
I1	I	COMPIN−/Current Source Out		8	10	10
I2	I	COMPIN0+		9	11	11
I3	I	COMPOUT/COMPIN2+		10	12	12
I4	I	COMPIN3+			13	13
I5	I	COMPIN4+			14	14
I6	I	COMPIN5+			15	15
I7	I	COMPOUT			16	16
D4	O				29	33
D5	O				30	34
D6	O				31	35
D7	O				32	36
C0	I/O				39	43
C1	I/O				40	44
C2	I/O				1	1
C3	I/O				2	2
C4	I/O					21
C5	I/O					22
C6	I/O					23
C7	I/O					24
V_{CC}				ô	ô	ô
GND				23	33	37
CKI				5	7	7
$\overline{\text{RESET}}$				24	34	38

Absolute Maximum Ratings

If Military/Aerospace specified devices are required, please contact the National Semiconductor Sales Office/Distributors for availability and specifications.

Supply Voltage (V_{CC})	7V
Voltage at Any Pin	−0.3V to V_{CC} + 0.3V
Total Current into V_{CC} Pin (Source)	100 mA
Total Current out of GND Pin (Sink)	110 mA
Storage Temperature Range	−65°C to +140°C

Note: *Absolute maximum ratings indicate limits beyond which damage to the device may occur. DC and AC electrical specifications are not ensured when operating the device at absolute maximum ratings.*

DC Electrical Characteristics 98XEK: 0°C ≤ T_A ≤ +70°C unless otherwise specified

Parameter	Conditions	Min	Typ	Max	Units
Operating Voltage COP98XEK		2.5		4.0	V
COP98XEKH		4.0		6.0	V
Power Supply Ripple (Note 1)	Peak-to-Peak			0.1 V_{CC}	V
Supply Current (Note 2)					
CKI = 10 MHz	V_{CC} = 6.0V, t_c = 1 μs			10.0	mA
CKI = 4 MHz	V_{CC} = 2.5V, t_c = 2.5 μs			1.7	mA
HALT Current (Note 3)	V_{CC} = 6.0V, CKI = 0 MHz		<4	8	μA
	V_{CC} = 4.0V, CKI = 0 MHz		<3	4	μA
IDLE Current (Note 2)					
CKI = 10 MHz	V_{CC} = 6.0V, t_c = 1 μs		0.4	1.7	mA
CKI = 4 MHz	V_{CC} = 2.5V, t_c = 2.5 μs		0.2	0.5	mA
Input Levels (V_{IH}, V_{IL})					
$\overline{\text{RESET}}$					
Logic High		0.8 V_{CC}			V
Logic Low				0.2 V_{CC}	V
CKI, All Other Inputs					
Logic High		0.7 V_{CC}			V
Logic Low				0.2 V_{CC}	V
Hi-Z Input Leakage	V_{CC} = 6.0V	−1		+1	μA
Input Pullup Current	V_{CC} = 6.0V, V_{IN} = 0V	−40		−250	μA
G and L Port Input Hysteresis (Note 6)				0.35 V_{CC}	V
Output Current Levels					
D Outputs					
Source	V_{CC} = 4.0V, V_{OH} = 3.3V	−0.4			mA
	V_{CC} = 2.5V, V_{OH} = 1.8V	−0.2			mA
Sink	V_{CC} = 4.0V, V_{OL} = 1V	10			mA
	V_{CC} = 2.5V, V_{OL} = 0.4V	2.0			mA
All Others					
Source (Weak Pull-Up Mode)	V_{CC} = 4.0V, V_{OH} = 2.7V	−10		−110	μA
	V_{CC} = 2.5V, V_{OH} = 1.8V	−2.5		−33	μA
Source (Push-Pull Mode)	V_{CC} = 4.0V, V_{OH} = 3.3V	−0.4			mA
	V_{CC} = 2.5V, V_{OH} = 1.8V	−0.2			mA
Sink (Push-Pull Mode)	V_{CC} = 4.0V, V_{OL} = 0.4V	1.6			mA
	V_{CC} = 2.5V, V_{OL} = 0.4V	0.7			mA
TRI-STATE Leakage	V_{CC} = 6.0V	−1		+1	μA
Allowable Sink/Source Current per Pin (Note 6)					
D Outputs (Sink)				15	mA
All others				3	mA
Maximum Input Current without Latchup (Note 4)	Room Temp			±100	mA
RAM Retention Voltage, V_r	500 ns Rise and Fall Time (Min)	2			V
Input Capacitance				7	pF
Load Capacitance on D2				1000	pF

AC Electrical Characteristics 98XEK: $0°C \leq T_A \leq +70°C$ unless otherwise specified

Parameter	Conditions	Min	Typ	Max	Units
Instruction Cycle Time (t_c)					
Crystal, Resonator,	$4.0V \leq V_{CC} \leq 6.0V$	1.0		DC	μs
	$2.5V \leq V_{CC} < 4.0V$	2.5		DC	μs
R/C Oscillator	$4.0V \leq V_{CC} \leq 6.0V$	3.0		DC	μs
	$2.5V \leq V_{CC} < 4.0V$	7.5		DC	μs
Inputs					
t_{SETUP}	$4.0V \leq V_{CC} \leq 6.0V$	200			ns
	$2.5V \leq V_{CC} < 4.0V$	500			ns
t_{HOLD}	$4.0V \leq V_{CC} \leq 6.0V$	60			ns
	$2.5V \leq V_{CC} < 4.0V$	150			ns
Output Propagation Delay (Note 5)	$R_L = 2.2k, C_L = 100$ pF				
t_{PD1}, t_{PD0}					
SO, SK	$4.0V \leq V_{CC} \leq 6.0V$			0.7	μs
	$2.5V \leq V_{CC} < 4.0V$			1.75	μs
All Others	$4.0V \leq V_{CC} \leq 6.0V$			1	μs
	$2.5V \leq V_{CC} < 4.0V$			2.5	μs
MICROWIRE™ Setup Time (t_{UWS}) (Note 5)	$V_{CC} \geq 4.0V$	20			ns
MICROWIRE Hold Time (t_{UWH}) (Note 5)	$V_{CC} \geq 4.0V$	56			ns
MICROWIRE Output Propagation Delay (t_{UPD})	$V_{CC} \geq 4.0V$			220	ns
Input Pulse Width (Note 6)					
Interrupt Input High Time		1.0			t_c
Interrupt Input Low Time		1.0			t_c
Timer 1, 2, 3 Input High Time		1.0			t_c
Timer 1, 2, 3 Input Low Time		1.0			t_c
Reset Pulse Width		1.0			μs

t_c = Instruction Cycle Time

Note 1: Maximum rate of voltage change must be < 0.5 V/ms.

Note 2: Supply and IDLE currents are measured with CKI driven with a square wave Oscillator, CKO driven 180° out of phase with CKI, inputs connected to V_{CC} and outputs driven low but not connected to a load.

Note 3: The HALT mode will stop CKI from oscillating in the RC and the Crystal configurations. Measurement of I_{DD} HALT is done with device neither sourcing nor sinking current; with L, C, and G0–G5 programmed as low outputs and not driving a load; all outputs programmed low and not driving a load; all inputs tied to V_{CC}; clock monitor and comparator disabled. Parameter refers to HALT mode entered via setting bit 7 of the G Port data register. Part will pull up CKI during HALT in crystal clock mode.

Note 4: Pins G6 and $\overline{\text{RESET}}$ are designed with a high voltage input network. These pins allow input voltages > V_{CC} and the pins will have sink current to V_{CC} when biased at voltages > V_{CC} (the pins do not have source current when biased at a voltage below V_{CC}). The effective resistance to V_{CC} is 750Ω (typical). These two pins will not latch up. The voltage at the pins must be limited to < 14V. **WARNING: Voltages in excess of 14V will cause damage to the pins. This warning excludes ESD transients.**

Note 5: The output propagation delay is referenced to the end of the instruction cycle where the output change occurs.

Note 6: Parameter characterized but not tested.

Absolute Maximum Ratings

If Military/Aerospace specified devices are required, please contact the National Semiconductor Sales Office/Distributors for availability and specifications.

Supply Voltage (V_{CC})	7V
Voltage at Any Pin	−0.3V to V_{CC} + 0.3V
Total Current into V_{CC} Pin (Source)	100 mA
Total Current out of GND Pin (Sink)	110 mA
Storage Temperature Range	−65°C to +140°C

Note: *Absolute maximum ratings indicate limits beyond which damage to the device may occur. DC and AC electrical specifications are not ensured when operating the device at absolute maximum ratings.*

DC Electrical Characteristics 88XEK: −40°C ≤ T_A ≤ +85°C unless otherwise specified

Parameter	Conditions	Min	Typ	Max	Units
Operating Voltage		2.5		6.0	V
Power Supply Ripple (Note 1)	Peak-to-Peak			0.1 V_{CC}	V
Supply Current (Note 2)					
CKI = 10 MHz	V_{CC} = 6.0V, t_c = 1 μs			10.0	mA
CKI = 4 MHz	V_{CC} = 2.5V, t_c = 2.5 μs			1.7	mA
HALT Current (Note 3)	V_{CC} = 6.0V, CKI = 0 MHz		<4	10	μA
IDLE Current (Note 2)					
CKI = 10 MHz	V_{CC} = 6.0V, t_c = 1 μs		0.4	1.7	mA
CKI = 4 MHz	V_{CC} = 6.0V, t_c = 2.5 μs		0.2	0.5	mA
Input Levels (V_{IH}, V_{IL})					
$\overline{\text{RESET}}$					
Logic High		0.8 V_{CC}			V
Logic Low				0.2 V_{CC}	V
CKI, All Other Inputs					
Logic High		0.7 V_{CC}			V
Logic Low				0.2 V_{CC}	V
Hi-Z Input Leakage	V_{CC} = 6.0V	−2		+2	μA
Input Pullup Current	V_{CC} = 6.0V, V_{IN} = 0V	−40		−250	μA
G and L Port Input Hysteresis (Note 6)				0.35 V_{CC}	V
Output Current Levels					
D Outputs					
Source	V_{CC} = 4.0V, V_{OH} = 3.3V	−0.4			mA
	V_{CC} = 2.5V, V_{OH} = 1.8V	−0.2			mA
Sink	V_{CC} = 4.0V, V_{OL} = 1V	10			mA
	V_{CC} = 2.5V, V_{OL} = 0.4V	2.0			mA
All Others					
Source (Weak Pull-Up Mode)	V_{CC} = 4.0V, V_{OH} = 2.7V	−10		−100	μA
	V_{CC} = 2.5V, V_{OH} = 1.8V	−2.5		−33	μA
Source (Push-Pull Mode)	V_{CC} = 4.0V, V_{OH} = 3.3V	−0.4			mA
	V_{CC} = 2.5V, V_{OH} = 1.8V	−0.2			mA
Sink (Push-Pull Mode)	V_{CC} = 4.0V, V_{OL} = 0.4V	1.6			mA
	V_{CC} = 2.5V, V_{OL} = 0.4V	0.7			mA
TRI-STATE Leakage	V_{CC} = 6.0V	−2		+2	μA
Allowable Sink/Source Current per Pin (Note 6)					
D Outputs (Sink)				15	mA
All others				3	mA
Maximum Input Current without Latchup (Note 4)	Room Temp			±100	mA
RAM Retention Voltage, V_r	500 ns Rise and Fall Time (min)	2			V
Input Capacitance				7	pF
Load Capacitance on D2				1000	pF

AC Electrical Characteristics 88XEK: $-40°C \le T_A \le +85°C$ unless otherwise specified

Parameter	Conditions	Min	Typ	Max	Units
Instruction Cycle Time (t_c)					
Crystal, Resonator,	$4.0V \le V_{CC} \le 6.0V$	1.0		DC	μs
	$2.5V \le V_{CC} < 4.0V$	2.5		DC	μs
R/C Oscillator	$4.0V \le V_{CC} \le 6.0V$	3.0		DC	μs
	$2.5V \le V_{CC} \le 4.0V$	7.5		DC	μs
Inputs					
t_{SETUP}	$4.0V \le V_{CC} \le 6.0V$	200			ns
	$2.5V \le V_{CC} < 4.0V$	500			ns
t_{HOLD}	$4.0V \le V_{CC} \le 6.0V$	60			ns
	$2.5V \le V_{CC} < 4.0V$	150			ns
Output Propagation Delay (Note 5)	$R_L = 2.2k$, $C_L = 100$ pF				
t_{PD1}, t_{PD0}					
SO, SK	$4.0V \le V_{CC} \le 6.0V$			0.7	μs
	$2.5V \le V_{CC} < 4.0V$			1.75	μs
All Others	$4.0V \le V_{CC} \le 6.0V$			1.0	μs
	$2.5V \le V_{CC} < 4.0V$			2.5	μs
MICROWIRE Setup Time (t_{UWS}) (Note 5)		20			ns
MICROWIRE Hold Time (t_{UWH}) (Note 5)		56			ns
MICROWIRE Output Propagation Delay (t_{UPD})				220	ns
Input Pulse Width (Note 6)					
Interrupt Input High Time		1.0			t_c
Interrupt Input Low Time		1.0			t_c
Timer 1, 2, 3 Input High Time		1.0			t_c
Timer 1, 2, 3 Input Low Time		1.0			t_c
Reset Pulse Width		1.0			μs

t_c = Instruction Cycle Time

Note 1: Maximum rate of voltage change must be < 0.5 V/ms.

Note 2: Supply and IDLE currents are measured with CKI driven with a square wave Oscillator, CKO driven 180° out of phase with CKI, inputs connected to V_{CC} and outputs driven low but not connected to a load.

Note 3: The HALT mode will stop CKI from oscillating in the RC and the Crystal configurations. Measurement of I_{DD} HALT is done with device neither sourcing nor sinking current; with L, C, G0, and G2–G5 programmed as low outputs and not driving a load; all outputs programmed low and not driving a load; all inputs tied to V_{CC}; clock monitor and comparator disabled. Parameter refers to HALT mode entered via setting bit 7 of the G Port data register. Part will pull up CKI during HALT in crystal clock mode.

Note 4: Pins G6 and $\overline{\text{RESET}}$ are designed with a high voltage input network. These pins allow input voltages > V_{CC} and the pins will have sink current to V_{CC} when biased at voltages > V_{CC} (the pins do not have source current when biased at a voltage below V_{CC}). The effective resistance to V_{CC} is 750Ω (typical). These two pins will not latch up. The voltage at the pins must be limited to < 14V. **WARNING: Voltages in excess of 14V will cause damage to the pins. This warning excludes ESD transients.**

Note 5: The output propagation delay is referenced to the end of the instruction cycle where the output change occurs.

Note 6: Parameter characterized but not tested.

Absolute Maximum Ratings

If Military/Aerospace specified devices are required, please contact the National Semiconductor Sales Office/Distributors for availability and specifications.

Supply Voltage (V_{CC})	7V
Voltage at Any Pin	$-0.3V$ to $V_{CC} + 0.3V$
Total Current into V_{CC} Pin (Source)	100 mA
Total Current out of GND Pin (Sink)	110 mA
Storage Temperature Range	$-65°C$ to $+140°C$

Note: *Absolute maximum ratings indicate limits beyond which damage to the device may occur. DC and AC electrical specifications are not ensured when operating the device at absolute maximum ratings.*

DC Electrical Characteristics 68XEK: $-55°C \le T_A \le +125°C$ unless otherwise specified

Parameter	Conditions	Min	Typ	Max	Units
Operating Voltage		4.5		5.5	V
Power Supply Ripple (Note 1)	Peak-to-Peak			0.1 V_{CC}	V
Supply Current (Note 2) CKI = 10 MHz	$V_{CC} = 5.5V$, $t_c = 1\ \mu s$			12.5	mA
HALT Current (Note 3)	$V_{CC} = 5.5V$, CKI = 0 MHz		<10	30	μA
IDLE Current (Note 2) CKI = 10 MHz	$V_{CC} = 5.5V$, $t_c = 1\ \mu s$			1.7	mA
Input Levels (V_{IH}, V_{IL})					
$\overline{\text{RESET}}$					
Logic High		0.8 V_{CC}			V
Logic Low				0.2 V_{CC}	V
CKI, All Other Inputs					
Logic High		0.7 V_{CC}			V
Logic Low				0.2 V_{CC}	V
Hi-Z Input Leakage	$V_{CC} = 5.5V$	−5		+5	μA
Input Pullup Current	$V_{CC} = 5.5V$, $V_{IN} = 0V$	−35		−400	μA
G and L Port Input Hysteresis (Note 6)				0.35 V_{CC}	V
Output Current Levels					
D Outputs					
Source	$V_{CC} = 4.5V$, $V_{OH} = 3.3V$	−0.4			mA
Sink	$V_{CC} = 4.5V$, $V_{OL} = 1.0V$	9.0			mA
All Others					
Source (Weak Pull-Up Mode)	$V_{CC} = 4.5V$, $V_{OH} = 2.7V$	−9.0		−110	μA
Source (Push-Pull Mode)	$V_{CC} = 4.5V$, $V_{OH} = 3.3V$	−0.4			mA
Sink (Push-Pull Mode)	$V_{CC} = 4.5V$, $V_{OL} = 0.4V$	1.6			mA
TRI-STATE Leakage	$V_{CC} = 5.5V$	−5.0		+5	μA
Allowable Sink/Source Current per Pin					
D Outputs (Sink)				12	mA
All others				2.5	mA
Maximum Input Current without Latchup (Note 4)	Room Temp			±100	mA
RAM Retention Voltage, V_r	500 ns Rise and and Fall Time (min)	2	1.5		V
Input Capacitance				7	pF
Load Capacitance on D2				1000	pF

AC Electrical Characteristics 68XEK: $-55°C \leq T_A \leq +125°C$ unless otherwise specified

Parameter	Conditions	Min	Typ	Max	Units
Instruction Cycle Time (t_c)					
Crystal, Resonator	$4.5V \leq V_{CC} \leq 5.5V$	1.0		DC	μs
R/C Oscillator	$4.5V \leq V_{CC} < 5.5V$	3.0		DC	μs
Inputs					
t_{SETUP}	$4.5V \leq V_{CC} \leq 5.5V$	200			ns
t_{HOLD}	$4.5V \leq V_{CC} \leq 5.5V$	60			ns
Output Propagation Delay (Note 5)	$R_L = 2.2k, C_L = 100$ pF				
t_{PD1}, t_{PD0}					
SO, SK	$4.5V \leq V_{CC} \leq 5.5V$			0.7	μs
All Others	$4.5V \leq V_{CC} \leq 5.5V$			1.0	μs
MICROWIRE Setup Time (t_{UWS}) (Note 5)	$V_{CC} \geq 4.5V$	20			ns
MICROWIRE Hold Time (t_{UWH}) (Note 5)	$V_{CC} \geq 4.5V$	56			ns
MICROWIRE Output Propagation Delay (t_{UPD})	$V_{CC} \geq 4.5V$			220	ns
Input Pulse Width (Note 6)					
Interrupt Input High Time		1.0			t_c
Interrupt Input Low Time		1.0			t_c
Timer 1, 2, 3 Input High Time		1.0			t_c
Timer 1, 2, 3 Input Low Time		1.0			t_c
Reset Pulse Width		1.0			μs

t_c = Instruction Cycle Time

Note 1: Maximum rate of voltage change must be < 0.5 V/ms.

Note 2: Supply and IDLE currents are measured with CKI driven with a square wave Oscillator, CKO driven 180° out of phase with CKI, inputs connected to V_{CC} and outputs driven low but not connected to a load.

Note 3: The HALT mode will stop CKI from oscillating in the RC and the Crystal configurations. Measurement of I_{DD} HALT is done with device neither sourcing nor sinking current; with L, C, G0, and G2–G5 programmed as low outputs and not driving a load; all outputs programmed low and not driving a load; all inputs tied to V_{CC}; clock monitor and comparator disabled. Parameter refers to HALT mode entered via setting bit 7 of the G Port data register. Part will pull up CKI during HALT in crystal clock mode.

Note 4: Pins G6 and $\overline{\text{RESET}}$ are designed with a high voltage input network. These pins allow input voltages > V_{CC} and the pins will have sink current to V_{CC} when biased at voltages > V_{CC} (the pins do not have source current when biased at a voltage below V_{CC}). The effective resistance to V_{CC} is 750Ω (typical). These two pins will not latch up. The voltage at the pins must be limited to < 14V. **WARNING: Voltages in excess of 14V will cause damage to the pins. This warning excludes ESD transients.**

Note 5: The output propagation delay is referenced to the end of the instruction cycle where the output change occurs.

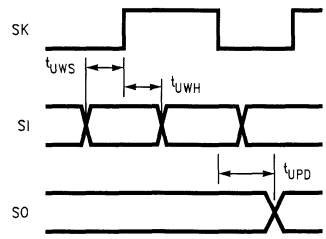
Note 6: Parameter characterized but not tested.

Analog Function Block AC and DC Characteristics V_{CC} = 5.0V, −40°C ≤ T_A ≤ +85°C

Parameter	Conditions	Min	Typ	Max	Units
Input Offset Voltage	0.4V < V_{IN} < V_{CC} − 1.5V		±10	±25	mV
Input Common Mode Voltage Range (Note 7)		0.4		V_{CC} − 1.5	V
V_{CC}/2 Reference	4.0V < V_{CC} < 6.0V	0.5 V_{CC} − 0.04	0.5 V_{CC}	0.5 V_{CC} + 0.04	V
DC Supply Current for Comparator (when enabled)	V_{CC} = 6.0V			250	μA
DC Supply Current for V_{CC}/2 Reference (when enabled)	V_{CC} = 6.0V		50	80	μA
DC Supply Current for Constant Current Source (when enabled)	V_{CC} = 6.0V			200	μA
Constant Current Source	4.0V < V_{CC} < 6.0V	10	20	40	μA
Current Source Variation over Common Mode Range	4.0V < V_{CC} < 6.0V Temp = Constant			±2	μA
Current Source Enable Time			1.5	2	μs
Comparator Response Time	100 mV Overdrive, 100 pF Load			1	μs

While performance characteristics are given at V_{CC} = 5.0V, the analog function block will operate over the entire 2.5V–6.0V V_{CC} range. Accuracy of the V_{CC}/2 reference and the constant current source is not guaranteed beyond the specified limits.

Note 7: The device is capable of operating over a common mode voltage range of 0 to V_{CC} − 1.5V, however increased offset voltage will be observed between 0V and 0.4V.

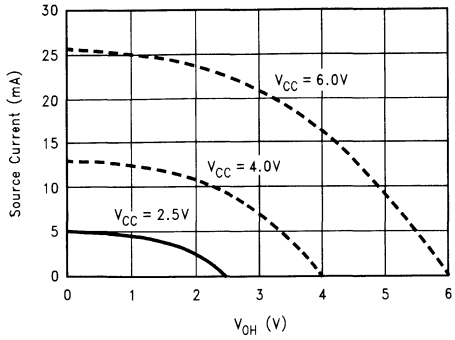


TL/DD/12094–18

FIGURE 3. MICROWIRE/PLUS Timing

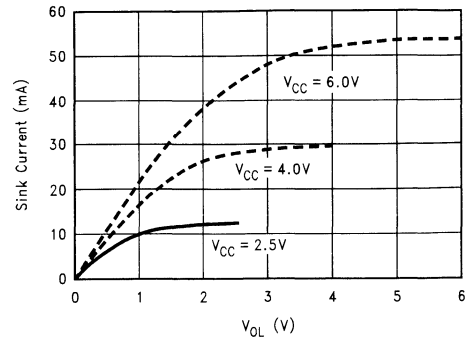
Typical Performance Characteristics ($-55°C \le T_A = +125°C$)

Port D Source Current



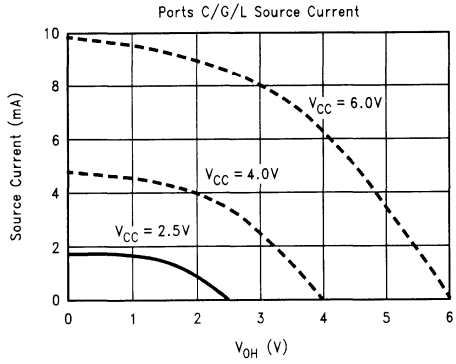
TL/DD/12094–19

Port D Sink Current



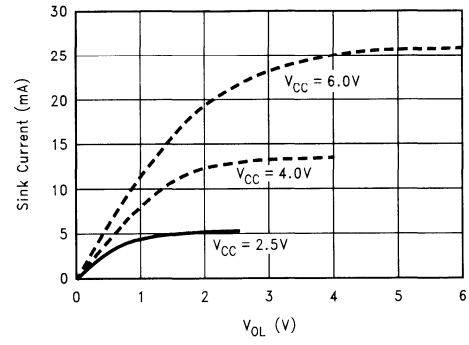
TL/DD/12094–20

Ports C/G/L Source Current



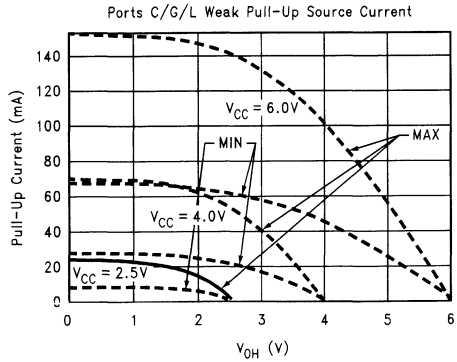
TL/DD/12094–21

Ports C/G/L Sink Current



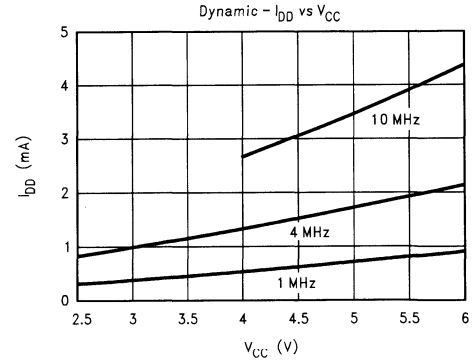
TL/DD/12094–22

Ports C/G/L Weak Pull-Up Source Current



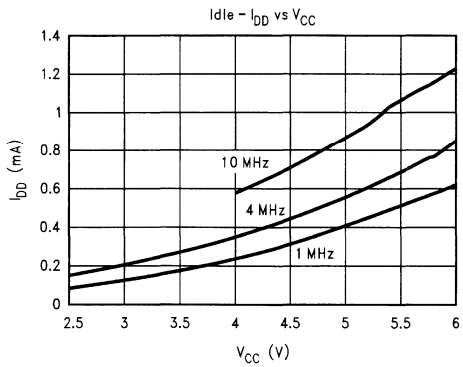
TL/DD/12094–23

Dynamic – I_{DD} vs V_{CC}

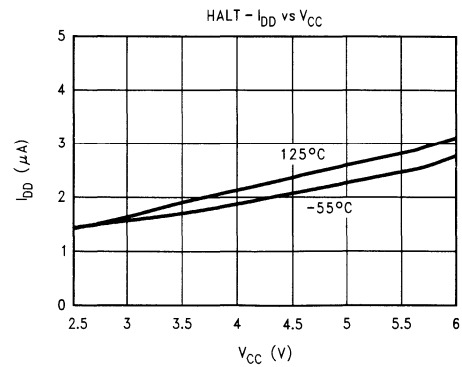


TL/DD/12094–24

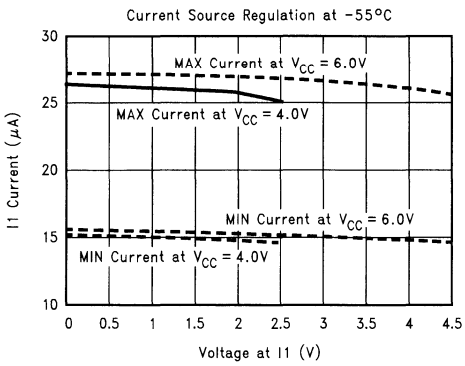
Typical Performance Characteristics ($-55°C \le T_A = +125°C$) (Continued)



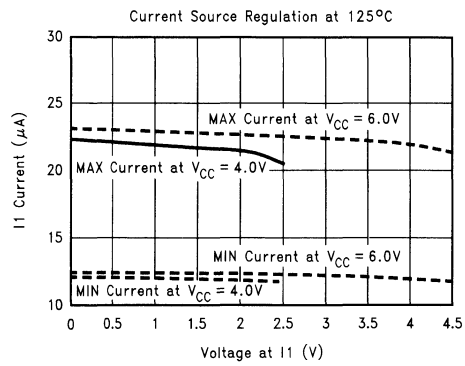
TL/DD/12094-25



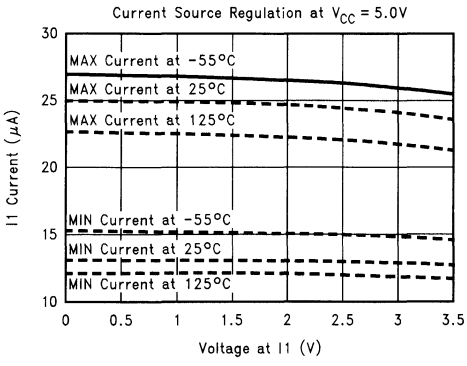
TL/DD/12094-26



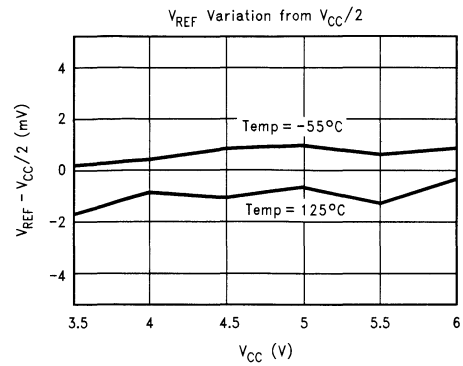
TL/DD/12094-27



TL/DD/12094-28



TL/DD/12094-29



TL/DD/12094-30

5

Pin Descriptions

V_{CC} and GND are the power supply pins. All V_{CC} and GND pins must be connected.

CKI is the clock input. This can come from an R/C generated oscillator, or a crystal oscillator (in conjunction with CKO). See Oscillator Description section.

$\overline{\text{RESET}}$ is the master reset input. See Reset Description section.

The device contains three bidirectional 8-bit I/O ports (C, G and L), where each individual bit may be independently configured as an input (Schmitt Trigger inputs on ports L and G), output or TRI-STATE under program control. Three data memory address locations are allocated for each of these I/O ports. Each I/O port has two associated 8-bit memory mapped registers, the CONFIGURATION register and the output DATA register. A memory mapped address is also reserved for the input pins of each I/O port. (See the memory map for the various addresses associated with the I/O ports.) *Figure 4* shows the I/O port configurations. The DATA and CONFIGURATION registers allow for each port bit to be individually configured under software control as shown below:

CONFIGURATION Register	DATA Register	Port Set-Up
0	0	Hi-Z Input (TRI-STATE Output)
0	1	Input with Weak Pull-Up
1	0	Push-Pull Zero Output
1	1	Push-Pull One Output

PORT L is an 8-bit I/O port. All L-pins have Schmitt triggers on the inputs.

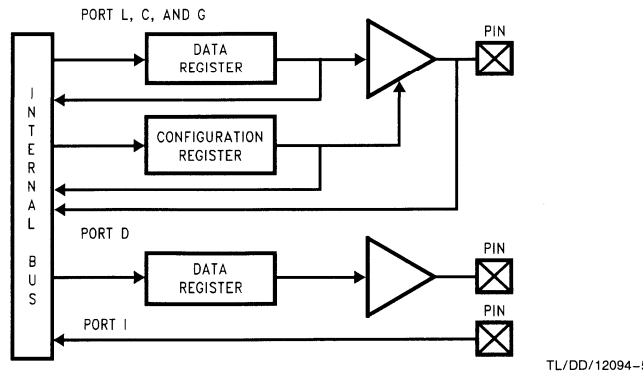
The Port L supports Multi-Input Wake Up on all eight pins. L4 and L5 are used for the timer input functions T2A and T2B. L6 and L7 are used for the timer input functions T3A and T3B.

The Port L has the following alternate features:

- L0 MIWU
- L1 MIWU
- L2 MIWU
- L3 MIWU
- L4 MIWU or T2A
- L5 MIWU or T2B
- L6 MIWU or T3A
- L7 MIWU or T3B

Port G is an 8-bit port with 5 I/O pins (G0, G2–G5), an input pin (G6), and a dedicated output pin (G7). Pins G0 and G2–G6 all have Schmitt Triggers on their inputs. Pin G1 serves as the dedicated WDOUT WATCHDOG output, while pin G7 is either input or output depending on the oscillator mask option selected. With the crystal oscillator option selected, G7 serves as the dedicated output pin for the CKO clock output. With the single-pin R/C oscillator mask option selected, G7 serves as a general purpose input pin but is also used to bring the device out of HALT mode with a low to high transition on G7. There are two registers associated with the G Port, a data register and a configuration register. Therefore, each of the 5 I/O bits (G0, G2–G5) can be individually configured under software control.

Since G6 is an input only pin and G7 is the dedicated CKO clock output pin (crystal clock option) or general purpose input (R/C clock option), the associated bits in the data and configuration registers for G6 and G7 are used for special purpose functions as outlined on the next page. Reading the G6 and G7 data bits will return zeros.



TL/DD/12094–5

FIGURE 4. I/O Port Configurations

Pin Descriptions (Continued)

Note that the chip will be placed in the HALT mode by writing a "1" to bit 7 of the Port G Data Register. Similarly the chip will be placed in the IDLE mode by writing a "1" to bit 6 of the Port G Data Register.

Writing a "1" to bit 6 of the Port G Configuration Register enables the MICROWIRE/PLUS to operate with the alternate phase of the SK clock. The G7 configuration bit, if set high, enables the clock start up delay after HALT when the R/C clock configuration is used.

	Config Reg.	Data Reg.
G7	CLKDLY	HALT
G6	Alternate SK	IDLE

Port G has the following alternate features:

- G0 INTR (External Interrupt Input)
- G2 T1B (Timer T1 Capture Input)
- G3 T1A (Timer T1 I/O)
- G4 SO (MICROWIRE Serial Data Output)
- G5 SK (MICROWIRE Serial Clock)
- G6 SI (MICROWIRE Serial Data Input)

Port G has the following dedicated functions:

- G1 WDOUT WATCHDOG and/or Clock Monitor dedicated output
- G7 CKO Oscillator dedicated output or general purpose input

Port C is an 8-bit I/O port. The 40-pin device does not have a full complement of Port C pins. The unavailable pins are not terminated. A read operation for these unterminated pins will return unpredicatable values.

PORT I is an eight-bit Hi-Z input port. The 28-pin device does not have a full complement of Port I pins. The unavailable pins are not terminated i.e., they are floating. A read operation for these unterminated pins will return unpredictable values. The user must ensure that the software takes this into account by either masking or restricting the accesses to bit operations. The unterminated Port I pins will draw power only when addressed.

Port I is an eight-bit Hi-Z input port.

Port I0–I7 are used for the analog function block.

The Port I has the following alternate features:

- I0 COMPIN1+ (Comparator Positive Input 1)
- I1 COMPIN− (Comparator Negative Input/Current Source Out)
- I2 COMPIN0+ (Comparator Positive Input 0)
- I3 COMPOUT/COMPIN2+ (Comparator Output/ Comparator Positive Input 2))
- I4 COMPIN3+ (Comparator Positive Input 3)
- I5 COMPIN4+ (Comparator Positive Input 4)
- I6 COMPIN5+ (Comparator Positive Input 5)
- I7 COMPOUT (Comparator Output)

Port D is an 8-bit output port that is preset high when $\overline{\text{RESET}}$ goes low. The user can tie two or more D port outputs (except D2) together in order to get a higher drive.

Note: Care must be exercised with the D2 pin operation. At RESET, the external loads on this pin must ensure that the output voltages stay above 0.8 V_{CC} to prevent the chip from entering special modes. Also keep the external loading on D2 to <1000 pF.

Functional Description

The architecture of the device is modified Harvard architecture. With the Harvard architecture, the control store program memory (ROM) is separated from the data store memory (RAM). Both ROM and RAM have their own separate addressing space with separate address buses. The architecture, though based on Harvard architecture, permits transfer of data from ROM to RAM.

CPU REGISTERS

The CPU can do an 8-bit addition, subtraction, logical or shift operation in one instruction (t_c) cycle time.

There are six CPU registers:

A is the 8-bit Accumulator Register

PC is the 15-bit Program Counter Register

PU is the upper 7 bits of the program counter (PC)
PL is the lower 8 bits of the program counter (PC)

B is an 8-bit RAM address pointer, which can be optionally post auto incremented or decremented.

X is an 8-bit alternate RAM address pointer, which can be optionally post auto incremented or decremented.

SP is the 8-bit stack pointer, which points to the subroutine/ interrupt stack (in RAM). The SP is initialized to RAM address 06F with reset.

S is the 8-bit Data Segment Address Register used to extend the lower half of the address range (00 to 7F) into 256 data segments of 128 bytes each.

All the CPU registers are memory mapped with the exception of the Accumulator (A) and the Program Counter (PC).

PROGRAM MEMORY

The program memory consists of 8192 bytes of ROM. These bytes may hold program instructions or constant data (data tables for the LAID instruction, jump vectors for the JID instruction, and interrupt vectors for the VIS instruction). The program memory is addressed by the 15-bit program counter (PC). All interrupts in the devices vector to program memory location 0FF Hex.

DATA MEMORY

The data memory address space includes the on-chip RAM and data registers, the I/O registers (Configuration, Data and Pin), the control registers, the MICROWIRE/PLUS SIO shift register, and the various registers, and counters associated with the timers (with the exception of the IDLE timer). Data memory is addressed directly by the instruction or indirectly by the B, X, SP pointers and S register.

The data memory consists of 256 bytes of RAM. Sixteen bytes of RAM are mapped as "registers" at addresses 0F0 to 0FF Hex. These registers can be loaded immediately, and also decremented and tested with the DRSZ (decrement register and skip if zero) instruction. The memory pointer registers X, SP, B and S are memory mapped into this space at address locations 0FC to 0FF Hex respectively, with the other registers being available for general usage.

The instruction set permits any bit in memory to be set, reset or tested. All I/O and registers (except A and PC) are memory mapped; therefore, I/O bits and register bits can be directly and individually set, reset and tested. The accumulator (A) bits can also be directly and individually tested.

Note: RAM contents are undefined upon power-up.

Data Memory Segment RAM Extension

Data memory address 0FF is used as a memory mapped location for the Data Segment Address Register (S).

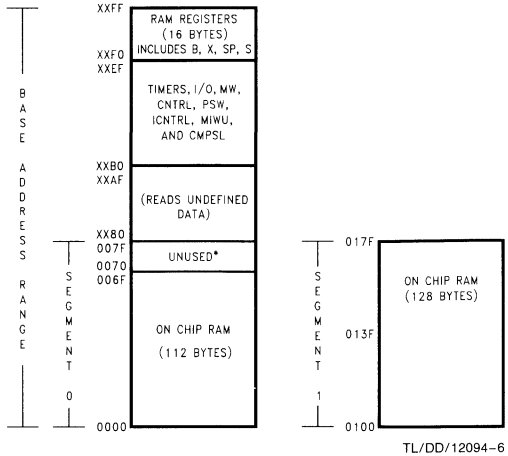
The data store memory is either addressed directly by a single byte address within the instruction, or indirectly relative to the reference of the B, X, or SP pointers (each contains a single-byte address). This single-byte address allows an addressing range of 256 locations from 00 to FF hex. The upper bit of this single-byte address divides the data store memory into two separate sections as outlined previously. With the exception of the RAM register memory from address locations 00F0 to 00FF, all RAM memory is memory mapped with the upper bit of the single-byte address being equal to zero. This allows the upper bit of the single-byte address to determine whether or not the base address range (from 0000 to 00FF) is extended. If this upper bit equals one (representing address range 0080 to 00FF), then address extension does not take place. Alternatively, if this upper bit equals zero, then the data segment extension register S is used to extend the base address range (from 0000 to 007F) from XX00 to XX7F, where XX represents the 8 bits from the S register. Thus the 128-byte data segment extensions are located from addresses 0100 to 017F for data segment 1, 0200 to 027F for data segment 2, etc., up to FF00 to FF7F for data segment 255. The base address range from 0000 to 007F represents data segment 0.

Figure 5 illustrates how the S register data memory extension is used in extending the lower half of the base address range (00 to 7F hex) into 256 data segments of 128 bytes each, with a total addressing range of 32 kbytes from XX00 to XX7F. This organization allows a total of 256 data segments of 128 bytes each with an additional upper base segment of 128 bytes. Furthermore, all addressing modes are available for all data segments. The S register must be changed under program control to move from one data segment (128 bytes) to another. However, the upper base segment (containing the 16 memory registers, I/O registers, control registers, etc.) is always available regardless of the contents of the S register, since the upper base segment (address range 0080 to 00FF) is independent of data segment extension.

The instructions that utilize the stack pointer (SP) always reference the stack as part of the base segment (Segment 0), regardless of the contents of the S register. The S register is not changed by these instructions. Consequently, the stack (used with subroutine linkage and interrupts) is always located in the base segment. The stack pointer will be initialized to point at data memory location 006F as a result of reset.

The 128 bytes of RAM contained in the base segment are split between the lower and upper base segments. The first 112 bytes of RAM are resident from address 0000 to 006F in the lower base segment, while the remaining 16 bytes of RAM represent the 16 data memory registers located at addresses 00F0 to 00FF of the upper base segment. No RAM is located at the upper sixteen addresses (0070 to 007F) of the lower base segment.

Additional RAM beyond these initial 128 bytes, however, will always be memory mapped in groups of 128 bytes (or less) at the data segment address extensions (XX00 to XX7F) of the lower base segment. The additional 128 bytes of RAM are memory mapped at address locations 0100 to 017F hex.



*Reads as all ones.

FIGURE 5. RAM Organization

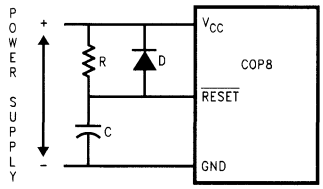
Reset

The $\overline{\text{RESET}}$ input when pulled low initializes the microcontroller. Initialization will occur whenever the $\overline{\text{RESET}}$ input is pulled low. Upon initialization, the data and configuration registers for ports L, G and C are cleared, resulting in these Ports being initialized to the TRI-STATE mode. Pin G1 of the G Port is an exception (as noted below) since pin G1 is dedicated as the WATCHDOG and/or Clock Monitor error output pin. Port D is set high. The PC, PSW, ICNTRL, CNTRL, T2CNTRL and T3CNTRL control registers are cleared. The Comparator Select Register is cleared. The S register is initialized to zero. The Multi-Input Wakeup registers WKEN and WKEDG are cleared. Wakeup register WKPND is unknown. The stack pointer, SP, is initialized to 6F hex.

The device comes out of reset with both the WATCHDOG logic and the Clock Monitor detector armed, with the WATCHDOG service window bits set and the Clock Monitor bit set. The WATCHDOG and Clock Monitor circuits are inhibited during reset. The WATCHDOG service window bits being initialized high default to the maximum WATCHDOG service window of 64k t_C clock cycles. The Clock Monitor bit being initialized high will cause a Clock Monitor error following reset if the clock has not reached the minimum specified frequency at the termination of reset. A Clock Monitor error will cause an active low error output on pin G1. This error output will continue until 16 t_C–32 t_C clock cycles following the clock frequency reaching the minimum specified value, at which time the G1 output will enter the TRI-STATE mode.

The external RC network shown in *Figure 6* should be used to ensure that the $\overline{\text{RESET}}$ pin is held low until the power supply to the chip stabilizes.

Reset (Continued)



TL/DD/12094–7

RC > 5 × Power Supply Rise Time

FIGURE 6. Recommended Reset Circuit

Oscillator Circuits

The chip can be driven by a clock input on the CKI input pin which can be between DC and 10 MHz. The CKO output clock is on pin G7 (crystal configuration). The CKI input frequency is divided down by 10 to produce the instruction cycle clock ($1/t_c$).

Note: External clocks with frequencies above about 4 MHz require the user to drive the CKO (G7) pin with a signal 180 degrees out of phase with CKI.

Figure 7 shows the Crystal and R/C oscillator diagrams.

CRYSTAL OSCILLATOR

CKI and CKO can be connected to make a closed loop crystal (or resonator) controlled oscillator.

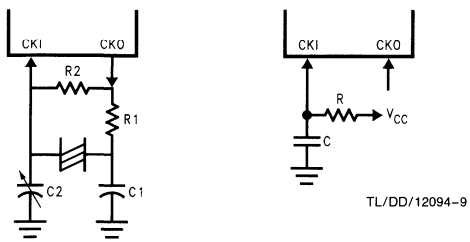
Table A shows the component values required for various standard crystal values.

R/C OSCILLATOR

By selecting CKI as a single pin oscillator input, a single pin R/C oscillator circuit can be connected to it. CKO is available as a general purpose input, and/or HALT restart input.

Note: Use of the R/C oscillator option will result in higher electromagnetic emissions.

Table B shows the variation in the oscillator frequencies as functions of the component (R and C) values.



TL/DD/12094–8

TL/DD/12094–9

FIGURE 7. Crystal and R/C Oscillator Diagrams

TABLE A. Crystal Oscillator Configuration, T_A = 25°C

R1 (kΩ)	R2 (MΩ)	C1 (pF)	C2 (pF)	CKI Freq (MHz)	Conditions
0	1	30	30–36	10	V_{CC} = 5V
0	1	30	30–36	4	V_{CC} = 5V
0	1	200	100–150	0.455	V_{CC} = 5V

TABLE B. RC Oscillator Configuration, T_A = 25°C

R (kΩ)	C (pF)	CKI Freq (MHz)	Instr. Cycle (μs)	Conditions
3.3	82	2.2 to 2.7	3.7 to 4.6	V_{CC} = 5V
5.6	100	1.1 to 1.3	7.4 to 9.0	V_{CC} = 5V
6.8	100	0.9 to 1.1	8.8 to 10.8	V_{CC} = 5V

Note: 3k ≤ R ≤ 200k

50 pF ≤ C ≤ 200 pF

Control Registers

CNTRL Register (Address X'00EE)

The Timer1 (T1) and MICROWIRE/PLUS control register contains the following bits:

SL1 & SL0 Select the MICROWIRE/PLUS clock divide by (00 = 2, 01 = 4, 1x = 8)

IEDG External interrupt edge polarity select (0 = Rising edge, 1 = Falling edge)

MSEL Selects G5 and G4 as MICROWIRE/PLUS signals SK and SO respectively

T1C0 Timer T1 Start/Stop control in timer modes 1 and 2

Timer T1 Underflow Interrupt Pending Flag in timer mode 3

T1C1 Timer T1 mode control bit

T1C2 Timer T1 mode control bit

T1C3 Timer T1 mode control bit

T1C3	T1C2	T1C1	T1C0	MSEL	IEDG	SL1	SL0

Bit 7 Bit 0

Control Registers (Continued)

PSW Register (Address X'00EF)

The PSW register contains the following select bits:

- GIE Global interrupt enable (enables interrupts)
- EXEN Enable external interrupt
- BUSY MICROWIRE/PLUS busy shifting flag
- EXPND External interrupt pending
- T1ENA Timer T1 Interrupt Enable for Timer Underflow or T1A Input capture edge
- T1PNDA Timer T1 Interrupt Pending Flag (Autoreload RA in mode 1, T1 Underflow in Mode 2, T1A capture edge in mode 3)
- C Carry Flag
- HC Half Carry Flag

HC	C	T1PNDA	T1ENA	EXPND	BUSY	EXEN	GIE

Bit 7 Bit 0

The Half-Carry bit is also affected by all the instructions that affect the Carry flag. The SC (Set Carry) and RC (Reset Carry) instructions will respectively set or clear both the carry flags. In addition to the SC and RC instructions, ADC, SUBC, RRC and RLC instructions affect the carry and Half Carry flags.

ICNTRL Register (Address X'00E8)

The ICNTRL register contains the following bits:

- T1ENB Timer T1 Interrupt Enable for T1B Input capture edge
- T1PNDB Timer T1 Interrupt Pending Flag for T1B capture edge
- μWEN Enable MICROWIRE/PLUS interrupt
- μWPND MICROWIRE/PLUS interrupt pending
- T0EN Timer T0 Interrupt Enable (Bit 12 toggle)
- T0PND Timer T0 Interrupt pending
- LPEN L Port Interrupt Enable (Multi-Input Wakeup/Interrupt)

  Bit 7 could be used as a flag

Unused	LPEN	T0PND	T0EN	μWPND	μWEN	T1PNDB	T1ENB

Bit 7 Bit 0

T2CNTRL Register (Address X'00C6)

The T2CNTRL register contains the following bits:

- T2ENB Timer T2 Interrupt Enable for T2B Input capture edge
- T2PNDB Timer T2 Interrupt Pending Flag for T2B capture edge
- T2ENA Timer T2 Interrupt Enable for Timer Underflow or T2A Input capture edge
- T2PNDA Timer T2 Interrupt Pending Flag (Autoreload RA in mode 1, T2 Underflow in mode 2, T2A capture edge in mode 3)
- T2C0 Timer T2 Start/Stop control in timer modes 1 and 2 Timer T2 Underflow Interrupt Pending Flag in timer mode 3
- T2C1 Timer T2 mode control bit
- T2C2 Timer T2 mode control bit
- T2C3 Timer T2 mode control bit

T2C3	T2C2	T2C1	T2C0	T2PNDA	T2ENA	T2PNDB	T2ENB

Bit 7 Bit 0

T3CNTRL Register (Address X'00B6)

The T3CNTRL register contains the following bits:

- T3ENB Timer T3 Interrupt Enable for T3B
- T3PNDB Timer T3 Interrupt Pending Flag for T3B pin (T3B capture edge)
- T3ENA Timer T3 Interrupt Enable for Timer Underflow or T3A pin
- T3PNDA Timer T3 Interrupt Pending Flag (Autoload RA in mode 1, T3 Underflow in mode 2, T3a capture edge in mode 3)
- T3C0 Timer T3 Start/Stop control in timer modes 1 and 2

  Timer T3 Underflow Interrupt Pending Flag in timer mode 3
- T3C1 Timer T3 mode control bit
- T3C2 Timer T3 mode control bit
- T3C3 Timer T3 mode control bit

T3C3	T3C2	T3C1	T3C0	T3PNDA	T3ENA	T3PNDB	T3ENB

Bit 7 Bit 0

Timers

The device contains a very versatile set of timers (T0, T1, T2, T3). All timers and associated autoreload/capture registers power up containing random data.

TIMER T0 (IDLE TIMER)

The device supports applications that require maintaining real time and low power with the IDLE mode. This IDLE mode support is furnished by the IDLE timer T0, which is a 16-bit timer. The Timer T0 runs continuously at the fixed rate of the instruction cycle clock, t_C. The user cannot read or write to the IDLE Timer T0, which is a count down timer.

The Timer T0 supports the following functions:

- Exit out of the Idle Mode (See Idle Mode description)
- WATCHDOG logic (See WATCHDOG description)
- Start up delay out of the HALT mode

The IDLE Timer T0 can generate an interrupt when the thirteenth bit toggles. This toggle is latched into the T0PND pending flag, and will occur every 4 ms at the maximum clock frequency (t_C = 1 μs). A control flag T0EN allows the interrupt from the thirteenth bit of Timer T0 to be enabled or disabled. Setting T0EN will enable the interrupt, while resetting it will disable the interrupt.

Timers (Continued)

TIMER T1, TIMER T2 AND TIMER T3

The device has a set of three powerful timer/counter blocks, T1, T2 and T3. The associated features and functioning of a timer block are described by referring to the timer block Tx. Since the three timer blocks, T1, T2 and T3 are identical, all comments are equally applicable to any of the three timer blocks.

Each timer block consists of a 16-bit timer, Tx, and two supporting 16-bit autoreload/capture registers, RxA and RxB. Each timer block has two pins associated with it, TxA and TxB. The pin TxA supports I/O required by the timer block, while the pin TxB is an input to the timer block. The powerful and flexible timer block allows the device to easily perform all timer functions with minimal software overhead. The timer block has three operating modes: Processor Independent PWM mode, External Event Counter mode, and Input Capture mode.

The control bits TxC3, TxC2, and TxC1 allow selection of the different modes of operation.

Mode 1. Processor Independent PWM Mode

As the name suggests, this mode allows the device to generate a PWM signal with very minimal user intervention. The user only has to define the parameters of the PWM signal (ON time and OFF time). Once begun, the timer block will continuously generate the PWM signal completely independent of the microcontroller. The user software services the timer block only when the PWM parameters require updating.

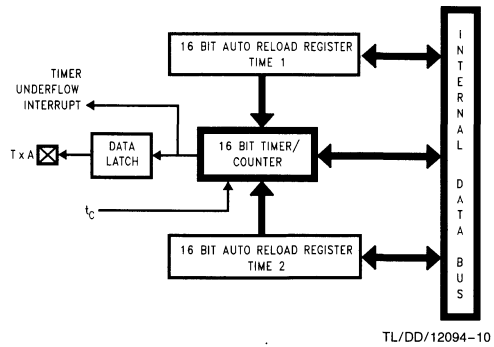
In this mode the timer Tx counts down at a fixed rate of t_C. Upon every underflow the timer is alternately reloaded with the contents of supporting registers, RxA and RxB. The very first underflow of the timer causes the timer to reload from the register RxA. Subsequent underflows cause the timer to be reloaded from the registers alternately beginning with the register RxB.

The Tx Timer control bits, TxC3, TxC2 and TxC1 set up the timer for PWM mode operation.

Figure 8 shows a block diagram of the timer in PWM mode. The underflows can be programmed to toggle the TxA output pin. The underflows can also be programmed to generate interrupts.

Underflows from the timer are alternately latched into two pending flags, TxPNDA and TxPNDB. The user must reset these pending flags under software control. Two control enable flags, TxENA and TxENB, allow the interrupts from the timer underflow to be enabled or disabled. Setting the timer enable flag TxENA will cause an interrupt when a timer underflow causes the RxA register to be reloaded into the timer. Setting the timer enable flag TxENB will cause an interrupt when a timer underflow causes the RxB register to be reloaded into the timer. Resetting the timer enable flags will disable the associated interrupts.

Either or both of the timer underflow interrupts may be enabled. This gives the user the flexibility of interrupting once per PWM period on either the rising or falling edge of the PWM output. Alternatively, the user may choose to interrupt on both edges of the PWM output.



TL/DD/12094–10

FIGURE 8. Timer in PWM Mode

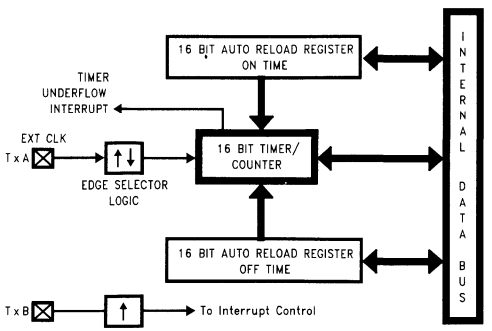
Mode 2. External Event Counter Mode

This mode is quite similar to the processor independent PWM mode described above. The main difference is that the timer, Tx, is clocked by the input signal from the TxA pin. The Tx timer control bits, TxC3, TxC2 and TxC1 allow the timer to be clocked either on a positive or negative edge from the TxA pin. Underflows from the timer are latched into the TxPNDA pending flag. Setting the TxENA control flag will cause an interrupt when the timer underflows.

In this mode the input pin TxB can be used as an independent positive edge sensitive interrupt input if the TxENB control flag is set. The occurrence of a positive edge on the TxB input pin is latched into the TxPNDB flag.

Figure 9 shows a block diagram of the timer in External Event Counter mode.

Note: The PWM output is not available in this mode since the TxA pin is being used as the counter input clock.



TL/DD/12094–11

FIGURE 9. Timer in External Event Counter Mode

Mode 3. Input Capture Mode

The device can precisely measure external frequencies or time external events by placing the timer block, Tx, in the input capture mode.

In this mode, the timer Tx is constantly running at the fixed t_C rate. The two registers, RxA and RxB, act as capture registers. Each register acts in conjunction with a pin. The register RxA acts in conjunction with the TxA pin and the register RxB acts in conjunction with the TxB pin.

Timers (Continued)

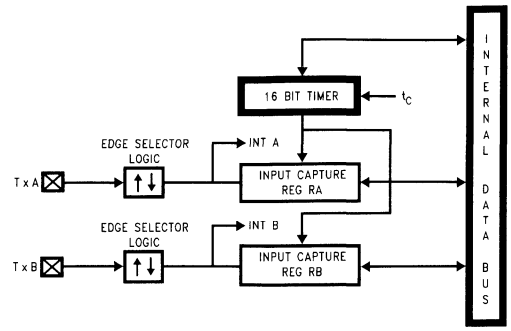
The timer value gets copied over into the register when a trigger event occurs on its corresponding pin. Control bits, TxC3, TxC2 and TxC1, allow the trigger events to be specified either as a positive or a negative edge. The trigger condition for each input pin can be specified independently.

The trigger conditions can also be programmed to generate interrupts. The occurrence of the specified trigger condition on the TxA and TxB pins will be respectively latched into the pending flags, TxPNDA and TxPNDB. The control flag TxENA allows the interrupt on TxA to be either enabled or disabled. Setting the TxENA flag enables interrupts to be generated when the selected trigger condition occurs on the TxA pin. Similarly, the flag TxENB controls the interrupts from the TxB pin.

T2 has additional flexibility because T2B can be internally connected to the comparator output of the Analog Function Block. This allows the user to capture the time of a comparator event without using external pins.

Underflows from the timer can also be programmed to generate interrupts. Underflows are latched into the timer TxC0 pending flag (the TxC0 control bit serves as the timer underflow interrupt pending flag in the Input Capture mode). Consequently, the TxC0 control bit should be reset when entering the Input Capture mode. The timer underflow interrupt is enabled with the TxENA control flag. When a TxA interrupt occurs in the Input Capture mode, the user must check both the TxPNDA and TxC0 pending flags in order to determine whether a TxA input capture or a timer underflow (or both) caused the interrupt.

Figure 10 shows a block diagram of the timer in Input Capture mode.



TL/DD/12094–12

FIGURE 10. Timer in Input Capture Mode

TIMER CONTROL FLAGS

The timers T1, T2 and T3 have indentical control structures. The control bits and their functions are summarized below.

TxC0	Timer Start/Stop control in Modes 1 and 2 (Processor Independent PWM and External Event Counter), where 1 = Start, 0 = Stop Timer Underflow Interrupt Pending Flag in Mode 3 (Input Capture)
TxPNDA	Timer Interrupt Pending Flag
TxPNDB	Timer Interrupt Pending Flag
TxENA	Timer Interrupt Enable Flag
TxENB	Timer Interrupt Enable Flag 1 = Timer Interrupt Enabled 0 = Timer Interrupt Disabled
TxC3	Timer mode control
TxC2	Timer mode control
TxC1	Timer mode control

Timers (Continued)

The timer mode control bits (TxC3, TxC2 and TxC1) are detailed below:

TxC3	TxC2	TxC1	Timer Mode	Interrupt A Source	Interrupt B Source	Timer Counts On
0	0	0	MODE 2 (External Event Counter)	Timer Underflow	Pos. TxB Edge	TxA Pos. Edge
0	0	1	MODE 2 (External Event Counter)	Timer Underflow	Pos. TxB Edge	TxA Neg. Edge
1	0	1	MODE 1 (PWM) TxA Toggle	Autoreload RA	Autoreload RB	t_C
1	0	0	MODE 1 (PWM) No TxA Toggle	Autoreload RA	Autoreload RB	t_C
0	1	0	MODE 3 (Capture) Captures: TxA Pos. Edge TxB Pos. Edge	Pos. TxA Edge or Timer Underflow	Pos. TxB Edge	t_C
1	1	0	MODE 3 (Capture) Captures: TxA Pos. Edge TxB Neg. Edge	Pos. TxA Edge or Timer Underflow	Neg. TxB Edge	t_C
0	1	1	MODE 3 (Capture) Captures: TxA Neg. Edge TxB Pos. Edge	Neg. TxA Edge or Timer Underflow	Pos. TxB Edge	t_C
1	1	1	MODE 3 (Capture) Captures: TxA Neg. Edge TxB Neg. Edge	Neg. TxA Edge or Timer Underflow	Neg. TxB Edge	t_C

Power Save Modes

The device offers the user two power save modes of operation: HALT and IDLE. In the HALT mode, all microcontroller activities are stopped. In the IDLE mode, the on-board oscillator circuitry the WATCHDOG logic, the Clock Monitor and timer T0 are active but all other microcontroller activities are stopped. In either mode, all on-board RAM, registers, I/O states, and timers (with the exception of T0) are unaltered.

HALT MODE

The device can be placed in the HALT mode by writing a "1" to the HALT flag (G7 data bit). All microcontroller activities, including the clock and timers, are stopped. The WATCHDOG logic is disabled during the HALT mode. However, the clock monitor circuitry if enabled remains active and will cause the WATCHDOG output pin (WDOUT) to go low. If the HALT mode is used and the user does not want to activate the WDOUT pin, the Clock Monitor should be disabled after the device comes out of reset (resetting the Clock Monitor control bit with the first write to the WDSVR register). In the HALT mode, the power requirements of the device are minimal and the applied voltage (V_{CC}) may be decreased to V_r (V_r = 2.0V) without altering the state of the machine.

The device supports three different ways of exiting the HALT mode. The first method of exiting the HALT mode is with the Multi-Input Wakeup feature on the L port. The second method is with a low to high transition on the CKO (G7) pin. This method precludes the use of the crystal clock configuration (since CKO becomes a dedicated output), and so may be used with an RC clock configuration. The third method of exiting the HALT mode is by pulling the $\overline{\text{RESET}}$ pin low.

Since a crystal or ceramic resonator may be selected as the oscillator, the Wakeup signal is not allowed to start the chip running immediately since crystal oscillators and ceramic resonators have a delayed start up time to reach full amplitude and frequency stability. The IDLE timer is used to generate a fixed delay to ensure that the oscillator has indeed stabilized before allowing instruction execution. In this case, upon detecting a valid Wakeup signal, only the oscillator circuitry is enabled. The IDLE timer is loaded with a value of 256 and is clocked with the t_C instruction cycle clock. The t_C clock is derived by dividing the oscillator clock down by a factor of 10. The Schmitt trigger following the CKI inverter on the chip ensures that the IDLE timer is clocked only when the oscillator has a sufficiently large amplitude to meet the Schmitt trigger specifications. This Schmitt trigger is not part of the oscillator closed loop. The startup timeout from the IDLE timer enables the clock signals to be routed to the rest of the chip.

If an RC clock option is being used, the fixed delay is introduced optionally. A control bit, CLKDLY, mapped as configuration bit G7, controls whether the delay is to be introduced or not. The delay is included if CLKDLY is set, and excluded if CLKDLY is reset. The CLKDLY bit is cleared on reset.

Power Save Modes (Continued)

The device has two mask options associated with the HALT mode. The first mask option enables the HALT mode feature, while the second mask option disables the HALT mode. With the HALT mode enable mask option, the device will enter and exit the HALT mode as described above. With the HALT disable mask option, the device cannot be placed in the HALT mode (writing a "1" to the HALT flag will have no effect, the HALT flag will remain "0").

The WATCHDOG detector circuit is inhibited during the HALT mode. However, the clock monitor circuit if enabled remains active during HALT mode in order to ensure a clock monitor error if the device inadvertently enters the HALT mode as a result of a runaway program or power glitch.

IDLE MODE

The device is placed in the IDLE mode by writing a "1" to the IDLE flag (G6 data bit). In this mode, all activities, except the associated on-board oscillator circuitry, the WATCHDOG logic, the clock monitor and the IDLE Timer T0, are stopped.

As with the HALT mode, the device can be returned to normal operation with a reset, or with a Multi-Input Wakeup from the L Port. Alternately, the microcontroller resumes normal operation from the IDLE mode when the thirteenth bit (representing 4.096 ms at internal clock frequency of 1 MHz, $t_C = 1\ \mu s$) of the IDLE Timer toggles.

This toggle condition of the thirteenth bit of the IDLE Timer T0 is latched into the T0PND pending flag.

The user has the option of being interrupted with a transition on the thirteenth bit of the IDLE Timer T0. The interrupt can be enabled or disabled via the T0EN control bit. Setting the T0EN flag enables the interrupt and vice versa.

The user can enter the IDLE mode with the Timer T0 interrupt enabled. In this case, when the T0PND bit gets set, the device will first execute the Timer T0 interrupt service routine and then return to the instruction following the "Enter Idle Mode" instruction.

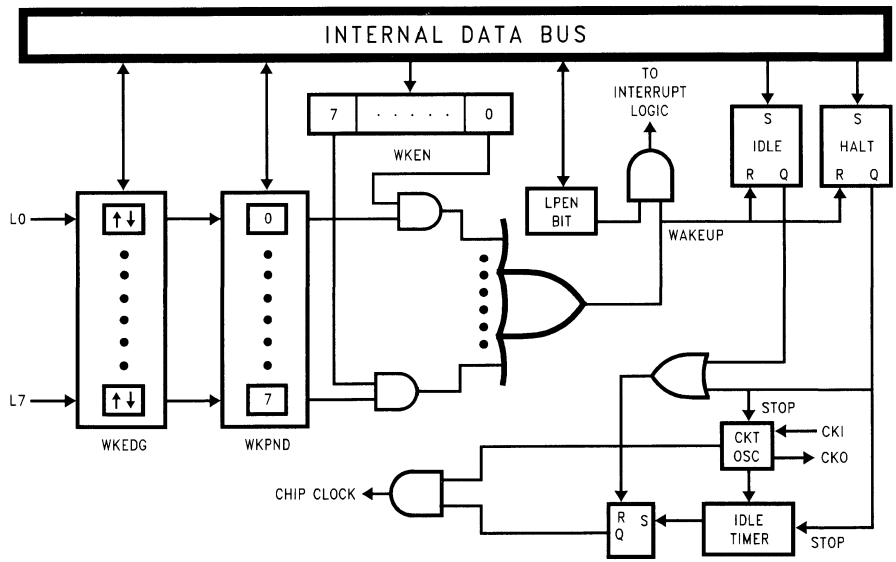
Alternatively, the user can enter the IDLE mode with the IDLE Timer T0 interrupt disabled. In this case, the device will resume normal operation with the instruction immediately following the "Enter IDLE Mode" instruction.

Note: It is necessary to program two NOP instructions following both the set HALT mode and set IDLE mode instructions. These NOP instructions are necessary to allow clock resynchronization following the HALT or IDLE modes.

Multi-Input Wakeup

The Multi-Input Wakeup feature is ued to return (wakeup) the device from either the HALT or IDLE modes. Alternately Multi-Input Wakeup/Interrupt feature may also be used to generate up to 8 edge selectable external interrupts.

Figure 11 shows the Multi-Input Wakeup logic.



TL/DD/12094-13

FIGURE 11. Multi-Input Wake Up Logic

Multi-Input Wakeup (Continued)

The Multi-Input Wakeup feature utilizes the L Port. The user selects which particular L port bit (or combination of L Port bits) will cause the device to exit the HALT or IDLE modes. The selection is done through the Reg: WKEN. The Reg: WKEN is an 8-bit read/write register, which contains a control bit for every L port bit. Setting a particular WKEN bit enables a Wakeup from the associated L port pin.

The user can select whether the trigger condition on the selected L Port pin is going to be either a positive edge (low to high transition) or a negative edge (high to low transition). This selection is made via the Reg: WKEDG, which is an 8-bit control register with a bit assigned to each L Port pin. Setting the control bit will select the trigger condition to be a negative edge on that particular L Port pin. Resetting the bit selects the trigger condition to be a positive edge. Changing an edge select entails several steps in order to avoid a pseudo Wakeup condition as a result of the edge change. First, the associated WKEN bit should be reset, followed by the edge select change in WKEDG. Next, the associated WKPND bit should be cleared, followed by the associated WKEN bit being re-enabled.

An example may serve to clarify this procedure. Suppose we wish to change the edge select from positive (low going high) to negative (high going low) for L Port bit 5, where bit 5 has previously been enabled for an input interrupt. The program would be as follows:

```
RBIT  5, WKEN
SBIT  5, WKEDG
RBIT  5, WKPND
SBIT  5, WKEN
```

If the L port bits have been used as outputs and then changed to inputs with Multi-Input Wakeup/Interrupt, a safety procedure should also be followed to avoid inherited pseudo wakeup conditions. After the selected L port bits have been changed from output to input but before the associated WKEN bits are enabled, the associated edge select bits in WKEDG should be set or reset for the desired edge selects, followed by the associated WKPND bits being cleared.

This same procedure should be used following reset, since the L port inputs are left floating as a result of reset.

The occurrence of the selected trigger condition for Multi-Input Wakeup is latched into a pending register called WKPND. The respective bits of the WKPND register will be set on the occurrence of the selected trigger edge on the corresponding Port L pin. The user has the responsibility of clearing these pending flags. Since WKPND is a pending register for the occurrence of selected wakeup conditions, the device will not enter the HALT mode if any Wakeup bit is both enabled and pending. Consequently, the user has the responsibility of clearing the pending flags before attempting to enter the HALT mode.

WKEN, WKPND and WKEDG are all read/write registers, and are cleared at reset.

PORT L INTERRUPTS

Port L provides the user with an additional eight fully selectable, edge sensitive interrupts which are all vectored into the same service subroutine.

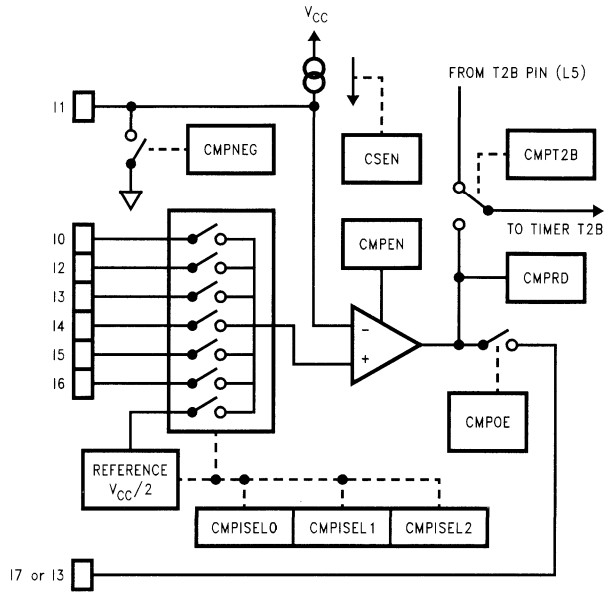
The interrupt from Port L shares logic with the wake up circuitry. The register WKEN allows interrupts from Port L to be individually enabled or disabled. The register WKEDG specifies the trigger condition to be either a positive or a negative edge. Finally, the register WKPND latches in the pending trigger conditions.

The GIE (Global Interrupt Enable) bit enables the interrupt function.

A control flag, LPEN, functions as a global interrupt enable for Port L interrupts. Setting the LPEN flag will enable interrupts and vice versa. A separate global pending flag is not needed since the register WKPND is adequate.

Since Port L is also used for waking the device out of the HALT or IDLE modes, the user can elect to exit the HALT or IDLE modes either with or without the interrupt enabled. If he elects to disable the interrupt, then the device will restart execution from the instruction immediately following the instruction that placed the microcontroller in the HALT or IDLE modes. In the other case, the device will first execute the interrupt service routine and then revert to normal operation. (See HALT MODE for clock option wakeup information.)

Analog Function Block



TL/DD/12094–14

FIGURE 12. COP888EK Analog Function Block

This device contains an analog function block with the intent to provide a function which allows for single slope, low cost, A/D conversion of up to 6 channels. See Application Note 983, Simple, Cost Effective A/D Conversion using COP888EK, for further information on this application.

CMPSL REGISTER (ADDRESS X'00B7)

The CMPSL register contains the following bits:

CMPNEG — Will drive I1 to a low level. This bit can be used to discharge an external capacitor. This bit is disabled if the comparator is not enabled (CMPEN = 0).

CMPEN — Enable the comparator ("1" = enable).

CSEN — Enables the internal constant current source. This current source provides a nominal 20 μA constant current at the I1 pin. This current can be used to ensure a linear charging rate on an external capacitor. This bit has no affect and the current source is disabled if the comparator is not enabled (CMPEN = 0).

CMPOE — Enables the comparator output to either pin I3 or pin I7 ("1" = enable) depending on the value of CMPISEL0/1/2.

CMPISEL0/1/2 — Will select one of seven possible sources (I0/I2/I3/I4/I5/I6/internal reference) as a positive input to the comparator (see Table I for more information.) Power savings can be realized by deselecting the internal reference when it is not in actual use.

CMPT2B — Selects the timer T2B input to be driven directly by the comparator output. If the comparator is disabled (CMPEN = 0), this function is disabled, i.e., the T2B input is connected to Port L5.

CMPT2B	CMPISEL2	CMPISEL1	CMPISEL0	CMPOE	CSEN	CMPEN	CMPNEG
Bit 7							Bit 0

The Comparator Select Register is cleared on RESET (the comparator is disabled). To save power the program should also disable the comparator before the μC enters the HALT/IDLE modes. Disabling the comparator will turn off the constant current source and the $V_{CC}/2$ reference, disconnect the comparator output from the T2B input and pin I3 or I7 and remove the low on I1 caused by CMPNEG.

It is often useful for the user's program to read the result of a comparator operation. Since I1 is always selected to be COMPIN− when the comparator is enabled (CMPEN = 1), the comparator output can be read internally by reading bit 1 (CMPRD) of register PORTI (RAM address 0 x D7).

The following table lists the comparator inputs and outputs vs. the value of the CMPISEL0/1/2 bits. The output will only be driven if the CMPOE bit is set to 1.

Analog Function Block (Continued)

TABLE I. Comparator Input Selection

Control Bit			Comparator Input Source		Comparator Output
CMPISEL2	CMPISEL1	CMPISEL0	Neg. Input	Pos. Input	
0	0	0	I1	I2	I3
0	0	1	I1	I2	I7
0	1	0	I1	I3	I7
0	1	1	I1	I0	I7
1	0	0	I1	I4	I7
1	0	1	I1	I5	I7
1	1	0	I1	I6	I7
1	1	1	I1	$V_{CC}/2$ Ref.	I7

Reset

The state of the Comparator Block immediately after RESET is as follows:

1. The CMPSL Register is set to all zeros
2. The Comparator is disabled
3. The Constant Current Source is disabled
4. CMPNEG is turned off
5. The Port I inputs are electrically isolated from the comparator
6. The T2B input is as normally selected by the T2CNTRL Register
7. CMPISEL0–CMPISEL2 are set to zero
8. All Port I inputs are selected to the default digital input mode

The comparator outputs have the same specification as Ports L and G except that the rise and fall times are symmetrical.

Interrupts

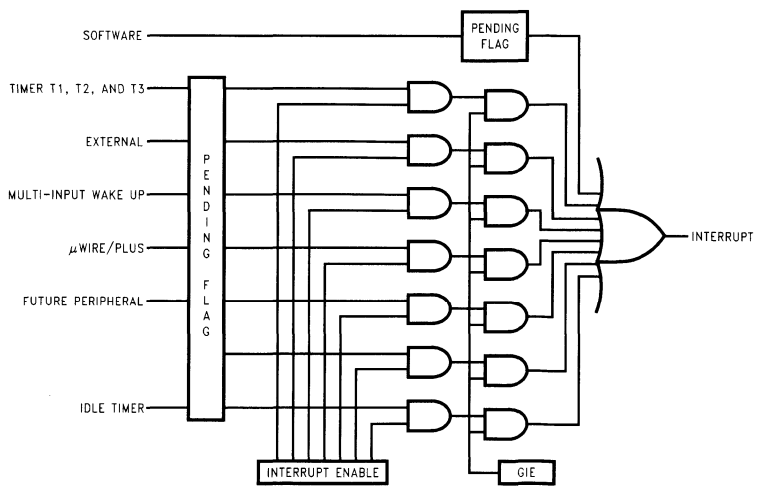
The device supports a vectored interrupt scheme. It supports a total of fourteen interrupt sources. The following table lists all the possible interrupt sources, their arbitration ranking and the memory locations reserved for the interrupt vector for each source.

Two bytes of program memory space are reserved for each interrupt source. All interrupt sources except the software interrupt are maskable. Each of the maskable interrupts have an Enable bit and a Pending bit. A maskable interrupt is active if its associated enable and pending bits are set. If GIE = 1 and an interrupt is active, then the processor will be interrupted as soon as it is ready to start executing an instruction except if the above conditions happen during the Software Trap service routine. This exception is described in the Software Trap sub-section.

The interruption process is accomplished with the INTR instruction (opcode 00), which is jammed inside the Instruction Register and replaces the opcode about to be executed. The following steps are performed for every interrupt:

1. The GIE (Global Interrupt Enable) bit is reset.
2. The address of the instruction about to be executed is pushed into the stack.
3. The PC (Program Counter) branches to address 00FF.

This procedure takes 7 t_C cycles to execute.



TL/DD/12094–15

FIGURE 13. Interrupt Block Diagram

Interrupts (Continued)

Arbitration Ranking	Source	Description	Vector* Address Hi-Low Byte
(1) Highest	Software	INTR Instruction	0yFE–0yFF
	Reserved		0yFC–0yFD
(2)	External	G0	0yFA–0yFB
(3)	Timer T0	Underflow	0yF8–0yF9
(4)	Timer T1	T1A/Underflow	0yF6–0yF7
(5)	Timer T1	T1B	0yF4–0yF5
(6)	MICROWIRE/PLUS	BUSY Low	0yF2–0yF3
	Reserved		0yF0–0yF1
(7)	Reserved		0yEE–0yEF
(8)	Reserved		0yEC–0yED
(9)	Timer T2	T2A/Underflow	0yEA–0yEB
(10)	Timer T2	T2B	0yE8–0yE9
(11)	Timer T3	T3A/Underflow	0yE6–0yE7
(12)	Timer T3	T3B	0yE4–0yE5
(13)	Port L/Wakeup	Port L Edge	0yE2–0yE3
(14) Lowest	Default	VIS Instr. Execution without Any Interrupts	0yE0–0yE1

*y is a variable which represents the VIS block. VIS and the vector table must be located in the same 256-byte block except if VIS is located at the last address of a block. In this case, the table must be in the next block.

At this time, since GIE = 0, other maskable interrupts are disabled. The user is now free to do whatever context switching is required by saving the context of the machine in the stack with PUSH instructions. The user would then program a VIS (Vector Interrupt Select) instruction in order to branch to the interrupt service routine of the highest priority interrupt enabled and pending at the time of the VIS. Note that this is not necessarily the interrupt that caused the branch to address location 00FF Hex prior to the context switching.

Thus, if an interrupt with a higher rank than the one which caused the interruption becomes active before the decision of which interrupt to service is made by the VIS, then the interrupt with the higher rank will override any lower ones and will be acknowledged. The lower priority interrupt(s) are still pending, however, and will cause another interrupt immediately following the completion of the interrupt service routine associated with the higher priority interrupt just serviced. This lower priority interrupt will occur immediately following the RETI (Return from Interrupt) instruction at the end of the interrupt service routine just completed.

Inside the interrupt service routine, the associated pending bit has to be cleared by software. The RETI (Return from Interrupt) instruction at the end of the interrupt service routine will set the GIE (Global Interrupt Enable) bit, allowing the processor to be interrupted again if another interrupt is active and pending.

The VIS instruction looks at all the active interrupts at the time it is executed and performs an indirect jump to the beginning of the service routine of the one with the highest rank.

The addresses of the different interrupt service routines, called vectors, are chosen by the user and stored in ROM in a table starting at 01E0 (assuming that VIS is located between 00FF and 01DF). The vectors are 15-bit wide and therefore occupy 2 ROM locations.

VIS and the vector table must be located in the same 256-byte block (0y00 to 0yFF) except if VIS is located at the last address of a block. In this case, the table must be in the next block. The vector table cannot be inserted in the first 256-byte block (y ≠ 0).

The vector of the maskable interrupt with the lowest rank is located at 0yE0 (Hi-Order byte) and 0yE1 (Lo-Order byte) and so forth in increasing rank number. The vector of the maskable interrupt with the highest rank is located at 0yFA (Hi-Order byte) and 0yFB (Lo-Order byte).

The Software Trap has the highest rank and its vector is located at 0yFE and 0yFF.

If, by accident, a VIS gets executed and no interrupt is active, then the PC (Program Counter) will branch to a vector located at 0yE0–0yE1.

WARNING

A Default VIS interrupt handler routine must be present. As a minimum, this handler should confirm that the GIE bit is cleared (this indicates that the interrupt sequence has been taken), take care of any required housekeeping, restore context and return. Some sort of Warm Restrart procedure should be implemented. These events can occur without any error on the part of the system designer or programmer.

Note: There is always the possibility of an interrupt occurring during an instruction which is attempting to reset the GIE bit or any other interrupt enable bit. If this occurs when a single cycle instruction is being used to reset the interrupt enable bit, the interrupt enable bit will be reset but an interrupt may still occur. This is because interrupt processing is started at the same time as the interrupt bit is being reset. To avoid this scenario, the user should always use a two, three, or four cycle instruction to reset interrupt enable bits.

Interrupts (Continued)

SOFTWARE TRAP

The Software Trap (ST) is a special kind of non-maskable interrupt which occurs when the INTR instruction (used to acknowledge interrupts) is fetched from ROM and placed inside the instruction register. This may happen when the PC is pointing beyond the available ROM address space or when the stack is over-popped.

When an ST occurs, the user can re-initialize the stack pointer and do a recovery procedure (similar to reset, but not necessarily containing all of the same initialization procedures) before restarting.

The occurrence of an ST is latched into the ST pending bit. The GIE bit is not affected and the ST pending bit **(not accessible by the user)** is used to inhibit other interrupts and to direct the program to the ST service routine with the VIS instruction. The RPND instruction is used to clear the software interrupt pending bit. This pending bit is also cleared on reset.

The ST has the highest rank among all interrupts.

Nothing (except another ST) can interrupt an ST being serviced.

WATCHDOG

The device contains a WATCHDOG and clock monitor. The WATCHDOG is designed to detect the user program getting stuck in infinite loops resulting in loss of program control or "runaway" programs. The Clock Monitor is used to detect the absence of a clock or a very slow clock below a specified rate on the CKI pin.

The WATCHDOG consists of two independent logic blocks: WD UPPER and WD LOWER. WD UPPER establishes the upper limit on the service window and WD LOWER defines the lower limit of the service window.

Servicing the WATCHDOG consists of writing a specific value to a WATCHDOG Service Register named WDSVR which is memory mapped in the RAM. This value is composed of three fields, consisting of a 2-bit Window Select, a 5-bit Key Data field, and the 1-bit Clock Monitor Select field. Table II shows the WDSVR register.

TABLE II. WATCHDOG Service Register (WDSVR)

Window Select		Key Data					Clock Monitor
X	X	0	1	1	0	0	Y
7	6	5	4	3	2	1	0

The lower limit of the service window is fixed at 2048 instruction cycles. Bits 7 and 6 of the WDSVR register allow the user to pick an upper limit of the service window.

Table III shows the four possible combinations of lower and upper limits for the WATCHDOG service window. This flexibility in choosing the WATCHDOG service window prevents any undue burden on the user software.

Bits 5, 4, 3, 2 and 1 of the WDSVR register represent the 5-bit Key Data field. The key data is fixed at 01100. Bit 0 of the WDSVR Register is the Clock Monitor Select bit.

TABLE III. WATCHDOG Service Window Select

WDSVR Bit 7	WDSVR Bit 6	Service Window (Lower-Upper Limits)
0	0	2k–8k t_C Cycles
0	1	2k–16k t_C Cycles
1	0	2k–32k t_C Cycles
1	1	2k–64k t_C Cycles

Clock Monitor

The Clock Monitor aboard the device can be selected or deselected under program control. The Clock Monitor is guaranteed not to reject the clock if the instruction cycle clock ($1/t_C$) is greater or equal to 10 kHz. This equates to a clock input rate on CKI of greater or equal to 100 kHz.

WATCHDOG Operation

The WATCHDOG and Clock Monitor are disabled during reset. The device comes out of reset with the WATCHDOG armed, the WATCHDOG Window Select bits (bits 6, 7 of the WDSVR Register) set, and the Clock Monitor bit (bit 0 of the WDSVR Register) enabled. Thus, a Clock Monitor error will occur after coming out of reset, if the instruction cycle clock frequency has not reached a minimum specified value, including the case where the oscillator fails to start.

The WDSVR register can be written to only once after reset and the key data (bits 5 through 1 of the WDSVR Register) must match to be a valid write. This write to the WDSVR register involves two irrevocable choices: (i) the selection of the WATCHDOG service window (ii) enabling or disabling of the Clock Monitor. Hence, the first write to WDSVR Register involves selecting or deselecting the Clock Monitor, select the WATCHDOG service window and match the WATCHDOG key data. Subsequent writes to the WDSVR register will compare the value being written by the user to the WATCHDOG service window value and the key data (bits 7 through 1) in the WDSVR Register. Table IV shows the sequence of events that can occur.

The user must service the WATCHDOG at least once before the upper limit of the service window expires. The WATCHDOG may not be serviced more than once in every lower limit of the service window. The user may service the WATCHDOG as many times as wished in the time period between the lower and upper limits of the service window. The first write to the WDSVR Register is also counted as a WATCHDOG service.

The WATCHDOG has an output pin associated with it. This is the WDOUT pin, on pin 1 of the port G. WDOUT is active low. The WDOUT pin is in the high impedance state in the inactive state. Upon triggering the WATCHDOG, the logic will pull the WDOUT (G1) pin low for an additional 16 t_C–32 t_C cycles after the signal level on WDOUT pin goes below the lower Schmitt trigger threshold. After this delay, the device will stop forcing the WDOUT output low.

The WATCHDOG service window will restart when the WDOUT pin goes high. It is recommended that the user tie the WDOUT pin back to V_{CC} through a resistor in order to pull WDOUT high.

WATCHDOG Operation (Continued)

A WATCHDOG service while the WDOUT signal is active will be ignored. The state of the WDOUT pin is not guaranteed on reset, but if it powers up low then the WATCHDOG will time out and WDOUT will enter high impedance state.

The Clock Monitor forces the G1 pin low upon detecting a clock frequency error. The Clock Monitor error will continue until the clock frequency has reached the minimum specified value, after which the G1 output will enter the high impedance TRI-STATE mode following 16 t_C–32 t_C clock cycles. The Clock Monitor generates a continual Clock Monitor error if the oscillator fails to start, or fails to reach the minimum specified frequency. The specification for the Clock Monitor is as follows:

$1/t_C > 10$ kHz—No clock rejection.

$1/t_C < 10$ Hz—Guaranteed clock rejection.

WATCHDOG AND CLOCK MONITOR SUMMARY

The following salient points regarding the WATCHDOG and CLOCK MONITOR should be noted:

- Both the WATCHDOG and CLOCK MONITOR detector circuits are inhibited during RESET.
- Following RESET, the WATCHDOG and CLOCK MONITOR are both enabled, with the WATCHDOG having he maximum service window selected.
- The WATCHDOG service window and CLOCK MONITOR enable/disable option can only be changed once, during the initial WATCHDOG service following RESET.
- The initial WATCHDOG service must match the key data value in the WATCHDOG Service register WDSVR in order to avoid a WATCHDOG error.
- Subsequent WATCHDOG services must match all three data fields in WDSVR in order to avoid WATCHDOG errors.
- The correct key data value cannot be read from the WATCHDOG Service register WDSVR. Any attempt to read this key data value of 01100 from WDSVR will read as key data value of all 0's.
- The WATCHDOG detector circuit is inhibited during both the HALT and IDLE modes.
- The CLOCK MONITOR detector circuit is active during both the HALT and IDLE modes. Consequently, the device inadvertently entering the HALT mode will be detected as a CLOCK MONITOR error (provided that the CLOCK MONITOR enable option has been selected by the program).
- With the single-pin R/C oscillator mask option selected and the CLKDLY bit reset, the WATCHDOG service window will resume following HALT mode from where it left off before entering the HALT mode.
- With the crystal oscillator mask option selected, or with the single-pin R/C oscillator mask option selected and the CLKDLY bit set, the WATCHDOG service window will be set to its selected value from WDSVR following HALT. Consequently, the WATCHDOG should not be serviced for at least 2048 instruction cycles following HALT, but must be serviced within the selected window to avoid a WATCHDOG error.
- The IDLE timer T0 is not initialized with RESET.
- The user can sync in to the IDLE counter cycle with an IDLE counter (T0) interrupt or by monitoring the T0PND flag. The T0PND flag is set whenever the thirteenth bit of the IDLE counter toggles (every 4096 instruction cycles). The user is responsible for resetting the T0PND flag.
- A hardware WATCHDOG service occurs just as the device exits the IDLE mode. Consequently, the WATCHDOG should not be serviced for at least 2048 instruction cycles following IDLE, but must be serviced within the selected window to avoid a WATCHDOG error.
- Following RESET, the initial WATCHDOG service (where the service window and the CLOCK MONITOR enable/disable must be selected) may be programmed anywhere within the maximum service window (65,536 instruction cycles) initialized by RESET. Note that this initial WATCHDOG service may be programmed within the initial 2048 instruction cycles without causing a WATCHDOG error.

Detection of Illegal Conditions

The device can detect various illegal conditions resulting from coding errors, transient noise, power supply voltage drops, runaway programs, etc.

Reading of undefined ROM gets zeros. The opcode for software interrupt is zero. If the program fetches instructions from undefined ROM, this will force a software interrupt, thus signaling that an illegal condition has occurred.

The subroutine stack grows down for each call (jump to subroutine), interrupt, or PUSH, and grows up for each return or POP. The stack pointer is initialized to RAM location 06F Hex during reset. Consequently, if there are more returns than calls, the stack pointer will point to addresses 070 and 071 Hex (which are undefined RAM). Undefined RAM from addresses 070 to 07F (Segment 0), 140 to 17F (Segment 1), and all other segments (i.e., Segments 2 . . . etc.) is read as all 1's, which in turn will cause the program to return to address 7FFF Hex. This is an undefined ROM location and the instruction fetched (all 0's) from this location will generate a software interrupt signaling an illegal condition.

Thus, the chip can detect the following illegal conditions:

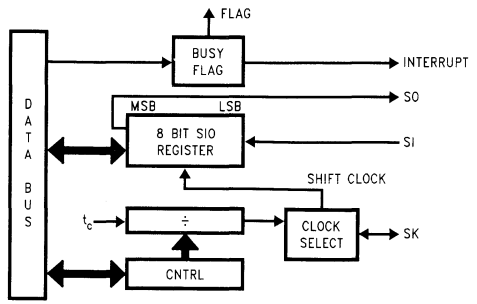
a. Executing from undefined ROM

b. Over "POP"ing the stack by having more returns than calls.

When the software interrupt occurs, the user can re-initialize the stack pointer and do a recovery procedure before restarting (this recovery program is probably similar to that following reset, but might not contain the same program initialization procedures). The recovery program should reset the software interrupt pending bit using the RPND instruction.

MICROWIRE/PLUS

MICROWIRE/PLUS is a serial synchronous communications interface. The MICROWIRE/PLUS capability enables the device to interface with any of National Semiconductor's MICROWIRE peripherals (i.e. A/D converters, display drivers, E^2PROMs etc.) and with other microcontrollers which support the MICROWIRE interface. It consists of an 8-bit serial shift register (SIO) with serial data input (SI), serial data output (SO) and serial shift clock (SK). *Figure 14* shows a block diagram of the MICROWIRE/PLUS logic.



TL/DD/12094–16

FIGURE 14. MICROWIRE/PLUS Block Diagram

The shift clock can be selected from either an internal source or an external source. Operating the MICROWIRE/PLUS arrangement with the internal clock source is called the Master mode of operation. Similarly, operating the MICROWIRE/PLUS arrangement with an external shift clock is called the Slave mode of operation.

The CNTRL register is used to configure and control the MICROWIRE/PLUS mode. To use the MICROWIRE/PLUS, the MSEL bit in the CNTRL register is set to one. In the master mode, the SK clock rate is selected by the two bits, SL0 and SL1, in the CNTRL register. Table V details the different clock rates that may be selected.

TABLE IV. WATCHDOG Service Actions

Key Data	Window Data	Clock Monitor	Action
Match	Match	Match	Valid Service: Restart Service Window
Don't Care	Mismatch	Don't Care	Error: Generate WATCHDOG Output
Mismatch	Don't Care	Don't Care	Error: Generate WATCHDOG Output
Don't Care	Don't Care	Mismatch	Error: Generate WATCHDOG Output

TABLE V. MICROWIRE/PLUS Master Mode Clock Select

SL1	SL0	SK
0	0	$2 \times t_c$
0	1	$4 \times t_c$
1	x	$8 \times t_c$

Where t_c is the instruction cycle clock

5

MICROWIRE/PLUS (Continued)

MICROWIRE/PLUS OPERATION

Setting the BUSY bit in the PSW register causes the MICROWIRE/PLUS to start shifting the data. It gets reset when eight data bits have been shifted. The user may reset the BUSY bit by software to allow less than 8 bits to shift. If enabled, an interrupt is generated when eight data bits have been shifted. The device may enter the MICROWIRE/PLUS mode either as a Master or as a Slave. *Figure 15* shows how two microcontroller devices and several peripherals may be interconnected using the MICROWIRE/PLUS arrangements.

Warning:

The SIO register should only be loaded when the SK clock is low. Loading the SIO register while the SK clock is high will result in undefined data in the SIO register. SK clock is normally low when not shifting.

Setting the BUSY flag when the input SK clock is high in the MICROWIRE/PLUS slave mode may cause the current SK clock for the SIO shift register to be narrow. For safety, the BUSY flag should only be set when the input SK clock is low.

MICROWIRE/PLUS Master Mode Operation

In the MICROWIRE/PLUS Master mode of operation the shift clock (SK) is generated internally by the device. The MICROWIRE Master always initiates all data exchanges. The MSEL bit in the CNTRL register must be set to enable the SO and SK functions onto the G Port. The SO and SK pins must also be selected as outputs by setting appropriate bits in the Port G configuration register. Table VI summarizes the bit settings required for Master mode of operation.

MICROWIRE/PLUS Slave Mode Operation

In the MICROWIRE/PLUS Slave mode of operation the SK clock is generated by an external source. Setting the MSEL bit in the CNTRL register enables the SO and SK functions onto the G Port. The SK pin must be selected as an input and the SO pin is selected as an output pin by setting and resetting the appropriate bits in the Port G configuration register. Table VI summarizes the settings required to enter the Slave mode of operation.

The user must set the BUSY flag immediately upon entering the Slave mode. This will ensure that all data bits sent by the Master will be shifted properly. After eight clock pulses the BUSY flag will be cleared and the sequence may be repeated.

Alternate SK Phase Operation

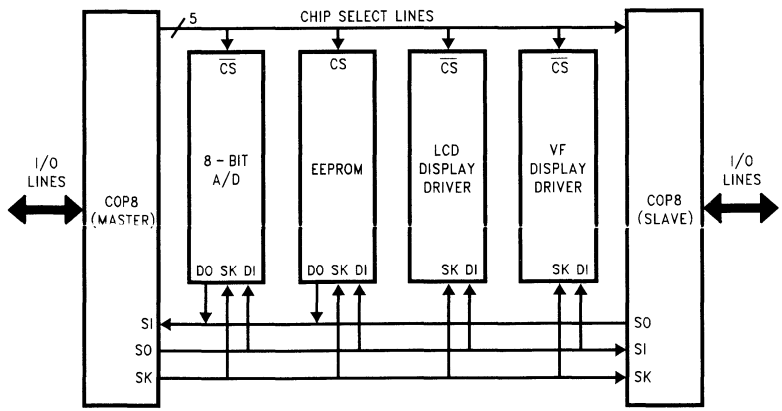
The device allows either the normal SK clock or an alternate phase SK clock to shift data in and out of the SIO register. In both the modes the SK is normally low. In the normal mode data is shifted in on the rising edge of the SK clock and the data is shifted out on the falling edge of the SK clock. The SIO register is shifted on each falling edge of the SK clock. In the alternate SK phase operation, data is shifted in on the falling edge of the SK clock and shifted out on the rising edge of the SK clock.

A control flag, SKSEL, allows either the normal SK clock or the alternate SK clock to be selected. Resetting SKSEL causes the MICROWIRE/PLUS logic to be clocked from the normal SK signal. Setting the SKSEL flag selects the alternate SK clock. The SKSEL is mapped into the G6 configuration bit. The SKSEL flag will power up in the reset condition, selecting the normal SK signal.

TABLE VI

This table assumes that the control flag MSEL is set.

G4 (SO) Config. Bit	G5 (SK) Config. Bit	G4 Fun.	G5 Fun.	Operation
1	1	SO	Int. SK	MICROWIRE/PLUS Master
0	1	TRI-STATE	Int. SK	MICROWIRE/PLUS Master
1	0	SO	Ext. SK	MICROWIRE/PLUS Slave
0	0	TRI-STATE	Ext. SK	MICROWIRE/PLUS Slave



TL/DD/12094–17

FIGURE 15. MICROWIRE/PLUS Application

Memory Map

All RAM, ports and registers (except A and PC) are mapped into data memory address space.

Address S/ADD REG	Contents
0000 to 006F	On-Chip RAM bytes (112 bytes)
0070 to 007F	Unused RAM Address Space (Reads As All Ones)
xx80 to xxAF	Unused RAM Address Space (Reads Undefined Data)
xxB0	Timer T3 Lower Byte
XXB1	Timer T3 Upper Byte
xxB2	Timer T3 Autoload Register T3RA Lower Byte
xxB3	Timer T3 Autoload Register T3RA Upper Byte
xxB4	Timer T3 Autoload Register T3RB Lower Byte
xxB5	Timer T3 Autoload Register T3RB Upper Byte
xxB6	Timer T3 Control Register
xxB7	Comparator Select Register (CMPSL)
xxB8–xxBF	Reserved
xxC0	Timer T2 Lower Byte
xxC1	Timer T2 Upper Byte
xxC2	Timer T2 Autoload Register T2RA Lower Byte
xxC3	Timer T2 Autoload Register T2RA Upper Byte
xxC4	Timer T2 Autoload Register T2RB Lower Byte
xxC5	Timer T2 Autoload Register T2RB Upper Byte
xxC6	Timer T2 Control Register
xxC7	WATCHDOG Service Register (Reg:WDSVR)
xxC8	MIWU Edge Select Register (Reg:WKEDG)
xxC9	MIWU Enable Register (Reg:WKEN)
xxCA	MIWU Pending Register (Reg:WKPND)
xxCB	Reserved
xxCC	Reserved
xxCD to xxCF	Reserved

Address S/ADD REG	Contents
xxD0	Port L Data Register
xxD1	Port L Configuration Register
xxD2	Port L Input Pins (Read Only)
xxD3	Reserved for Port L
xxD4	Port G Data Register
xxD5	Port G Configuration Register
xxD6	Port G Input Pins (Read Only)
xxD7	Port I Input Pins (Read Only)
xxD8	Port C Data Register
xxD9	Port C Configuration Register
xxDA	Port C Input Pins (Read Only)
xxDB	Reserved for Port C
xxDC	Port D
xxDD to DF	Reserved
xxE0 to xxE5	Reserved
xxE6	Timer T1 Autoload Register T1RB Lower Byte
xxE7	Timer T1 Autoload Register T1RB Upper Byte
xxE8	ICNTRL Register
xxE9	MICROWIRE/PLUS Shift Register
xxEA	Timer T1 Lower Byte
xxEB	Timer T1 Upper Byte
xxEC	Timer T1 Autoload Register T1RA Lower Byte
xxED	Timer T1 Autoload Register T1RA Upper Byte
xxEE	CNTRL Control Register
xxEF	PSW Register
xxF0 to FB	On-Chip RAM Mapped as Registers
xxFC	X Register
xxFD	SP Register
xxFE	B Register
xxFF	S Register
0100–017F	On-Chip 128 RAM Bytes

Reading memory locations 0070H–007FH (Segment 0) will return all ones. Reading unused memory locations 0080H–00AFH (Segment 0) will return undefined data. Reading memory locations from other unused Segments (i.e., Segment 2, Segment 3, ... etc.) will return all ones.

Addressing Modes

There are ten addressing modes, six for operand addressing and four for transfer of control.

OPERAND ADDRESSING MODES

Register Indirect

This is the "normal" addressing mode. The operand is the data memory addressed by the B pointer or X pointer.

Register Indirect (with auto post increment or decrement of pointer)

This addressing mode is used with the LD and X instructions. The operand is the data memory addressed by the B pointer or X pointer. This is a register indirect mode that automatically post increments or decrements the B or X register after executing the instruction.

Direct

The instruction contains an 8-bit address field that directly points to the data memory for the operand.

Immediate

The instruction contains an 8-bit immediate field as the operand.

Short Immediate

This addressing mode is used with the Load B Immediate instruction. The instruction contains a 4-bit immediate field as the operand.

Indirect

This addressing mode is used with the LAID instruction. The contents of the accumulator are used as a partial address (lower 8 bits of PC) for accessing a data operand from the program memory.

TRANSFER OF CONTROL ADDRESSING MODES

Relative

This mode is used for the JP instruction, with the instruction field being added to the program counter to get the new program location. JP has a range from −31 to +32 to allow a 1-byte relative jump (JP + 1 is implemented by a NOP instruction). There are no "pages" when using JP, since all 15 bits of PC are used.

Absolute

This mode is used with the JMP and JSR instructions, with the instruction field of 12 bits replacing the lower 12 bits of the program counter (PC). This allows jumping to any location in the current 4k program memory segment.

Absolute Long

This mode is used with the JMPL and JSRL instructions, with the instruction field of 15 bits replacing the entire 15 bits of the program counter (PC). This allows jumping to any location in the current 4k program memory space.

Indirect

This mode is used with the JID instruction. The contents of the accumulator are used as a partial address (lower 8 bits of PC) for accessing a location in the program memory. The contents of this program memory location serve as a partial address (lower 8 bits of PC) for the jump to the next instruction.

Note: The VIS is a special case of the Indirect Transfer of Control addressing mode, where the double byte vector associated with the interrupt is transferred from adjacent addresses in the program memory into the program counter (PC) in order to jump to the associated interrupt service routine.

Instruction Set

Register and Symbol Definition

Registers	
A	8-Bit Accumulator Register
B	8-Bit Address Register
X	8-Bit Address Register
S	8-Bit Segment Register
SP	8-Bit Stack Pointer Register
PC	15-Bit Program Counter Register
PU	Upper 7 Bits of PC
PL	Lower 8 Bits of PC
C	1 Bit of PSW Register for Carry
HC	1 Bit of PSW Register for Half Carry
GIE	1 Bit of PSW Register for Global Interrupt Enable
VU	Interrupt Vector Upper Byte
VL	Interrupt Vector Lower Byte

Symbols	
[B]	Memory Indirectly Addressed by B Register
[X]	Memory Indirectly Addressed by X Register
MD	Direct Addressed Memory
Mem	Direct Addressed Memory or [B]
Meml	Direct Addressed Memory or [B] or Immediate Data
Imm	8-Bit Immediate Data
Reg	Register Memory: Addresses F0 to FF (Includes B, X and SP)
Bit	Bit Number (0 to 7)
←	Loaded with
↔	Exchanged with

Instruction Set (Continued)

INSTRUCTION SET

ADD	A,Meml	ADD	A ← A + Meml
ADC	A,Meml	ADD with Carry	A ← A + Meml + C, C ← Carry HC ← Half Carry
SUBC	A,Meml	Subtract with Carry	A ← A − $\overline{\text{Meml}}$ + C, C ← Carry HC ← Half Carry
AND	A,Meml	Logical AND	A ← A and Meml
ANDSZ	A,Imm	Logical AND Immed., Skip if Zero	Skip next if (A and Imm) = 0
OR	A,Meml	Logical OR	A ← A or Meml
XOR	A,Meml	Logical EXclusive OR	A ← A xor Meml
IFEQ	MD,Imm	IF EQual	Compare MD and Imm, Do next if MD = Imm
IFEQ	A,Meml	IF EQual	Compare A and Meml, Do next if A = Meml
IFNE	A,Meml	IF Not Equal	Compare A and Meml, Do next if A ≠ Meml
IFGT	A,Meml	IF Greater Than	Compare A and Meml, Do next if A > Meml
IFBNE	#	If B Not Equal	Do next if lower 4 bits of B ≠ Imm
DRSZ	Reg	Decrement Reg., Skip if Zero	Reg ← Reg − 1, Skip if Reg = 0
SBIT	#,Mem	Set BIT	1 to bit, Mem (bit = 0 to 7 immediate)
RBIT	#,Mem	Reset BIT	0 to bit, Mem
IFBIT	#,Mem	IF BIT	If bit in A or Mem is true do next instruction
RPND		Reset PeNDing Flag	Reset Software Interrupt Pending Flag
X	A,Mem	EXchange A with Memory	A ⟷ Mem
X	A,[X]	EXchange A with Memory [X]	A ⟷ [X]
LD	A,Meml	LoaD A with Memory	A ← Meml
LD	A,[X]	LoaD A with Memory [X]	A ← [X]
LD	B,Imm	LoaD B with Immed.	B ← Imm
LD	Mem,Imm	LoaD Memory Immed	Mem ← Imm
LD	Reg,Imm	LoaD Register Memory Immed.	Reg ← Imm
X	A, [B ±]	EXchange A with Memory [B]	A ⟷ [B], (B ← B ±1)
X	A, [X ±]	EXchange A with Memory [X]	A ⟷ [X], (X ← ±1)
LD	A, [B±]	LoaD A with Memory [B]	A ← [B], (B ← B ±1)
LD	A, [X±]	LoaD A with Memory [X]	A ← [X], (X ← X±1)
LD	[B±],Imm	LoaD Memory [B] Immed.	[B] ← Imm, (B ← B±1)
CLR	A	CLeaR A	A ← 0
INC	A	INCrement A	A ← A + 1
DEC	A	DECrement A	A ← A − 1
LAID		Load A InDirect from ROM	A ← ROM (PU,A)
DCOR	A	Decimal CORrect A	A ← BCD correction of A (follows ADC, SUBC)
RRC	A	Rotate A Right thru C	C → A7 → . . . → A0 → C
RLC	A	Rotate A Left thru C	C ← A7 ← . . . ← A0 ← C
SWAP	A	SWAP nibbles of A	A7 . . . A4 ⟷ A3 . . . A0
SC		Set C	C ← 1, HC ← 1
RC		Reset C	C ← 0, HC ← 0
IFC		IF C	IF C is true, do next instruction
IFNC		IF Not C	If C is not true, do next instruction
POP	A	POP the stack into A	SP ← SP + 1, A ← [SP]
PUSH	A	PUSH A onto the stack	[SP] ← A, SP ← SP − 1
VIS		Vector to Interrupt Service Routine	PU ← [VU], PL ← [VL]
JMPL	Addr.	Jump absolute Long	PC ← ii (ii = 15 bits, 0 to 32k)
JMP	Addr.	Jump absolute	PC9 . . . 0 ← i (i = 12 bits)
JP	Disp.	Jump relative short	PC ← PC + r (r is −31 to +32, except 1)
JSRL	Addr.	Jump SubRoutine Long	[SP] ← PL, [SP−1] ← PU,SP−2, PC ← ii
JSR	Addr	Jump SubRoutine	[SP] ← PL, [SP−1] ← PU,SP−2, PC9 . . . 0 ← i
JID		Jump InDirect	PL ← ROM (PU,A)
RET		RETurn from subroutine	SP + 2, PL ← [SP], PU ← [SP−1]
RETSK		RETurn and SKip	SP + 2, PL ← [SP],PU ← [SP−1]
RETI		RETurn from Interrupt	SP + 2, PL ← [SP],PU ← [SP−1],GIE ← 1
INTR		Generate an Interrupt	[SP] ← PL, [SP−1] ← PU, SP−2, PC ← 0FF
NOP		No OPeration	PC ← PC + 1

5

Instruction Execution Time

Most instructions are single byte (with immediate addressing mode instructions taking two bytes).

Most single byte instructions take one cycle time to execute.

- Skipped instructions require x number of cycles to be skipped, where x equals the number of bytes in the skipped instruction opcode.

See the BYTES and CYCLES per INSTRUCTION table for details.

Bytes and Cycles per Instruction

The following table shows the number of bytes and cycles for each instruction in the format of byte/cycle.

Arithmetic and Logic Instructions

	[B]	Direct	Immed.
ADD	1/1	3/4	2/2
ADC	1/1	3/4	2/2
SUBC	1/1	3/4	2/2
AND	1/1	3/4	2/2
OR	1/1	3/4	2/2
XOR	1/1	3/4	2/2
IFEQ	1/1	3/4	2/2
IFNE	1/1	3/4	2/2
IFGT	1/1	3/4	2/2
IFBNE	1/1		
DRSZ		1/3	
SBIT	1/1	3/4	
RBIT	1/1	3/4	
IFBIT	1/1	3/4	

RPND	1/1

Instructions Using A & C

CLRA	1/1
INCA	1/1
DECA	1/1
LAID	1/3
DCOR	1/1
RRCA	1/1
RLCA	1/1
SWAPA	1/1
SC	1/1
RC	1/1
IFC	1/1
IFNC	1/1
PUSHA	1/3
POPA	1/3
ANDSZ	2/2

Transfer of Control Instructions

JMPL	3/4
JMP	2/3
JP	1/3
JSRL	3/5
JSR	2/5
JID	1/3
VIS	1/5
RET	1/5
RETSK	1/5
RETI	1/5
INTR	1/7
NOP	1/1

Memory Transfer Instructions

	Register Indirect		Direct	Immed.	Register Indirect Auto Incr. & Decr.		
	[B]	[X]			[B+, B−]	[X+, X−]	
X A,*	1/1	1/3	2/3		1/2	1/3	
LD A,*	1/1	1/3	2/3	2/2	1/2	1/3	
LD B, Imm				1/1			(IF B < 16)
LD B, Imm				2/2			(IF B > 15)
LD Mem, Imm	2/2		3/3		2/2		
LD Reg, Imm			2/3				
IFEQ MD, Imm			3/3				

* = > Memory location addressed by B or X or directly.

Opcode Table

Upper Nibble																Lower Nibble
F	E	D	C	B	A	9	8	7	6	5	4	3	2	1	0	
JP−15	JP−31	LD 0F0, # i	DRSZ 0F0	RRCA	RC	ADC A,#i	ADC A,[B]	IFBIT 0,[B]	ANDSZ A, #i	LD B,#0F	IFBNE 0	JSR x000−x0FF	JMP x000−x0FF	JP+17	INTR	0
JP−14	JP−30	LD 0F1, # i	DRSZ 0F1	*	SC	SUBC A, #i	SUB A,[B]	IFBIT 1,[B]	*	LD B,#0E	IFBNE 1	JSR x100−x1FF	JMP x100−x1FF	JP+18	JP+2	1
JP−13	JP−29	LD 0F2, # i	DRSZ 0F2	X A,[X+]	X A,[B+]	IFEQ A,#i	IFEQ A,[B]	IFBIT 2,[B]	*	LD B,#0D	IFBNE 2	JSR x200−x2FF	JMP x200−x2FF	JP+19	JP+3	2
JP−12	JP−28	LD 0F3, # i	DRSZ 0F3	X A,[X−]	X A,[B−]	IFGT A,#i	IFGT A,[B]	IFBIT 3,[B]	*	LD B,#0C	IFBNE 3	JSR x300−x3FF	JMP x300−x3FF	JP+20	JP+4	3
JP−11	JP−27	LD 0F4, # i	DRSZ 0F4	VIS	LAID	ADD A,#i	ADD A,[B]	IFBIT 4,[B]	CLRA	LD B,#0B	IFBNE 4	JSR x400−x4FF	JMP x400−x4FF	JP+21	JP+5	4
JP−10	JP−26	LD 0F5, # i	DRSZ 0F5	RPND	JID	AND A,#i	AND A,[B]	IFBIT 5,[B]	SWAPA	LD B,#0A	IFBNE 5	JSR x500−x5FF	JMP x500−x5FF	JP+22	JP+6	5
JP−9	JP−25	LD 0F6, # i	DRSZ 0F6	X A,[X]	X A,[B]	XOR A,#i	XOR A,[B]	IFBIT 6,[B]	DCORA	LD B,#09	IFBNE 6	JSR x600−x6FF	JMP x600−x6FF	JP+23	JP+7	6
JP−8	JP−24	LD 0F7, # i	DRSZ 0F7	*	*	OR A,#i	OR A,[B]	IFBIT 7,[B]	PUSHA	LD B,#08	IFBNE 7	JSR x700−x7FF	JMP x700−x7FF	JP+24	JP+8	7
JP−7	JP−23	LD 0F8, # i	DRSZ 0F8	NOP	RLCA	LD A,#i	IFC	SBIT 0,[B]	RBIT 0,[B]	LD B,#07	IFBNE 8	JSR x800−x8FF	JMP x800−x8FF	JP+25	JP+9	8
JP−6	JP−22	LD 0F9, # i	DRSZ 0F9	IFNE A,[B]	IFEQ Md,#i	IFNE A,#i	IFNC	SBIT 1,[B]	RBIT 1,[B]	LD B,#06	IFBNE 9	JSR x900−x9FF	JMP x900−x9FF	JP+26	JP+10	9
JP−5	JP−21	LD 0FA, # i	DRSZ 0FA	LD A,[X+]	LD A,[B+]	LD [B+],#i	INCA	SBIT 2,[B]	RBIT 2,[B]	LD B,#05	IFBNE 0A	JSR xA00−xAFF	JMP xA00−xAFF	JP+27	JP+11	A
JP−4	JP−20	LD 0FB, # i	DRSZ 0FB	LD A,[X−]	LD A,[B−]	LD [B−],#i	DECA	SBIT 3,[B]	RBIT 3,[B]	LD B,#04	IFBNE 0B	JSR xB00−xBFF	JMP xB00−xBFF	JP+28	JP+12	B
JP−3	JP−19	LD 0FC, # i	DRSZ 0FC	LD Md,#i	JMPL	X A,Md	POPA	SBIT 4,[B]	RBIT 4,[B]	LD B,#03	IFBNE 0C	JSR xC00−xCFF	JMP xC00−xCFF	JP+29	JP+13	C
JP−2	JP−18	LD 0FD, # i	DRSZ 0FD	DIR	JSRL	LD A,Md	RETSK	SBIT 5,[B]	RBIT 5,[B]	LD B,#02	IFBNE 0D	JSR xD00−xDFF	JMP xD00−xDFF	JP+30	JP+14	D
JP−1	JP−17	LD 0FE, # i	DRSZ 0FE	LD A,[X]	LD A,[B]	LD [B],#i	RET	SBIT 6,[B]	RBIT 6,[B]	LD B,#01	IFBNE 0E	JSR xE00−xEFF	JMP xE00−xEFF	JP+31	JP+15	E
JP−0	JP−16	LD 0FF, # i	DRSZ 0FF	*	*	LD B,#i	RETI	SBIT 7,[B]	RBIT 7,[B]	LD B,#00	IFBNE 0F	JSR xF00−xFFF	JMP xF00−xFFF	JP+32	JP+16	F

Where,

i is the immediate data

Md is a directly addressed memory location

* is an unused opcode

Note: The opcode 60 Hex is also the opcode for IFBIT #i,A

Mask Options

The mask programmable options are shown below. The options are programmed at the same time as the ROM pattern submission.

```
OPTION 1: CLOCK CONFIGURATION
  = 1     Crystal Oscillator (CKI/10)
              G7 (CKO) is clock generator
              output to crystal/resonator
              CKI is the clock input
  = 2     Single-pin RC controlled
              oscillator (CKI/10)
              G7 is available as a HALT
              restart and/or general purpose
              input
OPTION 2: HALT
  = 1     Enable HALT mode
  = 2     Disable HALT mode
OPTION 3: BONDING OPTIONS
  = 1     44-Pin PLCC
  = 2     40-Pin DIP
  = 3     N/A
  = 4     28-Pin DIP
  = 5     28-Pin SO
```

Development Support

SUMMARY

- iceMASTER™: IM-COP8/400—Full feature in-circuit emulation for all COP8 products. A full set of COP8 Basic and Feature Family device and package specific probes are available.
- COP8 Debug Module: Moderate cost in-circuit emulation and development programming unit.
- COP8 Evaluation and Programming Unit: EPU-COP888GG—low cost in-circuit simulation and development programming unit.
- Assembler: COP8-DEV-IBMA. A DOS installable cross development Assembler, Linker, Librarian and Utility Software Development Tool Kit.
- C Compiler: COP8C. A DOS installable cross development Software Tool Kit.
- OTP/EPROM Programmer Support: Covering needs from engineering prototype, pilot production to full production environments.

Development Support (Continued)

IceMASTER (IM) IN-CIRCUIT EMULATION

The iceMASTER IM-COP8/400 is a full feature, PC based, in-circuit emulation tool developed and marketed by MetaLink Corporation to support the whole COP8 family of products. National is a resale vendor for these products.

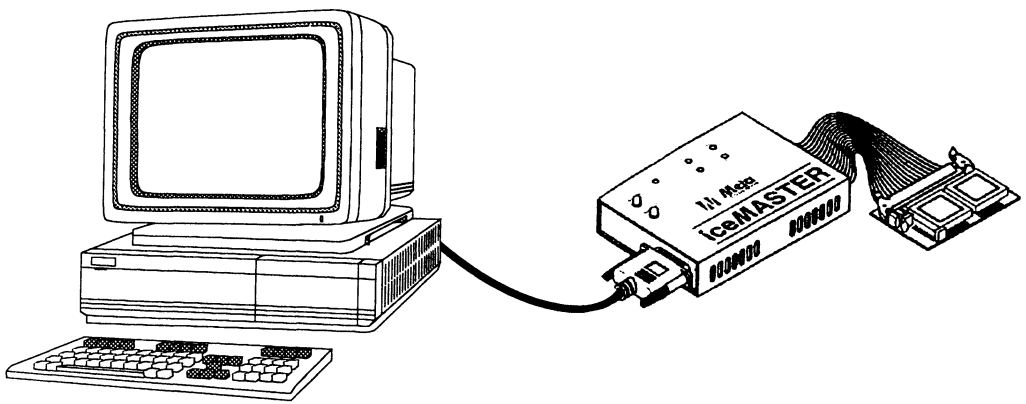
See *Figure 19* for configuration.

The iceMASTER IM-COP8/400 with its device specific COP8 Probe provides a rich feature set for developing, testing and maintaining product:

- Real-time in-circuit emulation; full 2.4V–5.5V operation range, full DC-10 MHz clock. Chip options are programmable or jumper selectable.
- Direct connection to application board by package compatible socket or surface mount assembly.
- Full 32 kbyte of loadable programming space that overlays (replaces) the on-chip ROM or EPROM. On-chip RAM and I/O blocks are used directly or recreated on the probe as necessary.
- Full 4k frame synchronous trace memory. Address, instruction, and 8 unspecified, circuit connectable trace lines. Display can be HLL source (e.g., C source), assembly or mixed.
- A full 64k hardware configurable break, trace on, trace off control, and pass count increment events.
- Tool set integrated interactive symbolic debugger—supports both assembler (COFF) and C Compiler (.COD) linked object formats.
- Real time performance profiling analysis; selectable bucket definition.
- Watch windows, content updated automatically at each execution break.
- Instruction by instruction memory/register changes displayed on source window when in single step operation.
- Single base unit and debugger software reconfigurable to support the entire COP8 family; only the probe personality needs to change. Debugger software is processor customized, and reconfigured from a master model file.
- Processor specific symbolic display of registers and bit level assignments, configured from master model file.
- Halt/Idle mode notification.
- On-Line HELP customized to specific processor using master model file.
- Includes a copy of COP8-DEV-IBMA assembler and linker SDK.

IM Order-Information

Base Unit	
IM-COP8/400-1	iceMASTER Base Unit, 110V Power Supply
IM-COP8/400-2	iceMASTER Base Unit, 220V Power Supply
iceMASTER Probe	
MHW-884EK28DWPC	28 DIP
MHW-888EK40DWPC	40 SO
MHW-888EG44PWPC	44 PLCC
Adapter for SO Package	
MHW-SOIC 28	28 SO



TL/DD/12094–31

FIGURE 19. COP8 iceMASTER Environment

Development Support (Continued)

iceMASTER DEBUG MODULE (DM)

The iceMASTER Debug Module is a PC based, combination in-circuit emulation tool and COP8 based OTP/EPROM programming tool developed and marketed by MetaLink Corporation to support the whole COP8 family of products. National is a resale vendor for these products.

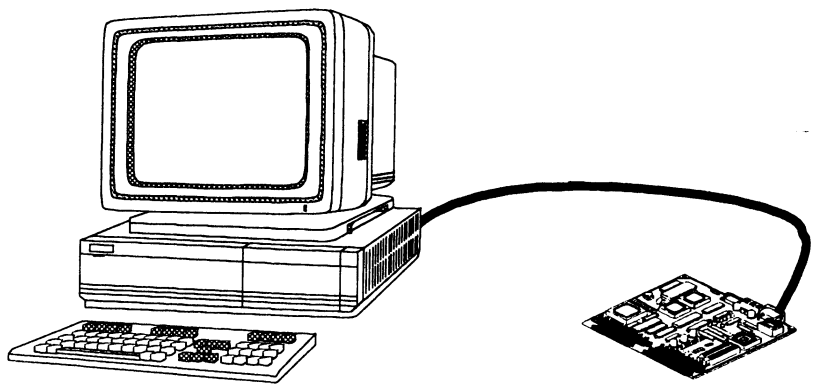
See *Figure 20* for configuration.

The iceMASTER Debug Module is a moderate cost development tool. It has the capability of in-circuit emulation for a specific COP8 microcontroller and in addition serves as a programming tool for COP8 OTP and EPROM product families. Summary of features is as follows:

- Real-time in-circuit emulation; full operating voltage range operation, full DC-10 MHz clock.
- All processor I/O pins can be cabled to an application development board with package compatible cable to socket and surface mount assembly.
- Full 32 kbyte of loadable programming space that overlays (replaces) the on-chip ROM or EPROM. On-chip RAM and I/O blocks are used directly or recreated as necessary.
- 100 frames of synchronous trace memory. The display can be HLL source (C source), assembly or mixed. The most recent history prior to a break is available in the trace memory.
- Configured break points; uses INTR instruction which is modestly intrusive.
- Software—only supported features are selectable.
- Tool set integrated interactive symbolic debugger—supports both assembler (COFF) and C Compiler (.COD) SDK linked object formats.
- Instruction by instruction memory/register changes displayed when in single step operation.
- Debugger software is processor customized, and reconfigured from a master model file.
- Processor specific symbolic display of registers and bit level assignments, configured from master model file.
- Halt/Idle mode notification.
- Programming menu supports full product line of programmable OTP and EPROM COP8 products. Program data is taken directly from the overlay RAM.
- Programming of 44 PLCC and 68 PLCC parts requires external programming adapters.
- Includes wallmount power supply.
- On-board V_{PP} generator from 5V input or connection to external supply supported. Requires V_{PP} level adjustment per the family programming specification (correct level is provided on an on-screen pop-down display).
- On-line HELP customized to specific processor using master model file.
- Includes a copy of COP8-DEV-IBMA assembler and linker SDK.

DM Order Information

Debug Module Unit	
COP8-DM/888EK	
Cable Adapters	
DM-COP8/28D	28 DIP
DM-COP8/40D	40 DIP
DM-COP8/44P	44 PLCC
Adapter for SO Package	
DM-COP8/28D-SO	28 SO



TL/DD/12094–32

FIGURE 20. COP8-DM Environment

Development Support (Continued)

COP8 ASSEMBLER/LINKER SOFTWARE DEVELOPMENT TOOL KIT

National Semiconductor offers a relocatable COP8 macro cross assembler, linker, librarian and utility software development tool kit. Features are summarized as follows:

- Basic and Feature Family instruction set by "device" type.
- Nested macro capability.
- Extensive set of assembler directives.
- Supported on PC/DOS platform.
- Generates National standard COFF output files.
- Integrated Linker and Librarian.
- Integrated utilities to generate ROM code file outputs.
- DUMPCOFF utility.

This product is integrated as a part of MetaLink tools as a development kit, fully supported by the MetaLink debugger. It may be ordered separately or it is bundled with the MetaLink products at no additional cost.

Order Information

Assembler SDK:	
COP8-DEV-IBMA	Assembler SDK on installable 3.5" PC/DOS Floppy Disk Drive format. Periodic upgrades and most recent version is available on National's BBS and Internet.

COP8 C COMPILER

A C Compiler is developed and marketed by Byte Craft Limited. The COP8C compiler is a fully integrated development tool specifically designed to support the compact embedded configuration of the COP8 family of products.

Features are summarized as follows:

- ANSI C with some restrictions and extensions that optimize development for the COP8 embedded application.
- BITS data type extension. Register declaration #pragma with direct bit level definitions.
- C language support for interrupt routines.
- Expert system, rule based code generation and optimization.
- Performs consistency checks against the architectural definitions of the target COP8 device.
- Generates program memory code.
- Supports linking of compiled object or COP8 assembled object formats.
- Global optiomization of linked code.
- Symbolic debug load format fully source level supported by the MetaLink debugger.

Approved List

Manufacturer	North America	Europe	Asia
BP Microsystems	(800) 225-2102 (713) 688-4600 Fax: (713) 688-0920	+49-8152-4183 +49-8856-932616	+852-234-16611 +852-2710-8121
Data I/O	(800) 426-1045 (206) 881-6444 Fax: (206) 882-1043	+44-0734-440011	Call North America
HI-LO	(510) 623-8860	Call Asia	+886-2-764-0215 Fax: +886-2-756-6403
ICE Technology	(800) 624-8949 (919) 430-7915	+44-1226-767404 Fax: 0-1226-370-434	
MetaLink	(800) 638-2423 (602) 926-0797 Fax: (602) 693-0681	+49-80 9156 96-0 Fax: +49-80 9123 86	+852-737-1800
Systems General	(408) 263-6667	+41-1-9450300	+886-2-917-3005 Fax: +886-2-911-1283
Needhams	(916) 924-8037 Fax: (916) 924-8065		

Development Support (Continued)

SINGLE CHIP OTP/EMULATOR SUPPORT

The COP8 family is supported by single chip OTP emulators. For detailed information refer to the emulator specific datasheet and the emulator selection table below:

OTP Emulator Ordering Information

Device Number	Clock Option	Package	Emulates
COP87L84EKN-XE	Crystal	28 DIP	COP884EK
COP87L84EKM-XE	Crystal	28 SO	COP884EK
COP87L88EKN-XE	Crystal	40 DIP	COP888EK
COP87L88EKV-XE	Crystal	44 PLCC	COP888EK

INDUSTRY WIDE OTP/EPROM PROGRAMMING SUPPORT

Programming support, in addition to the MetaLink development tools, is provided by a full range of independent approved vendors to meet the needs from the engineering laboratory to full production.

AVAILABLE LITERATURE

For more information, please see the COP8 Basic Family User's Manual, Literature Number 620895, COP8 Feature Family User's Manual, Literature Number 620897 and National's Family of 8-bit Microcontrollers COP8 Selection Guide, Literature Number 630009.

DIAL-A-HELPER SERVICE

Dial-A-Helper is a service provided by the Microcontroller Applications group. The Dial-A-Helper is an Electronic Information System that may be accessed as a Bulletin Board System (BBS) via data modem, as an FTP site on the Internet via standard FTP client application or as an FTP site on the Internet using a standard Internet browser such as Netscape or Mosaic.

The Dial-A-Helper system provides access to an automated information storage and retrieval system. The system capabilities include a MESSAGE SECTION (electronic mail, when accessed as a BBS) for communications to and from the Microcontroller Applications Group and a FILE SECTION which consists of several file areas where valuable application software and utilities could be found.

DIAL-A-HELPER BBS via a Standard Modem

Modem: CANADA/U.S.: (800) NSC-MICRO
(800) 672-6427
EUROPE: (+49) 0-814-1-351332
Baud: 14.4k
Set-up: Length: 8-Bit
Parity: None
Stop Bit: 1
Operation: 24 Hours, 7 Days

DIAL-A-HELPER via FTP

ftp nscmicro.nsc.com
user: anonymous
password: username@yourhost.site.domain

DIAL-A-HELPER via a WorldWide Web Browser

ftp://nscmicro.nsc.com

National Semiconductor on the WorldWide Web

See us on the WorldWide Web at: http://www.national.com

CUSTOMER RESPONSE CENTER

Complete product information and technical support is available from National's customer response centers.

CANADA/U.S.:	Tel:	(800) 272-9959
	email:	support @tevm2.nsc.com
EUROPE:	email:	europe.support@nsc.com
	Deutsch Tel:	+49 (0) 180-530 85 85
	English Tel:	+49 (0) 180-532 78 32
	Français Tel:	+49 (0) 180-532 93 58
	Italiano Tel:	+49 (0) 180-534 16 80
JAPAN:	Tel:	+81-043-299-2309
S.E. ASIA:	Beijing Tel:	(+86) 10-6856-8601
	Shanghai Tel:	(+86) 21-6415-4092
	Hong Kong Tel:	(+852) 2737-1600
	Korea Tel:	(+82) 2-3771-6909
	Malaysia Tel:	(+60-4) 644-9061
	Singapore Tel:	(+65) 255-2226
	Taiwan Tel:	+886-2-521-3288
AUSTRALIA:	Tel:	(+61) 3-9558-9999
INDIA:	Tel:	(+91) 80-559-9467

National Semiconductor

ADVANCE INFORMATION

COP688GD/COP888GD/COP988GD 8-Bit Microcontroller with A/D Converter

General Description

The COP888 family of microcontrollers uses an 8-bit single chip core architecture fabricated with National Semiconductor's M²CMOS™ process technology. The COP888GD is a member of this expandable 8-bit core processor family of microcontrollers. (Continued)

Key Features

- 8-channel A/D converter with prescaler and both differential and single ended modes
- Three 16-bit timers, each with two 16-bit registers supporting:
  - — Processor Independent PWM mode
  - — External Event counter mode
  - — Input Capture mode
- Quiet design (low radiated emissions)
- 16 kbytes on-board ROM
- 256 bytes on-board RAM

Additional Peripheral Features

- Multi-Input Wakeup (MIWU) with optional interrupts (8)
- WATCHDOG™ and clock monitor logic
- MICROWIRE/PLUS™ serial I/O

I/O Features

- Memory mapped I/O
- Software selectable I/O options (TRI-STATE® Output, Push-Pull Output, Weak Pull Up Input, High Impedance Input)
- Schmitt trigger inputs on ports G and L
- Package:
  - — 44 PLCC with 40 I/O pins
  - — 40 DIP with 36 I/O pins

CPU/Instruction Set Features

- 1 μs instruction cycle time
- Twelve multi-source vectored interrupts servicing
  - — External Interrupt
  - — Idle Timer T0
  - — Three Timers (each with 2 Interrupts)
  - — MICROWIRE/PLUS
  - — Multi-Input Wake Up
  - — Software Trap
  - — Default VIS (default interrupt)
- Versatile and easy to use instruction set
- 8-bit Stack Pointer (SP)—stack in RAM
- Two 8-bit Register Indirect Data Memory Pointers (B and X)

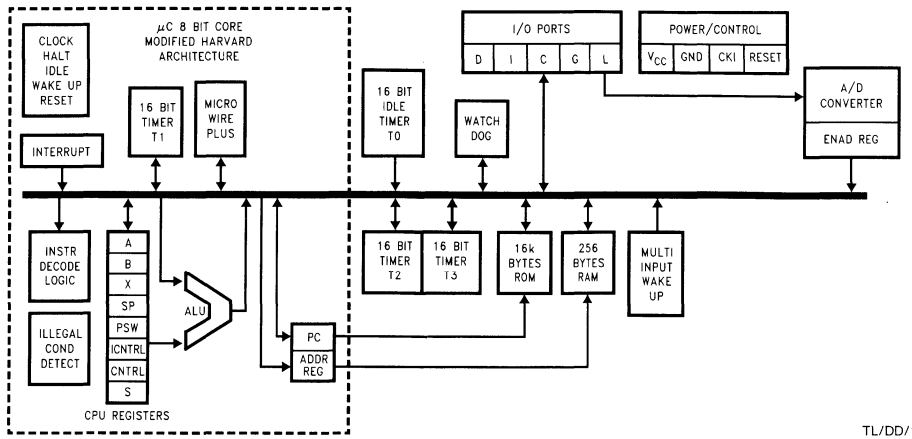
Fully Static CMOS

- Two power saving modes: HALT and IDLE
- Low current drain (typically < 1 μA)
- Single supply operation: 2.5V to 5.5V
- Temperature ranges: 0°C to +70°C, −40°C to +85°C, −55°C to +125°C

Development Support

- Emulation and OTP devices
- Real time emulation and full program debug offered by MetaLink Development System

Block Diagram



TL/DD/12870–1

FIGURE 1. Block Diagram

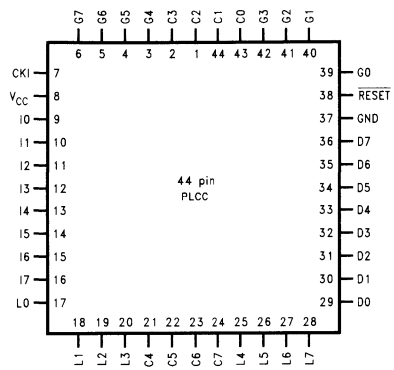
5

General Description (Continued)

It is a fully static part, fabricated using double-metal silicon gate microCMOS technology. Features include an 8-bit memory mapped architecture, MICROWIRE/PLUS serial I/O, three 16-bit timer/counters supporting three modes (Processor Independent PWM generation, External Event counter, and Input Capture mode capabilities), 8 channel analog to digital converter, and two power saving modes (HALT and IDLE), both with a multi-sourced wakeup/interrupt capability. This multi-sourced interrupt capability may also be used independent of the HALT or IDLE modes. The device includes an IDLE Timer with 5 selectable interrupt periods which can be used to wake the device from IDLE mode. Each I/O pin has software selectable configurations. The devices operate over a voltage range of 2.5V to 5.5V. High throughput is achieved with an efficient, regular instruction set operating at a maximum rate of 1 μs per instruction. Low radiated emissions are achieved by gradual turn-on output drivers and internal I_{CC} smoothing filters on the chip logic and crystal oscillator.

Connection Diagrams

Plastic Chip Carrier

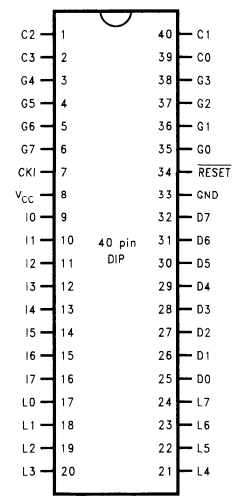


TL/DD/12870–2

Top View

Order Number COP688GD-XXX/V, COP888GD-XXX/V, COP988GD-XXX/V, or COP988GDH-XXX/V
See NS Plastic Chip Package Number V44A

Dual-In-Line Package



TL/DD/12870–3

Top View

Order Number COP688GD-XXX/N, COP888GD-XXX/N, COP988GD-XXX/N, or COP988GDH-XXX/N
See NS Molded Package Number N40A

FIGURE 2. Connection Diagrams

National Semiconductor

COP884CG/COP888CG 8-Bit Microcontroller with UART and Three Multi-Function Timers

General Description

The COP888 family of microcontrollers uses an 8-bit single chip core architecture fabricated with National Semiconductor's M²CMOS™ process technology. The COP888CG is a member of this expandable 8-bit core processor family of microcontrollers. (Continued)

Key Features

- Full duplex UART
- Three 16-bit timers, each with two 16-bit registers supporting:
  - — Processor independent PWM mode
  - — External event counter mode
  - — Input capture mode
- Quiet design (low radiated emissions)
- 4 kbytes of on-chip ROM
- 192 bytes of on-chip RAM

Additional Peripheral Features

- Idle timer
- Multi-Input Wake Up (MIWU) with optional interrupts (8)
- Two analog comparators
- WATCHDOG™ and Clock Monitor logic
- MICROWIRE/PLUS™ serial I/O

I/O Features

- Memory mapped I/O
- Software selectable I/O options (TRI-STATE® output, push-pull output, weak pull-up input, high impedance input)
- High current outputs
- Schmitt trigger inputs on Port G
- Packages:
  - — 44 PLCC with 40 I/O pins
  - — 40 DIP with 36 I/O pins
  - — 28 DIP with 24 I/O pins
  - — 28 SO with 24 I/O pins

CPU/Instruction Set Feature

- 1 μs instruction cycle time
- Fourteen multi-source vectored interrupts servicing
  - — External interrupt with selectable edge
  - — Idle timer T0
  - — Three timers (each with 2 interrupts)
  - — MICROWIRE/PLUS
  - — Multi-Input WAke Up
  - — Software trap
  - — UART (2)
  - — Default VIS (default interrupt)
- Versatile and easy to use instruction set
- 8-bit Stack Pointer (SP)—stack in RAM
- Two 8-bit register indirect data memory pointers (B, X)

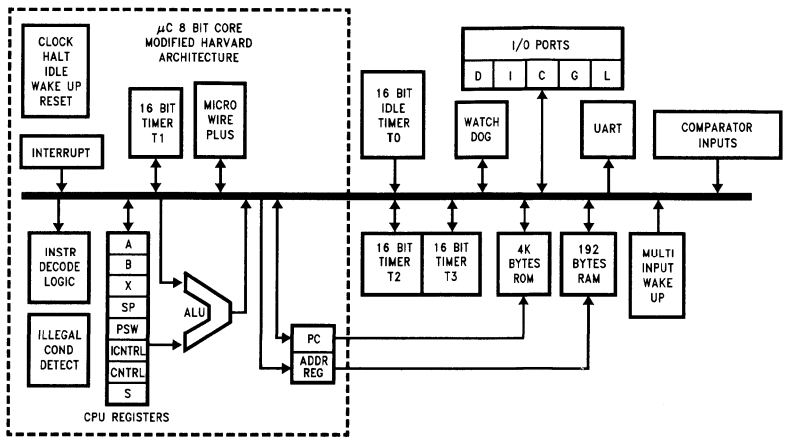
Fully Static CMOS

- Low current drain (typically < 1 μA)
- Single supply operation: 2.5V to 6.0V
- Temperature range: −40°C to +85°C

Development Support

- Emulation and OTP devices
- Real time emulation and full program debug offered by MetaLink Development System

Block Diagram



TL/DD/9765–1

FIGURE 1. Block Diagram

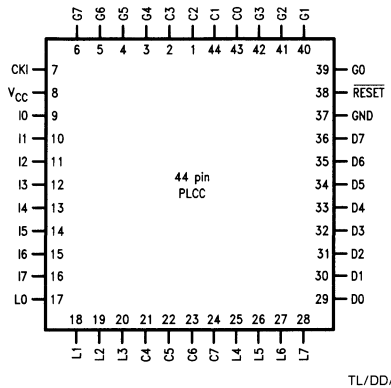
General Description (Continued)

They are fully static parts, fabricated using double-metal silicon gate microCMOS technology. Features include an 8-bit memory mapped architecture, MICROWIRE/PLUS serial I/O, three 16-bit timer/counters supporting three modes (Processor Independent PWM generation, External Event counter, and Input Capture mode capabilities), full duplex UART, two comparators, and two power savings modes (HALT and IDLE), both with a multi-sourced wakeup/interrupt capability. This multi-sourced interrupt capability may also be used independent of the HALT or IDLE modes. Each I/O pin has software selectable configurations. The device operates over a voltage range of 2.5V to 6V. High throughput is achieved with an efficient, regular instruction set operating at a maximum of 1 μs per instruction rate.

The device has reduced EMI emissions. Low radiated emissions are achieved by gradual turn-on output drivers and internal I_{CC} filters on the chip logic and crystal oscillator.

Connection Diagrams

Plastic Chip Carrier

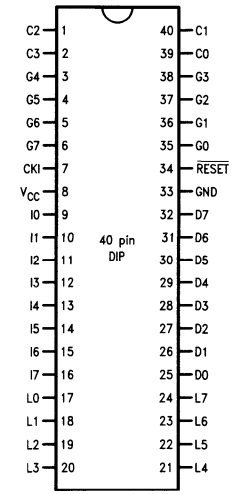


TL/DD/9765-2

Top View

Order Number COP888CG-XXX/V
See NS Plastic Chip Package Number V44A

Dual-In-Line Package

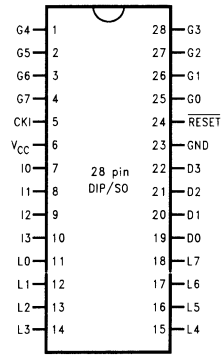


TL/DD/9765-4

Top View

Order Number COP888CG-XXX/N
See NS Molded Package Number N40A

Dual-In-Line Package



TL/DD/9765-5

Top View

Order Number COP884CG-XXX/N or COP884CG-XXX/WM
See NS Molded Package Number N28A OR M28B

FIGURE 2a. Connection Diagrams

Connection Diagrams (Continued)

Pinouts for 28-, 40- and 44-Pin Packages

Port	Type	Alt. Fun	Alt. Fun	28-Pin Pack.	40-Pin Pack.	44-Pin Pack.
L0	I/O	MIWU		11	17	17
L1	I/O	MIWU	CKX	12	18	18
L2	I/O	MIWU	TDX	13	19	19
L3	I/O	MIWU	RDX	14	20	20
L4	I/O	MIWU	T2A	15	21	25
L5	I/O	MIWU	T2B	16	22	26
L6	I/O	MIWU	T3A	17	23	27
L7	I/O	MIWU	T3B	18	24	28
G0	I/O	INT		25	35	39
G1	WDOUT			26	36	40
G2	I/O	T1B		27	37	41
G3	I/O	T1A		28	38	42
G4	I/O	SO		1	3	3
G5	I/O	SK		2	4	4
G6	I	SI		3	5	5
G7	I/CKO	HALT Restart		4	6	6
D0	O			19	25	29
D1	O			20	26	30
D2	O			21	27	31
D3	O			22	28	32
I0	I			7	9	9
I1	I	COMP1IN−		8	10	10
I2	I	COMP1IN+		9	11	11
I3	I	COMP1OUT		10	12	12
I4	I	COMP2IN−			13	13
I5	I	COMP2IN+			14	14
I6	I	COMP2OUT			15	15
I7	I				16	16
D4	O				29	33
D5	O				30	34
D6	O				31	35
D7	O				32	36
C0	I/O				39	43
C1	I/O				40	44
C2	I/O				1	1
C3	I/O				2	2
C4	I/O					21
C5	I/O					22
C6	I/O					23
C7	I/O					24
V_{CC}				6	8	8
GND				23	33	37
CKI				5	7	7
$\overline{\text{RESET}}$				24	34	38

Absolute Maximum Ratings

If Military/Aerospace specified devices are required, please contact the National Semiconductor Sales Office/Distributors for availability and specifications.

Supply Voltage (V_{CC})	7V
Voltage at Any Pin	−0.3V to V_{CC} + 0.3V
Total Current into V_{CC} Pin (Source)	100 mA
Total Current out of GND Pin (Sink)	110 mA
Storage Temperature Range	−65°C to +140°C

Note: *Absolute maximum ratings indicate limits beyond which damage to the device may occur. DC and AC electrical specifications are not ensured when operating the device at absolute maximum ratings.*

DC Electrical Characteristics $-40°C \le T_A \le +85°C$ unless otherwise specified

Parameter	Conditions	Min	Typ	Max	Units
Operating Voltage		2.5		6	V
Power Supply Ripple (Note 1)	Peak-to-Peak			0.1 V_{CC}	V
Supply Current (Note 2)					
CKI = 10 MHz	V_{CC} = 6V, t_c = 1 μs			8.0	mA
CKI = 4 MHz	V_{CC} = 6V, t_c = 2.5 μs			4.5	mA
CKI = 4 MHz	V_{CC} = 4.0V, t_c = 2.5 μs			2.5	mA
CKI = 1 MHz	V_{CC} = 4.0V, t_c = 10 μs			1.4	mA
HALT Current (Note 3)	V_{CC} = 6V, CKI = 0 MHz		<1	10	μA
	V_{CC} = 4.0V, CKI = 0 MHz		<0.5	6	μA
IDLE Current					
CKI = 10 MHz	V_{CC} = 6V, t_c = 1 μs			3.5	mA
CKI = 4 MHz	V_{CC} = 6V, t_c = 2.5 μs			2.5	mA
CKI = 1 MHz	V_{CC} = 4.0V, t_c = 10 μs			0.7	mA
Input Levels					
$\overline{\text{RESET}}$					
Logic High		0.8 V_{CC}			V
Logic Low				0.2 V_{CC}	V
CKI (External and Crystal Osc. Modes)					
Logic High		0.7 V_{CC}			V
Logic Low				0.2 V_{CC}	V
All Other Inputs					
Logic High		0.7 V_{CC}			V
Logic Low				0.2 V_{CC}	V
Hi-Z Input Leakage	V_{CC} = 6V	−2		+2	μA
Input Pullup Current	V_{CC} = 6V, V_{IN} = 0V	−40		−250	μA
G and L Port Input Hysteresis				0.35 V_{CC}	V
Output Current Levels					
D Outputs					
Source	V_{CC} = 4V, V_{OH} = 3.3V	−0.4			mA
	V_{CC} = 2.5V, V_{OH} = 1.8V	−0.2			mA
Sink	V_{CC} = 4V, V_{OL} = 1V	10			mA
	V_{CC} = 2.5V, V_{OL} = 0.4V	2.0			mA
All Others					
Source (Weak Pull-Up Mode)	V_{CC} = 4V, V_{OH} = 2.7V	−10		−100	μA
	V_{CC} = 2.5V, V_{OH} = 1.8V	−2.5		−33	μA
Source (Push-Pull Mode)	V_{CC} = 4V, V_{OH} = 3.3V	−0.4			mA
	V_{CC} = 2.5V, V_{OH} = 1.8V	−0.2			mA
Sink (Push-Pull Mode)	V_{CC} = 4V, V_{OL} = 0.4V	1.6			mA
	V_{CC} = 2.5V, V_{OL} = 0.4V	0.7			mA
TRI-STATE Leakage	V_{CC} = 6.0V	−2		+2	μA

Note 1: Rate of voltage change must be less then 0.5 V/ms.

Note 2: Supply current is measured after running 2000 cycles with a crystal/resonator oscillator, inputs at rails and outputs open.

Note 3: The HALT mode will stop CKI from oscillating in the RC and the Crystal configurations. Test conditions: All inputs tied to V_{CC}, L, C, and G0–G5 configured as outputs and set high. The D port set to zero. The clock monitor and the comparators are disabled.

DC Electrical Characteristics $-40°C \le T_A \le +85°C$ unless otherwise specified (Continued)

Parameter	Conditions	Min	Typ	Max	Units
Allowable Sink/Source Current per Pin					
D Outputs (Sink)				15	mA
All others				3	mA
Maximum Input Current without Latchup	$T_A = 25°C$			±100	mA
RAM Retention Voltage, V_r	500 ns Rise and Fall Time (Min)	2			V
Input Capacitance				7	pF
Load Capacitance on D2				1000	pF

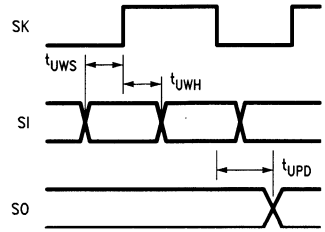
AC Electrical Characteristics $-40°C \le T_A \le +85°C$ unless otherwise specified

Parameter	Conditions	Min	Typ	Max	Units
Instruction Cycle Time (t_c)					
Crystal, Resonator,	$4V \le V_{CC} \le 6V$	1		DC	μs
R/C Oscillator	$2.5V \le V_{CC} < 4V$	2.5		DC	μs
	$4V \le V_{CC} \le 6V$	3		DC	μs
	$2.5V \le V_{CC} < 4V$	7.5		DC	μs
Inputs					
t_{SETUP}	$4V \le V_{CC} \le 6V$	200			ns
	$2.5V \le V_{CC} < 4V$	500			ns
t_{HOLD}	$4V \le V_{CC} \le 6V$	60			ns
	$2.5V \le V_{CC} < 4V$	150			ns
Output Propagation Delay (Note 4)	$R_L = 2.2k$, $C_L = 100$ pF				
t_{PD1}, t_{PD0}					
SO, SK	$4V \le V_{CC} \le 6V$			0.7	μs
	$2.5V \le V_{CC} < 4V$			1.75	μs
All Others	$4V \le V_{CC} \le 6V$			1	μs
	$2.5V \le V_{CC} < 4V$			2.5	μs
MICROWIRE™ Setup Time (t_{UWS})		20			ns
MICROWIRE Hold Time (t_{UWH})		56			ns
MICROWIRE Output Propagation Delay (t_{UPD})				220	ns
Input Pulse Width					
Interrupt Input High Time		1			t_c
Interrupt Input Low Time		1			t_c
Timer Input High Time		1			t_c
Timer Input Low Time		1			t_c
Reset Pulse Width		1			μs

Note 4: The output propagation delay is referenced to the end of the instruction cycle where the output change occurs.

Comparators AC and DC Characteristics V_{CC} = 5V, T_A = 25°C

Parameter	Conditions	Min	Typ	Max	Units
Input Offset Voltage	$0.4V \le V_{IN} \le V_{CC} - 1.5V$		±10	±25	mV
Input Common Mode Voltage Range		0.4		$V_{CC} - 1.5$	V
Low Level Output Current	V_{OL} = 0.4V	1.6			mA
High Level Output Current	V_{OH} = 4.6V	1.6			mA
DC Supply Current Per Comparator (When Enabled)				250	μA
Response Time	TBD mV Step, TBD mV Overdrive, 100 pF Load		1		μs



TL/DD/9765–7

FIGURE 2. MICROWIRE/PLUS Timing

Pin Descriptions

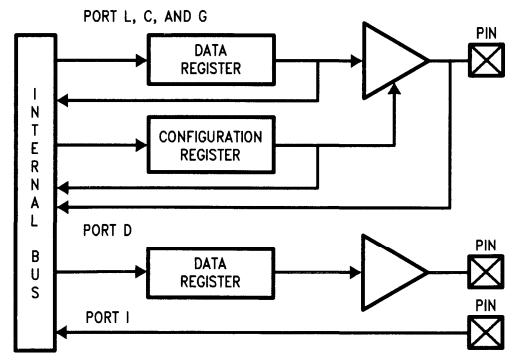
V_{CC} and GND are the power supply pins.

CKI is the clock input. This can come from an R/C generated oscillator, or a crystal oscillator (in conjunction with CKO). See Oscillator Description section.

$\overline{\text{RESET}}$ is the master reset input. See Reset Description section.

The device contains three bidirectional 8-bit I/O ports (C, G and L), where each individual bit may be independently configured as an input (Schmitt trigger inputs on ports L and G), output or TRI-STATE under program control. Three data memory address locations are allocated for each of these I/O ports. Each I/O port has two associated 8-bit memory mapped registers, the CONFIGURATION register and the output DATA register. A memory mapped address is also reserved for the input pins of each I/O port. (See the memory map for the various addresses associated with the I/O ports.) *Figure 3* shows the I/O port configurations. The DATA and CONFIGURATION registers allow for each port bit to be individually configured under software control as shown below:

CONFIGURATION Register	DATA Register	Port Set-Up
0	0	Hi-Z Input (TRI-STATE Output)
0	1	Input with Weak Pull-Up
1	0	Push-Pull Zero Output
1	1	Push-Pull One Output



TL/DD/9765–8

FIGURE 3. I/O Port Configurations

PORT L is an 8-bit I/O port. All L-pins have Schmitt triggers on the inputs.

The Port L supports Multi-Input Wake Up on all eight pins. L1 is used for the UART external clock. L2 and L3 are used for the UART transmit and receive. L4 and L5 are used for the timer input functions T2A and T2B. L6 and L7 are used for the timer input functions T3A and T3B.

The Port L has the following alternate features:

- L0 MIWU
- L1 MIWU or CKX
- L2 MIWU or TDX
- L3 MIWU or RDX
- L4 MIWU or T2A
- L5 MIWU or T2B
- L6 MIWU or T3A
- L7 MIWU or T3B

Port G is an 8-bit port with 5 I/O pins (G0, G2–G5), an input pin (G6), and two dedicated output pins (G1 and G7). Pins G0 and G2–G6 all have Schmitt Triggers on their inputs. Pin G1 serves as the dedicated WDOUT WATCHDOG output, while pin G7 is either input or output depending on the oscillator mask option selected. With the crystal oscillator option selected, G7 serves as the dedicated output pin for the CKO clock output. With the single-pin R/C oscillator mask option selected, G7 serves as a general purpose input pin but is also used to bring the device out of HALT mode with a low to high transition on G7. There are two registers associated with the G Port, a data register and a configuration register. Therefore, each of the 5 I/O bits (G0, G2–G5) can be individually configured under software control.

Pin Descriptions (Continued)

Since G6 is an input only pin and G7 is the dedicated CKO clock output pin (crystal clock option) or general purpose input (R/C clock option), the associated bits in the data and configuration registers for G6 and G7 are used for special purpose functions as outlined below. Reading the G6 and G7 data bits will return zeros.

Note that the chip will be placed in the HALT mode by writing a "1" to bit 7 of the Port G Data Register. Similarly the chip will be placed in the IDLE mode by writing a "1" to bit 6 of the Port G Data Register.

Writing a "1" to bit 6 of the Port G Configuration Register enables the MICROWIRE/PLUS to operate with the alternate phase of the SK clock. The G7 configuration bit, if set high, enables the clock start up delay after HALT when the R/C clock configuration is used.

	Config Reg.	Data Reg.
G7	CLKDLY	HALT
G6	Alternate SK	IDLE

Port G has the following alternate features:

- G0 INTR (External Interrupt Input)
- G2 T1B (Timer T1 Capture Input)
- G3 T1A (Timer T1 I/O)
- G4 SO (MICROWIRE™ Serial Data Output)
- G5 SK (MICROWIRE Serial Clock)
- G6 SI (MICROWIRE Serial Data Input)

Port G has the following dedicated functions:

- G1 WDOUT WATCHDOG and/or Clock Monitor dedicated output
- G7 CKO Oscillator dedicated output or general purpose input

Port C is an 8-bit I/O port. The 40-pin device does not have a full complement of Port C pins. The unavailable pins are not terminated. A read operation for these unterminated pins will return unpredicatable values.

PORT I is an eight-bit Hi-Z input port. The 28-pin device does not have a full complement of Port I pins. The unavailable pins are not terminated i.e., they are floating. A read operation for these unterminated pins will return unpredictable values. The user must ensure that the software takes this into account by either masking or restricting the accesses to bit operations. The unterminated Port I pins will draw power only when addressed.

Port I1–I3 are used for Comparator 1. Port I4–I6 are used for Comparator 2.

The Port I has the following alternate features.

- I1 COMP1 − IN (Comparator 1 Negative Input)
- I2 COMP1 + IN (Comparator 1 Positive Input)
- I3 COMP1OUT (Comparator 1 Output)
- I4 COMP2 − IN (Comparator 2 Negative Input)
- I5 COMP2 + IN (Comparator 2 Positive Input)
- I6 COMP2OUT (Comparator 2 Output)

Port D is an 8-bit output port that is preset high when $\overline{\text{RESET}}$ goes low. The user can tie two or more D port outputs together in order to get a higher drive.

Functional Description

The architecture of the device is modified Harvard architecture. With the Harvard architecture, the control store program memory (ROM) is separated from the data store memory (RAM). Both ROM and RAM have their own separate addressing space with separate address buses. The architecture, though based on Harvard architecture, permits transfer of data from ROM to RAM.

CPU REGISTERS

The CPU can do an 8-bit addition, subtraction, logical or shift operation in one instruction (t_C) cycle time.

There are six CPU registers:

A is the 8-bit Accumulator Register

PC is the 15-bit Program Counter Register

PU is the upper 7 bits of the program counter (PC)
PL is the lower 8 bits of the program counter (PC)

B is an 8-bit RAM address pointer, which can be optionally post auto incremented or decremented.

X is an 8-bit alternate RAM address pointer, which can be optionally post auto incremented or decremented.

SP is the 8-bit stack pointer, which points to the subroutine/interrupt stack (in RAM). The SP is initialized to RAM address 06F with reset.

S is the 8-bit Data Segment Address Register used to extend the lower half of the address range (00 to 7F) into 256 data segments of 128 bytes each.

All the CPU registers are memory mapped with the exception of the Accumulator (A) and the Program Counter (PC).

PROGRAM MEMORY

The program memory consists of 4096 bytes of ROM. These bytes may hold program instructions or constant data (data tables for the LAID instruction, jump vectors for the JID instruction, and interrupt vectors for the VIS instruction). The program memory is addressed by the 15-bit program counter (PC). All interrupts in the devices vector to program memory location 0FF Hex.

DATA MEMORY

The data memory address space includes the on-chip RAM and data registers, the I/O registers (Configuration, Data and Pin), the control registers, the MICROWIRE/PLUS SIO shift register, and the various registers, and counters associated with the timers (with the exception of the IDLE timer). Data memory is addressed directly by the instruction or indirectly by the B, X, SP pointers and S register.

The device has 192 bytes of RAM. Sixteen bytes of RAM are mapped as "registers" at addresses 0F0 to 0FF Hex. These registers can be loaded immediately, and also decremented and tested with the DRSZ (decrement register and skip if zero) instruction. The memory pointer registers X, SP, B and S are memory mapped into this space at address locations 0FC to 0FF Hex respectively, with the other registers being available for general usage.

The instruction set permits any bit in memory to be set, reset or tested. All I/O and registers (except A and PC) are memory mapped; therefore, I/O bits and register bits can be directly and individually set, reset and tested. The accumulator (A) bits can also be directly and individually tested.

Note: RAM contents are undefined upon power-up.

Data Memory Segment RAM Extension

Data memory address 0FF is used as a memory mapped location for the Data Segment Address Register (S).

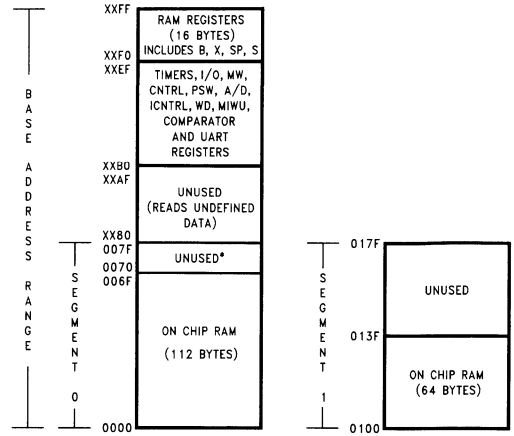
The data store memory is either addressed directly by a single byte address within the instruction, or indirectly relative to the reference of the B, X, or SP pointers (each contains a single-byte address). This single-byte address allows an addressing range of 256 locations from 00 to FF hex. The upper bit of this single-byte address divides the data store memory into two separate sections as outlined previously. With the exception of the RAM register memory from address locations 00F0 to 00FF, all RAM memory is memory mapped with the upper bit of the single-byte address being equal to zero. This allows the upper bit of the single-byte address to determine whether or not the base address range (from 0000 to 00FF) is extended. If this upper bit equals one (representing address range 0080 to 00FF), then address extension does not take place. Alternatively, if this upper bit equals zero, then the data segment extension register S is used to extend the base address range (from 0000 to 007F) from XX00 to XX7F, where XX represents the 8 bits from the S register. Thus the 128-byte data segment extensions are located from addresses 0100 to 017F for data segment 1, 0200 to 027F for data segment 2, etc., up to FF00 to FF7F for data segment 255. The base address range from 0000 to 007F represents data segment 0.

Figure 4 illustrates how the S register data memory extension is used in extending the lower half of the base address range (00 to 7F hex) into 256 data segments of 128 bytes each, with a total addressing range of 32 kbytes from XX00 to XX7F. This organization allows a total of 256 data segments of 128 bytes each with an additional upper base segment of 128 bytes. Furthermore, all addressing modes are available for all data segments. The S register must be changed under program control to move from one data segment (128 bytes) to another. However, the upper base segment (containing the 16 memory registers, I/O registers, control registers, etc.) is always available regardless of the contents of the S register, since the upper base segment (address range 0080 to 00FF) is independent of data segment extension.

The instructions that utilize the stack pointer (SP) always reference the stack as part of the base segment (Segment 0), regardless of the contents of the S register. The S register is not changed by these instructions. Consequently, the stack (used with subroutine linkage and interrupts) is always located in the base segment. The stack pointer will be initialized to point at data memory location 006F as a result of reset.

The 128 bytes of RAM contained in the base segment are split between the lower and upper base segments. The first 116 bytes of RAM are resident from address 0000 to 006F in the lower base segment, while the remaining 16 bytes of RAM represent the 16 data memory registers located at addresses 00F0 to 00FF of the upper base segment. No RAM is located at the upper sixteen addresses (0070 to 007F) of the lower base segment.

Additional RAM beyond these initial 128 bytes, however, will always be memory mapped in groups of 128 bytes (or less) at the data segment address extensions (XX00 to XX7F) of the lower base segment. The additional 64 bytes of RAM (beyond the initial 128 bytes) are memory mapped at address locations 0100 to 013F hex.



TL/DD/9765–9

*Reads as all ones.

FIGURE 4. RAM Organization

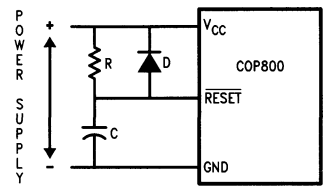
Reset

The $\overline{\text{RESET}}$ input when pulled low initializes the microcontroller. Initialization will occur whenever the $\overline{\text{RESET}}$ input is pulled low. Upon initialization, the data and configuration registers for ports L, G and C are cleared, resulting in these Ports being initialized to the TRI-STATE mode. Pin G1 of the G Port is an exception (as noted below) since pin G1 is dedicated as the WATCHDOG and/or Clock Monitor error output pin. Port D is set high. The PC, PSW, ICNTRL, CNTRL, T2CNTRL and T3CNTRL control registers are cleared. The UART registers PSR, ENU (except that TBMT bit is set), ENUR and ENUI are cleared. The Comparator Select Register is cleared. The S register is initialized to zero. The Multi-Input Wakeup registers WKEN, WKEDG and WKPND are cleared. The stack pointer, SP, is initialized to 6F Hex.

The device comes out of reset with both the WATCHDOG logic and the Clock Monitor detector armed, with the WATCHDOG service window bits set and the Clock Monitor bit set. The WATCHDOG and Clock Monitor circuits are inhibited during reset. The WATCHDOG service window bits being initialized high default to the maximum WATCHDOG service window of 64k t_C clock cycles. The Clock Monitor bit being initialized high will cause a Clock Monitor error following reset if the clock has not reached the minimum specified frequency at the termination of reset. A Clock Monitor error will cause an active low error output on pin G1. This error output will continue until 16 t_C–32 t_C clock cycles following the clock frequency reaching the minimum specified value, at which time the G1 output will enter the TRI-STATE mode.

The external RC network shown in *Figure 5* should be used to ensure that the $\overline{\text{RESET}}$ pin is held low until the power supply to the chip stabilizes.

Reset (Continued)



TL/DD/9765–10

RC > 5 × Power Supply Rise Time

FIGURE 5. Recommended Reset Circuit

Oscillator Circuits

The chip can be driven by a clock input on the CKI input pin which can be between DC and 10 MHz. The CKO output clock is on pin G7 (crystal configuration). The CKI input frequency is divided down by 10 to produce the instruction cycle clock ($1/t_C$).

Figure 6 shows the Crystal and R/C diagrams.

CRYSTAL OSCILLATOR

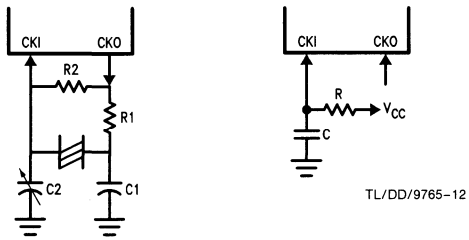
CKI and CKO can be connected to make a closed loop crystal (or resonator) controlled oscillator.

Table A shows the component values required for various standard crystal values.

R/C OSCILLATOR

By selecting CKI as a single pin oscillator input, a single pin R/C oscillator circuit can be connected to it. CKO is available as a general purpose input, and/or HALT restart input.

Table B shows the variation in the oscillator frequencies as functions of the component (R and C) values.



TL/DD/9765–12

TL/DD/9765–11

FIGURE 6. Crystal and R/C Oscillator Diagrams

TABLE A. Crystal Oscillator Configuration, T_A = 25°C

R1 (kΩ)	R2 (MΩ)	C1 (pF)	C2 (pF)	CKI Freq (MHz)	Conditions
0	1	30	30–36	10	V_{CC} = 5V
0	1	30	30–36	4	V_{CC} = 5.0V
0	1	200	100–150	0.455	V_{CC} = 5V

TABLE B. RC Oscillator Configuration, T_A = 25°C

R (kΩ)	C (pF)	CKI Freq (MHz)	Instr. Cycle (μs)	Conditions
3.3	82	2.2 to 2.7	3.7 to 4.6	V_{CC} = 5V
5.6	100	1.1 to 1.3	7.4 to 9.0	V_{CC} = 5V
6.8	100	0.9 to 1.1	8.8 to 10.8	V_{CC} = 5V

Note: 3k ≤ R ≤ 200k

50 pF ≤ C ≤ 200 pF

Control Registers

CNTRL Register (Address X'00EE)

The Timer1 (T1) and MICROWIRE/PLUS control register contains the following bits:

SL1 & SL0	Select the MICROWIRE/PLUS clock divide by (00 = 2, 01 = 4, 1x = 8)
IEDG	External interrupt edge polarity select (0 = Rising edge, 1 = Falling edge)
MSEL	Selects G5 and G4 as MICROWIRE/PLUS signals SK and SO respectively
T1C0	Timer T1 Start/Stop control in timer modes 1 and 2
	Timer T1 Underflow Interrupt Pending Flag in timer mode 3
T1C1	Timer T1 mode control bit
T1C2	Timer T1 mode control bit
T1C3	Timer T1 mode control bit

T1C3	T1C2	T1C1	T1C0	MSEL	IEDG	SL1	SL0

Bit 7 Bit 0

Control Registers (Continued)

PSW Register (Address X'00EF)

The PSW register contains the following select bits:

- GIE Global interrupt enable (enables interrupts)
- EXEN Enable external interrupt
- BUSY MICROWIRE/PLUS busy shifting flag
- EXPND External interrupt pending
- T1ENA Timer T1 Interrupt Enable for Timer Underflow or T1A Input capture edge
- T1PNDA Timer T1 Interrupt Pending Flag (Autoreload RA in mode 1, T1 Underflow in Mode 2, T1A capture edge in mode 3)
- C Carry Flag
- HC Half Carry Flag

HC	C	T1PNDA	T1ENA	EXPND	BUSY	EXEN	GIE

Bit 7 Bit 0

The Half-Carry bit is also affected by all the instructions that affect the Carry flag. The SC (Set Carry) and RC (Reset Carry) instructions will respectively set or clear both the carry flags. In addition to the SC and RC instructions, ADC, SUBC, RRC and RLC instructions affect the carry and Half Carry flags.

ICNTRL Register (Address X'00E8)

The ICNTRL register contains the following bits:

- T1ENB Timer T1 Interrupt Enable for T1B Input capture edge
- T1PNDB Timer T1 Interrupt Pending Flag for T1B capture edge
- μWEN Enable MICROWIRE/PLUS interrupt
- μWPND MICROWIRE/PLUS interrupt pending
- T0EN Timer T0 Interrupt Enable (Bit 12 toggle)
- T0PND Timer T0 Interrupt pending
- LPEN L Port Interrupt Enable (Multi-Input Wakeup/Interrupt)

  Bit 7 could be used as a flag

Unused	LPEN	T0PND	T0EN	μWPND	μWEN	T1PNDB	T1ENB

Bit 7 Bit 0

T2CNTRL Register (Address X'00C6)

The T2CNTRL register contains the following bits:

- T2ENB Timer T2 Interrupt Enable for T2B Input capture edge
- T2PNDB Timer T2 Interrupt Pending Flag for T2B capture edge
- T2ENA Timer T2 Interrupt Enable for Timer Underflow or T2A Input capture edge
- T2PNDA Timer T2 Interrupt Pending Flag (Autoreload RA in mode 1, T2 Underflow in mode 2, T2A capture edge in mode 3)
- T2C0 Timer T2 Start/Stop control in timer modes 1 and 2 Timer T2 Underflow Interrupt Pending Flag in timer mode 3
- T2C1 Timer T2 mode control bit
- T2C2 Timer T2 mode control bit
- T2C3 Timer T2 mode control bit

T2C3	T2C2	T2C1	T2C0	T2PNDA	T2ENA	T2PNDB	T2ENB

Bit 7 Bit 0

T3CNTRL Register (Address X'00B6)

The T3CNTRL register contains the following bits:

- T3ENB Timer T3 Interrupt Enable for T3B
- T3PNDB Timer T3 Interrupt Pending Flag for T3B pin (T3B capture edge)
- T3ENA Timer T3 Interrupt Enable for Timer Underflow or T3A pin
- T3PNDA Timer T3 Interrupt Pending Flag (Autoload RA in mode 1, T3 Underflow in mode 2, T3a capture edge in mode 3)
- T3C0 Timer T3 Start/Stop control in timer modes 1 and 2

  Timer T3 Underflow Interrupt Pending Flag in timer mode 3
- T3C1 Timer T3 mode control bit
- T3C2 Timer T3 mode control bit
- T3C3 Timer T3 mode control bit

T3C3	T3C2	T3C1	T3C0	T3PNDA	T3ENA	T3PNDB	T3ENB

Bit 7 Bit 0

Timers

The device contains a very versatile set of timers (T0, T1, T2, T3). All timers and associated autoreload/capture registers power up containing random data.

TIMER T0 (IDLE TIMER)

The device supports applications that require maintaining real time and low power with the IDLE mode. This IDLE mode support is furnished by the IDLE timer T0, which is a 16-bit timer. The Timer T0 runs continuously at the fixed rate of the instruction cycle clock, t_C. The user cannot read or write to the IDLE Timer T0, which is a count down timer.

The Timer T0 supports the following functions:

Exit out of the Idle Mode (See Idle Mode description)
WATCHDOG logic (See WATCHDOG description)
Start up delay out of the HALT mode

The IDLE Timer T0 can generate an interrupt when the thirteenth bit toggles. This toggle is latched into the T0PND pending flag, and will occur every 4 ms at the maximum clock frequency (t_C = 1 μs). A control flag T0EN allows the interrupt from the thirteenth bit of Timer T0 to be enabled or disabled. Setting T0EN will enable the interrupt, while resetting it will disable the interrupt.

Timers (Continued)

TIMER T1, TIMER T2 AND TIMER T3

The device has a set of three powerful timer/counter blocks, T1, T2 and T3. The associated features and functioning of a timer block are described by referring to the timer block Tx. Since the three timer blocks, T1, T2 and T3 are identical, all comments are equally applicable to any of the three timer blocks.

Each timer block consists of a 16-bit timer, Tx, and two supporting 16-bit autoreload/capture registers, RxA and RxB. Each timer block has two pins associated with it, TxA and TxB. The pin TxA supports I/O required by the timer block, while the pin TxB is an input to the timer block. The powerful and flexible timer block allows the device to easily perform all timer functions with minimal software overhead. The timer block has three operating modes: Processor Independent PWM mode, External Event Counter mode, and Input Capture mode.

The control bits TxC3, TxC2, and TxC1 allow selection of the different modes of operation.

Mode 1. Processor Independent PWM Mode

As the name suggests, this mode allows the device to generate a PWM signal with very minimal user intervention. The user only has to define the parameters of the PWM signal (ON time and OFF time). Once begun, the timer block will continuously generate the PWM signal completely independent of the microcontroller. The user software services the timer block only when the PWM parameters require updating.

In this mode the timer Tx counts down at a fixed rate of t_C. Upon every underflow the timer is alternately reloaded with the contents of supporting registers, RxA and RxB. The very first underflow of the timer causes the timer to reload from the register RxA. Subsequent underflows cause the timer to be reloaded from the registers alternately beginning with the register RxB.

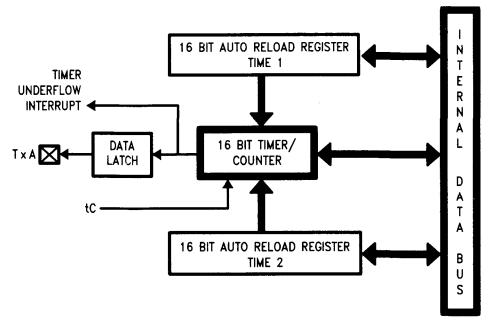
The Tx Timer control bits, TxC3, TxC2 and TxC1 set up the timer for PWM mode operation.

Figure 7 shows a block diagram of the timer in PWM mode.

The underflows can be programmed to toggle the TxA output pin. The underflows can also be programmed to generate interrupts.

Underflows from the timer are alternately latched into two pending flags, TxPNDA and TxPNDB. The user must reset these pending flags under software control. Two control enable flags, TxENA and TxENB, allow the interrupts from the timer underflow to be enabled or disabled. Setting the timer enable flag TxENA will cause an interrupt when a timer underflow causes the RxA register to be reloaded into the timer. Setting the timer enable flag TxENB will cause an interrupt when a timer underflow causes the RxB register to be reloaded into the timer. Resetting the timer enable flags will disable the associated interrupts.

Either or both of the timer underflow interrupts may be enabled. This gives the user the flexibility of interrupting once per PWM period on either the rising or falling edge of the PWM output. Alternatively, the user may choose to interrupt on both edges of the PWM output.



TL/DD/9765–14

FIGURE 7. Timer in PWM Mode

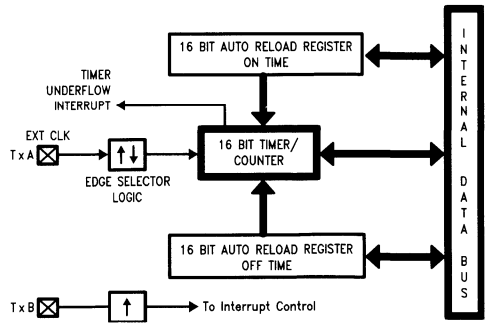
Mode 2. External Event Counter Mode

This mode is quite similar to the processor independent PWM mode described above. The main difference is that the timer, Tx, is clocked by the input signal from the TxA pin. The Tx timer control bits, TxC3, TxC2 and TxC1 allow the timer to be clocked either on a positive or negative edge from the TxA pin. Underflows from the timer are latched into the TxPNDA pending flag. Setting the TxENA control flag will cause an interrupt when the timer underflows.

In this mode the input pin TxB can be used as an independent positive edge sensitive interrupt input if the TxENB control flag is set. The occurrence of a positive edge on the TxB input pin is latched into the TxPNDB flag.

Figure 8 shows a block diagram of the timer in External Event Counter mode.

Note: The PWM output is not available in this mode since the TxA pin is being used as the counter input clock.



TL/DD/9765–15

FIGURE 8. Timer in External Event Counter Mode

Mode 3. Input Capture Mode

The device can precisely measure external frequencies or time external events by placing the timer block, Tx, in the input capture mode.

In this mode, the timer Tx is constantly running at the fixed t_C rate. The two registers, RxA and RxB, act as capture registers. Each register acts in conjunction with a pin. The register RxA acts in conjunction with the TxA pin and the register RxB acts in conjunction with the TxB pin.

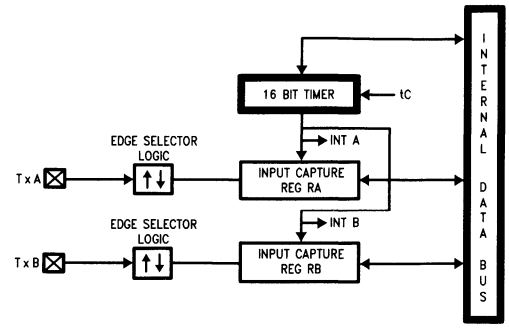
Timers (Continued)

The timer value gets copied over into the register when a trigger event occurs on its corresponding pin. Control bits, TxC3, TxC2 and TxC1, allow the trigger events to be specified either as a positive or a negative edge. The trigger condition for each input pin can be specified independently.

The trigger conditions can also be programmed to generate interrupts. The occurrence of the specified trigger condition on the TxA and TxB pins will be respectively latched into the pending flags, TxPNDA and TxPNDB. The control flag TxENA allows the interrupt on TxA to be either enabled or disabled. Setting the TxENA flag enables interrupts to be generated when the selected trigger condition occurs on the TxA pin. Similarly, the flag TxENB controls the interrupts from the TxB pin.

Underflows from the timer can also be programmed to generate interrupts. Underflows are latched into the timer TxC0 pending flag (the TxC0 control bit serves as the timer underflow interrupt pending flag in the Input Capture mode). Consequently, the TxC0 control bit should be reset when entering the Input Capture mode. The timer underflow interrupt is enabled with the TxENA control flag. When a TxA interrupt occurs in the Input Capture mode, the user must check both the TxPNDA and TxC0 pending flags in order to determine whether a TxA input capture or a timer underflow (or both) caused the interrupt.

Figure 9 shows a block diagram of the timer in Input Capture mode.



TL/DD/9765–16

FIGURE 9. Timer in Input Capture Mode

TIMER CONTROL FLAGS

The timers T1, T2 and T3 have indentical control structures. The control bits and their functions are summarized below.

TxC0	Timer Start/Stop control in Modes 1 and 2 (Processor Independent PWM and External Event Counter), where 1 = Start, 0 = Stop Timer Underflow Interrupt Pending Flag in Mode 3 (Input Capture)
TxPNDA	Timer Interrupt Pending Flag
TxPNDB	Timer Interrupt Pending Flag
TxENA	Timer Interrupt Enable Flag
TxENB	Timer Interrupt Enable Flag 1 = Timer Interrupt Enabled 0 = Timer Interrupt Disabled
TxC3	Timer mode control
TxC2	Timer mode control
TxC1	Timer mode control

Timers (Continued)

The timer mode control bits (TxC3, TxC2 and TxC1) are detailed below:

TxC3	TxC2	TxC1	Timer Mode	Interrupt A Source	Interrupt B Source	Timer Counts On
0	0	0	MODE 2 (External Event Counter)	Timer Underflow	Pos. TxB Edge	TxA Pos. Edge
0	0	1	MODE 2 (External Event Counter)	Timer Underflow	Pos. TxB Edge	TxA Neg. Edge
1	0	1	MODE 1 (PWM) TxA Toggle	Autoreload RA	Autoreload RB	t_C
1	0	0	MODE 1 (PWM) No TxA Toggle	Autoreload RA	Autoreload RB	t_C
0	1	0	MODE 3 (Capture) Captures: TxA Pos. Edge TxB Pos. Edge	Pos. TxA Edge or Timer Underflow	Pos. TxB Edge	t_C
1	1	0	MODE 3 (Capture) Captures: TxA Pos. Edge TxB Neg. Edge	Pos. TxA Edge or Timer Underflow	Neg. TxB Edge	t_C
0	1	1	MODE 3 (Capture) Captures: TxA Neg. Edge TxB Pos. Edge	Neg. TxB Edge or Timer Underflow	Pos. TxB Edge	t_C
1	1	1	MODE 3 (Capture) Captures: TxA Neg. Edge TxB Neg. Edge	Neg. TxA Edge or Timer Underflow	Neg. TxB Edge	t_C

Power Save Modes

The device offers the user two power save modes of operation: HALT and IDLE. In the HALT mode, all microcontroller activities are stopped. In the IDLE mode, the on-board oscillator circuitry the WATCHDOG logic, the Clock Monitor and timer T0 are active but all other microcontroller activities are stopped. In either mode, all on-board RAM, registers, I/O states, and timers (with the exception of T0) are unaltered.

HALT MODE

The device can be placed in the HALT mode by writing a "1" to the HALT flag (G7 data bit). All microcontroller activities, including the clock and timers, are stopped. The WATCHDOG logic is disabled during the HALT mode. However, the clock monitor circuitry if enabled remains active and will cause the WATCHDOG output pin (WDOUT) to go low. If the HALT mode is used and the user does not want to activate the WDOUT pin, the Clock Monitor should be disabled after the device comes out of reset (resetting the Clock Monitor control bit with the first write to the WDSVR register). In the HALT mode, the power requirements of the device are minimal and the applied voltage (V_{CC}) may be decreased to V_r (V_r = 2.0V) without altering the state of the machine.

The device supports three different ways of exiting the HALT mode. The first method of exiting the HALT mode is with the Multi-Input Wakeup feature on the L port. The second method is with a low to high transition on the CKO (G7) pin. This method precludes the use of the crystal clock configuration (since CKO becomes a dedicated output), and so may be used with an RC clock configuration. The third method of exiting the HALT mode is by pulling the $\overline{\text{RESET}}$ pin low.

Since a crystal or ceramic resonator may be selected as the oscillator, the Wakeup signal is not allowed to start the chip running immediately since crystal oscillators and ceramic resonators have a delayed start up time to reach full amplitude and frequency stability. The IDLE timer is used to generate a fixed delay to ensure that the oscillator has indeed stabilized before allowing instruction execution. In this case, upon detecting a valid Wakeup signal, only the oscillator circuitry is enabled. The IDLE timer is loaded with a value of 256 and is clocked with the t_C instruction cycle clock. The t_C clock is derived by dividing the oscillator clock down by a factor of 10. The Schmitt trigger following the CKI inverter on the chip ensures that the IDLE timer is clocked only when the oscillator has a sufficiently large amplitude to meet the Schmitt trigger specifications. This Schmitt trigger is not part of the oscillator closed loop. The startup timeout from the IDLE timer enables the clock signals to be routed to the rest of the chip.

If an RC clock option is being used, the fixed delay is introduced optionally. A control bit, CLKDLY, mapped as configuration bit G7, controls whether the delay is to be introduced or not. The delay is included if CLKDLY is set, and excluded if CLKDLY is reset. The CLKDLY bit is cleared on reset.

Power Save Modes (Continued)

The device has two mask options associated with the HALT mode. The first mask option enables the HALT mode feature, while the second mask option disables the HALT mode. With the HALT mode enable mask option, the device will enter and exit the HALT mode as described above. With the HALT disable mask option, the device cannot be placed in the HALT mode (writing a "1" to the HALT flag will have no effect).

The WATCHDOG detector circuit is inhibited during the HALT mode. However, the clock monitor circuit if enabled remains active during HALT mode in order to ensure a clock monitor error if the device inadvertently enters the HALT mode as a result of a runaway program or power glitch.

IDLE MODE

The device is placed in the IDLE mode by writing a "1" to the IDLE flag (G6 data bit). In this mode, all activities, except the associated on-board oscillator circuitry, the WATCHDOG logic, the clock monitor and the IDLE Timer T0, are stopped.

As with the HALT mode, the device can be returned to normal operation with a reset, or with a Multi-Input Wakeup from the L Port. Alternately, the microcontroller resumes normal operation from the IDLE mode when the thirteenth bit (representing 4.096 ms at internal clock frequency of 1 MHz, $t_C = 1\ \mu s$) of the IDLE Timer toggles.

This toggle condition of the thirteenth bit of the IDLE Timer T0 is latched into the T0PND pending flag.

The user has the option of being interrupted with a transition on the thirteenth bit of the IDLE Timer T0. The interrupt can be enabled or disabled via the T0EN control bit. Setting the T0EN flag enables the interrupt and vice versa.

The user can enter the IDLE mode with the Timer T0 interrupt enabled. In this case, when the T0PND bit gets set, the device will first execute the Timer T0 interrupt service routine and then return to the instruction following the "Enter Idle Mode" instruction.

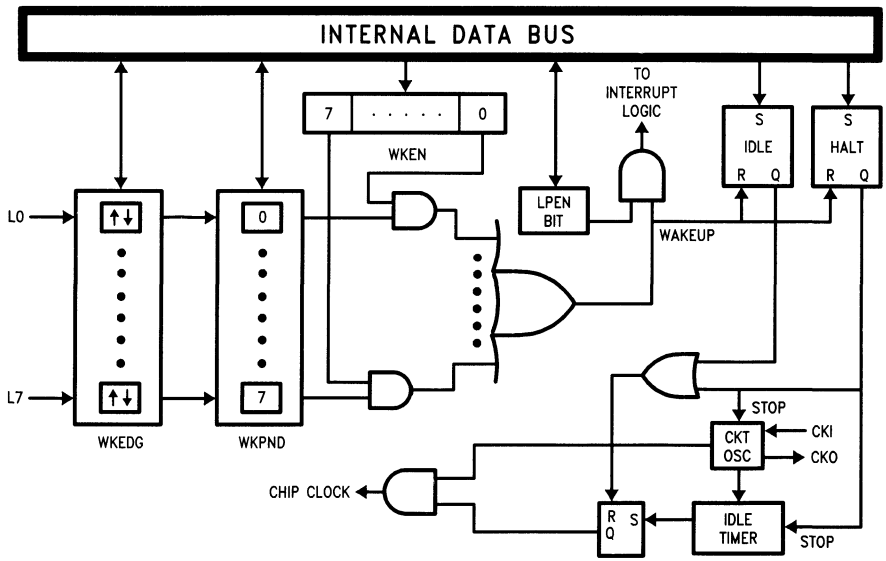
Alternatively, the user can enter the IDLE mode with the IDLE Timer T0 interrupt disabled. In this case, the device will resume normal operation with the instruction immediately following the "Enter IDLE Mode" instruction.

Note: It is necessary to program two NOP instructions following both the set HALT mode and set IDLE mode instructions. These NOP instructions are necessary to allow clock resynchronization following the HALT or IDLE modes.

Multi-Input Wakeup

The Multi-Input Wakeup feature is ued to return (wakeup) the device from either the HALT or IDLE modes. Alternately Multi-Input Wakeup/Interrupt feature may also be used to generate up to 8 edge selectable external interrupts.

Figure 10 shows the Multi-Input Wakeup logic.



TL/DD/9765–17

FIGURE 10. Multi-Input Wake Up Logic

Multi-Input Wakeup (Continued)

The Multi-Input Wakeup feature utilizes the L Port. The user selects which particular L port bit (or combination of L Port bits) will cause the device to exit the HALT or IDLE modes. The selection is done through the Reg: WKEN. The Reg: WKEN is an 8-bit read/write register, which contains a control bit for every L port bit. Setting a particular WKEN bit enables a Wakeup from the associated L port pin.

The user can select whether the trigger condition on the selected L Port pin is going to be either a positive edge (low to high transition) or a negative edge (high to low transition). This selection is made via the Reg: WKEDG, which is an 8-bit control register with a bit assigned to each L Port pin. Setting the control bit will select the trigger condition to be a negative edge on that particular L Port pin. Resetting the bit selects the trigger condition to be a positive edge. Changing an edge select entails several steps in order to avoid a pseudo Wakeup condition as a result of the edge change. First, the associated WKEN bit should be reset, followed by the edge select change in WKEDG. Next, the associated WKPND bit should be cleared, followed by the associated WKEN bit being re-enabled.

An example may serve to clarify this procedure. Suppose we wish to change the edge select from positive (low going high) to negative (high going low) for L Port bit 5, where bit 5 has previously been enabled for an input interrupt. The program would be as follows:

```
RBIT  5, WKEN
SBIT  5, WKEDG
RBIT  5, WKPND
SBIT  5, WKEN
```

If the L port bits have been used as outputs and then changed to inputs with Multi-Input Wakeup/Interrupt, a safety procedure should also be followed to avoid inherited pseudo wakeup conditions. After the selected L port bits have been changed from output to input but before the associated WKEN bits are enabled, the associated edge select bits in WKEDG should be set or reset for the desired edge selects, followed by the associated WKPND bits being cleared.

This same procedure should be used following reset, since the L port inputs are left floating as a result of reset.

The occurrence of the selected trigger condition for Multi-Input Wakeup is latched into a pending register called WKPND. The respective bits of the WKPND register will be set on the occurrence of the selected trigger edge on the corresponding Port L pin. The user has the responsibility of clearing these pending flags. Since WKPND is a pending register for the occurrence of selected wakeup conditions, the device will not enter the HALT mode if any Wakeup bit is both enabled and pending. Consequently, the user has the responsibility of clearing the pending flags before attempting to enter the HALT mode.

WKEN, WKPND and WKEDG are all read/write registers, and are cleared at reset.

PORT L INTERRUPTS

Port L provides the user with an additional eight fully selectable, edge sensitive interrupts which are all vectored into the same service subroutine.

The interrupt from Port L shares logic with the wake up circuitry. The register WKEN allows interrupts from Port L to be individually enabled or disabled. The register WKEDG specifies the trigger condition to be either a positive or a negative edge. Finally, the register WKPND latches in the pending trigger conditions.

The GIE (Global Interrupt Enable) bit enables the interrupt function.

A control flag, LPEN, functions as a global interrupt enable for Port L interrupts. Setting the LPEN flag will enable interrupts and vice versa. A separate global pending flag is not needed since the register WKPND is adequate.

Since Port L is also used for waking the device out of the HALT or IDLE modes, the user can elect to exit the HALT or IDLE modes either with or without the interrupt enabled. If he elects to disable the interrupt, then the device will restart execution from the instruction immediately following the instruction that placed the microcontroller in the HALT or IDLE modes. In the other case, the device will first execute the interrupt service routine and then revert to normal operation.

The Wakeup signal will not start the chip running immediately since crystal oscillators or ceramic resonators have a finite start up time. The IDLE Timer (T0) generates a fixed delay to ensure that the oscillator has indeed stabilized before allowing the device to execute instructions. In this case, upon detecting a valid Wakeup signal, only the oscillator circuitry and the IDLE Timer T0 are enabled. The IDLE Timer is loaded with a value of 256 and is clocked from the t_C instruction cycle clock. The t_C clock is derived by dividing down the oscillator clock by a factor of 10. A Schmitt trigger following the CKI on-chip inverter ensures that the IDLE timer is clocked only when the oscillator has a sufficiently large amplitude to meet the Schmitt trigger specifications. This Schmitt trigger is not part of the oscillator closed loop. The startup timeout from the IDLE timer enables the clock signals to be routed to the rest of the chip.

If the RC clock option is used, the fixed delay is under software control. A control flag, CLKDLY, in the G7 configuration bit allows the clock start up delay to be optionally inserted. Setting CLKDLY flag high will cause clock start up delay to be inserted and resetting it will exclude the clock start up delay. The CLKDLY flag is cleared during reset, so the clock start up delay is not present following reset with the RC clock options.

UART

The COP888CG contains a full-duplex software programmable UART. The UART *(Figure 11)* consists of a transmit shift register, a receiver shift register and seven addressable registers, as follows: a transmit buffer register (TBUF), a receiver buffer register (RBUF), a UART control and status register (ENU), a UART receive control and status register (ENUR), a UART interrupt and clock source register (ENUI), a prescaler select register (PSR) and baud (BAUD) register. The ENU register contains flags for transmit and receive functions; this register also determines the length of the data frame (7, 8 or 9 bits), the value of the ninth bit in transmission, and parity selection bits. The ENUR register flags framming, data overrun and parity errors while the UART is receiving.

Other functions of the ENUR register include saving the ninth bit received in the data frame, enabling or disabling the UART's attention mode of operation and providing additional receiver/transmitter status information via RCVG and XMTG bits. The determination of an internal or external clock source is done by the ENUI register, as well as selecting the number of stop bits and enabling or disabling transmit and receive interrupts. A control flag in this register can also select the UART mode of operation: asynchronous or synchronous.

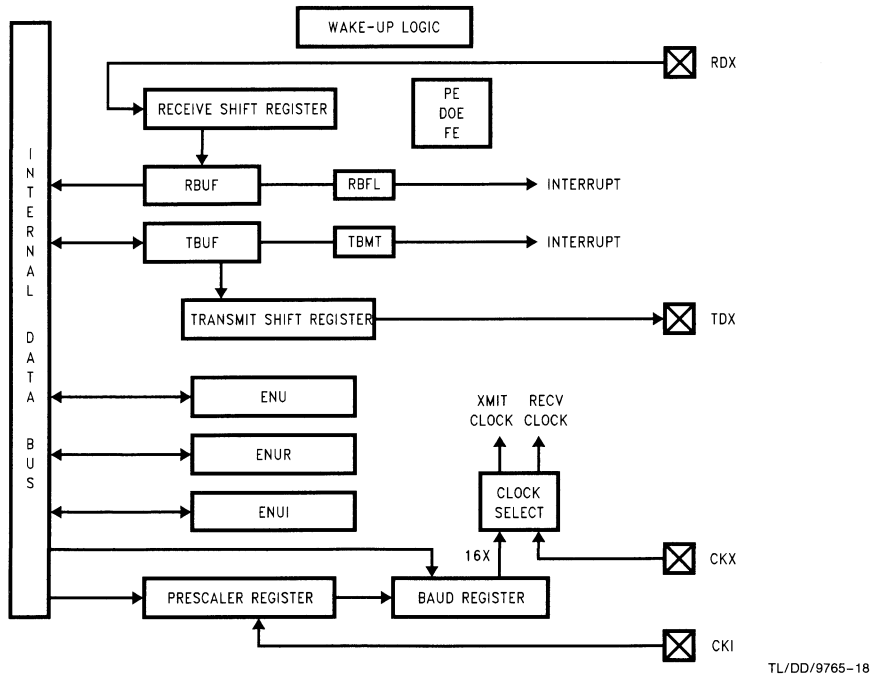


FIGURE 11. UART Block Diagram

UART (Continued)

UART CONTROL AND STATUS REGISTERS

The operation of the UART is programmed through three registers: ENU, ENUR and ENUI. The function of the individual bits in these registers is as follows:

ENU-UART Control and Status Register (Address at 0BA)

PEN	PSEL1	XBIT9/ PSEL0	CHL1	CHL0	ERR	RBFL	TBMT
0RW	0RW	0RW	0RW	0RW	0R	0R	1R

Bit 7 Bit 0

ENUR-UART Receive Control and Status Register (Address at 0BB)

DOE	FE	PE	SPARE	RBIT9	ATTN	XMTG	RCVG
0RD	0RD	0RD	0RW*	0R	0RW	0R	0R

Bit7 Bit0

ENUI-UART Interrupt and Clock Source Register (Address at 0BC)

STP2	STP78	ETDX	SSEL	XRCLK	XTCLK	ERI	ETI
0RW	0RW	0RW	0RW	0RW	0RW	0RW	0RW

Bit7 Bit0

*Bit is not used.

0 Bit is cleared on reset.

1 Bit is set to one on reset.

R Bit is read-only; it cannot be written by software.

RW Bit is read/write.

D Bit is cleared on read; when read by software as a one, it is cleared automatically. Writing to the bit does not affect its state.

DESCRIPTION OF UART REGISTER BITS

ENU—UART CONTROL AND STATUS REGISTER

TBMT: This bit is set when the UART transfers a byte of data from the TBUF register into the TSFT register for transmission. It is automatically reset when software writes into the TBUF register.

RBFL: This bit is set when the UART has received a complete character and has copied it into the RBUF register. It is automatically reset when software reads the character from RBUF.

ERR: This bit is a global UART error flag which gets set if any or a combination of the errors (DOE, FE, PE) occur.

CHL1, CHL0: These bits select the character frame format. Parity is not included and is generated/verified by hardware.

CHL1 = 0, CHL0 = 0 The frame contains eight data bits.
CHL1 = 0, CHL0 = 1 The frame contains seven data bits.
CHL1 = 1, CHL0 = 0 The frame contains nine data bits.
CHL1 = 1, CHL0 = 1 Loopback Mode selected. Transmitter output internally looped back to receiver input. Nine bit framing format is used.

XBIT9/PSEL0: Programs the ninth bit for transmission when the UART is operating with nine data bits per frame. For seven or eight data bits per frame, this bit in conjunction with PSEL1 selects parity.

PSEL1, PSEL0: Parity select bits.

PSEL1 = 0, PSEL0 = 0 Odd Parity (if Parity enabled)
PSEL1 = 0, PSEL0 = 1 Even Parity (if Parity enabled)
PSEL1 = 1, PSEL0 = 0 Mark(1) (if Parity enabled)
PSEL1 = 1, PSEL0 = 1 Space(0) (if Parity enabled)

PEN: This bit enables/disables Parity (7- and 8-bit modes only).

PEN = 0 Parity disabled.
PEN = 1 Parity enabled.

ENUR—UART RECEIVE CONTROL AND STATUS REGISTER

RCVG: This bit is set high whenever a framing error occurs and goes low when RDX goes high.

XMTG: This bit is set to indicate that the UART is transmitting. It gets reset at the end of the last frame (end of last Stop bit).

ATTN: ATTENTION Mode is enabled while this bit is set. This bit is cleared automatically on receiving a character with data bit nine set.

RBIT9: Contains the ninth data bit received when the UART is operating with nine data bits per frame.

SPARE: Reserved for future use.

PE: Flags a Parity Error.

PE = 0 Indicates no Parity Error has been detected since the last time the ENUR register was read.
PE = 1 Indicates the occurrence of a Parity Error.

FE: Flags a Framing Error.

FE = 0 Indicates no Framing Error has been detected since the last time the ENUR register was read.
FE = 1 Indicates the occurrence of a Framing Error.

DOE: Flags a Data Overrun Error.

DOE = 0 Indicates no Data Overrun Error has been detected since the last time the ENUR register was read.
DOE = 1 Indicates the occurrence of a Data Overrun Error.

ENUI—UART INTERRUPT AND CLOCK SOURCE REGISTER

ETI: This bit enables/disables interrupt from the transmitter section.

ETI = 0 Interrupt from the transmitter is disabled.
ETI = 1 Interrupt from the transmitter is enabled.

ERI: This bit enables/disables interrupt from the receiver section.

ERI = 0 Interrupt from the receiver is disabled.
ERI = 1 Interrupt from the receiver is enabled.

XTCLK: This bit selects the clock source for the transmitter-section.

XTCLK = 0 The clock source is selected through the PSR and BAUD registers.
XTCLK = 1 Signal on CKX (L1) pin is used as the clock.

XRCLK: This bit selects the clock source for the receiver section.

XRCLK = 0 The clock source is selected through the PSR and BAUD registers.
XRCLK = 1 Signal on CKX (L1) pin is used as the clock.

SSEL: UART mode select.

SSEL = 0 Asynchronous Mode.
SSEL = 1 Synchronous Mode.

UART (Continued)

ETDX: TDX (UART Transmit Pin) is the alternate function assigned to Port L pin L2; it is selected by setting ETDX bit. To simulate line break generation, software should reset ETDX bit and output logic zero to TDX pin through Port L data and configuration registers.

STP78: This bit is set to program the last Stop bit to be 7/8th of a bit in length.

STP2: This bit programs the number of Stop bits to be transmitted.

STP2 = 0 One Stop bit transmitted.
STP2 = 1 Two Stop bits transmitted.

Associated I/O Pins

Data is transmitted on the TDX pin and received on the RDX pin. TDX is the alternate function assigned to Port L pin L2; it is selected by setting ETDX (in the ENUI register) to one. RDX is an inherent function of Port L pin L3, requiring no setup.

The baud rate clock for the UART can be generated on-chip, or can be taken from an external source. Port L pin L1 (CKX) is the external clock I/O pin. The CKX pin can be either an input or an output, as determined by Port L Configuration and Data registers (Bit 1). As an input, it accepts a clock signal which may be selected to drive the transmitter and/or receiver. As an output, it presents the internal Baud Rate Generator output.

UART Operation

The UART has two modes of operation: asynchronous mode and synchronous mode.

ASYNCHRONOUS MODE

This mode is selected by resetting the SSEL (in the ENUI register) bit to zero. The input frequency to the UART is 16 times the baud rate.

The TSFT and TBUF registers double-buffer data for transmission. While TSFT is shifting out the current character on the TDX pin, the TBUF register may be loaded by software with the next byte to be transmitted. When TSFT finishes transmitting the current character the contents of TBUF are transferred to the TSFT register and the Transmit Buffer Empty Flag (TBMT in the ENU register) is set. The TBMT flag is automatically reset by the UART when software loads a new character into the TBUF register. There is also the XMTG bit which is set to indicate that the UART is transmitting. This bit gets reset at the end of the last frame (end of last Stop bit). TBUF is a read/write register.

The RSFT and RBUF registers double-buffer data being received. The UART receiver continually monitors the signal on the RDX pin for a low level to detect the beginning of a Start bit. Upon sensing this low level, it waits for half a bit time and samples again. If the RDX pin is still low, the receiver considers this to be a valid Start bit, and the remaining bits in the character frame are each sampled a single time, at the mid-bit position. Serial data input on the RDX pin is shifted into the RSFT register. Upon receiving the complete character, the contents of the RSFT register are copied into the RBUF register and the Received Buffer Full Flag (RBFL) is set. RBFL is automatically reset when software reads the character from the RBUF register. RBUF is a read only register. There is also the RCVG bit which is set high when a framing error occurs and goes low once RDX goes high. TBMT, XMTG, RBFL and RCVG are read only bits.

SYNCHRONOUS MODE

In this mode data is transferred synchronously with the clock. Data is transmitted on the rising edge and received on the falling edge of the synchronous clock.

This mode is selected by setting SSEL bit in the ENUI register. The input frequency to the UART is the same as the baud rate.

When an external clock input is selected at the CKX pin, data transmit and receive are performed synchronously with this clock through TDX/RDX pins.

If data transmit and receive are selected with the CKX pin as clock output, the device generates the synchronous clock output at the CKX pin. The internal baud rate generator is used to produce the synchronous clock. Data transmit and receive are performed synchronously with this clock.

FRAMING FORMATS

The UART supports several serial framing formats *(Figure 12)*. The format is selected using control bits in the ENU, ENUR and ENUI registers.

The first format (1, 1a, 1b, 1c) for data transmission (CHL0 = 1, CHL1 = 0) consists of Start bit, seven Data bits (excluding parity) and 7/8, one or two Stop bits. In applications using parity, the parity bit is generated and verified by hardware.

The second format (CHL0 = 0, CHL1 = 0) consists of one Start bit, eight Data bits (excluding parity) and 7/8, one or two Stop bits. Parity bit is generated and verified by hardware.

The third format for transmission (CHL0 = 0, CHL1 = 1) consists of one Start bit, nine Data bits and 7/8, one or two Stop bits. This format also supports the UART "ATTENTION" feature. When operating in this format, all eight bits of TBUF and RBUF are used for data. The ninth data bit is transmitted and received using two bits in the ENU and ENUR registers, called XBIT9 and RBIT9. RBIT9 is a read only bit. Parity is not generated or verified in this mode.

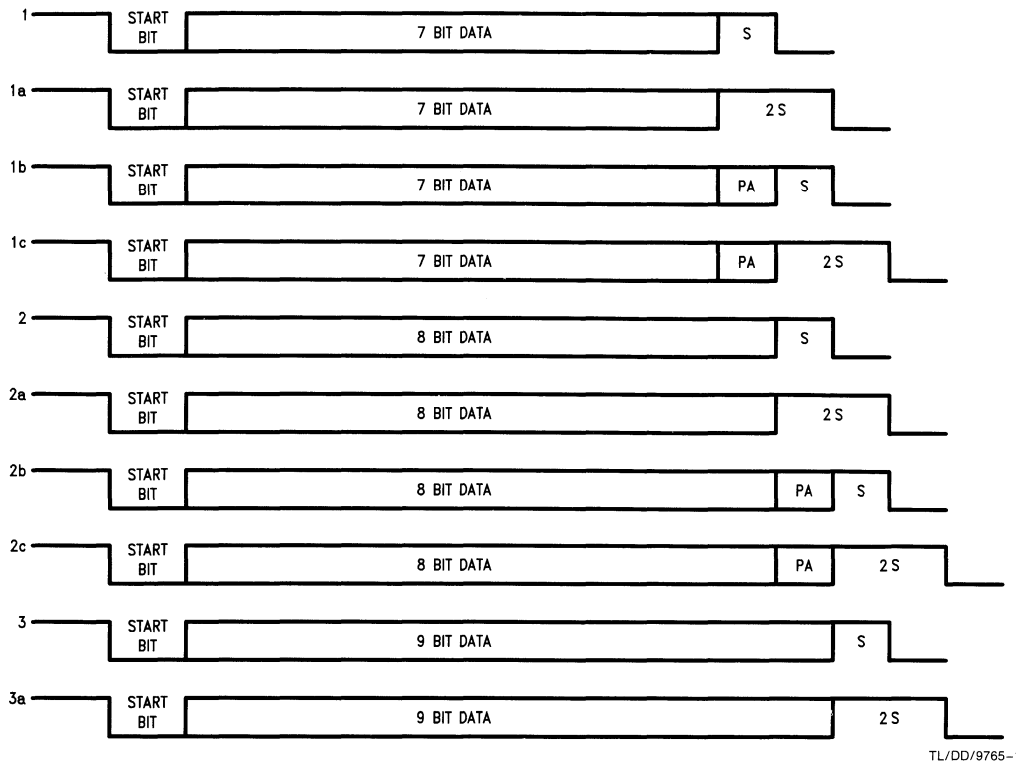
For any of the above framing formats, the last Stop bit can be programmed to be 7/8th of a bit in length. If two Stop bits are selected and the 7/8th bit is set (selected), the second Stop bit will be 7/8th of a bit in length.

The parity is enabled/disabled by PEN bit located in the ENU register. Parity is selected for 7- and 8-bit modes only. If parity is enabled (PEN = 1), the parity selection is then performed by PSEL0 and PSEL1 bits located in the ENU register.

Note that the XBIT9/PSEL0 bit located in the ENU register serves two mutually exclusive functions. This bit programs the ninth bit for transmission when the UART is operating with nine data bits per frame. There is no parity selection in this framing format. For other framing formats XBIT9 is not needed and the bit is PSEL0 used in conjunction with PSEL1 to select parity.

The frame formats for the receiver differ from the transmitter in the number of Stop bits required. The receiver only requires one Stop bit in a frame, regardless of the setting of the Stop bit selection bits in the control register. Note that an implicit assumption is made for full duplex UART operation that the framing formats are the same for the transmitter and receiver.

UART Operation (Continued)



TL/DD/9765–19

FIGURE 12. Framing Formats

UART INTERRUPTS

The UART is capable of generating interrupts. Interrupts are generated on Receive Buffer Full and Transmit Buffer Empty. Both interrupts have individual interrupt vectors. Two bytes of program memory space are reserved for each interrupt vector. The two vectors are located at addresses 0xEC to 0xEF Hex in the program memory space. The interrupts can be individually enabled or disabled using Enable Transmit Interrupt (ETI) and Enable Receive Interrupt (ERI) bits in the ENUI register.

The interrupt from the Transmitter is set pending, and remains pending, as long as both the TBMT and ETI bits are set. To remove this interrupt, software must either clear the ETI bit or write to the TBUF register (thus clearing the TBMT bit).

The interrupt from the receiver is set pending, and remains pending, as long as both the RBFL and ERI bits are set. To remove this interrupt, software must either clear the ERI bit or read from the RBUF register (thus clearing the RBFL bit).

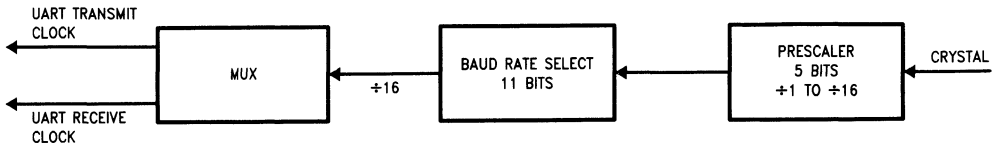
Baud Clock Generation

The clock inputs to the transmitter and receiver sections of the UART can be individually selected to come either from an external source at the CKX pin (port L, pin L1) or from a source selected in the PSR and BAUD registers. Internally, the basic baud clock is created from the oscillator frequency through a two-stage divider chain consisting of a 1–16 (increments of 0.5) prescaler and an 11-bit binary counter. *(Figure 13)* The divide factors are specified through two read/write registers shown in *Figure 14*. Note that the 11-bit Baud Rate Divisor spills over into the Prescaler Select Register (PSR). PSR is cleared upon reset.

As shown in Table I, a Prescaler Factor of 0 corresponds to NO CLOCK. NO CLOCK condition is the UART power down mode where the UART clock is turned off for power saving purpose. The user must also turn the UART clock off when a different baud rate is chosen.

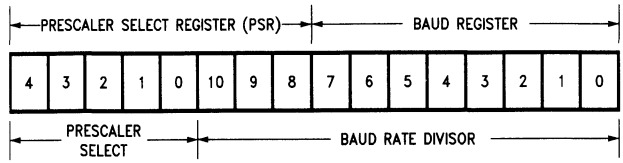
The correspondences between the 5-bit Prescaler Select and Prescaler factors are shown in Table I. Therer are many ways to calculate the two divisor factors, but one particularly effective method would be to achieve a 1.8432 MHz frequency coming out of the first stage. The 1.8432 MHz prescaler output is then used to drive the software programmable baud rate counter to create a x16 clock for the following baud rates: 110, 134.5, 150, 300, 600, 1200, 1800, 2400, 3600, 4800, 7200, 9600, 19200 and 38400 (Table II). Other baud rates may be created by using appropriate divisors. The x16 clock is then divided by 16 to provide the rate for the serial shift registers of the transmitter and receiver.

Baud Clock Generation (Continued)



TL/DD/9765-20

FIGURE 13. UART BAUD Clock Generation



TL/DD/9765-21

FIGURE 14. UART BAUD Clock Divisor Registers

TABLE I. Prescaler Factors

Prescaler Select	Prescaler Factor
00000	NO CLOCK
00001	1
00010	1.5
00011	2
00100	2.5
00101	3
00110	3.5
00111	4
01000	4.5
01001	5
01010	5.5
01011	6
01100	6.5
01101	7
01110	7.5
01111	8
10000	8.5
10001	9
10010	9.5
10011	10
10100	10.5
10101	11
10110	11.5
10111	12
11000	12.5
11001	13
11010	13.5
11011	14
11100	14.5
11101	15
11110	15.5
11111	16

TABLE II. Baud Rate Divisors (1.8432 MHz Prescaler Output)

Baud Rate	Baud Rate Divisor − 1 (N-1)
110 (110.03)	1046
134.5 (134.58)	855
150	767
300	383
600	191
1200	95
1800	63
2400	47
3600	31
4800	23
7200	15
9600	11
19200	5
38400	2

The entries in Table II assume a prescaler output of 1.8432 MHz. In the asynchronous mode the baud rate could be as high as 625k.

As an example, considering the Asynchronous Mode and a CKI clock of 4.608 MHz, the prescaler factor selected is:

$$4.608/1.8432 = 2.5$$

The 2.5 entry is available in Table I. The 1.8432 MHz prescaler output is then used with proper Baud Rate Divisor (Table II) to obtain different baud rates. For a baud rate of 19200 e.g., the entry in Table II is 5.

$N - 1 = 5$ ($N - 1$ is the value from Table II)

$N = 6$ (N is the Baud Rate Divisor)

$$\text{Baud Rate} = 1.8432\ \text{MHz}/(16 \times 6) = 19200$$

The divide by 16 is performed because in the asynchronous mode, the input frequency to the UART is 16 times the baud rate. The equation to calculate baud rates is given below.

The actual Baud Rate may be found from:

$$BR = Fc/(16 \times N \times P)$$

Baud Clock Generation (Continued)

Where:

BR is the Baud Rate

Fc is the CKI frequency

N is the Baud Rate Divisor (Table II).

P is the Prescaler Divide Factor selected by the value in the Prescaler Select Register (Table I)

Note: In the Synchronous Mode, the divisor 16 is replaced by two.

Example:

Asynchronous Mode:

Crystal Frequency = 5 MHz

Desired baud rate = 9600

Using the above equation N × P can be calculated first.

$$N \times P = (5 \times 10^6)/(16 \times 9600) = 32.552$$

Now 32.552 is divided by each Prescaler Factor (Table II) to obtain a value closest to an integer. This factor happens to be 6.5 (P = 6.5).

$$N = 32.552/6.5 = 5.008\ (N = 5)$$

The programmed value (from Table II) should be 4 (N − 1).

Using the above values calculated for N and P:

$$BR = (5 \times 10^6)/(16 \times 5 \times 6.5) = 9615.384$$

$$\%\ \text{error} = (9615.385 - 9600)/9600 = 0.16$$

Effect of HALT/IDLE

The UART logic is reinitialized when either the HALT or IDLE modes are entered. This reinitialization sets the TBMT flag and resets all read only bits in the UART control and status registers. Read/Write bits remain unchanged. The Transmit Buffer (TBUF) is not affected, but the Transmit Shift register (TSFT) bits are set to one. The receiver registers RBUF and RSFT are not affected.

The device will exit from the HALT/IDLE modes when the Start bit of a character is detected at the RDX (L3) pin. This feature is obtained by using the Multi-Input Wakeup scheme provided on the device.

Before entering the HALT or IDLE modes the user program must select the Wakeup source to be on the RDX pin. This selection is done by setting bit 3 of WKEN (Wakeup Enable) register. The Wakeup trigger condition is then selected to be high to low transition. This is done via the WKEDG register (Bit 3 is one.)

If the device is halted and crystal oscillator is used, the Wakeup signal will not start the chip running immediately because of the finite start up time requirement of the crystal oscillator. The idle timer (T0) generates a fixed (256 t_C) delay to ensure that the oscillator has indeed stabilized before allowing the device to execute code. The user has to consider this delay when data transfer is expected immediately after exiting the HALT mode.

Diagnostic

Bits CHARL0 and CHARL1 in the ENU register provide a loopback feature for diagnostic testing of the UART. When these bits are set to one, the following occur: The receiver input pin (RDX) is internally connected to the transmitter output pin (TDX); the output of the Transmitter Shift Register is "looped back" into the Receive Shift Register input. In this mode, data that is transmitted is immediately received. This feature allows the processor to verify the transmit and receive data paths of the UART.

Note that the framing format for this mode is the nine bit format; one Start bit, nine data bits, and 7/8, one or two Stop bits. Parity is not generated or verified in this mode.

Attention Mode

The UART Receiver section supports an alternate mode of operation, referred to as ATTENTION Mode. This mode of operation is selected by the ATTN bit in the ENUR register. The data format for transmission must also be selected as having nine Data bits and either 7/8, one or two Stop bits.

The ATTENTION mode of operation is intended for use in networking the device with other processors. Typically in such environments the messages consists of device addresses, indicating which of several destinations should receive them, and the actual data. This Mode supports a scheme in which addresses are flagged by having the ninth bit of the data field set to a 1. If the ninth bit is reset to a zero the byte is a Data byte.

While in ATTENTION mode, the UART monitors the communication flow, but ignores all characters until an address character is received. Upon receiving an address character, the UART signals that the character is ready by setting the RBFL flag, which in turn interrupts the processor if UART Receiver interrupts are enabled. The ATTN bit is also cleared automatically at this point, so that data characters as well as address characters are recognized. Software examines the contents of the RBUF and responds by deciding either to accept the subsequent data stream (by leaving the ATTN bit reset) or to wait until the next address character is seen (by setting the ATTN bit again).

Operation of the UART Transmitter is not affected by selection of this Mode. The value of the ninth bit to be transmitted is programmed by setting XBIT9 appropriately. The value of the ninth bit received is obtained by reading RBIT9. Since this bit is located in ENUR register where the error flags reside, a bit operation on it will reset the error flags.

Comparators

The device contains two differential comparators, each with a pair of inputs (positive and negative) and an output. Ports I1–I3 and I4–I6 are used for the comparators. The following is the Port I assignment:

- I1 Comparator1 negative input
- I2 Comparator1 positive input
- I3 Comparator1 output
- I4 Comparator2 negative input
- I5 Comparator2 positive input
- I6 Comparator2 output

A Comparator Select Register (CMPSL) is used to enable the comparators, read the outputs of the comparators internally, and enable the outputs of the comparators to the pins. Two control bits (enable and output enable) and one result bit are associated with each comparator. The comparator result bits (CMP1RD and CMP2RD) are read only bits which will read as zero if the associated comparator is not enabled. The Comparator Select Register is cleared with reset, resulting in the comparators being disabled. The comparators should also be disabled before entering either the HALT or IDLE modes in order to save power. The configuration of the CMPSL register is as follows:

Comparators (Continued)

CMPSL REGISTER (ADDRESS X'00B7)

The CMPSL register contains the following bits:

CMP1EN Enable comparator 1

CMP1RD Comparator 1 result (this is a read only bit, which will read as 0 if the comparator is not enabled)

CMP10E Selects pin I3 as comparator 1 output provided that CMPIEN is set to enable the comparator

CMP2EN Enable comparator 2

CMP2RD Comparator 2 result (this is a read only bit, which will read as 0 if the comparator is not enabled)

CMP20E Selects pin I6 as comparator 2 output provided that CMP2EN is set to enable the comparator

Unused	CMP20E	CMP2RD	CMP2EN	CMP10E	CMP1RD	CMP1EN	Unused
Bit 7							Bit 0

Note that the two unused bits of CMPSL may be used as software flags.

Comparator outputs have the same spec as Ports L and G except that the rise and fall times are symmetrical.

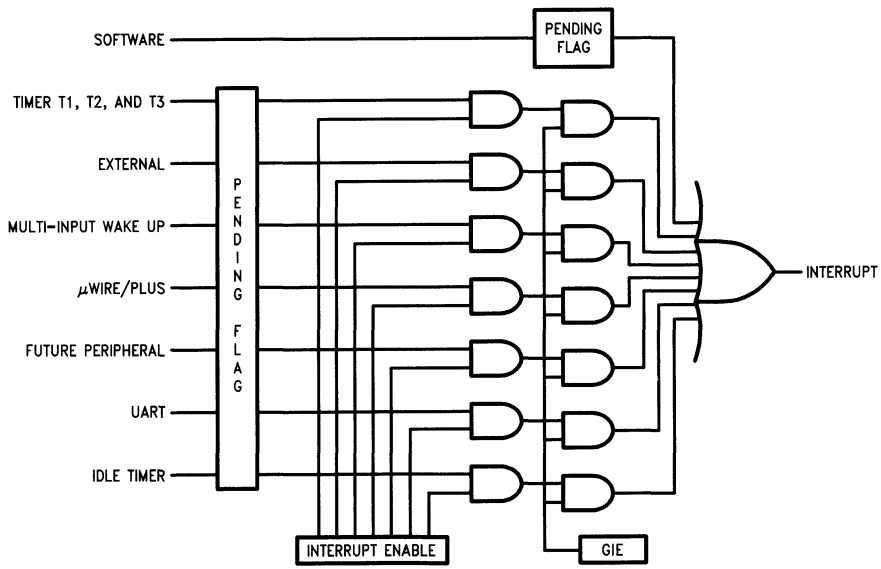
Interrupts

The device supports a vectored interrupt scheme. It supports a total of fourteen interrupt sources. The following table lists all the possible device interrupt sources, their arbitration ranking and the memory locations reserved for the interrupt vector for each source.

Two bytes of program memory space are reserved for each interrupt source. All interrupt sources except the software interrupt are maskable. Each of the maskable interrupts have an Enable bit and a Pending bit. A maskable interrupt is active if its associated enable and pending bits are set. If GIE = 1 and an interrupt is active, then the processor will be interrupted as soon as it is ready to start executing an instruction except if the above conditions happen during the Software Trap service routine. This exception is described in the Software Trap sub-section.

The interruption process is accomplished with the INTR instruction (opcode 00), which is jammed inside the Instruction Register and replaces the opcode about to be executed. The following steps are performed for every interrupt:

1. The GIE (Global Interrupt Enable) bit is reset.
2. The address of the instruction about to be executed is pushed into the stack.
3. The PC (Program Counter) branches to address 00FF. This procedure takes 7 t_C cycles to execute.



TL/DD/9765-22

FIGURE 15. Interrupt Block Diagram

Interrupts (Continued)

Arbitration Ranking	Source	Description	Vector Address Hi-Low Byte
(1) Highest	Software	INTR Instruction	0yFE–0yFF
	Reserved	for Future Use	0yFC–0yFD
(2)	External	Pin G0 Edge	0yFA–0yFB
(3)	Timer T0	Underflow	0yF8–0yF9
(4)	Timer T1	T1A/Underflow	0yF6–0yF7
(5)	Timer T1	T1B	0yF4–0yF5
(6)	MICROWIRE/PLUS	BUSY Goes Low	0yF2–0yF3
	Reserved	for Future Use	0yF0–0yF1
(7)	UART	Receive	0yEE–0yEF
(8)	UART	Transmit	0yEC–0yED
(9)	Timer T2	T2A/Underflow	0yEA–0yEB
(10)	Timer T2	T2B	0yE8–0yE9
(11)	Timer T3	T3A/Underflow	0yE6–0yE7
(12)	Timer T3	T3B	0yE4–0yE5
(13)	Port L/Wakeup	Port L Edge	0yE2–0yE3
(14) Lowest	Default	VIS Instr. Execution without Any Interrupts	0yE0–0yE1

y is VIS page, y $\neq$ 0.

At this time, since GIE = 0, other maskable interrupts are disabled. The user is now free to do whatever context switching is required by saving the context of the machine in the stack with PUSH instructions. The user would then program a VIS (Vector Interrupt Select) instruction in order to branch to the interrupt service routine of the highest priority interrupt enabled and pending at the time of the VIS. Note that this is not necessarily the interrupt that caused the branch to address location 00FF Hex prior to the context switching.

Thus, if an interrupt with a higher rank than the one which caused the interruption becomes active before the decision of which interrupt to service is made by the VIS, then the interrupt with the higher rank will override any lower ones and will be acknowledged. The lower priority interrupt(s) are still pending, however, and will cause another interrupt immediately following the completion of the interrupt service routine associated with the higher priority interrupt just serviced. This lower priority interrupt will occur immediately following the RETI (Return from Interrupt) instruction at the end of the interrupt service routine just completed.

Inside the interrupt service routine, the associated pending bit has to be cleared by software. The RETI (Return from Interrupt) instruction at the end of the interrupt service routine will set the GIE (Global Interrupt Enable) bit, allowing the processor to be interrupted again if another interrupt is active and pending.

The VIS instruction looks at all the active interrupts at the time it is executed and performs an indirect jump to the beginning of the service routine of the one with the highest rank.

The addresses of the different interrupt service routines, called vectors, are chosen by the user and stored in ROM in a table starting at 01E0 (assuming that VIS is located between 00FF and 01DF). The vectors are 15-bit wide and therefore occupy 2 ROM locations.

VIS and the vector table must be located in the same 256-byte block (0y00 to 0yFF) except if VIS is located at the last address of a block. In this case, the table must be in the next block. The vector table cannot be inserted in the first 256-byte block (y $\neq$ 0).

The vector of the maskable interrupt with the lowest rank is located at 0yE0 (Hi-Order byte) and 0yE1 (Lo-Order byte) and so forth in increasing rank number. The vector of the maskable interrupt with the highest rank is located at 0yFA (Hi-Order byte) and 0yFB (Lo-Order byte).

The Software Trap has the highest rank and its vector is located at 0yFE and 0yFF.

If, by accident, a VIS gets executed and no interrupt is active, then the PC (Program Counter) will branch to a vector located at 0yE0–0yE1.

WARNING; A Default VIS interrupt handler routine must be present. As a minimum, this handler should confirm that the GIE bit is cleared (this indicates that the interrupt sequence has been taken), take care of any required housekeeping, restore context and return. Some sort of Warm Restart procedure should be implemented. These events can occur without any error on the part of the system designer or programmer.

Note: There is always the possibility of an interrupt occurring during an instruction which is attempting to reset the GIE bit or any other interrupt enable bit. If this occurs when a single cycle instruction is being used to reset the interrupt enable bit, the interrupt enable bit will be reset but an interrupt may still occur. This is because interrupt processing is started at the same time as the interrupt bit is being reset. To avoid this scenario, the user should always use a two, three, or four cycle instruction to reset interrupt enable bits.

Figure 15 shows the Interrupt block diagram.

Interrupts (Continued)

SOFTWARE TRAP

The Software Trap (ST) is a special kind of non-maskable interrupt which occurs when the INTR instruction (used to acknowledge interrupts) is fetched from ROM and placed inside the instruction register. This may happen when the PC is pointing beyond the available ROM address space or when the stack is over-popped.

When an ST occurs, the user can re-initialize the stack pointer and do a recovery procedure (similar to reset, but not necessarily containing all of the same initialization procedures) before restarting.

The occurrence of an ST is latched into the ST pending bit. The GIE bit is not affected and the ST pending bit **(not accessible by the user)** is used to inhibit other interrupts and to direct the program to the ST service routine with the VIS instruction. The RPND instruction is used to clear the software interrupt pending bit. This pending bit is also cleared on reset.

The ST has the highest rank among all interrupts.

Nothing (except another ST) can interrupt an ST being serviced.

WATCHDOG

The device contains a WATCHDOG and clock monitor. The WATCHDOG is designed to detect the user program getting stuck in infinite loops resulting in loss of program control or "runaway" programs. The Clock Monitor is used to detect the absence of a clock or a very slow clock below a specified rate on the CKI pin.

The WATCHDOG consists of two independent logic blocks: WD UPPER and WD LOWER. WD UPPER establishes the upper limit on the service window and WD LOWER defines the lower limit of the service window.

Servicing the WATCHDOG consists of writing a specific value to a WATCHDOG Service Register named WDSVR which is memory mapped in the RAM. This value is composed of three fields, consisting of a 2-bit Window Select, a 5-bit Key Data field, and the 1-bit Clock Monitor Select field. Table III shows the WDSVR register.

The lower limit of the service window is fixed at 2048 instruction cycles. Bits 7 and 6 of the WDSVR register allow the user to pick an upper limit of the service window.

Table IV shows the four possible combinations of lower and upper limits for the WATCHDOG service window. This flexibility in choosing the WATCHDOG service window prevents any undue burden on the user software.

Bits 5, 4, 3, 2 and 1 of the WDSVR register represent the 5-bit Key Data field. The key data is fixed at 01100. Bit 0 of the WDSVR Register is the Clock Monitor Select bit.

TABLE III. WATCHDOG Service Register (WDSVR)

Window Select		Key Data					Clock Monitor
X	X	0	1	1	0	0	Y
7	6	5	4	3	2	1	0

TABLE IV. WATCHDOG Service Window Select

WDSVR Bit 7	WDSVR Bit 6	Service Window (Lower-Upper Limits)
0	0	2k–8k t_C Cycles
0	1	2k–16k t_C Cycles
1	0	2k–32k t_C Cycles
1	1	2k–64k t_C Cycles

Clock Monitor

The Clock Monitor aboard the device can be selected or deselected under program control. The Clock Monitor is guaranteed not to reject the clock if the instruction cycle clock ($1/t_C$) is greater or equal to 10 kHz. This equates to a clock input rate on CKI of greater or equal to 100 kHz.

WATCHDOG Operation

The WATCHDOG and Clock Monitor are disabled during reset. The device comes out of reset with the WATCHDOG armed, the WATCHDOG Window Select bits (bits 6, 7 of the WDSVR Register) set, and the Clock Monitor bit (bit 0 of the WDSVR Register) enabled. Thus, a Clock Monitor error will occur after coming out of reset, if the instruction cycle clock frequency has not reached a minimum specified value, including the case where the oscillator fails to start.

The WDSVR register can be written to only once after reset and the key data (bits 5 through 1 of the WDSVR Register) must match to be a valid write. This write to the WDSVR register involves two irrevocable choices: (i) the selection of the WATCHDOG service window (ii) enabling or disabling of the Clock Monitor. Hence, the first write to WDSVR Register involves selecting or deselecting the Clock Monitor, select the WATCHDOG service window and match the WATCHDOG key data. Subsequent writes to the WDSVR register will compare the value being written by the user to the WATCHDOG service window value and the key data (bits 7 through 1) in the WDSVR Register. Table V shows the sequence of events that can occur.

The user must service the WATCHDOG at least once before the upper limit of the service window expires. The WATCHDOG may not be serviced more than once in every lower limit of the service window. The user may service the WATCHDOG as many times as wished in the time period between the lower and upper limits of the service window. The first write to the WDSVR Register is also counted as a WATCHDOG service.

The WATCHDOG has an output pin associated with it. This is the WDOUT pin, on pin 1 of the port G. WDOUT is active low. The WDOUT pin is in the high impedance state in the inactive state. Upon triggering the WATCHDOG, the logic will pull the WDOUT (G1) pin low for an additional 16 t_C–32 t_C cycles after the signal level on WDOUT pin goes below the lower Schmitt trigger threshold. After this delay, the device will stop forcing the WDOUT output low.

WATCHDOG Operation (Continued)

The WATCHDOG service window will restart when the WDOUT pin goes high. It is recommended that the user tie the WDOUT pin back to V_{CC} through a resistor in order to pull WDOUT high.

A WATCHDOG service while the WDOUT signal is active will be ignored. The state of the WDOUT pin is not guaranteed on reset, but if it powers up low then the WATCHDOG will time out and WDOUT will enter high impedance state.

The Clock Monitor forces the G1 pin low upon detecting a clock frequency error. The Clock Monitor error will continue until the clock frequency has reached the minimum specified value, after which the G1 output will enter the high impedance TRI-STATE mode following 16 t_c–32 t_c clock cycles. The Clock Monitor generates a continual Clock Monitor error if the oscillator fails to start, or fails to reach the minimum specified frequency. The specification for the Clock Monitor is as follows:

$1/t_c > 10$ kHz—No clock rejection.

$1/t_c < 10$ Hz—Guaranteed clock rejection.

Watchdog and Clock Monitor Summary

The following salient points regarding the WATCHDOG and CLOCK MONITOR should be noted:

- Both the WATCHDOG and CLOCK MONITOR detector circuits are inhibited during RESET.
- Following RESET, the WATCHDOG and CLOCK MONITOR are both enabled, with the WATCHDOG having he maximum service window selected.
- The WATCHDOG service window and CLOCK MONITOR enable/disable option can only be changed once, during the initial WATCHDOG service following RESET.
- The initial WATCHDOG service must match the key data value in the WATCHDOG Service register WDSVR in order to avoid a WATCHDOG error.
- Subsequent WATCHDOG services must match all three data fields in WDSVR in order to avoid WATCHDOG errors.
- The correct key data value cannot be read from the WATCHDOG Service register WDSVR. Any attempt to read this key data value of 01100 from WDSVR will read as key data value of all 0's.
- The WATCHDOG detector circuit is inhibited during both the HALT and IDLE modes.
- The CLOCK MONITOR detector circuit is active during both the HALT and IDLE modes. Consequently, the device inadvertently entering the HALT mode will be detected as a CLOCK MONITOR error (provided that the CLOCK MONITOR enable option has been selected by the program).
- With the single-pin R/C oscillator mask option selected and the CLKDLY bit reset, the WATCHDOG service window will resume following HALT mode from where it left off before entering the HALT mode.
- With the crystal oscillator mask option selected, or with the single-pin R/C oscillator mask option selected and the CLKDLY bit set, the WATCHDOG service window will be set to its selected value from WDSVR following HALT. Consequently, the WATCHDOG should not be serviced for at least 2048 instruction cycles following HALT, but must be serviced within the selected window to avoid a WATCHDOG error.
- The IDLE timer T0 is not initialized with RESET.
- The user can sync in to the IDLE counter cycle with an IDLE counter (T0) interrupt or by monitoring the T0PND flag. The T0PND flag is set whenever the thirteenth bit of the IDLE counter toggles (every 4096 instruction cycles). The user is responsible for resetting the T0PND flag.
- A hardware WATCHDOG service occurs just as the device exits the IDLE mode. Consequently, the WATCHDOG should not be serviced for at least 2048 instruction cycles following IDLE, but must be serviced within the selected window to avoid a WATCHDOG error.
- Following RESET, the initial WATCHDOG service (where the service window and the CLOCK MONITOR enable/disable must be selected) may be programmed anywhere within the maximum service window (65,536 instruction cycles) initialized by RESET. Note that this initial WATCHDOG service may be programmed within the initial 2048 instruction cycles without causing a WATCHDOG error.

Detection of Illegal Conditions

The device can detect various illegal conditions resulting from coding errors, transient noise, power supply voltage drops, runaway programs, etc.

Reading of undefined ROM gets zeros. The opcode for software interrupt is zero. If the program fetches instructions from undefined ROM, this will force a software interrupt, thus signaling that an illegal condition has occurred.

The subroutine stack grows down for each call (jump to subroutine), interrupt, or PUSH, and grows up for each return or POP. The stack pointer is initialized to RAM location 06F Hex during reset. Consequently, if there are more returns than calls, the stack pointer will point to addresses 070 and 071 Hex (which are undefined RAM). Undefined RAM from addresses 070 to 07F (Segment 0), 140 to 17F (Segment 1), and all other segments (i.e., Segments 3 . . . etc.) is read as all 1's, which in turn will cause the program to return to address 7FFF Hex. This is an undefined ROM location and the instruction fetched (all 0's) from this location will generate a software interrupt signaling an illegal condition.

Thus, the chip can detect the following illegal conditions:

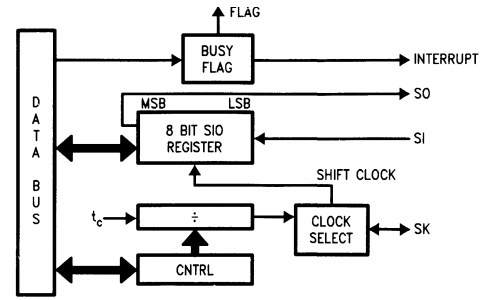
a. Executing from undefined ROM

b. Over "POP"ing the stack by having more returns than calls.

When the software interrupt occurs, the user can re-initialize the stack pointer and do a recovery procedure before restarting (this recovery program is probably similar to that following reset, but might not contain the same program initialization procedures). The recovery program should reset the software interrupt pending bit using the RPND instruction.

MICROWIRE/PLUS

MICROWIRE/PLUS is a serial synchronous communications interface. The MICROWIRE/PLUS capability enables the device to interface with any of National Semiconductor's MICROWIRE peripherals (i.e. A/D converters, display drivers, E^2PROMs etc.) and with other microcontrollers which support the MICROWIRE interface. It consists of an 8-bit serial shift register (SIO) with serial data input (SI), serial data output (SO) and serial shift clock (SK). *Figure 12* shows a block diagram of the MICROWIRE/PLUS logic.



TL/DD/9765–23

FIGURE 16. MICROWIRE/PLUS Block Diagram

The shift clock can be selected from either an internal source or an external source. Operating the MICROWIRE/PLUS arrangement with the internal clock source is called the Master mode of operation. Similarly, operating the MICROWIRE/PLUS arrangement with an external shift clock is called the Slave mode of operation.

The CNTRL register is used to configure and control the MICROWIRE/PLUS mode. To use the MICROWIRE/PLUS, the MSEL bit in the CNTRL register is set to one. In the master mode, the SK clock rate is selected by the two bits, SL0 and SL1, in the CNTRL register. Table VI details the different clock rates that may be selected.

TABLE V. WATCHDOG Service Actions

Key Data	Window Data	Clock Monitor	Action
Match	Match	Match	Valid Service: Restart Service Window
Don't Care	Mismatch	Don't Care	Error: Generate WATCHDOG Output
Mismatch	Don't Care	Don't Care	Error: Generate WATCHDOG Output
Don't Care	Don't Care	Mismatch	Error: Generate WATCHDOG Output

TABLE VI. MICROWIRE/PLUS Master Mode Clock Select

SL1	SL0	SK
0	0	$2 \times t_c$
0	1	$4 \times t_c$
1	x	$8 \times t_c$

Where t_c is the instruction cycle clock

MICROWIRE/PLUS (Continued)

MICROWIRE/PLUS OPERATION

Setting the BUSY bit in the PSW register causes the MICROWIRE/PLUS to start shifting the data. It gets reset when eight data bits have been shifted. The user may reset the BUSY bit by software to allow less than 8 bits to shift. If enabled, an interrupt is generated when eight data bits have been shifted. The device may enter the MICROWIRE/PLUS mode either as a Master or as a Slave. *Figure 13* shows how two devices, microcontrollers and several peripherals may be interconnected using the MICROWIRE/PLUS arrangements.

Warning:

The SIO register should only be loaded when the SK clock is low. Loading the SIO register while the SK clock is high will result in undefined data in the SIO register. SK clock is normally low when not shifting.

Setting the BUSY flag when the input SK clock is high in the MICROWIRE/PLUS slave mode may cause the current SK clock for the SIO shift register to be narrow. For safety, the BUSY flag should only be set when the input SK clock is low.

MICROWIRE/PLUS Master Mode Operation

In the MICROWIRE/PLUS Master mode of operation the shift clock (SK) is generated internally. The MICROWIRE Master always initiates all data exchanges. The MSEL bit in the CNTRL register must be set to enable the SO and SK functions onto the G Port. The SO and SK pins must also be selected as outputs by setting appropriate bits in the Port G configuration register. Table VII summarizes the bit settings required for Master mode of operation.

MICROWIRE/PLUS Slave Mode Operation

In the MICROWIRE/PLUS Slave mode of operation the SK clock is generated by an external source. Setting the MSEL bit in the CNTRL register enables the SO and SK functions onto the G Port. The SK pin must be selected as an input and the SO pin is selected as an output pin by setting and resetting the appropriate bit in the Port G configuration register. Table VII summarizes the settings required to enter the Slave mode of operation.

The user must set the BUSY flag immediately upon entering the Slave mode. This will ensure that all data bits sent by the Master will be shifted properly. After eight clock pulses the BUSY flag will be cleared and the sequence may be repeated.

Alternate SK Phase Operation

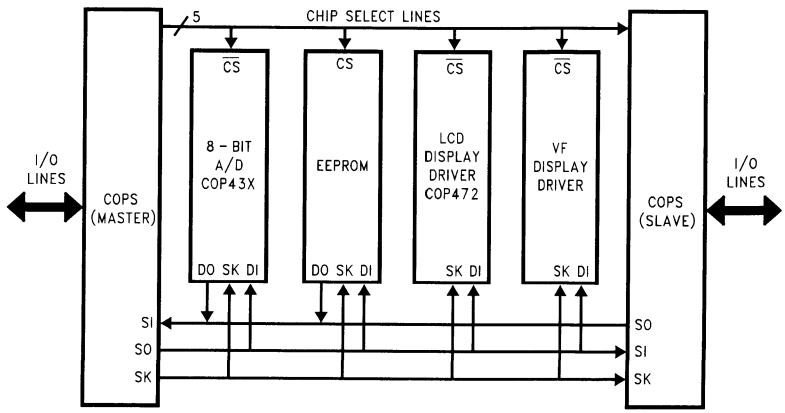
The device allows either the normal SK clock or an alternate phase SK clock to shift data in and out of the SIO register. In both the modes the SK is normally low. In the normal mode data is shifted in on the rising edge of the SK clock and the data is shifted out on the falling edge of the SK clock. The SIO register is shifted on each falling edge of the SK clock. In the alternate SK phase operation, data is shifted in on the falling edge of the SK clock and shifted out on the rising edge of the SK clock.

A control flag, SKSEL, allows either the normal SK clock or the alternate SK clock to be selected. Resetting SKSEL causes the MICROWIRE/PLUS logic to be clocked from the normal SK signal. Setting the SKSEL flag selects the alternate SK clock. The SKSEL is mapped into the G6 configuration bit. The SKSEL flag will power up in the reset condition, selecting the normal SK signal.

TABLE VII

This table assumes that the control flag MSEL is set.

G4 (SO) Config. Bit	G5 (SK) Config. Bit	G4 Fun.	G5 Fun.	Operation
1	1	SO	Int. SK	MICROWIRE/PLUS Master
0	1	TRI-STATE	Int. SK	MICROWIRE/PLUS Master
1	0	SO	Ext. SK	MICROWIRE/PLUS Slave
0	0	TRI-STATE	Ext. SK	MICROWIRE/PLUS Slave



TL/DD/9765-24

FIGURE 17. MICROWIRE/PLUS Application

Memory Map

All RAM, ports and registers (except A and PC) are mapped into data memory address space.

Address S/ADD REG	Contents
0000 to 006F	On-Chip RAM bytes (112 bytes)
0070 to 007F	Unused RAM Address Space (Reads As All Ones)
xx80 to xxAF	Unused RAM Address Space (Reads Undefined Data)
xxB0	Timer T3 Lower Byte
XXB1	Timer T3 Upper Byte
xxB2	Timer T3 Autoload Register T3RA Lower Byte
xxB3	Timer T3 Autoload Register T3RA Upper Byte
xxB4	Timer T3 Autoload Register T3RB Lower Byte
xxB5	Timer T3 Autoload Register T3RB Upper Byte
xxB6	Timer T3 Control Register
xxB7	Comparator Select Register (CMPSL)
xxB8	UART Transmit Buffer (TBUF)
xxB9	UART Receive Buffer (RBUF)
xxBA	UART Control and Status Register (ENU)
xxBB	UART Receive Control and Status Register (ENUR)
xxBC	UART Interrupt and Clock Source Register (ENUI)
xxBD	UART Baud Register (BAUD)
xxBE	UART Prescale Select Register (PSR)
xxBF	Reserved for UART
xxC0	Timer T2 Lower Byte
xxC1	Timer T2 Upper Byte
xxC2	Timer T2 Autoload Register T2RA Lower Byte
xxC3	Timer T2 Autoload Register T2RA Upper Byte
xxC4	Timer T2 Autoload Register T2RB Lower Byte
xxC5	Timer T2 Autoload Register T2RB Upper Byte
xxC6	Timer T2 Control Register
xxC7	WATCHDOG Service Register (Reg:WDSVR)
xxC8	MIWU Edge Select Register (Reg:WKEDG)
xxC9	MIWU Enable Register (Reg:WKEN)
xxCA	MIWU Pending Register (Reg:WKPND)
xxCB	Reserved
xxCC	Reserved
xxCD to xxCF	Reserved

Address S/ADD REG	Contents
xxD0	Port L Data Register
xxD1	Port L Configuration Register
xxD2	Port L Input Pins (Read Only)
xxD3	Reserved for Port L
xxD4	Port G Data Register
xxD5	Port G Configuration Register
xxD6	Port G Input Pins (Read Only)
xxD7	Port I Input Pins (Read Only)
xxD8	Port C Data Register
xxD9	Port C Configuration Register
xxDA	Port C Input Pins (Read Only)
xxDB	Reserved for Port C
xxDC	Port D
xxDD to DF	Reserved for Port D
xxE0 to xxE5	Reserved for EE Control Registers
xxE6	Timer T1 Autoload Register T1RB Lower Byte
xxE7	Timer T1 Autoload Register T1RB Upper Byte
xxE8	ICNTRL Register
xxE9	MICROWIRE/PLUS Shift Register
xxEA	Timer T1 Lower Byte
xxEB	Timer T1 Upper Byte
xxEC	Timer T1 Autoload Register T1RA Lower Byte
xxED	Timer T1 Autoload Register T1RA Upper Byte
xxEE	CNTRL Control Register
xxEF	PSW Register
xxF0 to FB	On-Chip RAM Mapped as Registers
xxFC	X Register
xxFD	SP Register
xxFE	B Register
xxFF	S Register
0100–013F	On-Chip 64 RAM Bytes

Reading memory locations 0070H–007FH (Segment 0) will return all ones. Reading unused memory locations 0080H–00AFH (Segment 0) will return undefined data. Reading unused memory locations 0140–017F (Segment 1) will return all ones. Reading memory locations from other Segments (i.e., Segment 2, Segment 3, ... etc.) will return all ones.

Addressing Modes

There are ten addressing modes, six for operand addressing and four for transfer of control.

OPERAND ADDRESSING MODES

Register Indirect

This is the "normal" addressing mode. The operand is the data memory addressed by the B pointer or X pointer.

Register Indirect (with auto post increment or decrement of pointer)

This addressing mode is used with the LD and X instructions. The operand is the data memory addressed by the B pointer or X pointer. This is a register indirect mode that automatically post increments or decrements the B or X register after executing the instruction.

Direct

The instruction contains an 8-bit address field that directly points to the data memory for the operand.

Immediate

The instruction contains an 8-bit immediate field as the operand.

Short Immediate

This addressing mode is used with the Load B Immediate instruction. The instruction contains a 4-bit immediate field as the operand.

Indirect

This addressing mode is used with the LAID instruction. The contents of the accumulator are used as a partial address (lower 8 bits of PC) for accessing a data operand from the program memory.

TRANSFER OF CONTROL ADDRESSING MODES

Relative

This mode is used for the JP instruction, with the instruction field being added to the program counter to get the new program location. JP has a range from −31 to +32 to allow a 1-byte relative jump (JP + 1 is implemented by a NOP instruction). There are no "pages" when using JP, since all 15 bits of PC are used.

Absolute

This mode is used with the JMP and JSR instructions, with the instruction field of 12 bits replacing the lower 12 bits of the program counter (PC). This allows jumping to any location in the current 4k program memory segment.

Absolute Long

This mode is used with the JMPL and JSRL instructions, with the instruction field of 15 bits replacing the entire 15 bits of the program counter (PC). This allows jumping to any location in the current 4k program memory space.

Indirect

This mode is used with the JID instruction. The contents of the accumulator are used as a partial address (lower 8 bits of PC) for accessing a location in the program memory. The contents of this program memory location serve as a partial address (lower 8 bits of PC) for the jump to the next instruction.

Note: The VIS is a special case of the Indirect Transfer of Control addressing mode, where the double byte vector associated with the interrupt is transferred from adjacent addresses in the program memory into the program counter (PC) in order to jump to the associated interrupt service routine.

Instruction Set

Register and Symbol Definition

Registers	
A	8-Bit Accumulator Register
B	8-Bit Address Register
X	8-Bit Address Register
SP	8-Bit Stack Pointer Register
PC	15-Bit Program Counter Register
PU	Upper 7 Bits of PC
PL	Lower 8 Bits of PC
C	1 Bit of PSW Register for Carry
HC	1 Bit of PSW Register for Half Carry
GIE	1 Bit of PSW Register for Global Interrupt Enable
VU	Interrupt Vector Upper Byte
VL	Interrupt Vector Lower Byte

Symbols	
[B]	Memory Indirectly Addressed by B Register
[X]	Memory Indirectly Addressed by X Register
MD	Direct Addressed Memory
Mem	Direct Addressed Memory or [B]
Meml	Direct Addressed Memory or [B] or Immediate Data
Imm	8-Bit Immediate Data
Reg	Register Memory: Addresses F0 to FF (Includes B, X and SP)
Bit	Bit Number (0 to 7)
←	Loaded with
⟷	Exchanged with

5

Instruction Set (Continued)

INSTRUCTION SET

ADD	A,Meml	ADD	A ← A + Meml
ADC	A,Meml	ADD with Carry	A ← A + Meml + C, C ← Carry HC ← Half Carry
SUBC	A,Meml	Subtract with Carry	A ← A − $\overline{\text{Meml}}$ + C, C ← Carry HC ← Half Carry
AND	A,Meml	Logical AND	A ← A and Meml
ANDSZ	A,Imm	Logical AND Immed., Skip if Zero	Skip next if (A and Imm) = 0
OR	A,Meml	Logical OR	A ← A or Meml
XOR	A,Meml	Logical EXclusive OR	A ← A xor Meml
IFEQ	MD,Imm	IF EQual	Compare MD and Imm, Do next if MD = Imm
IFEQ	A,Meml	IF EQual	Compare A and Meml, Do next if A = Meml
IFNE	A,Meml	IF Not Equal	Compare A and Meml, Do next if A ≠ Meml
IFGT	A,Meml	IF Greater Than	Compare A and Meml, Do next if A > Meml
IFBNE	#	If B Not Equal	Do next if lower 4 bits of B ≠ Imm
DRSZ	Reg	Decrement Reg., Skip if Zero	Reg ← Reg − 1, Skip if Reg = 0
SBIT	#,Mem	Set BIT	1 to bit, Mem (bit = 0 to 7 immediate)
RBIT	#,Mem	Reset BIT	0 to bit, Mem
IFBIT	#,Mem	IF BIT	If bit in A or Mem is true do next instruction
RPND		Reset PeNDing Flag	Reset Software Interrupt Pending Flag
X	A,Mem	EXchange A with Memory	A ⟷ Mem
X	A,[X]	EXchange A with Memory [X]	A ⟷ [X]
LD	A,Meml	LoaD A with Memory	A ← Meml
LD	A,[X]	LoaD A with Memory [X]	A ← [X]
LD	B,Imm	LoaD B with Immed.	B ← Imm
LD	Mem,Imm	LoaD Memory Immed	Mem ← Imm
LD	Reg,Imm	LoaD Register Memory Immed.	Reg ← Imm
X	A, [B ±]	EXchange A with Memory [B]	A ⟷ [B], (B ← B ±1)
X	A, [X ±]	EXchange A with Memory [X]	A ⟷ [X], (X ← ±1)
LD	A, [B±]	LoaD A with Memory [B]	A ← [B], (B ← B ±1)
LD	A, [X±]	LoaD A with Memory [X]	A ← [X], (X ← X±1)
LD	[B±],Imm	LoaD Memory [B] Immed.	[B] ← Imm, (B ← B±1)
CLR	A	CLeaR A	A ← 0
INC	A	INCrement A	A ← A + 1
DEC	A	DECrementA	A ← A − 1
LAID		Load A InDirect from ROM	A ← ROM (PU,A)
DCOR	A	Decimal CORrect A	A ← BCD correction of A (follows ADC, SUBC)
RRC	A	Rotate A Right thru C	C → A7 → . . . → A0 → C
RLC	A	Rotate A Left thru C	C ← A7 ← . . . ← A0 ← C
SWAP	A	SWAP nibbles of A	A7 . . . A4 ⟷ A3 . . . A0
SC		Set C	C ← 1, HC ← 1
RC		Reset C	C ← 0, HC ← 0
IFC		IF C	IF C is true, do next instruction
IFNC		IF Not C	If C is not true, do next instruction
POP	A	POP the stack into A	SP ← SP + 1, A ← [SP]
PUSH	A	PUSH A onto the stack	[SP] ← A, SP ← SP − 1
VIS		Vector to Interrupt Service Routine	PU ← [VU], PL ← [VL]
JMPL	Addr.	Jump absolute Long	PC ← ii (ii = 15 bits, 0 to 32k)
JMP	Addr.	Jump absolute	PC9 . . . 0 ← i (i = 12 bits)
JP	Disp.	Jump relative short	PC ← PC + r (r is −31 to +32, except 1)
JSRL	Addr.	Jump SubRoutine Long	[SP] ← PL, [SP−1] ← PU,SP−2, PC ← ii
JSR	Addr	Jump SubRoutine	[SP] ← PL, [SP−1] ← PU,SP−2, PC9 . . . 0 ← i
JID		Jump InDirect	PL ← ROM (PU,A)
RET		RETurn from subroutine	SP + 2, PL ← [SP], PU ← [SP−1]
RETSK		RETurn and SKip	SP + 2, PL ← [SP],PU ← [SP−1]
RETI		RETurn from Interrupt	SP + 2, PL ← [SP],PU ← [SP−1],GIE ← 1
INTR		Generate an Interrupt	[SP] ← PL, [SP−1] ← PU, SP−2, PC ← 0FF
NOP		No OPeration	PC ← PC + 1

Instruction Execution Time

Most instructions are single byte (with immediate addressing mode instructions taking two bytes).

Most single byte instructions take one cycle time to execute.

Skipped instructions require x number of cycles to be skipped, where x equals the number of bytes in the skipped instruction opcode.

See the BYTES and CYCLES per INSTRUCTION table for details.

Bytes and Cycles per Instruction

The following table shows the number of bytes and cycles for each instruction in the format of byte/cycle.

Arithmetic and Logic Instructions

	[B]	Direct	Immed.
ADD	1/1	3/4	2/2
ADC	1/1	3/4	2/2
SUBC	1/1	3/4	2/2
AND	1/1	3/4	2/2
OR	1/1	3/4	2/2
XOR	1/1	3/4	2/2
IFEQ	1/1	3/4	2/2
IFNE	1/1	3/4	2/2
IFGT	1/1	3/4	2/2
IFBNE	1/1		
DRSZ		1/3	
SBIT	1/1	3/4	
RBIT	1/1	3/4	
IFBIT	1/1	3/4	

RPND	1/1

Instructions Using A & C

CLRA	1/1
INCA	1/1
DECA	1/1
LAID	1/3
DCOR	1/1
RRCA	1/1
RLCA	1/1
SWAPA	1/1
SC	1/1
RC	1/1
IFC	1/1
IFNC	1/1
PUSHA	1/3
POPA	1/3
ANDSZ	2/2

Transfer of Control Instructions

JMPL	3/4
JMP	2/3
JP	1/3
JSRL	3/5
JSR	2/5
JID	1/3
VIS	1/5
RET	1/5
RETSK	1/5
RETI	1/5
INTR	1/7
NOP	1/1

Memory Transfer Instructions

	Register Indirect		Direct	Immed.	Register Indirect Auto Incr. & Decr.		
	[B]	[X]			[B+, B−]	[X+, X−]	
X A,*	1/1	1/3	2/3		1/2	1/3	
LD A,*	1/1	1/3	2/3	2/2	1/2	1/3	
LD B, Imm				1/1			(IF B < 16)
LD B, Imm				2/2			(IF B > 15)
LD Mem, Imm	2/2		3/3		2/2		
LD Reg, Imm			2/3				
IFEQ MD, Imm			3/3				

* = > Memory location addressed by B or X or directly.

Opcode Table

Upper Nibble Along X-Axis

Lower Nibble Along Y-Axis

F	E	D	C	B	A	9	8	
JP −15	JP −31	LD 0F0, # i	DRSZ 0F0	RRCA	RC	ADC A,#i	ADC A,[B]	0
JP −14	JP −30	LD 0F1, # i	DRSZ 0F1	*	SC	SUBC A, #i	SUB A,[B]	1
JP −13	JP −29	LD 0F2, # i	DRSZ 0F2	X A, [X+]	X A,[B+]	IFEQ A,#i	IFEQ A,[B]	2
JP −12	JP −28	LD 0F3, # i	DRSZ 0F3	X A, [X−]	X A,[B−]	IFGT A,#i	IFGT A,[B]	3
JP −11	JP −27	LD 0F4, # i	DRSZ 0F4	VIS	LAID	ADD A,#i	ADD A,[B]	4
JP −10	JP −26	LD 0F5, # i	DRSZ 0F5	RPND	JID	AND A,#i	AND A,[B]	5
JP −9	JP −25	LD 0F6, # i	DRSZ 0F6	X A,[X]	X A,[B]	XOR A,#i	XOR A,[B]	6
JP −8	JP −24	LD 0F7, # i	DRSZ 0F7	*	*	OR A,#i	OR A,[B]	7
JP −7	JP −23	LD 0F8, # i	DRSZ 0F8	NOP	RLCA	LD A,#i	IFC	8
JP −6	JP −22	LD 0F9, # i	DRSZ 0F9	IFNE A,[B]	IFEQ Md,#i	IFNE A,#i	IFNC	9
JP −5	JP −21	LD 0FA, # i	DRSZ 0FA	LD A,[X+]	LD A,[B+]	LD [B+],#i	INCA	A
JP −4	JP −20	LD 0FB, # i	DRSZ 0FB	LD A,[X−]	LD A,[B−]	LD [B−],#i	DECA	B
JP −3	JP −19	LD 0FC, # i	DRSZ 0FC	LD Md,#i	JMPL	X A,Md	POPA	C
JP −2	JP −18	LD 0FD, # i	DRSZ 0FD	DIR	JSRL	LD A,Md	RETSK	D
JP −1	JP −17	LD 0FE, # i	DRSZ 0FE	LD A,[X]	LD A,[B]	LD [B],#i	RET	E
JP −0	JP −16	LD 0FF, # i	DRSZ 0FF	*	*	LD B,#i	RETI	F

Opcode Table (Continued)

Upper Nibble Along X-Axis

Lower Nibble Along Y-Axis

7	6	5	4	3	2	1	0	
IFBIT 0,[B]	ANDSZ A, #i	LD B,#0F	IFBNE 0	JSR x000-x0FF	JMP x000-x0FF	JP +17	INTR	0
IFBIT 1,[B]	*	LD B,#0E	IFBNE 1	JSR x100-x1FF	JMP x100-x1FF	JP +18	JP + 2	1
IFBIT 2,[B]	*	LD B,#0D	IFBNE 2	JSR x200-x2FF	JMP x200-x2FF	JP +19	JP + 3	2
IFBIT 3,[B]	*	LD B,#0C	IFBNE 3	JSR x300-x3FF	JMP x300-x3FF	JP +20	JP + 4	3
IFBIT 4,[B]	CLRA	LD B,#0B	IFBNE 4	JSR x400-x4FF	JMP x400-x4FF	JP +21	JP + 5	4
IFBIT 5,[B]	SWAPA	LD B,#0A	IFBNE 5	JSR x500-x5FF	JMP x500-x5FF	JP +22	JP + 6	5
IFBIT 6,[B]	DCORA	LD B,#09	IFBNE 6	JSR x600-x6FF	JMP x600-x6FF	JP +23	JP + 7	6
IFBIT 7,[B]	PUSHA	LD B,#08	IFBNE 7	JSR x700-x7FF	JMP x700-x7FF	JP +24	JP + 8	7
SBIT 0,[B]	RBIT 0,[B]	LD B,#07	IFBNE 8	JSR x800-x8FF	JMP x800-x8FF	JP +25	JP + 9	8
SBIT 1,[B]	RBIT 1,[B]	LD B,#06	IFBNE 9	JSR x900-x9FF	JMP x900-x9FF	JP +26	JP + 10	9
SBIT 2,[B]	RBIT 2,[B]	LD B,#05	IFBNE 0A	JSR xA00-xAFF	JMP xA00-xAFF	JP +27	JP + 11	A
SBIT 3,[B]	RBIT 3,[B]	LD B,#04	IFBNE 0B	JSR xB00-xBFF	JMP xB00-xBFF	JP +28	JP + 12	B
SBIT 4,[B]	RBIT 4,[B]	LD B,#03	IFBNE 0C	JSR xC00-xCFF	JMP xC00-xCFF	JP +29	JP + 13	C
SBIT 5,[B]	RBIT 5,[B]	LD B,#02	IFBNE 0D	JSR xD00-xDFF	JMP xD00-xDFF	JP +30	JP + 14	D
SBIT 6,[B]	RBIT 6,[B]	LD B,#01	IFBNE 0E	JSR xE00-xEFF	JMP xE00-xEFF	JP +31	JP + 15	E
SBIT 7,[B]	RBIT 7,[B]	LD B,#00	IFBNE 0F	JSR xF00-xFFF	JMP xF00-xFFF	JP +32	JP + 16	F

Where,

i is the immediate data

Md is a directly addressed memory location

* is an unused opcode

Note: The opcode 60 Hex is also the opcode for IFBIT #i,A

Mask Options

The mask programmable options are shown below. The options are programmed at the same time as the ROM pattern submission.

```
OPTION 1: CLOCK CONFIGURATION
  = 1     Crystal Oscillator (CKI/l0)
              G7 (CKO) is clock generator
              output to crystal/resonator
              CKI is the clock input
  = 2     Single-pin RC controlled
              oscillator (CKI/l0)
              G7 is available as a HALT
              restart and/or general purpose
              input

OPTION 2: HALT
  = 1     Enable HALT mode
  = 2     Disable HALT mode
OPTION 3: BONDING OPTIONS
  = 1     44-Pin PLCC
  = 2     40-Pin DIP
  = 3     N/A
  = 4     28-Pin DIP
  = 5     28-Pin SO
```

5

Development Support

SUMMARY

- iceMASTER™: IM-COP8/400—Full feature in-circuit emulation for all COP8 products. A full set of COP8 Basic and Feature Family device and package specific probes are available.
- COP8 Debug Module: Moderate cost in-circuit emulation and development programming unit.
- COP8 Evaluation and Programming Unit: EPU-COP888GG—low cost in-circuit simulation and development programming unit.
- Assembler: COP8-DEV-IBMA. A DOS installable cross development Assembler, Linker, Librarian and Utility Software Development Tool Kit.
- C Compiler: COP8C. A DOS installable cross development Software Tool Kit.
- OTP/EPROM Programmer Support: Covering needs from engineering prototype, pilot production to full production environments.

IceMASTER (IM) IN-CIRCUIT EMULATION

The iceMASTER IM-COP8/400 is a full feature, PC based, in-circuit emulation tool development and marketed by MetaLink Corporation to support the whole COP8 family of products. National is a resale vendor for these products.

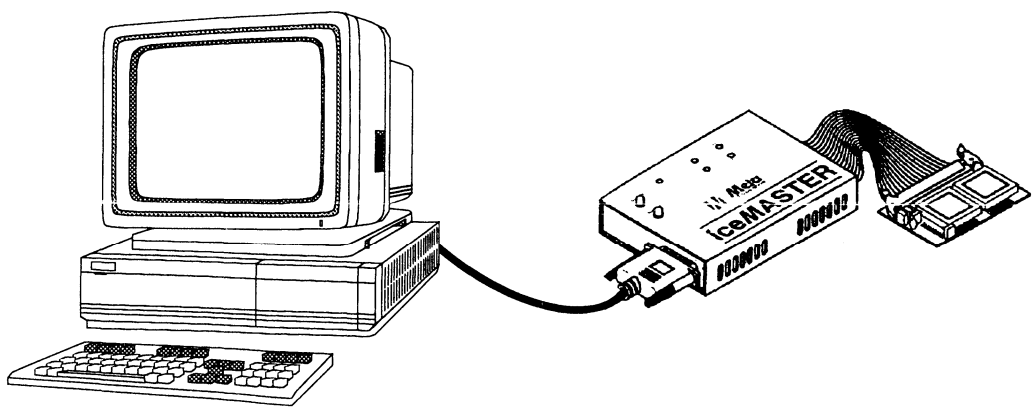
See *Figure 18* for configuration.

The iceMASTER IM-COP8/400 with its device specific COP8 Probe provides a rich feature set for developing, testing and maintaining product:

- Real-time in-circuit emulation; full 2.4V–5.5V operation range, full DC-10 MHz clock. Chip options are programmable or jumper selectable.
- Direct connection to application board by package compatible socket or surface assembly.
- Full 32 kbyte of loadable programming space that overlays (replaces) the on-chip ROM or EPROM. On-chip RAM and I/O blocks are used directly or recreated on the probe as necessary.
- Full 4k frame synchronous trace memory. Address, instruction, and 8 unspecified, circuit connectable trace lines. Display can be HLL source (e.g., C source), assembly or mixed.
- A full 64k hardware configurable break, trace on, trace off control, and pass count increment events.
- Tool set integrated interactive symbolic debugger—supports both assembler (COFF) and C Compiler (.COD) linked object formats.
- Real time performance profiling analysis; selectable bucket defintion.
- Watch windows, content updated automatically at each execution break.
- Instruction by instruction memory/register changes displayed on source window when in single step operation.
- Single base unit and debugger software reconfigurable to support the entire COP8 family; only the probe personality needs to change. Debugger software is processor customized, and reconfigured from a master model file.
- Processor specific symbolic display of registers and bit level assignments, configured from master model file.
- Halt/Idle mode notification.
- On-Line HELP customized to specific processor using master model file.
- Includes a copy of COP8-DEV-IBMA assembler and linker SDK.

IM Order Information

Base Unit	
IM-COP8/400-1	iceMASTER Base Unit, 110V Power Supply
IM-COP8/400-2	iceMASTER Base Unit, 220V Power Supply
iceMASTER Probe	
MHW-884EG28DWPC MHW-888EG40DWPC MHW-888EG44PWPC	28 DIP 40 DIP 44 PLCC
Adapter for SO Package	
MHW-SOIC28	28 SO



TL/DD/9765–26

FIGURE 18. COP8 iceMASTER Environment

Development Support (Continued)

iceMASTER DEBUG MODULE (DM)

The iceMASTER Debug Module is a PC based, combination in-circuit emulation tool and COP8 based OTP/EPROM programming tool developed and marketed by MetaLink Corporation to support the whole COP8 family of products. National is a resale vendor for these products.

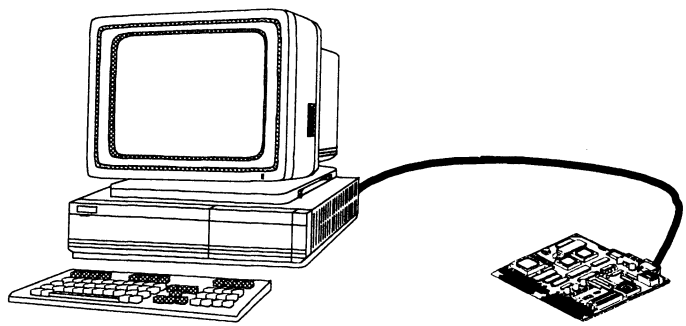
See *Figure 19* for configuration.

The iceMASTER Debug Module is a moderate cost development tool. It has the capability of in-circuit emulation for a specific COP8 microcontroller and in addition serves as a programming tool for COP8 OTP and EPROM product families. Summary of features is as follows:

- Real-time in-circuit emulation; full operating voltage range operation, full DC-10 MHz clock.
- All processor I/O pins can be cabled to an application development board with package compatible cable to socket and surface mount assembly.
- Full 32 kbyte of loadable programming space that overlays (replaces) the on-chip ROM or EPROM. On-chip RAM and I/O blocks are used directly or recreated as necessary.
- 100 frames of synchronous trace memory. The display can be HLL source (C source), assembly or mixed. The most recent history prior to a break is available in the trace memory.
- Configured break points; uses INTR instruction which is modestly intrusive.
- Software-only supported features are selectable.
- Tool set integrated interactive symbolic debugger—supports both assembler (COFF) and C Compiler (.COD) SDK linked object formats.
- Instruction by instruction memory/register changes displayed when in single step operation.
- Debugger software is processor customized, and reconfigured from a master model file.
- Processor specific symbolic display of registers and bit level assignments, configured from master model file.
- Halt/Idle mode notification.
- Programming menu supports full product line of programmable OTP and EPROM COP8 products. Program data is taken directly from the overlay RAM.
- Programming of 44 PLCC and 68 PLCC parts requires external programming adapters.
- Includes wallmount power supply.
- On-board V_{PP} generator from 5V input or connection to external supply supported. Requires V_{PP} level adjustment per the family programming specification (correct level is provided on an on-screen pop-down display).
- On-line HELP customized to specific processor using master model file.
- Includes a copy of COP8-DEV-IBMA assembler and linker SDK.

DM Order Information

Debug Module Unit	
COP8-DM/888GG	
Cable Adapters, Kits for COP888	
DM-COP8/28D	28 DIP
DM-COP8/40D	40 DIP
DM-COP8/44P	44 PLCC
Adapter for SO Package	
MHW-SOIC28	28 SO



TL/DD/9765-27

FIGURE 19. COP8-DM Environment

Development Support (Continued)

iceMASTER EVALUATION PROGRAMMING UNIT (EPU)

The iceMASTER EPU-COP888GG is a PC based, in-circuit simulation tool to support the feature family COP8 products.

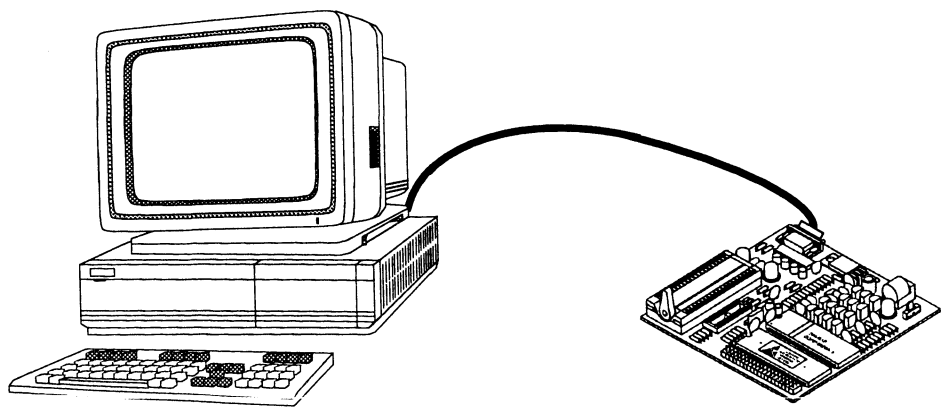
See Figure 20 for configuration.

The simulation capability is a very low cost means of evaluating the general COP8 architecture. In addition, the EPU has programming capability, with added adapters, for programming the whole COP8 product family of OTP and EPROM products. The product includes the following features:

- Non-real-time in-circuit simulation. Program overlay memory is PC resident; instructions are downloaded over RS-232 as executed. Approximate performance is 20 kHz.
- Includes a 40-pin DIP cable adapter. Other target packages are not supported. All processor I/O pins are cabled to the application development environment.
- Full 32 kbyte of loadable programming space that overlays (replaces) the on-chip ROM or EPROM. On-chip RAM and I/O blocks are used directly or recreated as necessary.
- On-chip timer and watch-dog execution are not well synchronized to the instruction simulation.
- 100 frames of synchronous trace memory. The display can be HLL source (e.g., C source), assembly or mixed. The most recent history prior to a break is available in the trace memory.
- Up to eight software configured break points; uses INTR instruction which is modestly intrusive.
- Common look-feel debugger software across all MetaLink products—only supported features are selectable.
- Tool set integrated interactive symbolic debugger—supports both assembler (COFF) and C Compiler (.COD) SDK linked object formats.
- Instruction by instruction memory/register changes displayed when in single step operation.
- Processor specific symbolic display of registers and bit level assignments, configured from master model file.
- Halt/Idle mode notification. Restart requires special handling.
- Programming menu supports full product line of programmable OTP and EPROM COP8 products. Only a 40 ZIF socket is available on the EPU unit. Adapters are available for other part package configurations.
- Integral wall mount power supply provides 5V and develops the required V_{PP} to program parts.
- Includes a copy of COP8-DEV-IBMA assembler, linker SDK.

EPU Order Information

Evaluation Programming Unit	
EPU-COP888GG	Evaluation Programming Unit with debugger and programmer control software with 40 ZIF programming socket.
General Programming Adapters	
COP8-PGMA-DS	28 and 20 DIP and SOIC adapter.
COP8-PGMA-DS44P	28 and 20 DIP and SOIC plus 44 PLCC adapter.



TL/DD/9765–28

FIGURE 20. EPU-COP8 Tool Environment

Development Support (Continued)

COP8 ASSEMBLER/LINKER SOFTWARE DEVELOPMENT TOOL KIT

National Semiconductor offers a relocatable COP8 macro cross assembler, linker, librarian and utility software development tool kit. Features are summarized as follows:

- Basic and Feature Family instruction set by "device" type.
- Nested macro capability.
- Extensive set of assembler directives.
- Supported on PC/DOS platform.
- Generates National standard COFF output files.
- Integrated Linker and Librarian.
- Integrated utilities to generate ROM code file outputs.
- DUMPCOFF utility.

This product is integrated as a part of MetaLink tools as a development kit, fully supported by the MetaLink debugger. It may be ordered separately or it is bundled with the MetaLink products at no additional cost.

Order Information

Assembler SDK	
COP8-DEV-IBMA	Assembler SDK on installable 3.5" PC/DOS Floppy Disk Drive format. Periodic upgrades and most recent version is available on National's BBS and Internet.

COP8 C COMPILER

A C Compiler is developed and marketed by Byte Craft Limited. The COP8C compiler is a fully integrated development tool specifically designed to support the compact embedded configuration of the COP8 family of products.

Features are summarized as follows:

- ANSI C with some restrictions and extensions that optimize development for the COP8 embedded application.
- BITS data type extension. Register declaration #pragma with direct bit level definitions.
- C language support for interrupt routines.
- Expert system, rule based code generation and optimization.
- Performs consistency checks against the architectural definitions of the target COP8 device.
- Generates program memory code.
- Supports linking of compiled object or COP8 assembled object formats.
- Global optimization of linked code.
- Symbolic debug load format fully source level supported by the MetaLink debugger.

SINGLE CHIP OTP/EMULATOR SUPPORT

The COP8 family is supported by single chip OTP emulators. For detailed information refer to the emulator specific datasheet and the emulator selection table below:

Approved List

Manufacturer	North America	Europe	Asia
BP Microsystems	(800) 225-2102 (713) 688-4600 Fax: (713) 688-0920	+49-8152-4183 +49-8856-932616	+852-234-16611 +852-2710-8121
Data I/O	(800) 426-1045 (206) 881-6444 Fax: (206) 882-1043	+44-0734-440011	Call North America
HI-LO	(510) 623-8860	Call Asia	+886-2-764-0215 Fax: +886-2-756-6403
ICE Technology	(800) 624-8949 (919) 430-7915	+44-1226-767404 Fax: 0-1226-370-434	
MetaLink	(800) 638-2423 (602) 926-0797 Fax: (602) 693-0681	+49-80 9156 96-0 Fax: +49-80 9123 86	+852-737-1800
Systems General	(408) 263-6667	+41-1-9450300	+886-2-917-3005 Fax: +886-2-911-1283
Needhams	(916) 924-8037 Fax: (916) 924-8065		

5

Development Support (Continued)

OTP Emulator Ordering Information

Device Number	Clock Option	Package	Emulates
COP87L84EGN-XE	Crystal	28 DIP	COP884CG
COP87L84EGM-XE	Crystal	28 SO	COP884CG
COP87L88EGN-XE	Crystal	40 DIP	COP888CG
COP87L88EGV-XE	Crystal	44 PLCC	COP888CG

INDUSTRY WIDE OTP/EPROM PROGRAMMING SUPPORT

Programming support, in addition to the MetaLink development tools, is provided by a full range of independent approved vendors to meet the needs from the engineering laboratory to full production.

AVAILABLE LITERATURE

For more information, please see the COP8 Basic Family User's Manual, Literature Number 620895, COP8 Feature Family User's Manual, Literature Number 620897 and National's Family of 8-bit Microcontrollers COP8 Selection Guide, Literature Number 630009.

DIAL-A-HELPER SERVICE

Dial-A-Helper is a service provided by the Microcontroller Applications group. The Dial-A-Helper is an Electronic Information System that may be accessed as a Bulletin Board System (BBS) via data modem, as an FTP site on the Internet via standard FTP client application or as an FTP site on the Internet using a standard Internet browser such as Netscape or Mosaic.

The Dial-A-Helper system provides access to an automated information storage and retrieval system. The system capabilities include a MESSAGE SECTION (electronic mail, when accessed as a BBS) for communication to and from the Microcontroller Applications Group and a FILE SECTION which consists of several file areas where valuable application software and utilities could be found.

DIAL-A-HELPER BBS via a Standard Modem

Modem: CANADA/U.S.: (800) NSC-MICRO
(800) 672-6427
EUROPE: (+49) 0-8141-351332
Baud: 14.4k
Set-up: Length: 8-Bit
Parity: None
Stop Bit: 1
Operation: 24 Hours, 7 Days

DIAL-A-HELPER via FTP

ftp nscmicro.nsc.com
user: anonymous
password: username@yourhost.site.domain

DIAL-A-HELPER via a WorldWide Web Browser

ftp://nscmicro.nsc.com

National Semiconductor on the WorldWide Web

See us on the WorldWide Web at: http://www.national.com

CUSTOMER RESPONSE CENTER

Complete product information and technical support is available from National's customer response centers.

CANADA/U.S.:	Tel:	(800) 272-9959
	email:	support @tevm2.nsc.com
EUROPE:	email:	europe.support@nsc.com
	Deutsch Tel:	+49 (0) 180-530 85 85
	English Tel:	+49 (0) 180-532 78 32
	Français Tel:	+49 (0) 180-532 93 58
	Italiano Tel:	+49 (0) 180-534 16 80
JAPAN:	Tel:	+81-043-299-2309
S.E. ASIA:	Beijing Tel:	(+86) 10-6856-8601
	Shanghai Tel:	(+86) 21
	Hong Kong Tel:	(+852) 2737-1600
	Korea Tel:	(+82) 2-3771-6909
	Malaysia Tel:	(+60-4) 644-9061
	Singapore Tel:	(+65) 255-2226
	Taiwan Tel:	+886-2-521-3288
AUSTRALIA:	Tel:	(+61) 3-9558-9999
INDIA:	Tel:	(+91) 80-559-9467

National Semiconductor

COP688CS/COP684CS/COP888CS/COP884CS/ COP988CS/COP984CS 8-Bit Microcontroller with UART and One Multi-Function Timer

General Description

The COP888 family of microcontrollers uses an 8-bit single chip core architecture fabricated with National Semiconductor's M²CMOS™ process technology. The COP888CS is a member of this expandable 8-bit core processor family of microcontrollers. (Continued)

Key Features

- Full duplex UART
- One 16-bit timer, with two 16-bit registers supporting:
  - — Processor Independent PWM mode
  - — External Event counter mode
  - — Input Capture mode
- 4 kbytes of on-chip ROM
- 192 bytes on-board RAM

Additional Peripheral Features

- Idle Timer
- Multi-Input Wakeup (MIWU) with optional interrupts (8)
- One analog comparator
- WATCHDOG™ and Clock Monitor logic
- MICROWIRE/PLUS™ serial I/O

I/O Features

- Memory mapped I/O
- Software selectable I/O options (TRI-STATE® Output, Push-Pull Output, Weak Pull Up Input, High Impedance Input)
- High current outputs
- Schmitt trigger inputs on Port G
- Packages:
  - — 44 PLCC with 40 I/O pins
  - — 40 DIP with 36 I/O pins
  - — 28 DIP with 24 I/O pins
  - — 28 SO with 24 I/O pins

CPU/Instruction Set Feature

- 1 μs instruction cycle time
- Ten multi-source vectored interrupts servicing
  - — External interrupt with selectable edge
  - — Idle Timer T0
  - — Timer (2 interrupts)
  - — MICROWIRE/PLUS
  - — Multi-Input Wake Up
  - — Software Trap
  - — UART (2)
  - — Default VIS (default interrupt)
- Versatile instruction set
- 8-bit Stack Pointer (SP)—stack in RAM
- Two 8-bit Register Indirect Data Memory Pointers (B, X)

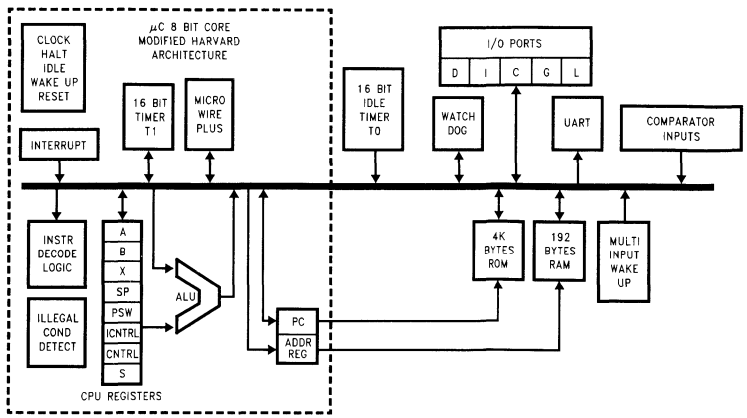
Fully Static CMOS

- Low current drain (typically < 1 μA)
- Single supply operation: 2.5V–6.0V
- Temperature ranges: 0°C to +70°C, −40°C to +85°C, −55°C to +125°C

Development Support

- Emulation and OTP devices
- Real time emulation and full program debug offered by MetaLink Development System

Block Diagram



TL/DD/10830–1

FIGURE 1. Block Diagram

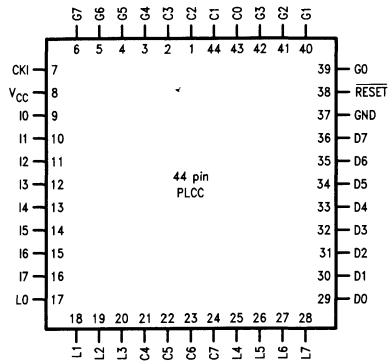
5

General Description (Continued)

It is a fully static part, fabricated using double-metal silicon gate microCMOS technology. Features include an 8-bit memory mapped architecture, MICROWIRE/PLUS serial I/O, one 16-bit timer/counter supporting three modes (Processor Independent PWM generation, External Event counter, and Input Capture mode capabilities), full duplex UART, one comparator, and two power savings modes (HALT and IDLE), both with a multi-sourced wakeup/interrupt capability. This multi-sourced interrupt capability may also be used independent of the HALT or IDLE modes. Each I/O pin has software selectable configurations. The device operates over a voltage range of 2.5V to 6V. High throughput is achieved with an efficient, regular instruction set operating at a maximum of 1 μs per instruction rate.

Connection Diagrams

Plastic Chip Carrier

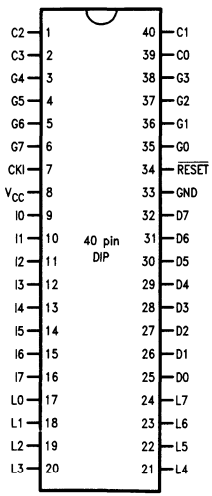


TL/DD/10830-2

Top View

Order Number COP888CS-XXX/V,
COP988CS-XXX/V or COP988CSH-XXX/V
See NS Package Number V44A

Dual-In-Line Package

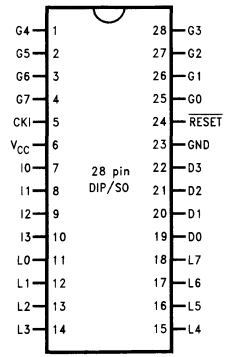


TL/DD/10830-3

Top View

Order Number COP888CS-XXX/N,
COP988CS-XXX/N or COP988CSH-XXX/N
See NS Package Number N40A

Dual-In-Line Package



TL/DD/10830-5

Top View

Order Number COP884CS-XXX/N,
COP984CS-XXX/N or COP984CSH-XXX/N
See NS Package Number N28B

Order Number COP884CS-XXX/WM,
COP984CS-XXX/WM or COP984CSH-XXX/WM
See NS Package Number M28B

FIGURE 2. Connection Diagrams

Connection Diagrams (Continued)

Pinouts for 28-, 40- and 44-Pin Packages

Port	Type	Alt. Fun	Alt. Fun	28-Pin Pack.	40-Pin Pack.	44-Pin Pack.
L0	I/O	MIWU		11	17	17
L1	I/O	MIWU	CKX	12	18	18
L2	I/O	MIWU	TDX	13	19	19
L3	I/O	MIWU	RDX	14	20	20
L4	I/O	MIWU		15	21	25
L5	I/O	MIWU		16	22	26
L6	I/O	MIWU		17	23	27
L7	I/O	MIWU		18	24	28
G0	I/O	INT		25	35	39
G1	WDOUT			26	36	40
G2	I/O	T1B		27	37	41
G3	I/O	T1A		28	38	42
G4	I/O	SO		1	3	3
G5	I/O	SK		2	4	4
G6	I	SI		3	5	5
G7	I/CKO	HALT Restart		4	6	6
D0	O			19	25	29
D1	O			20	26	30
D2	O			21	27	31
D3	O			22	28	32
I0	I			7	9	9
I1	I	COMP1IN−		8	10	10
I2	I	COMP1IN+		9	11	11
I3	I	COMP1OUT		10	12	12
I4	I				13	13
I5	I				14	14
I6	I				15	15
I7	I				16	16
D4	O				29	33
D5	O				30	34
D6	O				31	35
D7	O				32	36
C0	I/O				39	43
C1	I/O				40	44
C2	I/O				1	1
C3	I/O				2	2
C4	I/O					21
C5	I/O					22
C6	I/O					23
C7	I/O					24
V_{CC}				6	8	8
GND				23	33	37
CKI				5	7	7
$\overline{\text{RESET}}$				24	34	38

Absolute Maximum Ratings

If Military/Aerospace specified devices are required, please contact the National Semiconductor Sales Office/Distributors for availability and specifications.

Supply Voltage (V_{CC})	7V
Voltage at Any Pin	−0.3V to V_{CC} + 0.3V
Total Current into V_{CC} Pin (Source)	100 mA
Total Current out of GND Pin (Sink)	110 mA
Storage Temperature Range	−65°C to +140°C

Note: *Absolute maximum ratings indicate limits beyond which damage to the device may occur. DC and AC electrical specifications are not ensured when operating the device at absolute maximum ratings.*

DC Electrical Characteristics

98XCS: 0°C ≤ T_A ≤ +70°C unless otherwise specified

Parameter	Conditions	Min	Typ	Max	Units
Operating Voltage COP98XCS		2.5		4.0	V
COP98XCSH		4.0		6.0	V
Power Supply Ripple (Note 1)	Peak-to-Peak			0.1 V_{CC}	V
Supply Current (Note 2)					
CKI = 10 MHz	V_{CC} = 6V, t_c = 1 μs			12.5	mA
CKI = 4 MHz	V_{CC} = 6V, t_c = 2.5 μs			5.5	mA
CKI = 4 MHz	V_{CC} = 4V, t_c = 2.5 μs			2.5	mA
CKI = 1 MHz	V_{CC} = 4V, t_c = 10 μs			1.4	mA
HALT Current (Note 3)	V_{CC} = 6V, CKI = 0 MHz		<0.7	8	μA
	V_{CC} = 4V, CKI = 0 MHz		<0.3	4	μA
IDLE Current					
CKI = 10 MHz	V_{CC} = 6V, t_c = 1 μs			3.5	mA
CKI = 4 MHz	V_{CC} = 6V, t_c = 2.5 μs			2.5	mA
CKI = 1 MHz	V_{CC} = 4V, t_c = 10 μs			0.7	mA
Input Levels					
$\overline{\text{RESET}}$					
Logic High		0.8 V_{CC}			V
Logic Low				0.2 V_{CC}	V
CKI (External and Crystal Osc. Modes)					
Logic High		0.7 V_{CC}			V
Logic Low				0.2 V_{CC}	V
All Other Inputs					
Logic High		0.7 V_{CC}			V
Logic Low				0.2 V_{CC}	V
Hi-Z Input Leakage	V_{CC} = 6V	−1		+1	μA
Input Pullup Current	V_{CC} = 6V, V_{IN} = 0V	−40		−250	μA
G and L Port Input Hysteresis				0.35 V_{CC}	V
Output Current Levels					
D Outputs					
Source	V_{CC} = 4V, V_{OH} = 3.3V	−0.4			mA
	V_{CC} = 2.5V, V_{OH} = 1.8V	−0.2			mA
Sink	V_{CC} = 4V, V_{OL} = 1V	10			mA
	V_{CC} = 2.5V, V_{OL} = 0.4V	2.0			mA
All Others					
Source (Weak Pull-Up Mode)	V_{CC} = 4V, V_{OH} = 2.7V	−10		−100	μA
	V_{CC} = 2.5V, V_{OH} = 1.8V	−2.5		−33	μA
Source (Push-Pull Mode)	V_{CC} = 4V, V_{OH} = 3.3V	−0.4			mA
	V_{CC} = 2.5V, V_{OH} = 1.8V	−0.2			mA
Sink (Push-Pull Mode)	V_{CC} = 4V, V_{OL} = 0.4V	1.6			mA
	V_{CC} = 2.5V, V_{OL} = 0.4V	0.7			mA
TRI-STATE Leakage	V_{CC} = 6.0V	−1		+1	μA

Note 1: Rate of voltage change must be less then 0.5 V/ms.

Note 2: Supply current is measured after running 2000 cycles with a square wave CKI input, CKO open, inputs at rails and outputs open.

Note 3: The HALT mode will stop CKI from oscillating in the RC and the Crystal configurations. Test conditions: All inputs tied to V_{CC}, L, C and G0–G5 configured as outputs and set high. The D port set to zero. The clock monitor and the comparators are disabled.

DC Electrical Characteristics 98XCS: $0°C \le T_A \le +70°C$ unless otherwise specified (Continued)

Parameter	Conditions	Min	Typ	Max	Units
Allowable Sink/Source Current per Pin					
D Outputs (Sink)				15	mA
All others				3	mA
Maximum Input Current without Latchup (Note 5)	$T_A = 25°C$			±100	mA
RAM Retention Voltage, V_r	500 ns Rise and Fall Time (Min)	2			V
Input Capacitance				7	pF
Load Capacitance on D2				1000	pF

AC Electrical Characteristics 98XCS: $0°C \le T_A \le +70°C$ unless otherwise specified

Parameter	Conditions	Min	Typ	Max	Units
Instruction Cycle Time (t_c)	$4V \le V_{CC} \le 6V$	1		DC	μs
Crystal, Resonator,	$2.5V \le V_{CC} < 4V$	2.5		DC	μs
R/C Oscillator	$4V \le V_{CC} \le 6V$	3		DC	μs
	$2.5V \le V_{CC} < 4V$	7.5		DC	μs
Inputs					
t_{SETUP}	$4V \le V_{CC} \le 6V$	200			ns
	$2.5V \le V_{CC} < 4V$	500			ns
t_{HOLD}	$4V \le V_{CC} \le 6V$	60			ns
	$2.5V \le V_{CC} < 4V$	150			ns
Output Propagation Delay (Note 6)	$R_L = 2.2k, C_L = 100$ pF				
t_{PD1}, t_{PD0}					
SO, SK	$4V \le V_{CC} \le 6V$			0.7	μs
	$2.5V \le V_{CC} < 4V$			1.75	μs
All Others	$4V \le V_{CC} \le 6V$			1	μs
	$2.5V \le V_{CC} < 4V$			2.5	μs
MICROWIRE™ Setup Time (t_{UWS})		20			ns
MICROWIRE Hold Time (t_{UWH})		56			ns
MICROWIRE Output Propagation Delay (t_{UPD})				220	ns
Input Pulse Width					
Interrupt Input High Time		1			t_c
Interrupt Input Low Time		1			t_c
Timer Input High Time		1			t_c
Timer Input Low Time		1			t_c
Reset Pulse Width		1			μs

Note 5: Pins G6 and $\overline{\text{RESET}}$ are designed with a high voltage input network for factory testing. These pins allow input voltages greater than V_{CC} and the pins will have sink current to V_{CC} when biased at voltages greater than V_{CC} (the pins do not have source current when biased at a voltage below V_{CC}). The effective resistance to V_{CC} is 750Ω (typical). These two pins will not latch up. The voltage at the pins must be limited to less than 14V.

Notye 6: The output propagation delay is referenced to the end of the instruction cycle where the output change occurs.

5

Absolute Maximum Ratings

If Military/Aerospace specified devices are required, please contact the National Semiconductor Sales Office/Distributors for availability and specifications.

Supply Voltage (V_{CC})	7V
Voltage at Any Pin	−0.3V to V_{CC} + 0.3V
Total Current into V_{CC} Pin (Source)	100 mA
Total Current out of GND Pin (Sink)	110 mA
Storage Temperature Range	−65°C to +140°C

Note: *Absolute maximum ratings indicate limits beyond which damage to the device may occur. DC and AC electrical specifications are not ensured when operating the device at absolute maximum ratings.*

DC Electrical Characteristics

88XCS: $-40°C \le T_A \le +85°C$ unless otherwise specified

Parameter	Conditions	Min	Typ	Max	Units
Operating Voltage		2.5		6	V
Power Supply Ripple (Note 1)	Peak-to-Peak			0.1 V_{CC}	V
Supply Current (Note 2)					
CKI = 10 MHz	V_{CC} = 6V, t_c = 1 μs			12.5	mA
CKI = 4 MHz	V_{CC} = 6V, t_c = 2.5 μs			5.5	mA
HALT Current (Note 3)	V_{CC} = 6V, CKI = 0 MHz		<1	10	μA
IDLE Current					
CKI = 10 MHz	V_{CC} = 6V, t_c = 1 μs			3.5	mA
CKI = 4 MHz	V_{CC} = 6V, t_c = 2.5 μs			2.5	mA
Input Levels					
$\overline{\text{RESET}}$					
Logic High		0.8 V_{CC}			V
Logic Low				0.2 V_{CC}	V
CKI (External and Crystal Osc. Modes)					
Logic High		0.7 V_{CC}			V
Logic Low				0.2 V_{CC}	V
All Other Inputs					
Logic High		0.7 V_{CC}			V
Logic Low				0.2 V_{CC}	V
Hi-Z Input Leakage	V_{CC} = 6V	−2		+2	μA
Input Pullup Current	V_{CC} = 6V, V_{IN} = 0V	−40		−250	μA
G and L Port Input Hysteresis				0.35 V_{CC}	V
Output Current Levels					
D Outputs					
Source	V_{CC} = 4V, V_{OH} = 3.3V	−0.4			mA
	V_{CC} = 2.5V, V_{OH} = 1.8V	−0.2			mA
Sink	V_{CC} = 4V, V_{OL} = 1V	10			mA
	V_{CC} = 2.5V, V_{OL} = 0.4V	2.0			mA
All Others					
Source (Weak Pull-Up Mode)	V_{CC} = 4V, V_{OH} = 2.7V	−10		−100	μA
	V_{CC} = 2.5V, V_{OH} = 1.8V	−2.5		−33	μA
Source (Push-Pull Mode)	V_{CC} = 4V, V_{OH} = 3.3V	−0.4			mA
	V_{CC} = 2.5V, V_{OH} = 1.8V	−0.2			mA
Sink (Push-Pull Mode)	V_{CC} = 4V, V_{OL} = 0.4V	1.6			mA
	V_{CC} = 2.5V, V_{OL} = 0.4V	0.7			mA
TRI-STATE Leakage		−2		+2	μA

Note 1: Rate of voltage change must be less then 0.5 V/ms.

Note 2: Supply current is measured after running 2000 cycles with a square wave CKI input, CKO open, inputs at rails and outputs open.

Note 3: The HALT mode will stop CKI from oscillating in the RC and the Crystal configurations. Test conditions: All inputs tied to V_{CC}, L, C and G0–G5 configured as outputs and set high. The D port set to zero. The clock monitor and the comparators are disabled.

DC Electrical Characteristics 88XCS: $-40°C \le T_A \le +85°C$ unless otherwise specified (Continued)

Parameter	Conditions	Min	Typ	Max	Units
Allowable Sink/Source Current per Pin					
D Outputs (Sink)				15	mA
All others				3	mA
Maximum Input Current without Latchup (Note 5)	$T_A = 25°C$			±100	mA
RAM Retention Voltage, V_r	500 ns Rise and Fall Time (Min)	2			V
Input Capacitance				7	pF
Load Capacitance on D2				1000	pF

AC Electrical Characteristics 88XCS: $-40°C \le T_A \le +85°C$ unless otherwise specified

Parameter	Conditions	Min	Typ	Max	Units
Instruction Cycle Time (t_c)	$4V \le V_{CC} \le 6V$	1		DC	μs
Crystal, Resonator,	$2.5V \le V_{CC} < 4V$	2.5		DC	μs
R/C Oscillator	$4V \le V_{CC} \le 6V$	3		DC	μs
	$2.5V \le V_{CC} < 4V$	7.5		DC	μs
Inputs					
t_{SETUP}	$4V \le V_{CC} \le 6V$	200			ns
	$2.5V \le V_{CC} < 4V$	500			ns
t_{HOLD}	$4V \le V_{CC} \le 6V$	60			ns
	$2.5V \le V_{CC} < 4V$	150			ns
Output Propagation Delay (Note 6)	$R_L = 2.2k$, $C_L = 100$ pF				
t_{PD1}, t_{PD0}					
SO, SK	$4V \le V_{CC} \le 6V$			0.7	μs
	$2.5V \le V_{CC} < 4V$			1.75	μs
All Others	$4V \le V_{CC} \le 6V$			1	μs
	$2.5V \le V_{CC} < 4V$			2.5	μs
MICROWIRE Setup Time (t_{UWS})		20			ns
MICROWIRE Hold Time (t_{UWH})		56			ns
MICROWIRE Output Propagation Delay (t_{UPD})				220	ns
Input Pulse Width					
Interrupt Input High Time		1			t_c
Interrupt Input Low Time		1			t_c
Timer Input High Time		1			t_c
Timer Input Low Time		1			t_c
Reset Pulse Width		1			μs

Note 5: Pins G6 and $\overline{\text{RESET}}$ are designed with a high voltage input network for factory testing. These pins allow input voltages greater than V_{CC} and the pins will have sink current to V_{CC} when biased at voltages greater than V_{CC} (the pins do not have source current when biased at a voltage below V_{CC}). The effective resistance to V_{CC} is 750Ω (typical). These two pins will not latch up. The voltage at the pins must be limited to less than 14V.

Note 6: The output propagation delay is referenced to the end of the instruction cycle where the output change occurs.

5

Absolute Maximum Ratings

If Military/Aerospace specified devices are required, please contact the National Semiconductor Sales Office/Distributors for availability and specifications.

Supply Voltage (V_{CC})	7V
Voltage at Any Pin	−0.3V to V_{CC} + 0.3V
Total Current into V_{CC} Pin (Source)	100 mA
Total Current out of GND Pin (Sink)	110 mA
Storage Temperature Range	−65°C to +140°C

Note: *Absolute maximum ratings indicate limits beyond which damage to the device may occur. DC and AC electrical specifications are not ensured when operating the device at absolute maximum ratings.*

DC Electrical Characteristics 68XCS: −55°C < T_A ≤ +125°C unless otherwise specified

Parameter	Conditions	Min	Typ	Max	Units
Operating Voltage		4.5		5.5	V
Power Supply Ripple (Note 1)	Peak-to-Peak			0.1 V_{CC}	V
Supply Current (Note 2)					
CKI = 10 MHz	V_{CC} = 5.5V, t_c = 1 μs			12.5	mA
CKI = 4 MHz	V_{CC} = 5.5V, t_c = 2.5 μs			5.5	mA
HALT Current (Note 3)	V_{CC} = 5.5V, CKI = 0 MHz		<10	30	μA
IDLE Current					
CKI = 10 MHz	V_{CC} = 5.5V, t_c = 1 μs			3.5	mA
CKI = 4 MHz	V_{CC} = 5.5V, t_c = 2.5 μs			2.5	mA
Input Levels					
$\overline{\text{RESET}}$					
Logic High		0.8 V_{CC}			V
Logic Low				0.2 V_{CC}	V
CKI (External and Crystal Osc. Modes)					
Logic High		0.7 V_{CC}			V
Logic Low				0.2 V_{CC}	V
All Other Inputs					
Logic High		0.7 V_{CC}			V
Logic Low				0.2 V_{CC}	V
Hi-Z Input Leakage	V_{CC} = 5.5V, V_{IN} = 0V	−5		+5	μA
Input Pullup Current	V_{CC} = 5.5V, V_{IN} = 0V	−35		−400	μA
G and L Port Input Hysteresis				0.35 V_{CC}	V
Output Current Levels					
D Outputs					
Source	V_{CC} = 4.5V, V_{OH} = 3.8V	−0.4			mA
Sink	V_{CC} = 4.5V, V_{OL} = 1V	9			mA
All Others					
Source (Weak Pull-Up Mode)	V_{CC} = 4.5V, V_{OH} = 3.2V	−9		140	μA
Source (Push-Pull Mode)	V_{CC} = 4.5V, V_{OH} = 3.8V	−0.4			mA
Sink (Push-Pull Mode)	V_{CC} = 4.5V, V_{OL} = 0.4V	1.4			mA
TRI-STATE Leakage	V_{CC} = 5.5V	−5		+5	μA

Note 1: Rate of voltage change must be less then 0.5 V/ms.

Note 2: Supply current is measured after running 2000 cycles with a square wave CKI input, CKO open, inputs at rails and outputs open.

Note 3: The HALT mode will stop CKI from oscillating in the RC and the Crystal configurations. Test conditions: All inputs tied to V_{CC}, L, C and G0–G5 configured as outputs and set high. The D port set to zero. The clock monitor and the comparators are disabled.

DC Electrical Characteristics 68XCS: $-55°C \le T_A \le +125°C$ unless otherwise specified (Continued)

Parameter	Conditions	Min	Typ	Max	Units
Allowable Sink/Source Current per Pin					
D Outputs (Sink)				12	mA
All others				2.5	mA
Maximum Input Current without Latchup (Note 5)	$T_A = 25°C$			±100	mA
RAM Retention Voltage, V_r	500 ns Rise and Fall Time (Min)	2			V
Input Capacitance				7	pF
Load Capacitance on D2				1000	pF

AC Electrical Characteristics 68XCS: $-55°C \le T_A \le +125°C$ unless otherwise specified

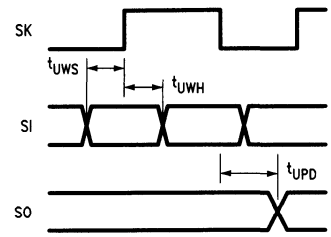
Parameter	Conditions	Min	Typ	Max	Units
Instruction Cycle Time (t_c)	$4.5V \le V_{CC} \le 5.5V$	1		DC	μs
Crystal, Resonator, R/C Oscillator	$4.5V \le V_{CC} \le 5.5V$	3		DC	μs
Inputs					
t_{SETUP}	$4.5V \le V_{CC} \le 5.5V$	200			ns
t_{HOLD}	$4.5V \le V_{CC} \le 5.5V$	60			ns
Output Propagation Delay (Note 6)	$R_L = 2.2k$, $C_L = 100$ pF				
t_{PD1}, t_{PD0}					
SO, SK	$4.5V \le V_{CC} \le 5.5V$			0.7	μs
All Others	$4.5V \le V_{CC} \le 5.5V$			1	μs
MICROWIRE Setup Time (t_{UWS})		20			ns
MICROWIRE Hold Time (t_{UWH})		56			ns
MICROWIRE Output Propagation Delay (t_{UPD})				220	ns
Input Pulse Width					
Interrupt Input High Time		1			t_c
Interrupt Input Low Time		1			t_c
Timer Input High Time		1			t_c
Timer Input Low Time		1			t_c
Reset Pulse Width		1			μs

Note 5: Pins G6 and $\overline{\text{RESET}}$ are designed with a high voltage input network for factory testing. These pins allow input voltages greater than V_{CC} and the pins will have sink current to V_{CC} when biased at voltages greater than V_{CC} (the pins do not have source current when biased at a voltage below V_{CC}). The effective resistance to V_{CC} is 750Ω (typical). These two pins will not latch up. The voltage at the pins must be limited to less than 14V.

Note 6: The output propagation delay is referenced to the end of the instruction cycle where the output change occurs.

Comparator AC and DC Characteristics $V_{CC} = 5V, T_A = 25°C$

Parameter	Conditions	Min	Typ	Max	Units
Input Offset Voltage	$0.4V \le V_{IN} \le V_{CC} - 1.5V$		±10	±25	mV
Input Common Mode Voltage Range		0.4		$V_{CC} - 1.5$	V
Low Level Output Current	$V_{OL} = 0.4V$	1.6			mA
High Level Output Current	$V_{OH} = 4.6V$	1.6			mA
DC Supply Current (When Enabled)				250	μA
Response Time	TBD mV Step, TBD mV Overdrive, 100 pF Load		1		μs

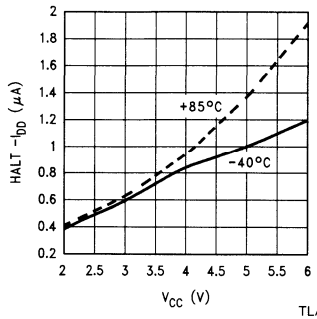


TL/DD/10830–6

FIGURE 3. MICROWIRE/PLUS Timing

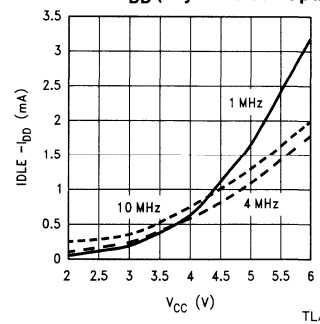
Typical Performance Characteristics (−40°C to +85°C)

Halt—I_{DD}



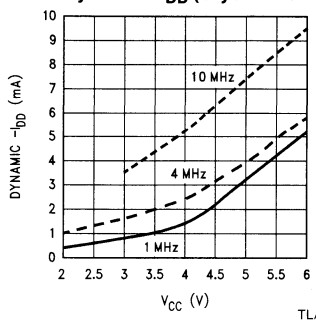
TL/DD/10830–24

Idle—I_{DD} (Crystal Clock Option)



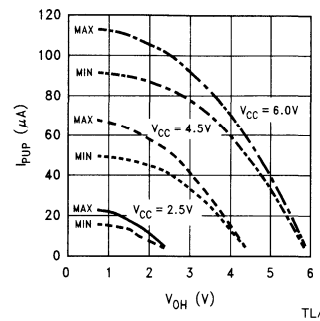
TL/DD/10830–25

Dynamic—I_{DD} (Crystal Clock Option)



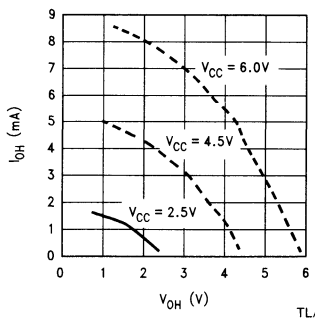
TL/DD/10830–26

Port L/C/G Weak Pull-Up Source Current



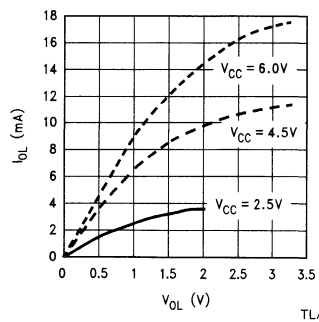
TL/DD/10830–27

Port L/C/G Push-Pull Source Current



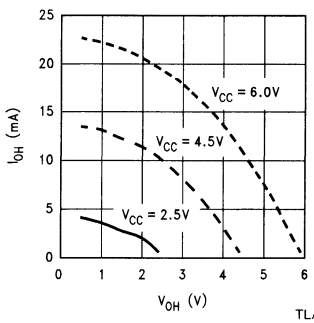
TL/DD/10830–28

Port L/C/G/ Push-Pull Sink Current



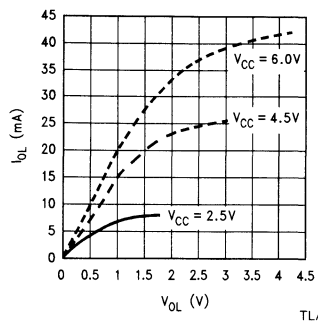
TL/DD/10830–29

Port D Source Current



TL/DD/10830–30

Port D Sink Current



TL/DD/10830–31

5

Pin Descriptions

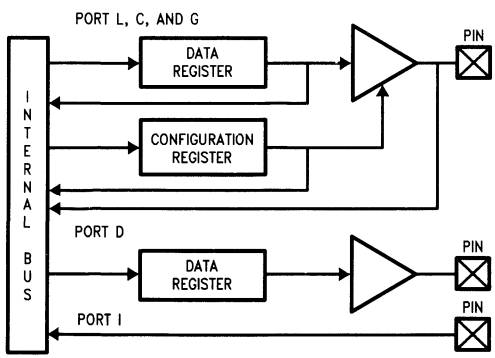
V_{CC} and GND are the power supply pins.

CKI is the clock input. This can come from an R/C generated oscillator, or a crystal oscillator (in conjunction with CKO). See Oscillator Description section.

$\overline{\text{RESET}}$ is the master reset input. See Reset Description section.

The device contains three bidirectional 8-bit I/O ports (C, G and L), where each individual bit may be independently configured as an input (Schmitt trigger inputs on ports L and G), output or TRI-STATE under program control. Three data memory address locations are allocated for each of these I/O ports. Each I/O port has two associated 8-bit memory mapped registers, the CONFIGURATION register and the output DATA register. A memory mapped address is also reserved for the input pins of each I/O port. (See the memory map for the various addresses associated with the I/O ports.) *Figure 4* shows the I/O port configurations. The DATA and CONFIGURATION registers allow for each port bit to be individually configured under software control as shown below:

CONFIGURATION Register	DATA Register	Port Set-Up
0	0	Hi-Z Input (TRI-STATE Output)
0	1	Input with Weak Pull-Up
1	0	Push-Pull Zero Output
1	1	Push-Pull One Output



TL/DD/10830–7

FIGURE 4. I/O Port Configurations

Port L is an 8-bit I/O port. All L-pins have Schmitt triggers on the inputs.

The Port L supports Multi-Input Wake Up on all eight pins. L1 is used for the UART external clock. L2 and L3 are used for the UART transmit and receive.

The Port L has the following alternate features:

- L0 MIWU
- L1 MIWU or CKX
- L2 MIWU or TDX
- L3 MIWU or RDX
- L4 MIWU
- L5 MIWU
- L6 MIWU
- L7 MIWU

Port G is an 8-bit port with 5 I/O pins (G0, G2–G5), an input pin (G6), and two dedicated output pins (G1 and G7). Pins G0 and G2–G6 all have Schmitt Triggers on their inputs. Pin G1 serves as the dedicated WDOUT WATCHDOG output, while pin G7 is either input or output depending on the oscillator mask option selected. With the crystal oscillator option selected, G7 serves as the dedicated output pin for the CKO clock output. With the single-pin R/C oscillator mask option selected, G7 serves as a general purpose input pin but is also used to bring the device out of HALT mode with a low to high transition on G7. There are two registers associated with the G Port, a data register and a configuration register. Therefore, each of the 5 I/O bits (G0, G2–G5) can be individually configured under software control.

Since G6 is an input only pin and G7 is the dedicated CKO clock output pin (crystal clock option) or general purpose input (R/C clock option), the associated bits in the data and configuration registers for G6 and G7 are used for special purpose functions as outlined below. Reading the G6 and G7 data bits will return zeros.

Note that the chip will be placed in the HALT mode by writing a "1" to bit 7 of the Port G Data Register. Similarly the chip will be placed in the IDLE mode by writing a "1" to bit 6 of the Port G Data Register.

Writing a "1" to bit 6 of the Port G Configuration Register enables the MICROWIRE/PLUS to operate with the alternate phase of the SK clock. The G7 configuration bit, if set high, enables the clock start up delay after HALT when the R/C clock configuration is used.

	Config Reg.	Data Reg.
G7	CLKDLY	HALT
G6	Alternate SK	IDLE

Port G has the following alternate features:

- G0 INTR (External Interrupt Input)
- G2 T1B (Timer T1 Capture Input)
- G3 T1A (Timer T1 I/O)
- G4 SO (MICROWIRE Serial Data Output)
- G5 SK (MICROWIRE Serial Clock)
- G6 SI (MICROWIRE Serial Data Input)

Port G has the following dedicated functions:

- G1 WDOUT WATCHDOG and/or Clock Monitor dedicated output
- G7 CKO Oscillator dedicated output or general purpose input

Port C is an 8-bit I/O port. The 40-pin device does not have a full complement of Port C pins. The unavailable pins are not terminated. A read operation for these unterminated pins will return unpredictable values.

Port I is an eight-bit Hi-Z input port. The 28-pin device does not have a full complement of Port I pins. The unavailable

Pin Descriptions (Continued)

pins are not terminated i.e., they are floating. A read operation for these unterminated pins will return unpredictable values. The user must ensure that the software takes this into account by either masking or restricting the accesses to bit operations. The unterminated Port I pins will draw power only when addressed.

Ports I1–I3 are used for Comparator 1.

Ports I1–I3 have the following alternate features.

I1 COMP1 − IN (Comparator 1 Negative Input)

I2 COMP1 + IN (Comparator 1 Positive Input)

I3 COMP1OUT (Comparator 1 Output)

Port D is an 8-bit output port that is preset high when $\overline{\text{RESET}}$ goes low. The user can tie two or more D port outputs (except D2) together in order to get a higher drive.

Note: Care must be exercised with the D2 pin operation. At RESET, the external loads on this pin must ensure that the output voltages stay above 0.8 V_{CC} to prevent the chip from entering special modes. Also keep the external loading on D2 to less than 1000 pF.

Functional Description

The architecture of the device is modified Harvard architecture. With the Harvard architecture, the control store program memory (ROM) is separated from the data store memory (RAM). Both ROM and RAM have their own separate addressing space with separate address buses. The architecture, though based on Harvard architecture, permits transfer of data from ROM to RAM.

CPU REGISTERS

The CPU can do an 8-bit addition, subtraction, logical or shift operation in one instruction (t_C) cycle time.

There are six CPU registers:

A is the 8-bit Accumulator Register

PC is the 15-bit Program Counter Register

PU is the upper 7 bits of the program counter (PC)
PL is the lower 8 bits of the program counter (PC)

B is an 8-bit RAM address pointer, which can be optionally post auto incremented or decremented.

X is an 8-bit alternate RAM address pointer, which can be optionally post auto incremented or decremented.

SP is the 8-bit stack pointer, which points to the subroutine/interrupt stack (in RAM). The SP is initialized to RAM address 06F with reset.

S is the 8-bit Data Segment Address Register used to extend the lower half of the address range (00 to 7F) into 256 data segments of 128 bytes each.

All the CPU registers are memory mapped with the exception of the Accumulator (A) and the Program Counter (PC).

PROGRAM MEMORY

Program memory consists of 4096 bytes of ROM. These bytes may hold program instructions or constant data (data tables for the LAID instruction, jump vectors for the JID instruction, and interrupt vectors for the VIS instruction). The program memory is addressed by the 15-bit program counter (PC). All interrupts vector to program memory location 0FF Hex.

DATA MEMORY

The data memory address space includes the on-chip RAM and data registers, the I/O registers (Configuration, Data and Pin), the control registers, the MICROWIRE/PLUS SIO shift register, and the various registers, and counters associated with the timers (with the exception of the IDLE timer). Data memory is addressed directly by the instruction or indirectly by the B, X, SP pointers and S register.

The device has 192 bytes of RAM. Sixteen bytes of RAM are mapped as "registers" at addresses 0F0 to 0FF Hex. These registers can be loaded immediately, and also decremented and tested with the DRSZ (decrement register and skip if zero) instruction. The memory pointer registers X, SP, B and S are memory mapped into this space at address locations 0FC to 0FF Hex respectively, with the other registers being available for general usage.

The instruction set permits any bit in memory to be set, reset or tested. All I/O and registers (except A and PC) are memory mapped; therefore, I/O bits and register bits can be directly and individually set, reset and tested. The accumulator (A) bits can also be directly and individually tested.

Note: RAM contents are undefined upon power-up.

Data Memory Segment RAM Extension

Data memory address 0FF is used as a memory mapped location for the Data Segment Address Register (S).

The data store memory is either addressed directly by a single byte address within the instruction, or indirectly relative to the reference of the B, X, or SP pointers (each contains a single-byte address). This single-byte address allows an addressing range of 256 locations from 00 to FF hex. The upper bit of this single-byte address divides the data store memory into two separate sections as outlined previously. With the exception of the RAM register memory from address locations 00F0 to 00FF, all RAM memory is memory mapped with the upper bit of the single-byte address being equal to zero. This allows the upper bit of the single-byte address to determine whether or not the base address range (from 0000 to 00FF) is extended. If this upper bit equals one (representing address range 0080 to 00FF), then address extension does not take place. Alternatively, if this upper bit equals zero, then the data segment extension register S is used to extend the base address range (from 0000 to 007F) from XX00 to XX7F, where XX represents the 8 bits from the S register. Thus the 128-byte data segment extensions are located from addresses 0100 to 017F for data segment 1, 0200 to 027F for data segment 2, etc., up to FF00 to FF7F for data segment 255. The base address range from 0000 to 007F represents data segment 0.

Figure 5 illustrates how the S register data memory extension is used in extending the lower half of the base address range (00 to 7F hex) into 256 data segments of 128 bytes each, with a total addressing range of 32 kbytes from XX00 to XX7F. This organization allows a total of 256 data segments of 128 bytes each with an additional upper base segment of 128 bytes. Furthermore, all addressing modes are available for all data segments. The S register must be changed under program control to move from one data segment (128 bytes) to another. However, the upper base segment (containing the 16 memory registers, I/O registers, control registers, etc.) is always available regardless of the

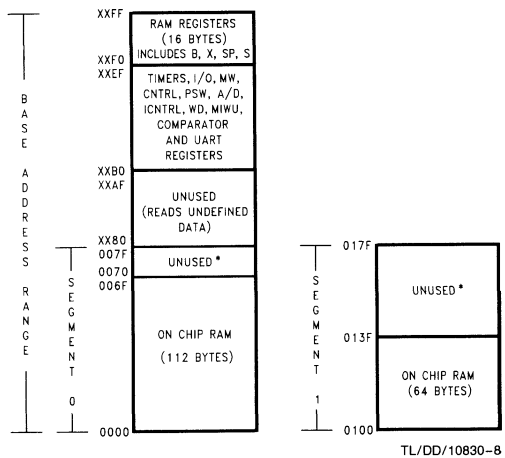
Data Memory Segment RAM Extension (Continued)

contents of the S register, since the upper base segment (address range 0080 to 00FF) is independent of data segment extension.

The instructions that utilize the stack pointer (SP) always reference the stack as part of the base segment (Segment 0), regardless of the contents of the S register. The S register is not changed by these instructions. Consequently, the stack (used with subroutine linkage and interrupts) is always located in the base segment. The stack pointer will be initialized to point at data memory location 006F as a result of reset.

The 128 bytes of RAM contained in the base segment are split between the lower and upper base segments. The first 112 bytes of RAM are resident from address 0000 to 006F in the lower base segment, while the remaining 16 bytes of RAM represent the 16 data memory registers located at addresses 00F0 to 00FF of the upper base segment. No RAM is located at the upper sixteen addresses (0070 to 007F) of the lower base segment.

Additional RAM beyond these initial 128 bytes, however, will always be memory mapped in groups of 128 bytes (or less) at the data segment address extensions (XX00 to XX7F) of the lower base segment. The additional 64 bytes of RAM (beyond the initial 128 bytes) are memory mapped at address locations 0100 to 013F hex.



*Reads as all ones.

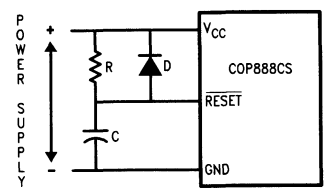
FIGURE 5. RAM Organization

Reset

The $\overline{\text{RESET}}$ input when pulled low initializes the microcontroller. Initialization will occur whenever the $\overline{\text{RESET}}$ input is pulled low. Upon initialization, the data and configuration registers for ports L, G and C are cleared, resulting in these Ports being initialized to the TRI-STATE mode. Pin G1 of the G Port is an exception (as noted below) since pin G1 is dedicated as the WATCHDOG and/or Clock Monitor error output pin. Port D is set high. The PC, PSW, ICNTRL, CNTRL, are cleared. The UART registers PSR, ENU (except that TBMT bit is set), ENUR and ENUI are cleared. The Comparator Select Register is cleared. The S register is initialized to zero. The Multi-Input Wakeup registers WKEN, WKEDG and WKPND are cleared. The stack pointer, SP, is initialized to 6F Hex.

The device comes out of reset with both the WATCHDOG logic and the Clock Monitor detector armed, with the WATCHDOG service window bits set and the Clock Monitor bit set. The WATCHDOG and Clock Monitor circuits are inhibited during reset. The WATCHDOG service window bits being initialized high default to the maximum WATCHDOG service window of 64k t_C clock cycles. The Clock Monitor bit being initialized high will cause a Clock Monitor error following reset if the clock has not reached the minimum specified frequency at the termination of reset. A Clock Monitor error will cause an active low error output on pin G1. This error output will continue until 16 t_C–32 t_C clock cycles following the clock frequency reaching the minimum specified value, at which time the G1 output will enter the TRI-STATE mode.

The external RC network shown in *Figure 6* should be used to ensure that the $\overline{\text{RESET}}$ pin is held low until the power supply to the chip stabilizes.



RC > 5 × Power Supply Rise Time

FIGURE 6. Recommended Reset Circuit

Oscillator Circuits

The chip can be driven by a clock input on the CKI input pin which can be between DC and 10 MHz. The CKO output clock is on pin G7 (crystal configuration). The CKI input frequency is divided down by 10 to produce the instruction cycle clock ($1/t_C$).

Figure 7 shows the Crystal and R/C diagrams.

CRYSTAL OSCILLATOR

CKI and CKO can be connected to make a closed loop crystal (or resonator) controlled oscillator.

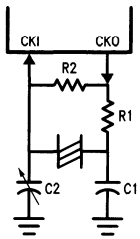
Table A shows the component values required for various standard crystal values.

Oscillator Circuits (Continued)

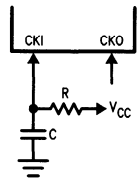
R/C OSCILLATOR

By selecting CKI as a single pin oscillator input, a single pin R/C oscillator circuit can be connected to it. CKO is available as a general purpose input, and/or HALT restart input.

Table B shows the variation in the oscillator frequencies as functions of the component (R and C) values.



TL/DD/10830-10



TL/DD/10830-11

FIGURE 7. Crystal and R/C Oscillator Diagrams

TABLE A. Crystal Oscillator Configuration, T_A = 25°C

R1 (kΩ)	R2 (MΩ)	C1 (pF)	C2 (pF)	CKI Freq (MHz)	Conditions
0	1	30	30–36	10	V_{CC} = 5V
0	1	30	30–36	4	V_{CC} = 5.0V
0	1	200	100–150	0.455	V_{CC} = 2.5V

TABLE B. RC Oscillator Configuration, T_A = 25°C

R (kΩ)	C (pF)	CKI Freq (MHz)	Instr. Cycle (μs)	Conditions
3.3	82	2.2 to 2.7	3.7 to 4.6	V_{CC} = 5V
5.6	100	1.1 to 1.3	7.4 to 9.0	V_{CC} = 5V
6.8	100	0.9 to 1.1	8.8 to 10.8	V_{CC} = 5V

Note: 3k ≤ R ≤ 200k
50 pF ≤ C < 200 pF

Control Registers

CNTRL Register (Address X'00EE)

The Timer1 (T1) and MICROWIRE/PLUS control register contains the following bits:

- SL1 & SL0 — Select the MICROWIRE/PLUS clock divide by (00 = 2, 01 = 4, 1x = 8)
- IEDG — External interrupt edge polarity select (0 = Rising edge, 1 = Falling edge)
- MSEL — Selects G5 and G4 as MICROWIRE/PLUS signals SK and SO respectively
- T1C0 — Timer T1 Start/Stop control in timer modes 1 and 2
  Timer T1 Underflow Interrupt Pending Flag in timer mode 3
- T1C1 — Timer T1 mode control bit
- T1C2 — Timer T1 mode control bit
- T1C3 — Timer T1 mode control bit

T1C3	T1C2	T1C1	T1C0	MSEL	IEDG	SL1	SL0

Bit 7 Bit 0

PSW Register (Address X'00EF)

The PSW register contains the following select bits:

- GIE — Global interrupt enable (enables interrupts)
- EXEN — Enable external interrupt
- BUSY — MICROWIRE/PLUS busy shifting flag
- EXPND — External interrupt pending
- T1ENA — Timer T1 Interrupt Enable for Timer Underflow or T1A Input capture edge
- T1PNDA — Timer T1 Interrupt Pending Flag (Autoreload RA in mode 1, T1 Underflow in Mode 2, T1A capture edge in mode 3)
- C — Carry Flag
- HC — Half Carry Flag

HC	C	T1PNDA	T1ENA	EXPND	BUSY	EXEN	GIE

Bit 7 Bit 0

The Half-Carry bit is also affected by all the instructions that affect the Carry flag. The SC (Set Carry) and RC (Reset Carry) instructions will respectively set or clear both the carry flags. In addition to the SC and RC instructions, ADC, SUBC, RRC and RLC instructions affect the carry and Half Carry flags.

Control Registers (Continued)

ICNTRL Register (Address X'00E8)

The ICNTRL register contains the following bits:

T1ENB	Timer T1 Interrupt Enable for T1B Input capture edge
T1PNDB	Timer T1 Interrupt Pending Flag for T1B capture edge
μWEN	Enable MICROWIRE/PLUS interrupt
μWPND	MICROWIRE/PLUS interrupt pending
T0EN	Timer T0 Interrupt Enable (Bit 12 toggle)
T0PND	Timer T0 Interrupt pending
LPEN	L Port Interrupt Enable (Multi-Input Wakeup/Interrupt)
	Bit 7 could be used as a flag

Unused	LPEN	T0PND	T0EN	μWPND	μWEN	T1PNDB	T1ENB
Bit 7							Bit 0

Timers

The device contains a very versatile set of timers (T0, T1). All timers and associated autoreload/capture registers power up containing random data.

TIMER T0 (IDLE TIMER)

The device supports applications that require maintaining real time and low power with the IDLE mode. This IDLE mode support is furnished by the IDLE timer T0, which is a 16-bit timer. The Timer T0 runs continuously at the fixed rate of the instruction cycle clock, t_C. The user cannot read or write to the IDLE Timer T0, which is a count down timer.

The Timer T0 supports the following functions:

Exit out of the Idle Mode (See Idle Mode description)
WATCHDOG logic (See WATCHDOG description)
Start up delay out of the HALT mode

The IDLE Timer T0 can generate an interrupt when the thirteenth bit toggles. This toggle is latched into the T0PND pending flag, and will occur every 4 ms at the maximum clock frequency (t_C = 1 μs). A control flag T0EN allows the interrupt from the thirteenth bit of Timer T0 to be enabled or disabled. Setting T0EN will enable the interrupt, while resetting it will disable the interrupt.

TIMER T1

The device has a powerful timer/counter block.

The timer block consists of a 16-bit timer, T1, and two supporting 16-bit autoreload/capture registers, R1A and R1B. It has two pins associated with it, T1A and T1B. The pin T1A supports I/O required by the timer block, while the pin T1B is an input to the timer block. The powerful and flexible timer block allows the device to easily perform all timer functions with minimal software overhead. The timer block has three operating modes: Processor Independent PWM mode, External Event Counter mode, and Input Capture mode.

The control bits T1C3, T1C2, and T1C1 allow selection of the different modes of operation.

Mode 1. Processor Independent PWM Mode

As the name suggests, this mode allows the device to generate a PWM signal with very minimal user intervention.

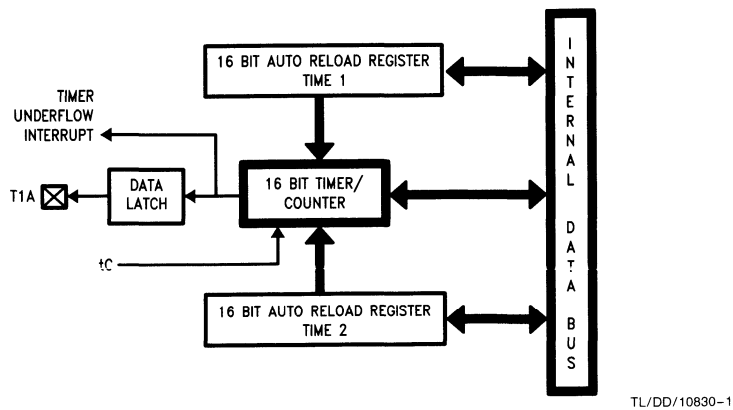
The user only has to define the parameters of the PWM signal (ON time and OFF time). Once begun, the timer block will continuously generate the PWM signal completely independent of the microcontroller. The user software services the timer block only when the PWM parameters require updating.

In this mode the timer T1 counts down at a fixed rate of t_C. Upon every underflow the timer is alternately reloaded with the contents of supporting registers, R1A and R1B. The very first underflow of the timer causes the timer to reload from the register R1A. Subsequent underflows cause the timer to be reloaded from the registers alternately beginning with the register R1B.

The T1 Timer control bits, T1C3, T1C2 and T1C1 set up the timer for PWM mode operation.

Figure 8 shows a block diagram of the timer in PWM mode.

The underflows can be programmed to toggle the T1A output pin. The underflows can also be programmed to generate interrupts.



TL/DD/10830–12

FIGURE 8. Timer in PWM Mode

Timers (Continued)

Underflows from the timer are alternately latched into two pending flags, T1PNDA and T1PNDB. The user must reset these pending flags under software control. Two control enable flags, T1ENA and T1ENB, allow the interrupts from the timer underflow to be enabled or disabled. Setting the timer enable flag T1ENA will cause an interrupt when a timer underflow causes the R1A register to be reloaded into the timer. Setting the timer enable flag T1ENB will cause an interrupt when a timer underflow causes the R1B register to be reloaded into the timer. Resetting the timer enable flags will disable the associated interrupts.

Either or both of the timer underflow interrupts may be enabled. This gives the user the flexibility of interrupting once per PWM period on either the rising or falling edge of the PWM output. Alternatively, the user may choose to interrupt on both edges of the PWM output.

Mode 2. External Event Counter Mode

This mode is quite similar to the processor independent PWM mode described above. The main difference is that the timer, T1, is clocked by the input signal from the T1A pin. The Tx timer control bits, T1C3, T1C2 and T1C1 allow the timer to be clocked either on a positive or negative edge from the T1A pin. Underflows from the timer are latched into the T1PNDA pending flag. Setting the T1ENA control flag will cause an interrupt when the timer underflows.

In this mode the input pin T1B can be used as an independent positive edge sensitive interrupt input if the T1ENB control flag is set. The occurrence of a positive edge on the T1B input pin is latched into the T1PNDB flag.

Figure 9 shows a block diagram of the timer in External Event Counter mode.

Note: The PWM output is not available in this mode since the T1A pin is being used as the counter input clock.

Mode 3. Input Capture Mode

The device can precisely measure external frequencies or time external events by placing the timer block, T1, in the input capture mode.

In this mode, the timer T1 is constantly running at the fixed t_C rate. The two registers, R1A and R1B, act as capture registers. Each register acts in conjunction with a pin. The register R1A acts in conjunction with the T1A pin and the register R1B acts in conjunction with the T1B pin.

The timer value gets copied over into the register when a trigger event occurs on its corresponding pin. Control bits, T1C3, T1C2 and T1C1, allow the trigger events to be specified either as a positive or a negative edge. The trigger condition for each input pin can be specified independently.

The trigger conditions can also be programmed to generate interrupts. The occurrence of the specified trigger condition on the T1A and T1B pins will be respectively latched into the pending flags, T1PNDA and T1PNDB. The control flag T1ENA allows the interrupt on T1A to be either enabled or disabled. Setting the T1ENA flag enables interrupts to be generated when the selected trigger condition occurs on the T1A pin. Similarly, the flag T1ENB controls the interrupts from the T1B pin.

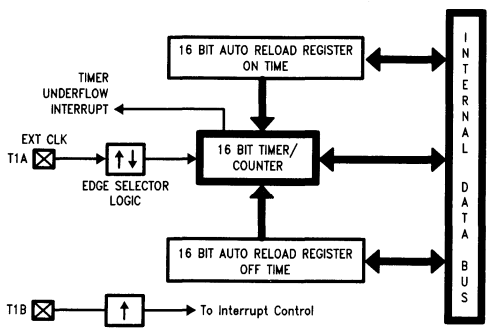
Underflows from the timer can also be programmed to generate interrupts. Underflows are latched into the timer T1C0 pending flag (the T1C0 control bit serves as the timer underflow interrupt pending flag in the Input Capture mode). Consequently, the T1C0 control bit should be reset when entering the Input Capture mode. The timer underflow interrupt is enabled with the T1ENA control flag. When a T1A interrupt occurs in the Input Capture mode, the user must check both the T1PNDA and T1C0 pending flags in order to determine whether a T1A input capture or a timer underflow (or both) caused the interrupt.

Figure 10 shows a block diagram of the timer in Input Capture mode.

TIMER CONTROL FLAGS

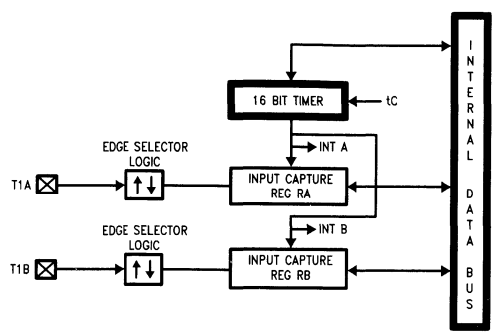
The control bits and their functions are summarized below.

T1C0	Timer Start/Stop control in Modes 1 and 2 (Processor Independent PWM and External Event Counter), where 1 = Start, 0 = Stop Timer Underflow Interrupt Pending Flag in Mode 3 (Input Capture)
T1PNDA	Timer Interrupt Pending Flag
T1PNDB	Timer Interrupt Pending Flag
T1ENA	Timer Interrupt Enable Flag
T1ENB	Timer Interrupt Enable Flag 1 = Timer Interrupt Enabled 0 = Timer Interrupt Disabled
T1C3	Timer mode control
T1C2	Timer mode control
T1C1	Timer mode control



TL/DD/10830–13

FIGURE 9. Timer in External Event Counter Mode



TL/DD/10830–14

FIGURE 10. Timer in Input Capture Mode

Timers (Continued)

The timer mode control bits (T1C3, T1C2 and T1C1) are detailed below:

T1C3	T1C2	T1C1	Timer Mode	Interrupt A Source	Interrupt B Source	Timer Counts On
0	0	0	MODE 2 (External Event Counter)	Timer Underflow	Pos. T1B Edge	T1A Pos. Edge
0	0	1	MODE 2 (External Event Counter)	Timer Underflow	Pos. T1B Edge	T1A Neg. Edge
1	0	1	MODE 1 (PWM) T1A Toggle	Autoreload RA	Autoreload RB	t_C
1	0	0	MODE 1 (PWM) No T1A Toggle	Autoreload RA	Autoreload RB	t_C
0	1	0	MODE 3 (Capture) Captures: T1A Pos. Edge T1B Pos. Edge	Pos. T1A Edge or Timer Underflow	Pos. T1B Edge	t_C
1	1	0	MODE 3 (Capture) Captures: T1A Pos. Edge T1B Neg. Edge	Pos. T1A Edge or Timer Underflow	Neg. T1B Edge	t_C
0	1	1	MODE 3 (Capture) Captures: T1A Neg. Edge T1B Pos. Edge	Neg. T1B Edge or Timer Underflow	Pos. T1B Edge	t_C
1	1	1	MODE 3 (Capture) Captures: T1A Neg. Edge T1B Neg. Edge	Neg. T1A Edge or Timer Underflow	Neg. T1B Edge	t_C

Power Save Modes

The device offers the user two power save modes of operation: HALT and IDLE. In the HALT mode, all microcontroller activities are stopped. In the IDLE mode, the on-board oscillator circuitry the WATCHDOG logic, the Clock Monitor and timer T0 are active but all other microcontroller activities are stopped. In either mode, all on-board RAM, registers, I/O states, and timers (with the exception of T0) are unaltered.

HALT MODE

The device is placed in the HALT mode by writing a "1" to the HALT flag (G7 data bit). All microcontroller activities, including the clock and timers, are stopped. The WATCHDOG logic is disabled during the HALT mode. However, the clock monitor circuitry if enabled remains active and will cause the WATCHDOG output pin (WDOUT) to go low. If the HALT mode is used and the user does not want to activate the WDOUT pin, the Clock Monitor should be disabled after the device comes out of reset (resetting the Clock Monitor control bit with the first write to the WDSVR register). In the HALT mode, the power requirements of the device are minimal and the applied voltage (V_{CC}) may be decreased to V_r (V_r = 2.0V) without altering the state of the machine.

The device supports three different ways of exiting the HALT mode. The first method of exiting the HALT mode is with the Multi-Input Wakeup feature on the L port. The second method is with a low to high transition on the CKO (G7) pin. This method precludes the use of the crystal clock configuration (since CKO becomes a dedicated output), and so may be used with an RC clock configuration. The third method of exiting the HALT mode is by pulling the $\overline{\text{RESET}}$ pin low.

Since a crystal or ceramic resonator may be selected as the oscillator, the Wakeup signal is not allowed to start the chip running immediately since crystal oscillators and ceramic resonators have a delayed start up time to reach full amplitude and frequency stability. The IDLE timer is used to generate a fixed delay to ensure that the oscillator has indeed stabilized before allowing instruction execution. In this case, upon detecting a valid Wakeup signal, only the oscillator circuitry is enabled. The IDLE timer is loaded with a value of 256 and is clocked with the t_C instruction cycle clock. The t_C clock is derived by dividing the oscillator clock down by a factor of 10. The Schmitt trigger following the CKI inverter on the chip ensures that the IDLE timer is clocked only when the oscillator has a sufficiently large amplitude to meet the Schmitt trigger specifications. This Schmitt trigger is not part of the oscillator closed loop. The startup timeout from the IDLE timer enables the clock signals to be routed to the rest of the chip.

Power Save Modes (Continued)

If an RC clock option is being used, the fixed delay is introduced optionally. A control bit, CLKDLY, mapped as configuration bit G7, controls whether the delay is to be introduced or not. The delay is included if CLKDLY is set, and excluded if CLKDLY is reset. The CLKDLY bit is cleared on reset.

The device has two mask options associated with the HALT mode. The first mask option enables the HALT mode feature, while the second mask option disables the HALT mode. With the HALT mode enable mask option, the device will enter and exit the HALT mode as described above. With the HALT disable mask option, the device cannot be placed in the HALT mode (writing a "1" to the HALT flag will have no effect).

The WATCHDOG detector circuit is inhibited during the HALT mode. However, the clock monitor circuit if enabled remains active during HALT mode in order to ensure a clock monitor error if the device inadvertently enters the HALT mode as a result of a runaway program or power glitch.

IDLE MODE

The device is placed in the IDLE mode by writing a "1" to the IDLE flag (G6 data bit). In this mode, all activities, except the associated on-board oscillator circuitry, the WATCHDOG logic, the clock monitor and the IDLE Timer T0, are stopped.

As with the HALT mode, the device can be returned to normal operation with a reset, or with a Multi-Input Wakeup from the L Port. Alternately, the microcontroller resumes normal operation from the IDLE mode when the thirteenth bit (representing 4.096 ms at internal clock frequency of 1 MHz, t_C = 1 μs) of the IDLE Timer toggles.

This toggle condition of the thirteenth bit of the IDLE Timer T0 is latched into the T0PND pending flag.

The user has the option of being interrupted with a transition on the thirteenth bit of the IDLE Timer T0. The interrupt can be enabled or disabled via the T0EN control bit. Setting the T0EN flag enables the interrupt and vice versa.

The user can enter the IDLE mode with the Timer T0 interrupt enabled. In this case, when the T0PND bit gets set, the device will first execute the Timer T0 interrupt service routine and then return to the instruction following the "Enter Idle Mode" instruction.

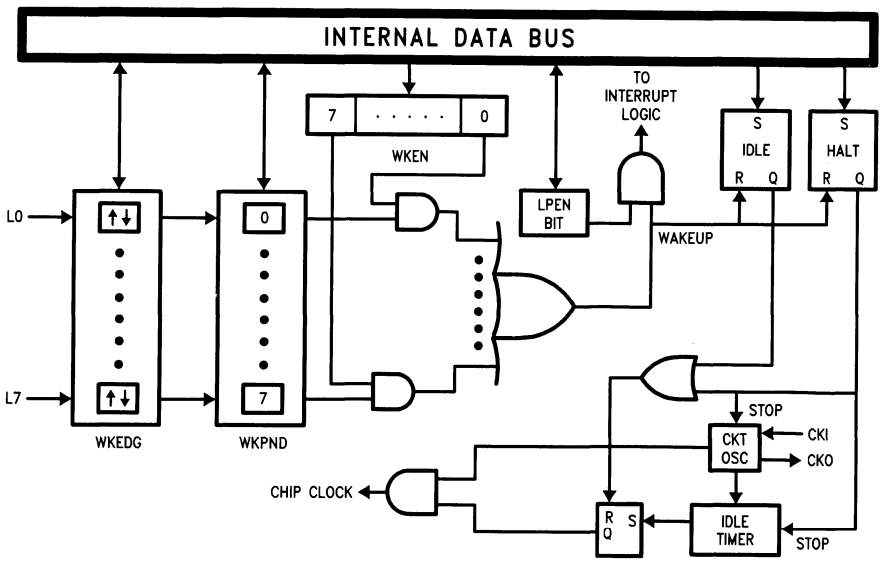
Alternatively, the user can enter the IDLE mode with the IDLE Timer T0 interrupt disabled. In this case, the device will resume normal operation with the instruction immediately following the "Enter IDLE Mode" instruction.

Note: It is necessary to program two NOP instructions following both the set HALT mode and set IDLE mode instructions. These NOP instructions are necessary to allow clock resynchronization following the HALT or IDLE modes.

Multi-Input Wakeup

The Multi-Input Wakeup feature is ued to return (wakeup) the device from either the HALT or IDLE modes. Alternately Multi-Input Wakeup/Interrupt feature may also be used to generate up to 8 edge selectable external interrupts.

Figure 11 shows the Multi-Input Wakeup logic.



TL/DD/10830–15

FIGURE 11. Multi-Input Wake Up Logic

Multi-Input Wakeup (Continued)

The Multi-Input Wakeup feature utilizes the L Port. The user selects which particular L port bit (or combination of L Port bits) will cause the device to exit the HALT or IDLE modes. The selection is done through the Reg: WKEN. The Reg: WKEN is an 8-bit read/write register, which contains a control bit for every L port bit. Setting a particular WKEN bit enables a Wakeup from the associated L port pin.

The user can select whether the trigger condition on the selected L Port pin is going to be either a positive edge (low to high transition) or a negative edge (high to low transition). This selection is made via the Reg: WKEDG, which is an 8-bit control register with a bit assigned to each L Port pin. Setting the control bit will select the trigger condition to be a negative edge on that particular L Port pin. Resetting the bit selects the trigger condition to be a positive edge. Changing an edge select entails several steps in order to avoid a pseudo Wakeup condition as a result of the edge change. First, the associated WKEN bit should be reset, followed by the edge select change in WKEDG. Next, the associated WKPND bit should be cleared, followed by the associated WKEN bit being re-enabled.

An example may serve to clarify this procedure. Suppose we wish to change the edge select from positive (low going high) to negative (high going low) for L Port bit 5, where bit 5 has previously been enabled for an input interrupt. The program would be as follows:

```
RBIT  5, WKEN
SBIT  5, WKEDG
RBIT  5, WKPND
SBIT  5, WKEN
```

If the L port bits have been used as outputs and then changed to inputs with Multi-Input Wakeup/Interrupt, a safety procedure should also be followed to avoid inherited pseudo wakeup conditions. After the selected L port bits have been changed from output to input but before the associated WKEN bits are enabled, the associated edge select bits in WKEDG should be set or reset for the desired edge selects, followed by the associated WKPND bits being cleared.

This same procedure should be used following reset, since the L port inputs are left floating as a result of reset.

The occurrence of the selected trigger condition for Multi-Input Wakeup is latched into a pending register called WKPND. The respective bits of the WKPND register will be set on the occurrence of the selected trigger edge on the corresponding Port L pin. The user has the responsibility of clearing these pending flags. Since WKPND is a pending register for the occurrence of selected wakeup conditions, the device will not enter the HALT mode if any Wakeup bit is both enabled and pending. Consequently, the user has the responsibility of clearing the pending flags before attempting to enter the HALT mode.

WKEN, WKPND and WKEDG are all read/write registers, and are cleared at reset.

PORT L INTERRUPTS

Port L provides the user with an additional eight fully selectable, edge sensitive interrupts which are all vectored into the same service subroutine.

The interrupt from Port L shares logic with the wake up circuitry. The register WKEN allows interrupts from Port L to be individually enabled or disabled. The register WKEDG specifies the trigger condition to be either a positive or a negative edge. Finally, the register WKPND latches in the pending trigger conditions.

The GIE (Global Interrupt Enable) bit enables the interrupt function.

A control flag, LPEN, functions as a global interrupt enable for Port L interrupts. Setting the LPEN flag will enable interrupts and vice versa. A separate global pending flag is not needed since the register WKPND is adequate.

Since Port L is also used for waking the device out of the HALT or IDLE modes, the user can elect to exit the HALT or IDLE modes either with or without the interrupt enabled. If he elects to disable the interrupt, then the device will restart execution from the instruction immediately following the instruction that placed the microcontroller in the HALT or IDLE modes. In the other case, the device will first execute the interrupt service routine and then revert to normal operation.

The Wakeup signal will not start the chip running immediately since crystal oscillators or ceramic resonators have a finite start up time. The IDLE Timer (T0) generates a fixed delay to ensure that the oscillator has indeed stabilized before allowing the device to execute instructions. In this case, upon detecting a valid Wakeup signal, only the oscillator circuitry and the IDLE Timer T0 are enabled. The IDLE Timer is loaded with a value of 256 and is clocked from the t_C instruction cycle clock. The t_C clock is derived by dividing down the oscillator clock by a factor of 10. A Schmitt trigger following the CKI on-chip inverter ensures that the IDLE timer is clocked only when the oscillator has a sufficiently large amplitude to meet the Schmitt trigger specifications. This Schmitt trigger is not part of the oscillator closed loop. The startup timeout from the IDLE timer enables the clock signals to be routed to the rest of the chip.

If the RC clock option is used, the fixed delay is under software control. A control flag, CLKDLY, in the G7 configuration bit allows the clock start up delay to be optionally inserted. Setting CLKDLY flag high will cause clock start up delay to be inserted and resetting it will exclude the clock start up delay. The CLKDLY flag is cleared during reset, so the clock start up delay is not present following reset with the RC clock options.

UART

The device contains a full-duplex software programmable UART. The UART *(Figure 12)* consists of a transmit shift register, a receiver shift register and seven addressable registers, as follows: a transmit buffer register (TBUF), a receiver buffer register (RBUF), a UART control and status register (ENU), a UART receive control and status register (ENUR), a UART interrupt and clock source register (ENUI), a prescaler select register (PSR) and baud (BAUD) register. The ENU register contains flags for transmit and receive functions; this register also determines the length of the data frame (7, 8 or 9 bits), the value of the ninth bit in transmission, and parity selection bits. The ENUR register flags framming, data overrun and parity errors while the UART is receiving.

Other functions of the ENUR register include saving the ninth bit received in the data frame, enabling or disabling the UART's attention mode of operation and providing additional receiver/transmitter status information via RCVG and XMTG bits. The determination of an internal or external clock source is done by the ENUI register, as well as selecting the number of stop bits and enabling or disabling transmit and receive interrupts. A control flag in this register can also select the UART mode of operation: asynchronous or synchronous.

UART CONTROL AND STATUS REGISTERS

The operation of the UART is programmed through three registers: ENU, ENUR and ENUI. The function of the individual bits in these registers is as follows:

ENU-UART Control and Status Register (Address at 0BA)

PEN	PSEL1	XBIT9/ PSEL0	CHL1	CHL0	ERR	RBFL	TBMT
0RW	0RW	0RW	0RW	0RW	0R	0R	1R

Bit 7 Bit 0

ENUR-UART Receive Control and Status Register (Address at 0BB)

DOE	FE	PE	SPARE	RBIT9	ATTN	XMTG	RCVG
0RD	0RD	0RD	0RW*	0R	0RW	0R	0R

Bit7 Bit0

ENUI-UART Interrupt and Clock Source Register (Address at 0BC)

STP2	STP78	ETDX	SSEL	XRCLK	XTCLK	ERI	ETI
0RW	0RW	0RW	0RW	0RW	0RW	0RW	0RW

Bit7 Bit0

*Bit is not used.

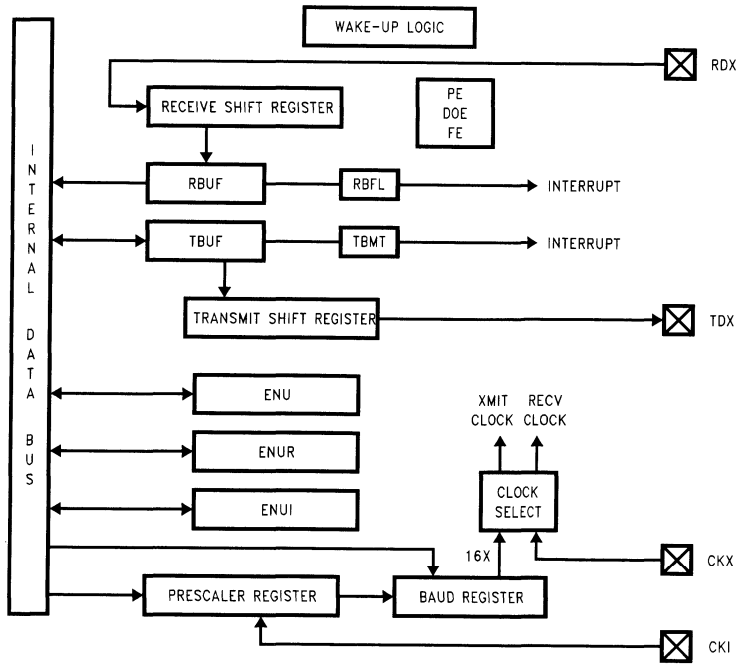
0 Bit is cleared on reset.

1 Bit is set to one on reset.

R Bit is read-only; it cannot be written by software.

RW Bit is read/write.

D Bit is cleared on read; when read by software as a one, it is cleared automatically. Writing to the bit does not affect its state.



TL/DD/10830–16

FIGURE 12. UART Block Diagram

UART (Continued)

DESCRIPTION OF UART REGISTER BITS

ENU—UART CONTROL AND STATUS REGISTER

TBMT: This bit is set when the UART transfers a byte of data from the TBUF register into the TSFT register for transmission. It is automatically reset when software writes into the TBUF register.

RBFL: This bit is set when the UART has received a complete character and has copied it into the RBUF register. It is automatically reset when software reads the character from RBUF.

ERR: This bit is a global UART error flag which gets set if any or a combination of the errors (DOE, FE, PE) occur.

CHL1, CHL0: These bits select the character frame format. Parity is not included and is generated/verified by hardware.

CHL1 = 0, CHL0 = 0 The frame contains eight data bits.
CHL1 = 0, CHL0 = 1 The frame contains seven data bits.
CHL1 = 1, CHL0 = 0 The frame contains nine data bits.
CHL1 = 1, CHL0 = 1 Loopback Mode selected. Transmitter output internally looped back to receiver input. Nine bit framing format is used.

XBIT9/PSEL0: Programs the ninth bit for transmission when the UART is operating with nine data bits per frame. For seven or eight data bits per frame, this bit in conjunction with PSEL1 selects parity.

PSEL1, PSEL0: Parity select bits.

PSEL1 = 0, PSEL0 = 0 Odd Parity (if Parity enabled)
PSEL1 = 0, PSEL0 = 1 Even Parity (if Parity enabled)
PSEL1 = 1, PSEL0 = 0 Mark(1) (if Parity enabled)
PSEL1 = 1, PSEL0 = 1 Space(0) (if Parity enabled)

PEN: This bit enables/disables Parity (7- and 8-bit modes only).

PEN = 0 Parity disabled.
PEN = 1 Parity enabled.

ENUR—UART RECEIVE CONTROL AND STATUS REGISTER

RCVG: This bit is set high whenever a framing error occurs and goes low when RDX goes high.

XMTG: This bit is set to indicate that the UART is transmitting. It gets reset at the end of the last frame (end of last Stop bit).

ATTN: ATTENTION Mode is enabled while this bit is set. This bit is cleared automatically on receiving a character with data bit nine set.

RBIT9: Contains the ninth data bit received when the UART is operating with nine data bits per frame.

SPARE: Reserved for future use

PE: Flags a Parity Error.

PE = 0 Indicates no Parity Error has been detected since the last time the ENUR register was read.
PE = 1 Indicates the occurence of a Parity Error.

FE: Flags a Framing Error.

FE = 0 Indicates no Framing Error has been detected since the last time the ENUR register was read.
FE = 1 Indicates the occurence of a Framing Error.

DOE: Flags a Data Overrun Error.

DOE = 0 Indicates no Data Overrun Error has been detected since the last time the ENUR register was read.
DOE = 1 Indicates the occurence of a Data Overrun Error.

ENUI—UART INTERRUPT AND CLOCK SOURCE REGISTER

ETI: This bit enables/disables interrupt from the transmitter section.

ETI = 0 Interrupt from the transmitter is disabled.
ETI = 1 Interrupt from the transmitter is enabled.

ERI: This bit enables/disables interrupt from the receiver section.

ERI = 0 Interrupt from the receiver is disabled.
ERI = 1 Interrupt from the receiver is enabled.

XTCLK: This bit selects the clock source for the transmitter-section.

XTCLK = 0 The clock source is selected through the PSR and BAUD registers.
XTCLK = 1 Signal on CKX (L1) pin is used as the clock.

XRCLK: This bit selects the clock source for the receiver section.

XRCLK = 0 The clock source is selected through the PSR and BAUD registers.
XRCLK = 1 Signal on CKX (L1) pin is used as the clock.

SSEL: UART mode select.

SSEL = 0 Asynchronous Mode.
SSEL = 1 Synchronous Mode.

ETDX: TDX (UART Transmit Pin) is the alternate function assigned to Port L pin L2; it is selected by setting ETDX bit. To simulate line break generation, software should reset ETDX bit and output logic zero to TDX pin through Port L data and configuration registers.

STP78: This bit is set to program the last Stop bit to be 7/8th of a bit in length.

STP2: This bit programs the number of Stop bits to be transmitted.

STP2 = 0 One Stop bit transmitted.
STP2 = 1 Two Stop bits transmitted.

Associated I/O Pins

Data is transmitted on the TDX pin and received on the RDX pin. TDX is the alternate function assigned to Port L pin L2; it is selected by setting ETDX (in the ENUI register) to one. RDX is an inherent function of Port L pin L3, requiring no setup.

The baud rate clock for the UART can be generated on-chip, or can be taken from an external source. Port L pin L1 (CKX) is the external clock I/O pin. The CKX pin can be either an input or an output, as determined by Port L Configuration and Data registers (Bit 1). As an input, it accepts a clock signal which may be selected to drive the transmitter and/or receiver. As an output, it presents the internal Baud Rate Generator output.

UART Operation

The UART has two modes of operation: asynchronous mode and synchronous mode.

ASYNCHRONOUS MODE

This mode is selected by resetting the SSEL (in the ENUI register) bit to zero. The input frequency to the UART is 16 times the baud rate.

The TSFT and TBUF registers double-buffer data for transmission. While TSFT is shifting out the current character on the TDX pin, the TBUF register may be loaded by software with the next byte to be transmitted. When TSFT finishes transmitting the current character the contents of TBUF are transferred to the TSFT register and the Transmit Buffer Empty Flag (TBMT in the ENU register) is set. The TBMT flag is automatically reset by the UART when software loads a new character into the TBUF register. There is also the XMTG bit which is set to indicate that the UART is transmitting. This bit gets reset at the end of the last frame (end of last Stop bit). TBUF is a read/write register.

The RSFT and RBUF registers double-buffer data being received. The UART receiver continually monitors the signal on the RDX pin for a low level to detect the beginning of a Start bit. Upon sensing this low level, it waits for half a bit time and samples again. If the RDX pin is still low, the receiver considers this to be a valid Start bit, and the remaining bits in the character frame are each sampled a single time, at the mid-bit position. Serial data input on the RDX pin is shifted into the RSFT register. Upon receiving the complete character, the contents of the RSFT register are copied into the RBUF register and the Received Buffer Full Flag (RBFL) is set. RBFL is automatically reset when software reads the character from the RBUF register. RBUF is a read only register. There is also the RCVG bit which is set high when a framing error occurs and goes low once RDX goes high. TBMT, XMTG, RBFL and RCVG are read only bits.

SYNCHRONOUS MODE

In this mode data is transferred synchronously with the clock. Data is transmitted on the rising edge and received on the falling edge of the synchronous clock.

This mode is selected by setting SSEL bit in the ENUI register. The input frequency to the UART is the same as the baud rate.

When an external clock input is selected at the CKX pin, data transmit and receive are performed synchronously with this clock through TDX/RDX pins.

If data transmit and receive are selected with the CKX pin as clock output, the μC generates the synchronous clock output at the CKX pin. The internal baud rate generator is used to produce the synchronous clock. Data transmit and receive are performed synchronously with this clock.

FRAMING FORMATS

The UART supports several serial framing formats *(Figure 13)*. The format is selected using control bits in the ENU, ENUR and ENUI registers.

The first format (1, 1a, 1b, 1c) for data transmission (CHL0 = 1, CHL1 = 0) consists of Start bit, seven Data bits (excluding parity) and 7/8, one or two Stop bits. In applications using parity, the parity bit is generated and verified by hardware.

The second format (CHL0 = 0, CHL1 = 0) consists of one Start bit, eight Data bits (excluding parity) and 7/8, one or two Stop bits. Parity bit is generated and verified by hardware.

The third format for transmission (CHL0 = 0, CHL1 = 1) consists of one Start bit, nine Data bits and 7/8, one or two Stop bits. This format also supports the UART "ATTENTION" feature. When operating in this format, all eight bits of TBUF and RBUF are used for data. The ninth data bit is transmitted and received using two bits in the ENU and ENUR registers, called XBIT9 and RBIT9. RBIT9 is a read only bit. Parity is not generated or verified in this mode.

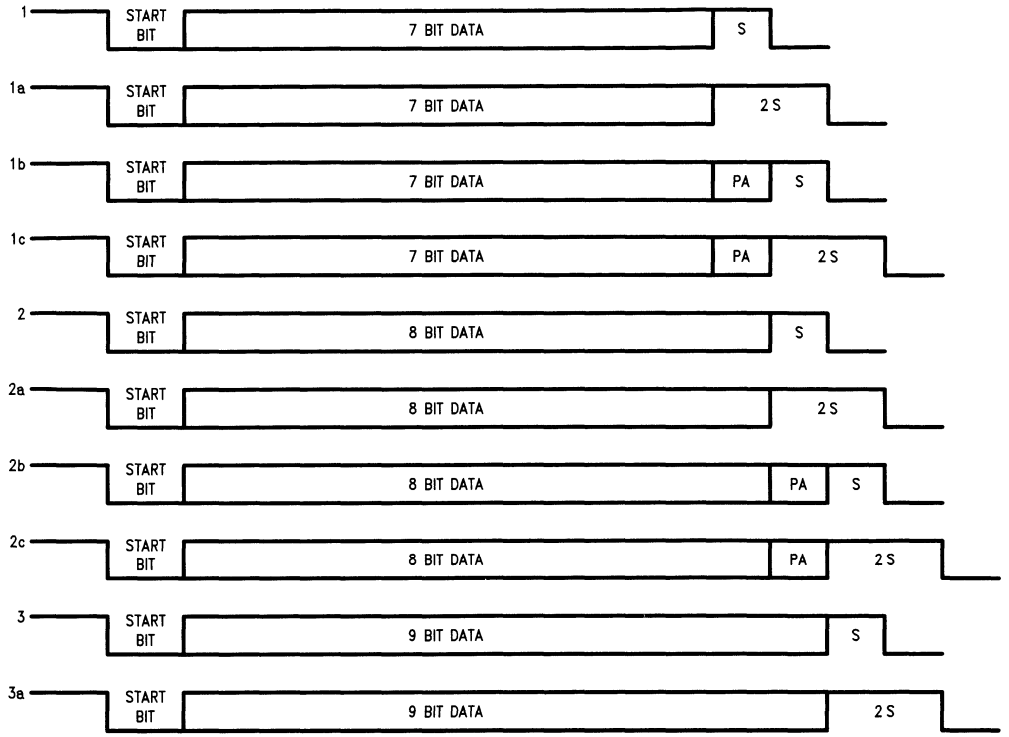
For any of the above framing formats, the last Stop bit can be programmed to be 7/8th of a bit in length. If two Stop bits are selected and the 7/8th bit is set (selected), the second Stop bit will be 7/8th of a bit in length.

The parity is enabled/disabled by PEN bit located in the ENU register. Parity is selected for 7- and 8-bit modes only. If parity is enabled (PEN = 1), the parity selection is then performed by PSEL0 and PSEL1 bits located in the ENU register.

Note that the XBIT9/PSEL0 bit located in the ENU register serves two mutually exclusive functions. This bit programs the ninth bit for transmission when the UART is operating with nine data bits per frame. There is no parity selection in this framing format. For other framing formats XBIT9 is not needed and the bit is PSEL0 used in conjunction with PSEL1 to select parity.

The frame formats for the receiver differ from the transmitter in the number of Stop bits required. The receiver only requires one Stop bit in a frame, regardless of the setting of the Stop bit selection bits in the control register. Note that an implicit assumption is made for full duplex UART operation that the framing formats are the same for the transmitter and receiver.

UART Operation (Continued)



TL/DD/10830–17

FIGURE 13. Framing Formats

UART INTERRUPTS

The UART is capable of generating interrupts. Interrupts are generated on Receive Buffer Full and Transmit Buffer Empty. Both interrupts have individual interrupt vectors. Two bytes of program memory space are reserved for each interrupt vector. The two vectors are located at addresses 0xEC to 0xEF Hex in the program memory space. The interrupts can be individually enabled or disabled using Enable Transmit Interrupt (ETI) and Enable Receive Interrupt (ERI) bits in the ENUI register.

The interrupt from the Transmitter is set pending, and remains pending, as long as both the TBMT and ETI bits are set. To remove this interrupt, software must either clear the ETI bit or write to the TBUF register (thus clearing the TBMT bit).

The interrupt from the receiver is set pending, and remains pending, as long as both the RBFL and ERI bits are set. To remove this interrupt, software must either clear the ERI bit or read from the RBUF register (thus clearing the RBFL bit).

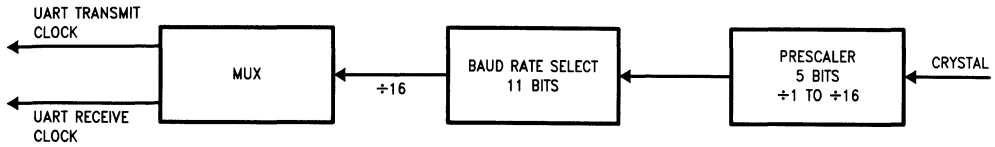
Baud Clock Generation

The clock inputs to the transmitter and receiver sections of the UART can be individually selected to come either from an external source at the CKX pin (port L, pin L1) or from a source selected in the PSR and BAUD registers. Internally, the basic baud clock is created from the oscillator frequency through a two-stage divider chain consisting of a 1–16 (increments of 0.5) prescaler and an 11-bit binary counter. *(Figure 14)* The divide factors are specified through two read/write registers shown in *Figure 15*. Note that the 11-bit Baud Rate Divisor spills over into the Prescaler Select Register (PSR). PSR is cleared upon reset.

As shown in Table I, a Prescaler Factor of 0 corresponds to NO CLOCK. NO CLOCK condition is the UART power down mode where the UART clock is turned off for power saving purpose. The user must also turn the UART clock off when a different baud rate is chosen.

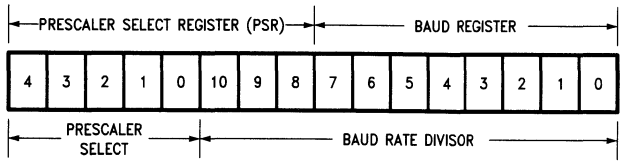
The correspondences between the 5-bit Prescaler Select and Prescaler factors are shown in Table I. Therer are many ways to calculate the two divisor factors, but one particularly effective method would be to achieve a 1.8432 MHz frequency coming out of the first stage. The 1.8432 MHz prescaler output is then used to drive the software programmable baud rate counter to create a x16 clock for the following baud rates: 110, 134.5, 150, 300, 600, 1200, 1800, 2400, 3600, 4800, 7200, 9600, 19200 and 38400 (Table II). Other baud rates may be created by using appropriate divisors. The x16 clock is then divided by 16 to provide the rate for the serial shift registers of the transmitter and receiver.

Baud Clock Generation (Continued)



TL/DD/10830–18

FIGURE 14. UART BAUD Clock Generation



TL/DD/10830–19

FIGURE 15. UART BAUD Clock Divisor Registers

TABLE I. Prescaler Factors

Prescaler Select	Prescaler Factor
00000	NO CLOCK
00001	1
00010	1.5
00011	2
00100	2.5
00101	3
00110	3.5
00111	4
01000	4.5
01001	5
01010	5.5
01011	6
01100	6.5
01101	7
01110	7.5
01111	8
10000	8.5
10001	9
10010	9.5
10011	10
10100	10.5
10101	11
10110	11.5
10111	12
11000	12.5
11001	13
11010	13.5
11011	14
11100	14.5
11101	15
11110	15.5
11111	16

TABLE II. Baud Rate Divisors (1.8432 MHz Prescaler Output)

Baud Rate	Baud Rate Divisor − 1 (N-1)
110 (110.03)	1046
134.5 (134.58)	855
150	767
300	383
600	191
1200	95
1800	63
2400	47
3600	31
4800	23
7200	15
9600	11
19200	5
38400	2

The entries in Table II assume a prescaler output of 1.8432 MHz. In the asynchronous mode the baud rate could be as high as 625k.

As an example, considering the Asynchronous Mode and a CKI clock of 4.608 MHz, the prescaler factor selected is:

$$4.608/1.8432 = 2.5$$

The 2.5 entry is available in Table I. The 1.8432 MHz prescaler output is then used with proper Baud Rate Divisor (Table II) to obtain different baud rates. For a baud rate of 19200 e.g., the entry in Table II is 5.

$N - 1 = 5$ ($N - 1$ is the value from Table II)

$N = 6$ (N is the Baud Rate Divisor)

$$\text{Baud Rate} = 1.8432\ \text{MHz}/(16 \times 6) = 19200$$

The divide by 16 is performed because in the asynchronous mode, the input frequency to the UART is 16 times the baud rate. The equation to calculate baud rates is given below.

The actual Baud Rate may be found from:

$$BR = Fc/(16 \times N \times P)$$

Baud Clock Generation (Continued)

Where:

BR is the Baud Rate

Fc is the CKI frequency

N is the Baud Rate Divisor (Table II).

P is the Prescaler Divide Factor selected by the value in the Prescaler Select Register (Table I)

Note: In the Synchronous Mode, the divisor 16 is replaced by two if internal Baud Rate generator is used. Replaced by one if external clock is used.

Example:

Asynchronous Mode:

Crystal Frequency = 5 MHz

Desired baud rate = 9600

Using the above equation N × P can be calculated first.

$$N \times P = (5 \times 10^6)/(16 \times 9600) = 32.552$$

Now 32.552 is divided by each Prescaler Factor (Table II) to obtain a value closest to an integer. This factor happens to be 6.5 (P = 6.5).

$$N = 32.552/6.5 = 5.008 \ (N = 5)$$

The programmed value (from Table II) should be 4 (N − 1).

Using the above values calculated for N and P:

$$BR = (5 \times 10^6)/(16 \times 5 \times 6.5) = 9615.384$$

$$\% \text{ error} = (9615.385 - 9600)/9600 = 0.16$$

Effect of HALT/IDLE

The UART logic is reinitialized when either the HALT or IDLE modes are entered. This reinitialization sets the TBMT flag and resets all read only bits in the UART control and status registers. Read/Write bits remain unchanged. The Transmit Buffer (TBUF) is not affected, but the Transmit Shift register (TSFT) bits are set to one. The receiver registers RBUF and RSFT are not affected.

The μC will exit from the HALT/IDLE modes when the Start bit of a character is detected at the RDX (L3) pin. This feature is obtained by using the Multi-Input Wakeup scheme provided on the μC.

Before entering the HALT or IDLE modes the user program must select the Wakeup source to be on the RDX pin. This selection is done by setting bit 3 of WKEN (Wakeup Enable) register. The Wakeup trigger condition is then selected to be high to low transition. This is done via the WKEDG register (Bit 3 is zero.)

If the microcontroller is halted and crystal oscillator is used, the Wakeup signal will not start the chip running immediately because of the finite start up time requirement of the crystal oscillator. The idle timer (T0) generates a fixed delay to ensure that the oscillator has indeed stabilized before allowing the μC to execute code. The user has to consider this delay when data transfer is expected immediately after exiting the HALT mode.

Diagnostic

Bits CHARL0 and CHARL1 in the ENU register provide a loopback feature for diagnostic testing of the UART. When these bits are set to one, the following occur: The receiver input pin (RDX) is internally connected to the transmitter output pin (TDX); the output of the Transmitter Shift Register is "looped back" into the Receive Shift Register input. In this mode, data that is transmitted is immediately received. This feature allows the processor to verify the transmit and receive data paths of the UART.

Note that the framing format for this mode is the nine bit format; one Start bit, nine data bits, and 7/8, one or two Stop bits. Parity is not generated or verified in this mode.

Attention Mode

The UART Receiver section supports an alternate mode of operation, referred to as ATTENTION Mode. This mode of operation is selected by the ATTN bit in the ENUR register. The data format for transmission must also be selected as having nine Data bits and either 7/8, one or two Stop bits.

The ATTENTION mode of operation is intended for use in networking the COP888CS with other processors. Typically in such environments the messages consists of device addresses, indicating which of several destinations should receive them, and the actual data. This Mode supports a scheme in which addresses are flagged by having the ninth bit of the data field set to a 1. If the ninth bit is reset to a zero the byte is a Data byte.

While in ATTENTION mode, the UART monitors the communication flow, but ignores all characters until an address character is received. Upon receiving an address character, the UART signals that the character is ready by setting the RBFL flag, which in turn interrupts the processor if UART Receiver interrupts are enabled. The ATTN bit is also cleared automatically at this point, so that data characters as well as address characters are recognized. Software examines the contents of the RBUF and responds by deciding either to accept the subsequent data stream (by leaving the ATTN bit reset) or to wait until the next address character is seen (by setting the ATTN bit again).

Operation of the UART Transmitter is not affected by selection of this Mode. The value of the ninth bit to be transmitted is programmed by setting XBIT9 appropriately. The value of the ninth bit received is obtained by reading RBIT9. Since this bit is located in ENUR register where the error flags reside, a bit operation on it will reset the error flags.

Comparator

The device contains one differential comparator, with a pair of inputs (positive and negative) and an output. Ports I1–I3 are used for the comparator. The following is the Port I assignment:

- I1 Comparator1 negative input
- I2 Comparator1 positive input
- I3 Comparator1 output

A Comparator Select Register (CMPSL) is used to enable the comparators, read the outputs of the comparator internally, and enable the output of the comparator to the pins. Two control bits (enable and output enable) and one result bit are associated with the comparator. The comparator result bit (CMP1RD) is read only bit which will read as zero if the comparator is not enabled. The Comparator Select Register is cleared with reset, resulting in the comparator being disabled. The comparator should also be disabled before entering either the HALT or IDLE modes in order to save power. The configuration of the CMPSL register is as follows:

Comparator (Continued)

CMPSL REGISTER (ADDRESS X'00B7)

The CMPSL register contains the following bits:

CMP1EN Enable comparator 1

CMP1RD Comparator 1 result (this is a read only bit, which will read as 0 if the comparator is not enabled)

CMP10E Selects pin I3 as comparator 1 output provided that CMPIEN is set to enable the comparator

Unused	Unused	Unused	Unused	CMP10E	CMP1RD	CMP1EN	Unused
Bit 7							Bit 0

Comparator outputs have the same spec as Ports L and G except that the rise and fall times are symmetrical.

Interrupts

The device supports a vectored interrupt scheme. It supports a total of fourteen interrupt sources. The following table lists all the possible interrupt sources, their arbitration ranking and the memory locations reserved for the interrupt vector for each source.

Two bytes of program memory space are reserved for each interrupt source. All interrupt sources except the software interrupt are maskable. Each of the maskable interrupts have an Enable bit and a Pending bit. A maskable interrupt is active if its associated enable and pending bits are set. If GIE = 1 and an interrupt is active, then the processor will be interrupted as soon as it is ready to start executing an instruction except if the above conditions happen during the Software Trap service routine. This exception is described in the Software Trap sub-section.

The interruption process is accomplished with the INTR instruction (opcode 00), which is jammed inside the Instruction Register and replaces the opcode about to be executed. The following steps are performed for every interrupt:

1. The GIE (Global Interrupt Enable) bit is reset.
2. The address of the instruction about to be executed is pushed into the stack.
3. The PC (Program Counter) branches to address 00FF. This procedure takes 7 t_C cycles to execute.

At this time, since GIE = 0, other maskable interrupts are disabled. The user is now free to do whatever context switching is required by saving the context of the machine in the stack with PUSH instructions. The user would then program a VIS (Vector Interrupt Select) instruction in order to branch to the interrupt service routine of the highest priority interrupt enabled and pending at the time of the VIS. Note that this is not necessarily the interrupt that caused the branch to address location 00FF Hex prior to the context switching.

Thus, if an interrupt with a higher rank than the one which caused the interruption becomes active before the decision of which interrupt to service is made by the VIS, then the interrupt with the higher rank will override any lower ones and will be acknowledged. The lower priority interrupt(s) are still pending, however, and will cause another interrupt immediately following the completion of the interrupt service routine associated with the higher priority interrupt just serviced. This lower priority interrupt will occur immediately following the RETI (Return from Interrupt) instruction at the end of the interrupt service routine just completed.

Inside the interrupt service routine, the associated pending bit has to be cleared by software. The RETI (Return from Interrupt) instruction at the end of the interrupt service rou-

Arbitration Ranking	Source	Description	Vector Address Hi-Low Byte
(1) Highest	Software	INTR Instruction	0yFE–0yFF
	Reserved	for Future Use	0yFC–0yFD
(2)	External	Pin G0 Edge	0yFA–0yFB
(3)	Timer T0	Underflow	0yF8–0yF9
(4)	Timer T1	T1A/Underflow	0yF6–0yF7
(5)	Timer T1	T1B	0yF4–0yF5
(6)	MICROWIRE/PLUS	BUSY Goes Low	0yF2–0yF3
	Reserved	for Future Use	0yF0–0yF1
(7)	UART	Receive	0yEE–0yEF
(8)	UART	Transmit	0yEC–0yED
(9)	Reserved		0yEA–0yEB
(10)	Reserved		0yE8–0yE9
(11)	Reserved		0yE6–0yE7
(12)	Reserved		0yE4–0yE5
(13)	Port L/Wakeup	Port L Edge	0yE2–0yE3
(14) Lowest	Default	VIS Instr. Execution without Any Interrupts	0yE0–0yE1

y is VIS page, y ≠ 0.

Interrupts (Continued)

tine will set the GIE (Global Interrupt Enable) bit, allowing the processor to be interrupted again if another interrupt is active and pending.

The VIS instruction looks at all the active interrupts at the time it is executed and performs an indirect jump to the beginning of the service routine of the one with the highest rank.

The addresses of the different interrupt service routines, called vectors, are chosen by the user and stored in ROM in a table starting at 01E0 (assuming that VIS is located between 00FF and 01DF). The vectors are 15-bit wide and therefore occupy 2 ROM locations.

VIS and the vector table must be located in the same 256-byte block (0y00 to 0yFF) except if VIS is located at the last address of a block. In this case, the table must be in the next block. The vector table cannot be inserted in the first 256-byte block (y $\neq$ 0).

The vector of the maskable interrupt with the lowest rank is located at 0yE0 (Hi-Order byte) and 0yE1 (Lo-Order byte) and so forth in increasing rank number. The vector of the maskable interrupt with the highest rank is located at 0yFA (Hi-Order byte) and 0yFB (Lo-Order byte).

The Software Trap has the highest rank and its vector is located at 0yFE and 0yFF.

If, by accident, a VIS gets executed and no interrupt is active, then the PC (Program Counter) will branch to a vector located at 0yE0–0yE1. This vector can point to the Software Trap (ST) interrupt service routine, or to another special service routine as desired.

If, by accident, a VIS gets executed and no interrupt is active, then the PC (Program Counter) will branch to a vector located at 0yE0–0yE1.

WARNING

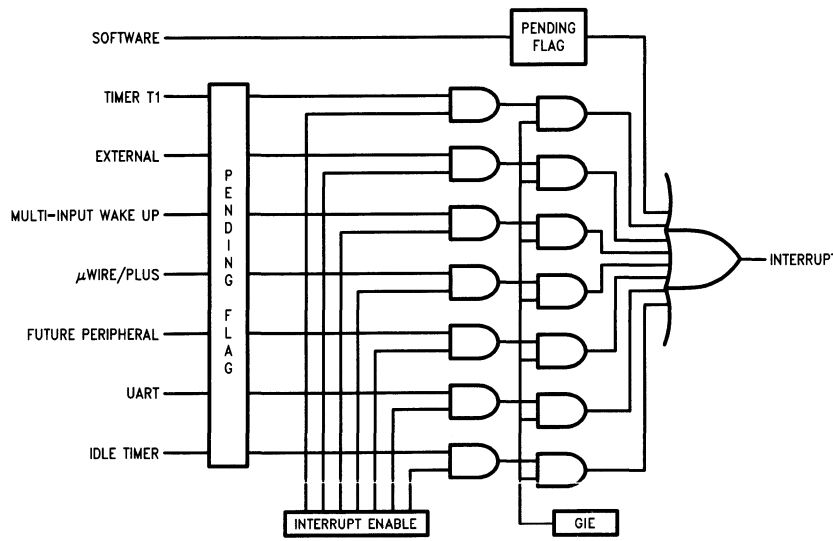
A Default VIS interrupt handler routine must be present. As a minimum, this handler should confirm that the GIE bit is cleared (this indicates that the interrupt sequence has been taken), take care of any required housekeeping, restore context and return. Some sort of Warm Restart procedure should be implemented. These events can occur without any error on the part of the system designer or programmer.

Note: There is always the possibility of an interrupt occurring during an instruction which is attempting to reset the GIE bit or any other interrupt enable bit. If this occurs when a single cycle instruction is being used to reset the interrupt enable bit, the interrupt enable bit will be reset but an interrupt may still occur. This is because interrupt processing is started at the same time as the interrupt bit is being reset. To avoid this scenario, the user should always use a two, three, or four cycle instruction to reset interrupt enable bits.

Figure 16 shows the Interrupt block diagram.

SOFTWARE TRAP

The Software Trap (ST) is a special kind of non-maskable interrupt which occurs when the INTR instruction (used to acknowledge interrupts) is fetched from ROM and placed inside the instruction register. This may happen when the PC is pointing beyond the available ROM address space or when the stack is over-popped.



TL/DD/10830–20

FIGURE 16. Interrupt Block Diagram

Interrupts (Continued)

When an ST occurs, the user can re-initialize the stack pointer and do a recovery procedure (similar to reset, but not necessarily containing all of the same initialization procedures) before restarting.

The occurrence of an ST is latched into the ST pending bit. The GIE bit is not affected and the ST pending bit **(not accessible by the user)** is used to inhibit other interrupts and to direct the program to the ST service routine with the VIS instruction. The RPND instruction is used to clear the software interrupt pending bit. This pending bit is also cleared on reset.

The ST has the highest rank among all interrupts.

Nothing (except another ST) can interrupt an ST being serviced.

WATCHDOG

The device contains a WATCHDOG and clock monitor. The WATCHDOG is designed to detect the user program getting stuck in infinite loops resulting in loss of program control or "runaway" programs. The Clock Monitor is used to detect the absence of a clock or a very slow clock below a specified rate on the CKI pin.

The WATCHDOG consists of two independent logic blocks: WD UPPER and WD LOWER. WD UPPER establishes the upper limit on the service window and WD LOWER defines the lower limit of the service window.

Servicing the WATCHDOG consists of writing a specific value to a WATCHDOG Service Register named WDSVR which is memory mapped in the RAM. This value is composed of three fields, consisting of a 2-bit Window Select, a 5-bit Key Data field, and the 1-bit Clock Monitor Select field. Table III shows the WDSVR register.

The lower limit of the service window is fixed at 2048 instruction cycles. Bits 7 and 6 of the WDSVR register allow the user to pick an upper limit of the service window.

Table IV shows the four possible combinations of lower and upper limits for the WATCHDOG service window. This flexibility in choosing the WATCHDOG service window prevents any undue burden on the user software.

Bits 5, 4, 3, 2 and 1 of the WDSVR register represent the 5-bit Key Data field. The key data is fixed at 01100. Bit 0 of the WDSVR Register is the Clock Monitor Select bit.

TABLE III. WATCHDOG Service Register (WDSVR)

Window Select		Key Data					Clock Monitor
X	X	0	1	1	0	0	Y
7	6	5	4	3	2	1	0

TABLE IV. WATCHDOG Service Window Select

WDSVR Bit 7	WDSVR Bit 6	Service Window (Lower-Upper Limits)
0	0	2k–8k t_C Cycles
0	1	2k–16k t_C Cycles
1	0	2k–32k t_C Cycles
1	1	2k–64k t_C Cycles

Clock Monitor

The Clock Monitor aboard the device can be selected or deselected under program control. The Clock Monitor is guaranteed not to reject the clock if the instruction cycle clock ($1/t_C$) is greater or equal to 10 kHz. This equates to a clock input rate on CKI of greater or equal to 100 kHz.

WATCHDOG Operation

The WATCHDOG and Clock Monitor are disabled during reset. The device comes out of reset with the WATCHDOG armed, the WATCHDOG Window Select bits (bits 6, 7 of the WDSVR Register) set, and the Clock Monitor bit (bit 0 of the WDSVR Register) enabled. Thus, a Clock Monitor error will occur after coming out of reset, if the instruction cycle clock frequency has not reached a minimum specified value, including the case where the oscillator fails to start.

The WDSVR register can be written to only once after reset and the key data (bits 5 through 1 of the WDSVR Register) must match to be a valid write. This write to the WDSVR register involves two irrevocable choices: (i) the selection of the WATCHDOG service window (ii) enabling or disabling of the Clock Monitor. Hence, the first write to WDSVR Register involves selecting or deselecting the Clock Monitor, select the WATCHDOG service window and match the WATCHDOG key data. Subsequent writes to the WDSVR register will compare the value being written by the user to the WATCHDOG service window value and the key data (bits 7 through 1) in the WDSVR Register. Table V shows the sequence of events that can occur.

The user must service the WATCHDOG at least once before the upper limit of the service window expires. The WATCHDOG may not be serviced more than once in every lower limit of the service window. The user may service the WATCHDOG as many times as wished in the time period between the lower and upper limits of the service window. The first write to the WDSVR Register is also counted as a WATCHDOG service.

The WATCHDOG has an output pin associated with it. This is the WDOUT pin, on pin 1 of the port G. WDOUT is active low. The WDOUT pin is in the high impedance state in the inactive state. Upon triggering the WATCHDOG, the logic will pull the WDOUT (G1) pin low for an additional 16 t_C–32 t_C cycles after the signal level on WDOUT pin goes below the lower Schmitt trigger threshold. After this delay, the device will stop forcing the WDOUT output low.

The WATCHDOG service window will restart when the WDOUT pin goes high. It is recommended that the user tie the WDOUT pin back to V_{CC} through a resistor in order to pull WDOUT high.

A WATCHDOG service while the WDOUT signal is active will be ignored. The state of the WDOUT pin is not guaranteed on reset, but if it powers up low then the WATCHDOG will time out and WDOUT will enter high impedance state.

The Clock Monitor forces the G1 pin low upon detecting a clock frequency error. The Clock Monitor error will continue until the clock frequency has reached the minimum specified value, after which the G1 output will enter the high impedance TRI-STATE mode following 16 t_C–32 t_C clock cycles. The Clock Monitor generates a continual Clock Monitor error if the oscillator fails to start, or fails to reach the minimum specified frequency. The specification for the Clock Monitor is as follows:

$1/t_C > 10$ kHz—No clock rejection.

$1/t_C < 10$ Hz—Guaranteed clock rejection.

WATCHDOG AND CLOCK MONITOR SUMMARY

The following salient points regarding the device WATCHDOG and CLOCK MONITOR should be noted:

- Both the WATCHDOG and Clock Monitor detector circuits are inhibited during RESET.
- Following RESET, the WATCHDOG and CLOCK MONITOR are both enabled, with the WATCHDOG having the maximum service window selected.
- The WATCHDOG service window and Clock Monitor enable/disable option can only be changed once, during the initial WATCHDOG service following RESET.
- The initial WATCHDOG service must match the key data value in the WATCHDOG Service register WDSVR in order to avoid a WATCHDOG error.
- Subsequent WATCHDOG services must match all three data fields in WDSVR in order to avoid WATCHDOG errors.
- The correct key data value cannot be read from the WATCHDOG Service register WDSVR. Any attempt to read this key data value of 01100 from WDSVR will read as key data value of all 0's.
- The WATCHDOG detector circuit is inhibited during both the HALT and IDLE modes.
- The Clock Monitor detector circuit is active during both the HALT and IDLE modes. Consequently, the device inadvertently entering the HALT mode wil be detected as a Clock Monitor error (provided that the Clock Monitor enable option has been selected by the program).
- With the single-pin R/C oscillator mask option selected and the CLKDLY bit reset, the WATCHDOG service window will resume following HALT mode from where it left off before entering the HALT mode.
- With the crystal oscillator mask option selected, or with the single-pin R/C oscillator mask option selected and the CLKDLY bit set, the WATCHDOG service window will

WATCHDOG Operation (Continued)

be set to its selected value from WDSVR following HALT. Consequently, the WATCHDOG should not be serviced for at least 2048 instruction cycles following HALT, but must be serviced within the selected window to avoid a WATCHDOG error.

- The IDLE timer T0 is not initialized with RESET.
- The user can sync in to the IDLE counter cycle with an IDLE counter (T0) interrupt or by monitoring the T0PND flag. The T0PND flag is set whenever the thirteenth bit of the IDLE counter toggles (every 4096 instruction cycles). The user is responsible for resetting the T0PND flag.
- A hardware WATCHDOG service occurs just as the device exits the IDLE mode. Consequently, the WATCHDOG should not be serviced for at least 2048 instruction cycles following IDLE, but must be serviced within the selected window to avoid a WATCHDOG error.
- Following RESET, the initial WATCHDOG service (where the service window and the CLOCK MONITOR enable/disable must be selected) may be programmed anywhere within the maximum service window (65,536 instruction cycles) initialized by RESET. Note that this initial WATCHDOG service may be programmed within the initial 2048 instruction cycles without causing a WATCHDOG error.

Detection of Illegal Conditions

The device can detect various illegal conditions resulting from coding errors, transient noise, power supply voltage drops, runaway programs, etc.

Reading of undefined ROM gets zeros. The opcode for software interrupt is zero. If the program fetches instructions from undefined ROM, this will force a software interrupt, thus signaling that an illegal condition has occurred.

The subroutine stack grows down for each call (jump to subroutine), interrupt, or PUSH, and grows up for each return or POP. The stack pointer is initialized to RAM location 06F Hex during reset. Consequently, if there are more returns than calls, the stack pointer will point to addresses 070 and 071 Hex (which are undefined RAM). Undefined RAM from addresses 070 to 07F (Segment 0), 140 to 17F (Segment 1), and all other segments (i.e., Segments 3 . . . etc.) is read as all 1's, which in turn will cause the program to return to address 7FFF Hex. This is an undefined ROM location and the instruction fetched (all 0's) from this location will generate a software interrupt signaling an illegal condition.

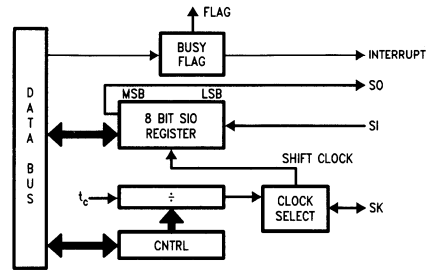
Thus, the chip can detect the following illegal conditions:

a. Executing from undefined ROM
b. Over "POP"ing the stack by having more returns than calls.

When the software interrupt occurs, the user can re-initialize the stack pointer and do a recovery procedure before restarting (this recovery program is probably similar to that following reset, but might not contain the same program initialization procedures). The recovery program should reset the software interrupt pending bit using the RPND instruction.

MICROWIRE/PLUS

MICROWIRE/PLUS is a serial synchronous communications interface. The MICROWIRE/PLUS capability enables the device to interface with any of National Semiconductor's MICROWIRE peripherals (i.e. A/D converters, display drivers, E^2PROMs etc.) and with other microcontrollers which support the MICROWIRE interface. It consists of an 8-bit serial shift register (SIO) with serial data input (SI), serial data output (SO) and serial shift clock (SK). *Figure 17* shows a block diagram of the MICROWIRE/PLUS logic.



TL/DD/10830–21

FIGURE 17. MICROWIRE/PLUS Block Diagram

The shift clock can be selected from either an internal source or an external source. Operating the MICROWIRE/PLUS arrangement with the internal clock source is called the Master mode of operation. Similarly, operating the MICROWIRE/PLUS arrangement with an external shift clock is called the Slave mode of operation.

The CNTRL register is used to configure and control the MICROWIRE/PLUS mode. To use the MICROWIRE/PLUS, the MSEL bit in the CNTRL register is set to one. In the master mode, the SK clock rate is selected by the two bits, SL0 and SL1, in the CNTRL register. Table VI details the different clock rates that may be selected.

TABLE V. WATCHDOG Service Actions

Key Data	Window Data	Clock Monitor	Action
Match	Match	Match	Valid Service: Restart Service Window
Don't Care	Mismatch	Don't Care	Error: Generate WATCHDOG Output
Mismatch	Don't Care	Don't Care	Error: Generate WATCHDOG Output
Don't Care	Don't Care	Mismatch	Error: Generate WATCHDOG Output

TABLE VI. MICROWIRE/PLUS
Master Mode Clock Select

SL1	SL0	SK
0	0	$2 \times t_c$
0	1	$4 \times t_c$
1	x	$8 \times t_c$

Where t_c is the instruction cycle clock

MICROWIRE/PLUS (Continued)

MICROWIRE/PLUS OPERATION

Setting the BUSY bit in the PSW register causes the MICROWIRE/PLUS to start shifting the data. It gets reset when eight data bits have been shifted. The user may reset the BUSY bit by software to allow less than 8 bits to shift. If enabled, an interrupt is generated when eight data bits have been shifted. The device may enter the MICROWIRE/PLUS mode either as a Master or as a Slave. *Figure 14* shows how two COP888CS microcontrollers and several peripherals may be interconnected using the MICROWIRE/PLUS arrangements.

Warning:

The SIO register should only be loaded when the SK clock is low. Loading the SIO register while the SK clock is high will result in undefined data in the SIO register. SK clock is normally low when not shifting.

Setting the BUSY flag when the input SK clock is high in the MICROWIRE/PLUS slave mode may cause the current SK clock for the SIO shift register to be narrow. For safety, the BUSY flag should only be set when the input SK clock is low.

MICROWIRE/PLUS Master Mode Operation

In the MICROWIRE/PLUS Master mode of operation the shift clock (SK) is generated internally. The MICROWIRE Master always initiates all data exchanges. The MSEL bit in the CNTRL register must be set to enable the SO and SK functions onto the G Port. The SO and SK pins must also be selected as outputs by setting appropriate bits in the Port G configuration register. Table VII summarizes the bit settings required for Master mode of operation.

MICROWIRE/PLUS Slave Mode Operation

In the MICROWIRE/PLUS Slave mode of operation the SK clock is generated by an external source. Setting the MSEL bit in the CNTRL register enables the SO and SK functions onto the G Port. The SK pin must be selected as an input and the SO pin is selected as an output pin by setting and resetting the appropriate bits in the Port G configuration register. Table VII summarizes the settings required to enter the Slave mode of operation.

The user must set the BUSY flag immediately upon entering the Slave mode. This will ensure that all data bits sent by the Master will be shifted properly. After eight clock pulses the BUSY flag will be cleared and the sequence may be repeated.

Alternate SK Phase Operation

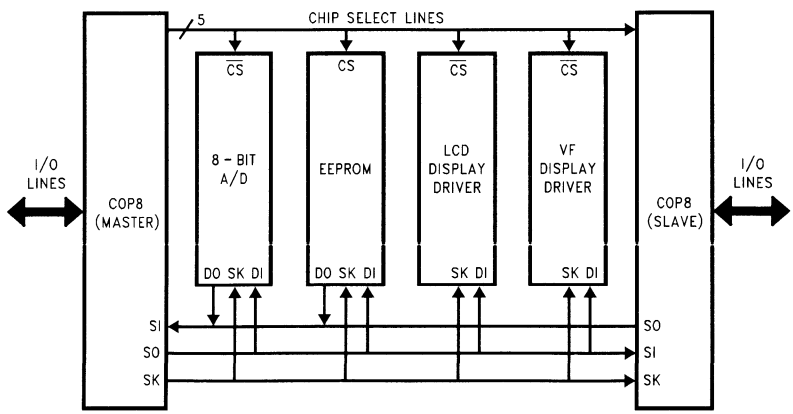
The device allows either the normal SK clock or an alternate phase SK clock to shift data in and out of the SIO register. In both the modes the SK is normally low. In the normal mode data is shifted in on the rising edge of the SK clock and the data is shifted out on the falling edge of the SK clock. The SIO register is shifted on each falling edge of the SK clock in the normal mode. In the alternate SK phase operation, data is shifted in on the falling edge of the SK clock and shifted out on the rising edge of the SK clock.

A control flag, SKSEL, allows either the normal SK clock or the alternate SK clock to be selected. Resetting SKSEL causes the MICROWIRE/PLUS logic to be clocked from the normal SK signal. Setting the SKSEL flag selects the alternate SK clock. The SKSEL is mapped into the G6 configuration bit. The SKSEL flag will power up in the reset condition, selecting the normal SK signal.

TABLE VII

This table assumes that the control flag MSEL is set.

G4 (SO) Config. Bit	G5 (SK) Config. Bit	G4 Fun.	G5 Fun.	Operation
1	1	SO	Int. SK	MICROWIRE/PLUS Master
0	1	TRI-STATE	Int. SK	MICROWIRE/PLUS Master
1	0	SO	Ext. SK	MICROWIRE/PLUS Slave
0	0	TRI-STATE	Ext. SK	MICROWIRE/PLUS Slave



TL/DD/10830-22

FIGURE 18. MICROWIRE/PLUS Application

Memory Map

All RAM, ports and registers (except A and PC) are mapped into data memory address space.

Address S/ADD REG	Contents
0000 to 006F	On-Chip RAM bytes (112 bytes)
0070 to 007F	Unused RAM Address Space (Reads As All Ones)
xx80 to xxAF	Unused RAM Address Space (Reads Undefined Data)
xxB0 to xxB6	Reserved
xxB7	Comparator Select Register (CMPSL)
xxB8	UART Transmit Buffer (TBUF)
xxB9	UART Receive Buffer (RBUF)
xxBA	UART Control and Status Register (ENU)
xxBB	UART Receive Control and Status Register (ENUR)
xxBC	UART Interrupt and Clock Source Register (ENUI)
xxBD	UART Baud Register (BAUD)
xxBE	UART Prescale Select Register (PSR)
xxBF	Reserved for UART
xxC0 to xxC6	Reserved
xxC7	WATCHDOG Service Register (Reg:WDSVR)
xxC8	MIWU Edge Select Register (Reg:WKEDG)
xxC9	MIWU Enable Register (Reg:WKEN)
xxCA	MIWU Pending Register (Reg:WKPND)
xxCB	Reserved
xxCC	Reserved
xxCD to xxCF	Reserved

Address S/ADD REG	Contents
xxD0	Port L Data Register
xxD1	Port L Configuration Register
xxD2	Port L Input Pins (Read Only)
xxD3	Reserved for Port L
xxD4	Port G Data Register
xxD5	Port G Configuration Register
xxD6	Port G Input Pins (Read Only)
xxD7	Port I Input Pins (Read Only)
xxD8	Port C Data Register
xxD9	Port C Configuration Register
xxDA	Port C Input Pins (Read Only)
xxDB	Reserved for Port C
xxDC	Port D
xxDD to DF	Reserved for Port D
xxE0 to xxE5	Reserved for EE Control Registers
xxE6	Timer T1 Autoload Register T1RB Lower Byte
xxE7	Timer T1 Autoload Register T1RB Upper Byte
xxE8	ICNTRL Register
xxE9	MICROWIRE/PLUS Shift Register
xxEA	Timer T1 Lower Byte
xxEB	Timer T1 Upper Byte
xxEC	Timer T1 Autoload Register T1RA Lower Byte
xxED	Timer T1 Autoload Register T1RA Upper Byte
xxEE	CNTRL Control Register
xxEF	PSW Register
xxF0 to FB	On-Chip RAM Mapped as Registers
xxFC	X Register
xxFD	SP Register
xxFE	B Register
xxFF	S Register
0100–013F	On-Chip RAM Bytes (64 bytes)

Reading memory locations 0070H–007FH (Segment 0) will return all ones. Reading unused memory locations 0080H–00AFH (Segment 0) will return undefined data. Reading unused memory locations 0140–017F (Segment 1) will return all ones. Reading memory locations from other Segments (i.e., Segment 2, Segment 3, ... etc.) will return all ones.

All reserved location reads undefined data.

Addressing Modes

The device has ten addressing modes, six for operand addressing and four for transfer of control.

OPERAND ADDRESSING MODES

Register Indirect

This is the "normal" addressing mode. The operand is the data memory addressed by the B pointer or X pointer.

Register Indirect (with auto post increment or decrement of pointer)

This addressing mode is used with the LD and X instructions. The operand is the data memory addressed by the B pointer or X pointer. This is a register indirect mode that automatically post increments or decrements the B or X register after executing the instruction.

Direct

The instruction contains an 8-bit address field that directly points to the data memory for the operand.

Immediate

The instruction contains an 8-bit immediate field as the operand.

Short Immediate

This addressing mode is used with the Load B Immediate instruction. The instruction contains a 4-bit immediate field as the operand.

Indirect

This addressing mode is used with the LAID instruction. The contents of the accumulator are used as a partial address (lower 8 bits of PC) for accessing a data operand from the program memory.

TRANSFER OF CONTROL ADDRESSING MODES

Relative

This mode is used for the JP instruction, with the instruction field being added to the program counter to get the new program location. JP has a range from −31 to +32 to allow a 1-byte relative jump (JP + 1 is implemented by a NOP instruction). There are no "pages" when using JP, since all 15 bits of PC are used.

Absolute

This mode is used with the JMP and JSR instructions, with the instruction field of 12 bits replacing the lower 12 bits of the program counter (PC). This allows jumping to any location in the current 4k program memory segment.

Absolute Long

This mode is used with the JMPL and JSRL instructions, with the instruction field of 15 bits replacing the entire 15 bits of the program counter (PC). This allows jumping to any location in the current 4k program memory space.

Indirect

This mode is used with the JID instruction. The contents of the accumulator are used as a partial address (lower 8 bits of PC) for accessing a location in the program memory. The contents of this program memory location serve as a partial address (lower 8 bits of PC) for the jump to the next instruction.

Note: The VIS is a special case of the Indirect Transfer of Control addressing mode, where the double byte vector associated with the interrupt is transferred from adjacent addresses in the program memory into the program counter (PC) in order to jump to the associated interrupt service routine.

Instruction Set

Register and Symbol Definition

Registers	
A	8-Bit Accumulator Register
B	8-Bit Address Register
X	8-Bit Address Register
SP	8-Bit Stack Pointer Register
PC	15-Bit Program Counter Register
PU	Upper 7 Bits of PC
PL	Lower 8 Bits of PC
C	1 Bit of PSW Register for Carry
HC	1 Bit of PSW Register for Half Carry
GIE	1 Bit of PSW Register for Global Interrupt Enable
VU	Interrupt Vector Upper Byte
VL	Interrupt Vector Lower Byte

Symbols	
[B]	Memory Indirectly Addressed by B Register
[X]	Memory Indirectly Addressed by X Register
MD	Direct Addressed Memory
Mem	Direct Addressed Memory or [B]
Meml	Direct Addressed Memory or [B] or Immediate Data
Imm	8-Bit Immediate Data
Reg	Register Memory: Addresses F0 to FF (Includes B, X and SP)
Bit	Bit Number (0 to 7)
←	Loaded with
⟷	Exchanged with

Instruction Set (Continued)

INSTRUCTION SET

ADD	A,Meml	ADD	A ← A + Meml
ADC	A,Meml	ADD with Carry	A ← A + Meml + C, C ← Carry HC ← Half Carry
SUBC	A,Meml	Subtract with Carry	A ← A − $\overline{\text{Meml}}$ + C, C ← Carry HC ← Half Carry
AND	A,Meml	Logical AND	A ← A and Meml
ANDSZ	A,Imm	Logical AND Immed., Skip if Zero	Skip next if (A and Imm) = 0
OR	A,Meml	Logical OR	A ← A or Meml
XOR	A,Meml	Logical EXclusive OR	A ← A xor Meml
IFEQ	MD,Imm	IF EQual	Compare MD and Imm, Do next if MD = Imm
IFEQ	A,Meml	IF EQual	Compare A and Meml, Do next if A = Meml
IFNE	A,Meml	IF Not Equal	Compare A and Meml, Do next if A ≠ Meml
IFGT	A,Meml	IF Greater Than	Compare A and Meml, Do next if A > Meml
IFBNE	#	If B Not Equal	Do next if lower 4 bits of B ≠ Imm
DRSZ	Reg	Decrement Reg., Skip if Zero	Reg ← Reg − 1, Skip if Reg = 0
SBIT	#,Mem	Set BIT	1 to bit, Mem (bit = 0 to 7 immediate)
RBIT	#,Mem	Reset BIT	0 to bit, Mem
IFBIT	#,Mem	IF BIT	If bit in A or Mem is true do next instruction
RPND		Reset PeNDing Flag	Reset Software Interrupt Pending Flag
X	A,Mem	EXchange A with Memory	A ⟷ Mem
LD	A,Meml	LoaD A with Memory	A ← Meml
LD	B,Imm	LoaD B with Immed.	B ← Imm
LD	Mem,Imm	LoaD Memory Immed	Mem ← Imm
LD	Reg,Imm	LoaD Register Memory Immed.	Reg ← Imm
X	A, [B ±]	EXchange A with Memory [B]	A ⟷ [B], (B ← B ±1)
X	A, [X ±]	EXchange A with Memory [X]	A ⟷ [X], (X ← ±1)
LD	A, [B±]	LoaD A with Memory [B]	A ← [B], (B ← B ±1)
LD	A, [X±]	LoaD A with Memory [X]	A ← [X], (X ← X±1)
LD	[B±],Imm	LoaD Memory [B] Immed.	[B] ← Imm, (B ← B±1)
CLR	A	CLeaR A	A ← 0
INC	A	INCrement A	A ← A + 1
DEC	A	DECrementA	A ← A − 1
LAID		Load A InDirect from ROM	A ← ROM (PU,A)
DCOR	A	Decimal CORrect A	A ← BCD correction of A (follows ADC, SUBC)
RRC	A	Rotate A Right thru C	C → A7 → . . . → A0 → C
RLC	A	Rotate A Left thru C	C ← A7 ← . . . ← A0 ← C
SWAP	A	SWAP nibbles of A	A7 . . . A4 ⟷ A3 . . . A0
SC		Set C	C ← 1, HC ← 1
RC		Reset C	C ← 0, HC ← 0
IFC		IF C	IF C is true, do next instruction
IFNC		IF Not C	If C is not true, do next instruction
POP	A	POP the stack into A	SP ← SP + 1, A ← [SP]
PUSH	A	PUSH A onto the stack	[SP] ← A, SP ← SP − 1
VIS		Vector to Interrupt Service Routine	PU ← [VU], PL ← [VL]
JMPL	Addr.	Jump absolute Long	PC ← ii (ii = 15 bits, 0 to 32k)
JMP	Addr.	Jump absolute	PC9 . . . 0 ← i (i = 12 bits)
JP	Disp.	Jump relative short	PC ← PC + r (r is −31 to +32, except 1)
JSRL	Addr.	Jump SubRoutine Long	[SP] ← PL, [SP−1] ← PU,SP−2, PC ← ii
JSR	Addr	Jump SubRoutine	[SP] ← PL, [SP−1] ← PU,SP−2, PC9 . . . 0 ← i
JID		Jump InDirect	PL ← ROM (PU,A)
RET		RETurn from subroutine	SP + 2, PL ← [SP], PU ← [SP−1]
RETSK		RETurn and SKip	SP + 2, PL ← [SP],PU ← [SP−1]
RETI		RETurn from Interrupt	SP + 2, PL ← [SP],PU ← [SP−1],GIE ← 1
INTR		Generate an Interrupt	[SP] ← PL, [SP−1] ← PU, SP−2, PC ← 0FF
NOP		No OPeration	PC ← PC + 1

5

Instruction Execution Time

Most instructions are single byte (with immediate addressing mode instructions taking two bytes).

Most single byte instructions take one cycle time to execute.

Skipped instructions require x number of cycles to be skipped, where x equals the number of bytes in the skipped instruction opcode.

See the BYTES and CYCLES per INSTRUCTION table for details.

Bytes and Cycles per Instruction

The following table shows the number of bytes and cycles for each instruction in the format of byte/cycle.

Arithmetic and Logic Instructions

	[B]	Direct	Immed.
ADD	1/1	3/4	2/2
ADC	1/1	3/4	2/2
SUBC	1/1	3/4	2/2
AND	1/1	3/4	2/2
OR	1/1	3/4	2/2
XOR	1/1	3/4	2/2
IFEQ	1/1	3/4	2/2
IFNE	1/1	3/4	2/2
IFGT	1/1	3/4	2/2
IFBNE	1/1		
DRSZ		1/3	
SBIT	1/1	3/4	
RBIT	1/1	3/4	
IFBIT	1/1	3/4	

RPND	1/1

Instructions Using A & C

CLRA	1/1
INCA	1/1
DECA	1/1
LAID	1/3
DCOR	1/1
RRCA	1/1
RLCA	1/1
SWAPA	1/1
SC	1/1
RC	1/1
IFC	1/1
IFNC	1/1
PUSHA	1/3
POPA	1/3
ANDSZ	2/2

Transfer of Control Instructions

JMPL	3/4
JMP	2/3
JP	1/3
JSRL	3/5
JSR	2/5
JID	1/3
VIS	1/5
RET	1/5
RETSK	1/5
RETI	1/5
INTR	1/7
NOP	1/1

Memory Transfer Instructions

	Register Indirect		Direct	Immed.	Register Indirect Auto Incr. & Decr.		
	[B]	[X]			[B+, B−]	[X+, X−]	
X A,*	1/1	1/3	2/3		1/2	1/3	
LD A,*	1/1	1/3	2/3	2/2	1/2	1/3	
LD B, Imm				1/1			(IF B < 16)
LD B, Imm				2/2			(IF B > 15)
LD Mem, Imm	2/2		3/3		2/2		
LD Reg, Imm			2/3				
IFEQ MD, Imm			3/3				

* = > Memory location addressed by B or X or directly.

Opcode Table

Upper Nibble Along X-Axis

Lower Nibble Along Y-Axis

F	E	D	C	B	A	9	8	
JP −15	JP −31	LD 0F0, # i	DRSZ 0F0	RRCA	RC	ADC A,#i	ADC A,[B]	0
JP −14	JP −30	LD 0F1, # i	DRSZ 0F1	*	SC	SUBC A, #i	SUB A,[B]	1
JP −13	JP −29	LD 0F2, # i	DRSZ 0F2	X A, [X+]	X A,[B+]	IFEQ A,#i	IFEQ A,[B]	2
JP −12	JP −28	LD 0F3, # i	DRSZ 0F3	X A, [X−]	X A,[B−]	IFGT A,#i	IFGT A,[B]	3
JP −11	JP −27	LD 0F4, # i	DRSZ 0F4	VIS	LAID	ADD A,#i	ADD A,[B]	4
JP −10	JP −26	LD 0F5, # i	DRSZ 0F5	RPND	JID	AND A,#i	AND A,[B]	5
JP −9	JP −25	LD 0F6, # i	DRSZ 0F6	X A,[X]	X A,[B]	XOR A,#i	XOR A,[B]	6
JP −8	JP −24	LD 0F7, # i	DRSZ 0F7	*	*	OR A,#i	OR A,[B]	7
JP −7	JP −23	LD 0F8, # i	DRSZ 0F8	NOP	RLCA	LD A,#i	IFC	8
JP −6	JP −22	LD 0F9, # i	DRSZ 0F9	IFNE A,[B]	IFEQ Md,#i	IFNE A,#i	IFNC	9
JP −5	JP −21	LD 0FA, # i	DRSZ 0FA	LD A,[X+]	LD A,[B+]	LD [B+],#i	INCA	A
JP −4	JP −20	LD 0FB, # i	DRSZ 0FB	LD A,[X−]	LD A,[B−]	LD [B−],#i	DECA	B
JP −3	JP −19	LD 0FC, # i	DRSZ 0FC	LD Md,#i	JMPL	X A,Md	POPA	C
JP −2	JP −18	LD 0FD, # i	DRSZ 0FD	DIR	JSRL	LD A,Md	RETSK	D
JP −1	JP −17	LD 0FE, # i	DRSZ 0FE	LD A,[X]	LD A,[B]	LD [B],#i	RET	E
JP −0	JP −16	LD 0FF, # i	DRSZ 0FF	*	*	LD B,#i	RETI	F

Opcode Table (Continued)

Upper Nibble Along X-Axis

Lower Nibble Along Y-Axis

7	6	5	4	3	2	1	0	
IFBIT 0,[B]	ANDSZ A, #i	LD B,#0F	IFBNE 0	JSR x000–x0FF	JMP x000–x0FF	JP +17	INTR	0
IFBIT 1,[B]	*	LD B,#0E	IFBNE 1	JSR x100–x1FF	JMP x100–x1FF	JP +18	JP + 2	1
IFBIT 2,[B]	*	LD B,#0D	IFBNE 2	JSR x200–x2FF	JMP x200–x2FF	JP +19	JP + 3	2
IFBIT 3,[B]	*	LD B,#0C	IFBNE 3	JSR x300–x3FF	JMP x300–x3FF	JP +20	JP + 4	3
IFBIT 4,[B]	CLRA	LD B,#0B	IFBNE 4	JSR x400–x4FF	JMP x400–x4FF	JP +21	JP + 5	4
IFBIT 5,[B]	SWAPA	LD B,#0A	IFBNE 5	JSR x500–x5FF	JMP x500–x5FF	JP +22	JP + 6	5
IFBIT 6,[B]	DCORA	LD B,#09	IFBNE 6	JSR x600–x6FF	JMP x600–x6FF	JP +23	JP + 7	6
IFBIT 7,[B]	PUSHA	LD B,#08	IFBNE 7	JSR x700–x7FF	JMP x700–x7FF	JP +24	JP + 8	7
SBIT 0,[B]	RBIT 0,[B]	LD B,#07	IFBNE 8	JSR x800–x8FF	JMP x800–x8FF	JP +25	JP + 9	8
SBIT 1,[B]	RBIT 1,[B]	LD B,#06	IFBNE 9	JSR x900–x9FF	JMP x900–x9FF	JP +26	JP + 10	9
SBIT 2,[B]	RBIT 2,[B]	LD B,#05	IFBNE 0A	JSR xA00–xAFF	JMP xA00–xAFF	JP +27	JP + 11	A
SBIT 3,[B]	RBIT 3,[B]	LD B,#04	IFBNE 0B	JSR xB00–xBFF	JMP xB00–xBFF	JP +28	JP + 12	B
SBIT 4,[B]	RBIT 4,[B]	LD B,#03	IFBNE 0C	JSR xC00–xCFF	JMP xC00–xCFF	JP +29	JP + 13	C
SBIT 5,[B]	RBIT 5,[B]	LD B,#02	IFBNE 0D	JSR xD00–xDFF	JMP xD00–xDFF	JP +30	JP + 14	D
SBIT 6,[B]	RBIT 6,[B]	LD B,#01	IFBNE 0E	JSR xE00–xEFF	JMP xE00–xEFF	JP +31	JP + 15	E
SBIT 7,[B]	RBIT 7,[B]	LD B,#00	IFBNE 0F	JSR xF00–xFFF	JMP xF00–xFFF	JP +32	JP + 16	F

Where,

i is the immediate data

Md is a directly addressed memory location

* is an unused opcode

Note: The opcode 60 Hex is also the opcode for IFBIT #i,A

Mask Options

The device mask programmable options are shown below. The options are programmed at the same time as the ROM pattern submission.

```
OPTION 1: CLOCK CONFIGURATION
  = 1     Crystal Oscillator (CKI/lO)
              G7 (CKO) is clock generator
              output to crystal/resonator
              CKI is the clock input
  = 2     Single-pin RC controlled
              oscillator (CKI/lO)
              G7 is available as a HALT
              restart and/or general purpose
              input

OPTION 2: HALT
  = 1     Enable HALT mode
  = 2     Disable HALT mode

OPTION 3: BONDING OPTIONS
  = 1     44-Pin PLCC
  = 2     40-Pin DIP
  = 3     NA
  = 4     28-Pin DIP
  = 5     28-Pin SO
```

Development Support

SUMMARY

- iceMASTER™: IM-COP8/400—Full feature in-circuit emulation for all COP8 products. A full set of COP8 Basic and Feature Family device and package specific probes are available.
- COP8 Debug Module: Moderate cost in-circuit emulation and development programming unit.
- COP8 Evaluation and Programming Unit: EPU-COP888GG—low cost in-circuit simulation and development programming unit.
- Assembler: COP8-DEV-IBMA. A DOS installable cross development Assembler, Linker, Librarian and Utility Software Development Tool Kit.
- C Compiler: COP8C. A DOS installable cross development Software Tool Kit.
- OTP/EPROM Programmer Support: Covering needs from engineering prototype, pilot production to full production environments.

Development Support (Continued)

IceMASTER (IM) IN-CIRCUIT EMULATION

The iceMASTER IM-COP8/400 is a full feature, PC based, in-circuit emulation tool developed and marketed by MetaLink Corporation to support the whole COP8 family of products. National is a resale vendor for these products.

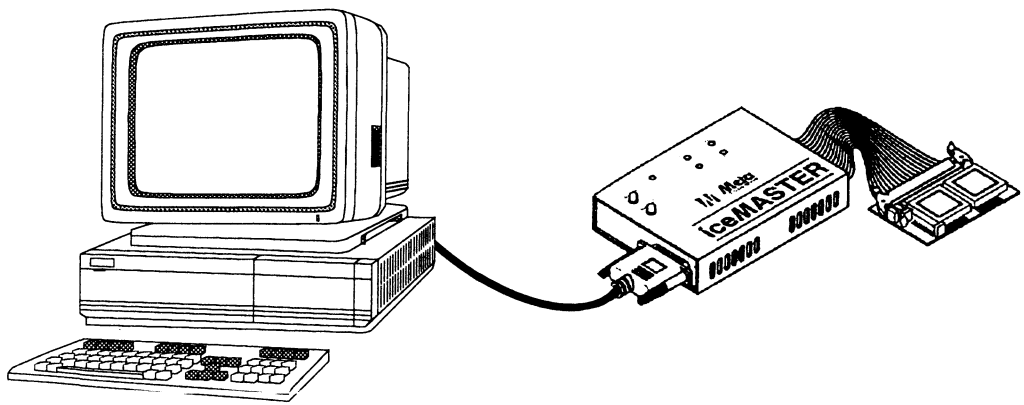
See *Figure 19* for configuration.

The iceMASTER IM-COP8/400 with its device specific COP8 Probe provides a rich feature set for developing, testing and maintaining product:

- Real-time in-circuit emulation; full 2.4V–5.5V operation range, full DC-10 MHz clock. Chip options are programmable or jumper selectable.
- Direct connection to application board by package compatible socket or surface mount assembly.
- Full 32 kbyte of loadable programming space that overlays (replaces) the on-chip ROM or EPROM. On-chip RAM and I/O blocks are used directly or recreated on the probe as necessary.
- Full 4k frame synchronous trace memory. Address, instruction, and 8 unspecified, circuit connectable trace lines. Display can be HLL source (e.g., C source), assembly or mixed.
- A full 64k hardware configurable break, trace on, trace off control, and pass count increment events.
- Tool set integrated interactive symbolic debugger—supports both assembler (COFF) and C Compiler (.COD) linked object formats.
- Real time performance profiling analysis; selectable bucket definition.
- Watch windows, content updated automatically at each execution break.
- Instruction by instruction memory/register changes displayed on source window when in single step operation.
- Single base unit and debugger software reconfigurable to support the entire COP8 family; only the probe personality needs to change. Debugger software is processor customized, and reconfigured from a master model file.
- Processor specific symbolic display of registers and bit level assignments, configured from master model file.
- Halt/Idle mode notification.
- On-Line HELP customized to specific processor using master model file.
- Includes a copy of COP8-DEV-IBMA assembler and linker SDK.

IM Order Information

Base Unit	
IM-COP8/400-1	iceMASTER Base Unit, 110V Power Supply
IM-COP8/400-2	iceMASTER Base Unit, 220V Power Supply
iceMASTER Probe	
MHW-884EG28DWPC	28 DIP
MHW-888EG40DWPC	40 DIP
MHW-888EG44PWPC	44 PLCC
Adapter for SO Package	
MHW-SOIC 28	28 SO



TL/DD/10830–32

FIGURE 19. COP8 iceMASTER Environment

Development Support (Continued)

iceMASTER DEBUG MODULE (DM)

The iceMASTER Debug Module is a PC based, combination in-circuit emulation tool and COP8 based OTP/EPROM programming tool developed and marketed by MetaLink Corporation to support the whole COP8 family of products. National is a resale vendor for these products.

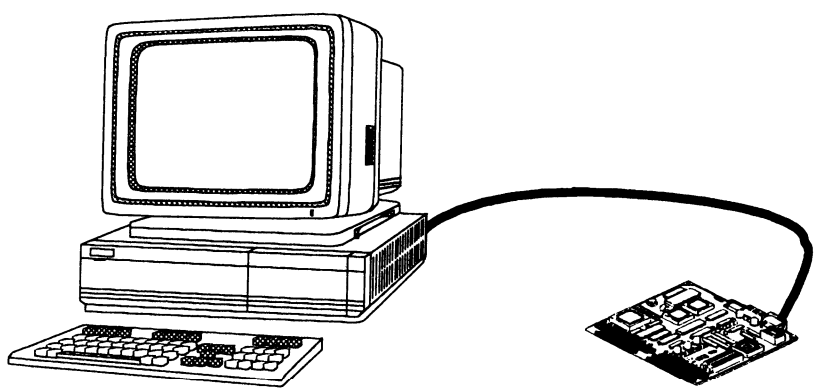
See *Figure 20* for configuration.

The iceMASTER Debug Module is a moderate cost development tool. It has the capability of in-circuit emulation for a specific COP8 microcontroller and in addition serves as a programming tool for COP8 OTP and EPROM product families. Summary of features is as follows:

- Real-time in-circuit emulation; full operating voltage range operation, full DC-10 MHz clock.
- All processor I/O pins can be cabled to an application development board with package compatible cable to socket and surface mount assembly.
- Full 32 kbyte of loadable programming space that overlays (replaces) the on-chip ROM or EPROM. On-chip RAM and I/O blocks are used directly or recreated as necessary.
- 100 frames of synchronous trace memory. The display can be HLL source (C source), assembly or mixed. The most recent history prior to a break is available in the trace memory.
- Configured break points; uses INTR instruction which is modestly intrusive.
- Software—only supported features are selectable.
- Tool set integrated interactive symbolic debugger—supports both assembler (COFF) and C Compiler (.COD) SDK linked object formats.
- Instruction by instruction memory/register changes displayed when in single step operation.
- Debugger software is processor customized, and reconfigured from a master model file.
- Processor specific symbolic display of registers and bit level assignments, configured from master model file.
- Halt/Idle mode notification.
- Programming menu supports full product line of programmable OTP and EPROM COP8 products. Program data is taken directly from the overlay RAM.
- Programming of 44 PLCC and 68 PLCC parts requires external programming adapters.
- Includes wallmount power supply.
- On-board V_{PP} generator from 5V input or connection to external supply supported. Requires V_{PP} level adjustment per the family programming specification (correct level is provided on an on-screen pop-down display).
- On-line HELP customized to specific processor using master model file.
- Includes a copy of COP8-DEV-IBMA assembler and linker SDK.

DM Order Information

Debug Module Unit	
COP8-DM/888GG	
Cable Adapters	
DM-COP8/28D	28 DIP
DM-COP8/40D	40 DIP
DM-COP8/44P	44 PLCC
Adapter for SO Package	
MHW-SOIC 28	28 SO



TL/DD/10830–33

FIGURE 20. COP8-DM Environment

Development Support (Continued)

COP8 ASSEMBLER/LINKER SOFTWARE DEVELOPMENT TOOL KIT

National Semiconductor offers a relocatable COP8 macro cross assembler, linker, librarian and utility software development tool kit. Features are summarized as follows:

- Basic and Feature Family instruction set by "device" type.
- Nested macro capability.
- Extensive set of assembler directives.
- Supported on PC/DOS platform.
- Generates National standard COFF output files.
- Integrated Linker and Librarian.
- Integrated utilities to generate ROM code file outputs.
- DUMPCOFF utility.

This product is integrated as a part of MetaLink tools as a development kit, fully supported by the MetaLink debugger. It may be ordered separately or it is bundled with the MetaLink products at no additional cost.

Order Information

Assembler SDK:	
COP8-DEV-IBMA	Assembler SDK on installable 3.5" PC/DOS Floppy Disk Drive format. Periodic upgrades and most recent version is available on National's BBS and Internet.

COP8 C COMPILER

A C Compiler is developed and marketed by Byte Craft Limited. The COP8C compiler is a fully integrated development tool specifically designed to support the compact embedded configuration of the COP8 family of products.

Features are summarized as follows:

- ANSI C with some restrictions and extensions that optimize development for the COP8 embedded application.
- BITS data type extension. Register declaration #pragma with direct bit level definitions.
- C language support for interrupt routines.
- Expert system, rule based code generation and optimization.
- Performs consistency checks against the architectural definitions of the target COP8 device.
- Generates program memory code.
- Supports linking of compiled object or COP8 assembled object formats.
- Global optimization of linked code.
- Symbolic debug load format fully source level supported by the MetaLink debugger.

SINGLE CHIP OTP/EMULATOR SUPPORT

The COP8 family is supported by single chip OTP emulators. For detailed information refer to the emulator specific datasheet and the emulator selection table below:

OTP Emulator Ordering Information

Device Number	Clock Option	Package	Emulates
COP87L84EGN-XE	Crystal	28 DIP	COP884CS
COP87L84EGM-XE	Crystal	28 SO	COP884CS
COP87L88EGN-XE	Crystal	40 N	COP888CS
COP87L88EGV-XE	Crystal	44 PLCC	COP888CS

INDUSTRY WIDE OTP/EPROM PROGRAMMING SUPPORT

Programming support, in addition to the MetaLink development tools, is provided by a full range of independent approved vendors to meet the needs from the engineering laboratory to full production.

Approved List

Manufacturer	North America	Europe	Asia
BP Microsystems	(800) 225-2102 (713) 688-4600 Fax: (713) 688-0920	+49-8152-4183 +49-8856-932616	+852-234-16611 +852-2710-8121
Data I/O	(800) 426-1045 (206) 881-6444 Fax: (206) 882-1043	+44-0734-440011	Call North America
HI-LO	(510) 623-8860	Call Asia	+886-2-764-0215 Fax: +886-2-756-6403
ICE Technology	(800) 624-8949 (919) 430-7915	+44-1226-767404 Fax: 0-1226-370-434	
MetaLink	(800) 638-2423 (602) 926-0797 Fax: (602) 693-0681	+49-80 9156 96-0 Fax: +49-80 9123 86	+852-737-1800
Systems General	(408) 263-6667	+41-1-9450300	+886-2-917-3005 Fax: +886-2-911-1283
Needhams	(916) 924-8037 Fax: (916) 924-8065		

Development Support (Continued)

AVAILABLE LITERATURE

For more information, please see the COP8 Basic Family User's Manual, Literature Number 620895, COP8 Feature Family User's Manual, Literature Number 620897 and National's Family of 8-bit Microcontrollers COP8 Selection Guide, Literature Number 630009.

DIAL-A-HELPER SERVICE

Dial-A-Helper is a service provided by the Microcontroller Applications group. The Dial-A-Helper is an Electronic Information System that may be accessed as a Bulletin Board System (BBS) via data modem, as an FTP site on the Internet via standard FTP client application or as an FTP site on the Internet using a standard Internet browser such as Netscape or Mosaic.

The Dial-A-Helper system provides access to an automated information storage and retrieval system. The system capabilities include a MESSAGE SECTION (electronic mail, when accessed as a BBS) for communications to and from the Microcontroller Applications Group and a FILE SECTION which consists of several file areas where valuable application software and utilities could be found.

DIAL-A-HELPER BBS via a Standard Modem

Modem: CANADA/U.S.: (800) NSC-MICRO
(800) 672-6427
EUROPE: (+49) 0-8141-351332
Baud: 14.4k
Set-up: Length: 8-Bit
Parity: None
Stop Bit: 1
Operation: 24 Hrs., 7 Days

DIAL-A-HELPER via FTP

ftp nscmicro.nsc.com
user: anonymous
password: username@yourhost.site.domain

DIAL-A-HELPER via a WorldWide Web Browser

ftp://nscmicro.nsc.com

National Semiconductor on the WorldWide Web

See us on the WorldWide Web at: http://www.national.com

CUSTOMER RESPONSE CENTER

Complete product information and technical support is available from National's customer response centers.

CANADA/U.S.:	Tel:	(800) 272-9959
	email:	support @tevm2.nsc.com
EUROPE:	email:	europe.support@nsc.com
	Deutsch Tel:	+49 (0) 180-530 85 85
	English Tel:	+49 (0) 180-532 78 32
	Français Tel:	+49 (0) 180-532 93 58
	Italiano Tel:	+49 (0) 180-534 16 80
JAPAN:	Tel:	+81-043-299-2309
S.E. ASIA:	Beijing Tel:	(+86) 10-6856-8601
	Shanghai Tel:	(+86) 21-6415-4092
	Hong Kong Tel:	(+852) 2737-1600
	Korea Tel:	(+82) 2-3771-6909
	Malaysia Tel:	(+60-4) 644-9061
	Singapore Tel:	(+65) 255-2226
	Taiwan Tel:	+886-2-521-3288
AUSTRALIA:	Tel:	(+61) 3-9558-9999
INDIA:	Tel:	(+91) 80-559-9467

National Semiconductor

COP688EG/COP684EG/COP888EG/COP884EG/ COP988EG/COP984EG 8-Bit Microcontroller with UART and Three Multi-Function Timers

General Description

The COP8™ feature family of microcontrollers use an 8-bit single-chip core architecture fabricated with National Semiconductor's M²CMOS™ process technology. The COP888EG/COP884EG are members of this expandable 8-bit core processor family of microcontrollers. (Continued)

Key Features

- Full duplex UART
- Three 16-bit timers, each with two 16-bit registers supporting
  - — Processor Independent PWM mode
  - — External Event counter mode
  - — Input Capture mode
- Quiet design (low radiated emissions)
- 8k bytes on-board ROM
- 256 bytes on-board RAM

Additional Peripheral Features

- Idle Timer
- Multi-Input Wake-Up (MIWU) with optional interrupts (8)
- Two analog comparators
- WATCHDOG™ and clock monitor logic
- MICROWIRE/PLUS™ serial I/O

I/O Features

- Memory mapped I/O
- Software selectable I/O options (TRI-STATE® Output, Push-Pull Output, Weak Pull-Up Input, High Impedance Input)
- Schmitt trigger inputs on ports G and L
- Packages:
  - — 28 SO or 28 DIP, each with 24 I/O pins
  - — 40 DIP with 36 I/O pins
  - — 44 PQFP with 40 I/O pins
  - — 44 PLCC with 40 I/O pins

CPU/Instruction Set Features

- 1 μs instruction cycle time
- Fourteen multi-source vectored interrupts servicing
  - — External Interrupt with selectable edge
  - — Idle Timer T0
  - — Three Timers (each with 2 Interrupts)
  - — MICROWIRE/PLUS
  - — Multi-Input Wake-Up
  - — Software Trap
  - — UART (2)
  - — Default VIS (default interrupt)
- Versatile and easy to use instruction set
- 8-bit Stack Pointer (SP) stack in RAM
- Two 8-bit Register Indirect Data Memory Pointers (B and X)

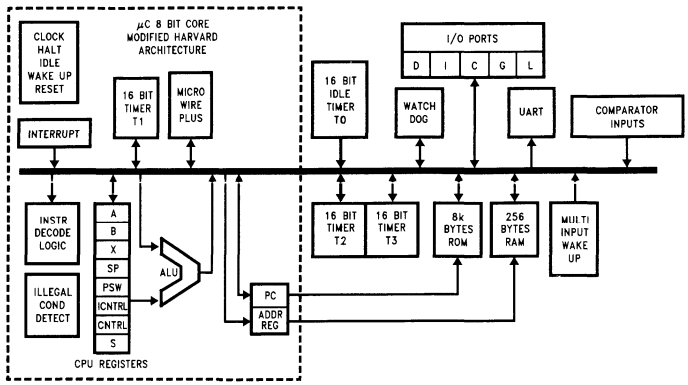
Fully Static CMOS

- Two power saving modes: HALT and IDLE
- Low current drain (typically <1 μA)
- Single supply operation: 2.5V to 6.0V
- Temperature ranges: 0°C to +70°C, −40°C to +85°C, −55°C to +125°C

Development Support

- Emulation and OTP devices
- Real time emulation and full program debug offered by MetaLink Development System

Block Diagram



TL/DD/11214–1

FIGURE 1. Block Diagram

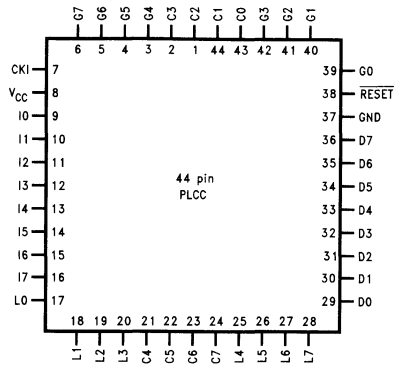
General Description (Continued)

They are fully static parts, fabricated using double-metal silicon gate microCMOS technology. Features include an 8-bit memory mapped architecture, MICROWIRE/PLUS serial I/O, three 16-bit timer/counters supporting three modes (Processor Independent PWM generation, External Event counter, and Input Capture mode capabilities), full duplex UART, two comparators, and two power savings modes (HALT and IDLE), both with a multi-sourced wakeup/interrupt capability. This multi-sourced interrupt capability may also be used independent of the HALT or IDLE modes. Each I/O pin has software selectable configurations. The device operates over a voltage range of 2.5V to 6V. High throughput is achieved with an efficient, regular instruction set operating at a maximum rate of 1 μs per instruction.

Low radiated emissions are achieved by gradual turn-on output drivers and internal I_{CC} smoothing filters on the chip logic and crystal oscillator.

Connection Diagrams

Plastic Chip Carrier

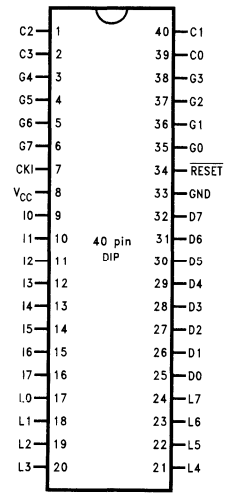


TL/DD/11214-2

Top View

Order Number COP888EG-XXX/V
See NS Plastic Chip Package Number V44A

Dual-In-Line Package

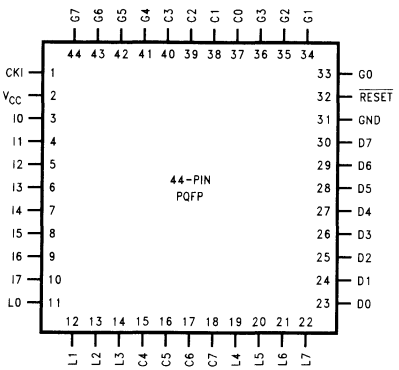


TL/DD/11214-3

Top View

Order Number COP888EG-XXX/N
See NS Molded Package Number N40A

Molded Plastic Quad Flat Package

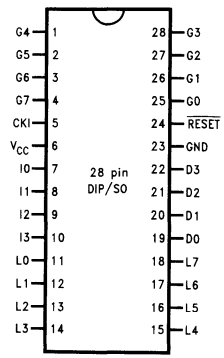


TL/DD/11214-30

Top View

Order Number COP888EG-XXX/VEJ
See NS Package Number VEJ44A

Dual-In-Line Package



TL/DD/11214-4

Top View

Order Number COP884EG-XXX/WM
or COP884EG-XXX/N
See NS Molded Package Number M28B or N28A

FIGURE 2a. Connection Diagrams

Connection Diagrams (Continued)

Pinouts for 28-, 40- and 44-Pin Packages

Port	Type	Alt. Fun	Alt. Fun	28-Pin DIP/SO	40-Pin DIP	44-Pin PLCC	44-Pin PQFP
L0	I/O	MIWU		11	17	17	11
L1	I/O	MIWU	CKX	12	18	18	12
L2	I/O	MIWU	TDX	13	19	19	13
L3	I/O	MIWU	RDX	14	20	20	14
L4	I/O	MIWU	T2A	15	21	25	19
L5	I/O	MIWU	T2B	16	22	26	20
L6	I/O	MIWU	T3A	17	23	27	21
L7	I/O	MIWU	T3B	18	24	28	22
G0	I/O	INT		25	35	39	33
G1	WDOUT			26	36	40	34
G2	I/O	T1B		27	37	41	35
G3	I/O	T1A		28	38	42	36
G4	I/O	SO		1	3	3	41
G5	I/O	SK		2	4	4	42
G6	I	SI		3	5	5	43
G7	I/CKO	HALT Restart		4	6	6	44
D0	O			19	25	29	23
D1	O			20	26	30	24
D2	O			21	27	31	25
D3	O			22	28	32	26
D4	O				29	33	7
D5	O				30	34	8
D6	O				31	35	9
D7	O				32	36	10
I0	I			7	9	9	27
I1	I	COMP1IN−		8	10	10	28
I2	I	COMP1IN+		9	11	11	29
I3	I	COMP1OUT		10	12	12	30
I4	I	COMP2IN−			13	13	3
I5	I	COMP2IN+			14	14	4
I6	I	COMP2OUT			15	15	5
I7	I				16	16	6
C0	I/O				39	43	37
C1	I/O				40	44	38
C2	I/O				1	1	39
C3	I/O				2	2	40
C4	I/O					21	15
C5	I/O					22	16
C6	I/O					23	17
C7	I/O					24	18
V_{CC}				6	8	8	2
GND				23	33	37	31
CKI				5	7	7	1
$\overline{\text{RESET}}$				24	34	38	32

Absolute Maximum Ratings

If Military/Aerospace specified devices are required, please contact the National Semiconductor Sales Office/Distributors for availability and specifications.

Supply Voltage (V_{CC})	7V
Voltage at Any Pin	−0.3V to V_{CC} + 0.3V
Total Current into V_{CC} Pin (Source)	100 mA
Total Current out of GND Pin (Sink)	110 mA
Storage Temperature Range	−65°C to +140°C

Note: *Absolute maximum ratings indicate limits beyond which damage to the device may occur. DC and AC electrical specifications are not ensured when operating the device at absolute maximum ratings.*

DC Electrical Characteristics 98XEG: 0°C ≤ T_A ≤ +70°C unless otherwise specified

Parameter	Conditions	Min	Typ	Max	Units
Operating Voltage COP98XEG		2.5		4.0	V
COP98XEGH		4.0		6.0	V
Power Supply Ripple (Note 1)	Peak-to-Peak			0.1 V_{CC}	V
Supply Current (Note 2)					
CKI = 10 MHz	V_{CC} = 6V, t_c = 1 μs			12.5	mA
CKI = 4 MHz	V_{CC} = 6V, t_c = 2.5 μs			5.5	mA
CKI = 4 MHz	V_{CC} = 4V, t_c = 2.5 μs			2.5	mA
CKI = 1 MHz	V_{CC} = 4V, t_c = 10 μs			1.4	mA
HALT Current (Note 3)	V_{CC} = 6V, CKI = 0 MHz		<0.7	8	μA
	V_{CC} = 4V, CKI = 0 MHz		<0.3	4	μA
IDLE Current					
CKI = 10 MHz	V_{CC} = 6V, t_c = 1 μs			3.5	mA
CKI = 4 MHz	V_{CC} = 6V, t_c = 2.5 μs			2.5	mA
CKI = 1 MHz	V_{CC} = 4V, t_c = 10 μs			0.7	mA
Input Levels					
$\overline{\text{RESET}}$					
Logic High		0.8 V_{CC}			V
Logic Low				0.2 V_{CC}	V
CKI (External and Crystal Osc. Modes)					
Logic High		0.7 V_{CC}			V
Logic Low				0.2 V_{CC}	V
All Other Inputs					
Logic High		0.7 V_{CC}			V
Logic Low				0.2 V_{CC}	V
Hi-Z Input Leakage	V_{CC} = 6V	−1		+1	μA
Input Pullup Current	V_{CC} = 6V, V_{IN} = 0V	−40		−250	μA
G and L Port Input Hysteresis				0.35 V_{CC}	V
Output Current Levels					
D Outputs					
Source	V_{CC} = 4V, V_{OH} = 3.3V	−0.4			mA
	V_{CC} = 2.5V, V_{OH} = 1.8V	−0.2			mA
Sink	V_{CC} = 4V, V_{OL} = 1V	10			mA
	V_{CC} = 2.5V, V_{OL} = 0.4V	2.0			mA
All Others					
Source (Weak Pull-Up Mode)	V_{CC} = 4V, V_{OH} = 2.7V	−10		−100	μA
	V_{CC} = 2.5V, V_{OH} = 1.8V	−2.5		−33	μA
Source (Push-Pull Mode)	V_{CC} = 4V, V_{OH} = 3.3V	−0.4			mA
	V_{CC} = 2.5V, V_{OH} = 1.8V	−0.2			mA
Sink (Push-Pull Mode)	V_{CC} = 4V, V_{OL} = 0.4V	1.6			mA
	V_{CC} = 2.5V, V_{OL} = 0.4V	0.7			mA
TRI-STATE Leakage	V_{CC} = 6.0V	−1		+1	μA

Note 1: Rate of voltage change must be less then 0.5 V/ms.

Note 2: Supply current is measured after running 2000 cycles with a square wave CKI input, CKO open, inputs at rails and outputs open.

Note 3: The HALT mode will stop CKI from oscillating in the RC and the Crystal configurations. Test conditions: All inputs tied to V_{CC}, L and G_0-G_5 configured as outputs and set high. The D port set to zero. The clock monitor and the comparators are disabled.

DC Electrical Characteristics 98XEG: 0°C ≤ T_A ≤ +70°C unless otherwise specified (Continued)

Parameter	Conditions	Min	Typ	Max	Units
Allowable Sink/Source Current per Pin					
D Outputs (Sink)				15	mA
All others				3	mA
Maximum Input Current without Latchup (Note 5)	T_A = 25°C			±100	mA
RAM Retention Voltage, V_r	500 ns Rise and Fall Time (Min)	2			V
Input Capacitance				7	pF
Load Capacitance on D2				1000	pF

AC Electrical Characteristics 98XEG: 0°C ≤ T_A ≤ +70°C unless otherwise specified

Parameter	Conditions	Min	Typ	Max	Units
Instruction Cycle Time (t_c)	4V ≤ V_{CC} ≤ 6V	1		DC	μs
Crystal, Resonator,	2.5V ≤ V_{CC} < 4V	2.5		DC	μs
R/C Oscillator	4V ≤ V_{CC} ≤ 6V	3		DC	μs
	2.5V ≤ V_{CC} < 4V	7.5		DC	μs
Inputs					
t_{SETUP}	4V ≤ V_{CC} ≤ 6V	200			ns
	2.5V ≤ V_{CC} < 4V	500			ns
t_{HOLD}	4V ≤ V_{CC} ≤ 6V	60			ns
	2.5V ≤ V_{CC} < 4V	150			ns
Output Propagation Delay (Note 6)	R_L = 2.2k, C_L = 100 pF				
t_{PD1}, t_{PD0}					
SO, SK	4V ≤ V_{CC} ≤ 6V			0.7	μs
	2.5V ≤ V_{CC} < 4V			1.75	μs
All Others	4V ≤ V_{CC} ≤ 6V			1	μs
	2.5V ≤ V_{CC} < 4V			2.5	μs
MICROWIRE™ Setup Time (t_{UWS})		20			ns
MICROWIRE Hold Time (t_{UWH})		56			ns
MICROWIRE Output Propagation Delay (t_{UPD})				220	ns
Input Pulse Width					
Interrupt Input High Time		1			t_c
Interrupt Input Low Time		1			t_c
Timer Input High Time		1			t_c
Timer Input Low Time		1			t_c
Reset Pulse Width		1			μs

Note 5: Pins G6 and $\overline{RESET}$ are designed with a high voltage input network for factory testing. These pins allow input voltages greater than V_{CC} and the pins will have sink current to V_{CC} when biased at voltages greater than V_{CC} (the pins do not have source current when biased at a voltage below V_{CC}). The effective resistance to V_{CC} is 750Ω (typical). These two pins will not latch up. The voltage at the pins must be limited to less than 14V.

Note 6: The output propagation delay is referenced to the end of the instruction cycle where the output change occurs.

Absolute Maximum Ratings

If Military/Aerospace specified devices are required, please contact the National Semiconductor Sales Office/Distributors for availability and specifications.

Supply Voltage (V_{CC})	7V
Voltage at Any Pin	−0.3V to V_{CC} + 0.3V
Total Current into V_{CC} Pin (Source)	100 mA
Total Current out of GND Pin (Sink)	110 mA
Storage Temperature Range	−65°C to +140°C

Note: *Absolute maximum ratings indicate limits beyond which damage to the device may occur. DC and AC electrical specifications are not ensured when operating the device at absolute maximum ratings.*

DC Electrical Characteristics 888EG: $-40°C \le T_A \le +85°C$ unless otherwise specified

Parameter	Conditions	Min	Typ	Max	Units
Operating Voltage		2.5		6	V
Power Supply Ripple (Note 1)	Peak-to-Peak			0.1 V_{CC}	V
Supply Current (Note 2)					
CKI = 10 MHz	V_{CC} = 6V, t_c = 1 μs			12.5	mA
CKI = 4 MHz	V_{CC} = 6V, t_c = 2.5 μs			5.5	mA
CKI = 4 MHz	V_{CC} = 4.0V, t_c = 2.5 μs			2.5	mA
CKI = 1 MHz	V_{CC} = 4.0V, t_c = 10 μs			1.4	mA
HALT Current (Note 3)	V_{CC} = 6V, CKI = 0 MHz		<1	10	μA
	V_{CC} = 4.0V, CKI = 0 MHz		<0.5	6	μA
IDLE Current					
CKI = 10 MHz	V_{CC} = 6V, t_c = 1 μs			3.5	mA
CKI = 4 MHz	V_{CC} = 6V, t_c = 2.5 μs			2.5	mA
CKI = 1 MHz	V_{CC} = 4.0V, t_c = 10 μs			0.7	mA
Input Levels					
$\overline{\text{RESET}}$					
Logic High		0.8 V_{CC}			V
Logic Low				0.2 V_{CC}	V
CKI (External and Crystal Osc. Modes)					
Logic High		0.7 V_{CC}			V
Logic Low				0.2 V_{CC}	V
All Other Inputs					
Logic High		0.7 V_{CC}			V
Logic Low				0.2 V_{CC}	V
Hi-Z Input Leakage	V_{CC} = 6V	−2		+2	μA
Input Pullup Current	V_{CC} = 6V, V_{IN} = 0V	−40		−250	μA
G and L Port Input Hysteresis				0.35 V_{CC}	V
Output Current Levels					
D Outputs					
Source	V_{CC} = 4V, V_{OH} = 3.3V	−0.4			mA
	V_{CC} = 2.5V, V_{OH} = 1.8V	−0.2			mA
Sink	V_{CC} = 4V, V_{OL} = 1V	10			mA
	V_{CC} = 2.5V, V_{OL} = 0.4V	2.0			mA
All Others					
Source (Weak Pull-Up Mode)	V_{CC} = 4V, V_{OH} = 2.7V	−10		−100	μA
	V_{CC} = 2.5V, V_{OH} = 1.8V	−2.5		−33	μA
Source (Push-Pull Mode)	V_{CC} = 4V, V_{OH} = 3.3V	−0.4			mA
	V_{CC} = 2.5V, V_{OH} = 1.8V	−0.2			mA
Sink (Push-Pull Mode)	V_{CC} = 4V, V_{OL} = 0.4V	1.6			mA
	V_{CC} = 2.5V, V_{OL} = 0.4V	0.7			mA
TRI-STATE Leakage	V_{CC} = 6.0V	−2		+2	μA

Note 1: Rate of voltage change must be less then 0.5 V/ms.

Note 2: Supply current is measured after running 2000 cycles with a crystal/resonator oscillator, inputs at rails and outputs open.

Note 3: The HALT mode will stop CKI from oscillating in the RC and the Crystal configurations. Test conditions: All inputs tied to V_{CC}, L, C, and G_0-G_5 configured as outputs and set high. The D port set to zero. The clock monitor and the comparators are disabled.

DC Electrical Characteristics 888EG: $-40°C \le T_A \le +85°C$ unless otherwise specified (Continued)

Parameter	Conditions	Min	Typ	Max	Units
Allowable Sink/Source Current per Pin					
D Outputs (Sink)				15	mA
All others				3	mA
Maximum Input Current without Latchup	$T_A = 25°C$			±100	mA
RAM Retention Voltage, V_r	500 ns Rise and Fall Time (Min)	2			V
Input Capacitance				7	pF
Load Capacitance on D2				1000	pF

AC Electrical Characteristics 888EG: $-40°C \le T_A \le +85°C$ unless otherwise specified

Parameter	Conditions	Min	Typ	Max	Units
Instruction Cycle Time (t_c)					
Crystal, Resonator,	$4V \le V_{CC} \le 6V$	1		DC	μs
R/C Oscillator	$2.5V \le V_{CC} < 4V$	2.5		DC	μs
	$4V \le V_{CC} \le 6V$	3		DC	μs
	$2.5V \le V_{CC} < 4V$	7.5		DC	μs
Inputs					
t_{SETUP}	$4V \le V_{CC} \le 6V$	200			ns
	$2.5V \le V_{CC} < 4V$	500			ns
t_{HOLD}	$4V \le V_{CC} \le 6V$	60			ns
	$2.5V \le V_{CC} < 4V$	150			ns
Output Propagation Delay (Note 4)	$R_L = 2.2k, C_L = 100$ pF				
t_{PD1}, t_{PD0}					
SO, SK	$4V \le V_{CC} \le 6V$			0.7	μs
	$2.5V \le V_{CC} < 4V$			1.75	μs
All Others	$4V \le V_{CC} \le 6V$			1	μs
	$2.5V \le V_{CC} < 4V$			2.5	μs
MICROWIRE™ Setup Time (t_{UWS})		20			ns
MICROWIRE Hold Time (t_{UWH})		56			ns
MICROWIRE Output Propagation Delay (t_{UPD})				220	ns
Input Pulse Width					
Interrupt Input High Time		1			t_c
Interrupt Input Low Time		1			t_c
Timer Input High Time		1			t_c
Timer Input Low Time		1			t_c
Reset Pulse Width		1			μs

t_c = Instruction cycle time.

Note 4: The output propagation delay is referenced to the end of the instruction cycle where the output change occurs.

Absolute Maximum Ratings

If Military/Aerospace specified devices are required, please contact the National Semiconductor Sales Office/Distributors for availability and specifications.

Supply Voltage (V_{CC})	7V
Voltage at Any Pin	−0.3V to V_{CC} + 0.3V
Total Current into V_{CC} Pin (Source)	100 mA
Total Current out of GND Pin (Sink)	110 mA
Storage Temperature Range	−65°C to +140°C

Note: *Absolute maximum ratings indicate limits beyond which damage to the device may occur. DC and AC electrical specifications are not ensured when operating the device at absolute maximum ratings.*

DC Electrical Characteristics 688EG: $-55°C \le T_A \le +125°C$ unless otherwise specified

Parameter	Conditions	Min	Typ	Max	Units
Operating Voltage		4.5		5.5	V
Power Supply Ripple (Note 1)	Peak-to-Peak			0.1 V_{CC}	V
Supply Current (Note 2)					
CKI = 10 MHz	V_{CC} = 5.5V, t_c = 1 μs			12.5	mA
CKI = 4 MHz	V_{CC} = 5.5V, t_c = 2.5 μs			5.5	mA
HALT Current (Note 3)	V_{CC} = 5.5V, CKI = 0 MHz		<10	30	μA
IDLE Current					
CKI = 10 MHz	V_{CC} = 5.5V, t_c = 1 μs			3.5	mA
CKI = 4 MHz	V_{CC} = 5.5V, t_c = 2.5 μs			2.5	mA
Input Levels					
$\overline{\text{RESET}}$					
Logic High		0.8 V_{CC}			V
Logic Low				0.2 V_{CC}	V
CKI (External and Crystal Osc. Modes)					
Logic High		0.7 V_{CC}			V
Logic Low				0.2 V_{CC}	V
All Other Inputs					
Logic High		0.7 V_{CC}			V
Logic Low				0.2 V_{CC}	V
Hi-Z Input Leakage	V_{CC} = 5.5V	−5		+5	μA
Input Pullup Current	V_{CC} = 5.5V, V_{IN} = 0V	−35		−400	μA
G and L Port Input Hysteresis				0.35 V_{CC}	V
Output Current Levels					
D Outputs					
Source	V_{CC} = 4.5V, V_{OH} = 3.3V	−0.4			mA
Sink	V_{CC} = 4.5V, V_{OL} = 1V	9			mA
All Others					
Source (Weak Pull-Up Mode)	V_{CC} = 4.5V, V_{OH} = 2.7V	−9		−140	μA
Source (Push-Pull Mode)	V_{CC} = 4.5V, V_{OH} = 3.3V	−0.4			mA
Sink (Push-Pull Mode)	V_{CC} = 4.5V, V_{OL} = 0.4V	1.4			mA
TRI-STATE Leakage	V_{CC} = 5.5V	−5		+5	μA

Note 1: Rate of voltage change must be less then 0.5 V/ms.

Note 2: Supply current is measured after running 2000 cycles with a crystal/resonator oscillator, inputs at rails and outputs open.

Note 3: The HALT mode will stop CKI from oscillating in the RC and the Crystal configurations. Test conditions: All inputs tied to V_{CC}, L, C, and G_0-G_5 configured as outputs and set high. The D port set to zero. The clock monitor and the comparators are disabled.

DC Electrical Characteristics 688EG: $-55°C \le T_A \le +125°C$ unless otherwise specified (Continued)

Parameter	Conditions	Min	Typ	Max	Units
Allowable Sink/Source Current per Pin					
D Outputs (Sink)				12	mA
All others				2.5	mA
Maximum Input Current without Latchup	$T_A = 25°C$			±100	mA
RAM Retention Voltage, V_r	500 ns Rise and Fall Time (Min)	2			V
Input Capacitance				7	pF
Load Capacitance on D2				1000	pF

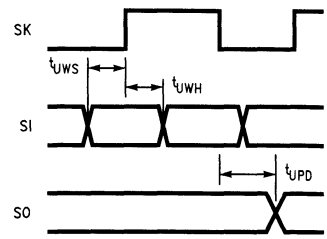
AC Electrical Characteristics 688EG: $-55°C \le T_A \le +125°C$ unless otherwise specified

Parameter	Conditions	Min	Typ	Max	Units
Instruction Cycle Time (t_c)					
Crystal, Resonator,	$V_{CC} \ge 4.5V$	1		DC	μs
R/C Oscillator	$V_{CC} \ge 4.5V$	3		DC	μs
Inputs					
t_{SETUP}	$V_{CC} \ge 4.5V$	200			ns
t_{HOLD}	$V_{CC} \ge 4.5V$	60			ns
Output Propagation Delay (Note 4)	$R_L = 2.2k$, $C_L = 100$ pF				
t_{PD1}, t_{PD0}					
SO, SK	$V_{CC} \ge 4.5V$			0.7	μs
All Others	$V_{CC} \ge 4.5V$			1	μs
MICROWIRE Setup Time (t_{UWS})		20			ns
MICROWIRE Hold Time (t_{UWH})		56			ns
MICROWIRE Output Propagation Delay (t_{UPD})				220	ns
Input Pulse Width					
Interrupt Input High Time		1			t_c
Interrupt Input Low Time		1			t_c
Timer Input High Time		1			t_c
Timer Input Low Time		1			t_c
Reset Pulse Width		1			μs

Note 4: The output propagation delay is referenced to the end of instruction cycle where the output change occurs.

Comparators AC and DC Characteristics $V_{CC} = 5V, T_A = 25°C$

Parameter	Conditions	Min	Typ	Max	Units
Input Offset Voltage	$0.4V \le V_{IN} \le V_{CC} - 1.5V$		±10	±25	mV
Input Common Mode Voltage Range		0.4		$V_{CC} - 1.5$	V
Low Level Output Current	$V_{OL} = 0.4V$	1.6			mA
High Level Output Current	$V_{OH} = 4.6V$	1.6			mA
DC Supply Current Per Comparator (When Enabled)				250	μA
Response Time	TBD mV Step, TBD mV Overdrive, 100 pF Load		1		μs

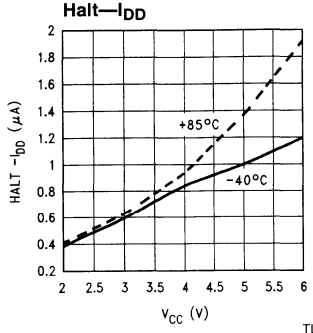


TL/DD/11214–5

FIGURE 2. MICROWIRE/PLUS Timing

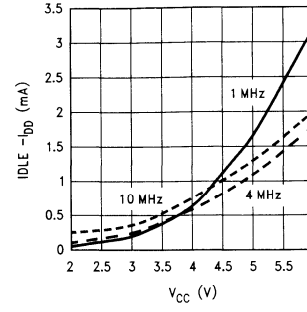
Typical Performance Characteristics ($-40°C \leq T_A \leq +85°C$)

Halt—I_{DD}



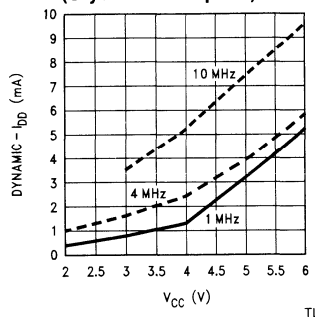
TL/DD/11214–7

Idle—I_{DD} (Crystal Clock Option)



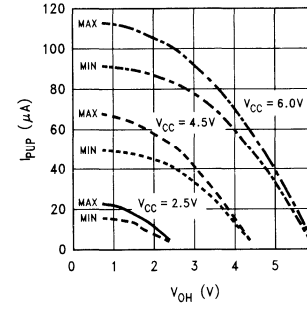
TL/DD/11214–8

Dynamic—I_{DD} vs V_{CC} (Crystal Clock Option)



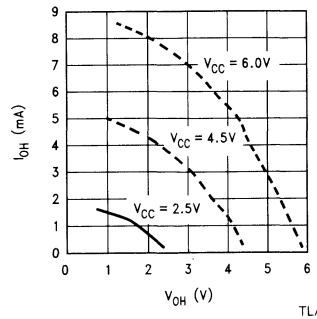
TL/DD/11214–9

Port L/C/G Weak Pull-Up Source Current



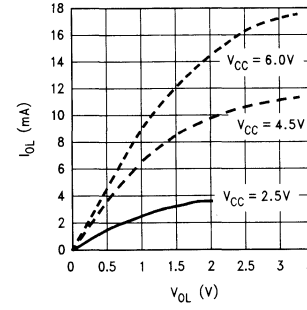
TL/DD/11214–10

Port L/C/G Push-Pull Source Current



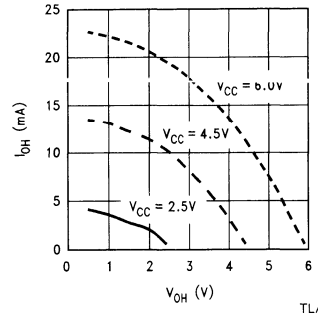
TL/DD/11214–11

Port L/C/G Push-Pull Sink Current



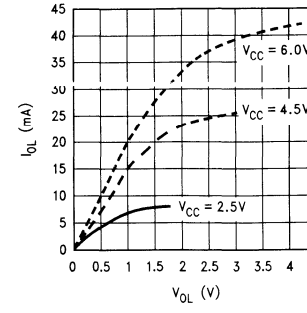
TL/DD/11214–12

Port D Source Current



TL/DD/11214–13

Port D Sink Current



TL/DD/11214–14

Pin Descriptions

V_{CC} and GND are the power supply pins.

CKI is the clock input. This can come from an R/C generated oscillator, or a crystal oscillator (in conjunction with CKO). See Oscillator Description section.

$\overline{\text{RESET}}$ is the master reset input. See Reset Description section.

The device contains three bidirectional 8-bit I/O ports (C, G and L), where each individual bit may be independently configured as an input (Schmitt trigger inputs on ports L and G), output or TRI-STATE under program control. Three data memory address locations are allocated for each of these I/O ports. Each I/O port has two associated 8-bit memory mapped registers, the CONFIGURATION register and the output DATA register. A memory mapped address is also reserved for the input pins of each I/O port. (See the memory map for the various addresses associated with the I/O ports.) *Figure 3* shows the I/O port configurations. The DATA and CONFIGURATION registers allow for each port bit to be individually configured under software control as shown below:

CONFIGURATION Register	DATA Register	Port Set-Up
0	0	Hi-Z Input (TRI-STATE Output)
0	1	Input with Weak Pull-Up
1	0	Push-Pull Zero Output
1	1	Push-Pull One Output

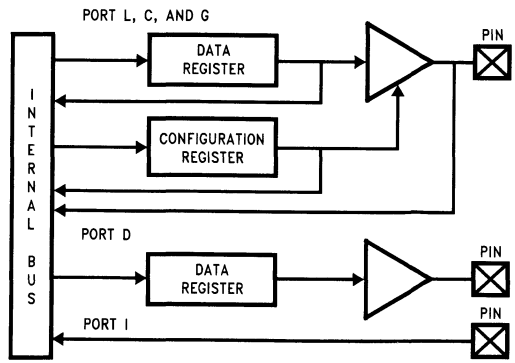
PORT L is an 8-bit I/O port. All L-pins have Schmitt triggers on the inputs.

The Port L supports Multi-Input Wake Up on all eight pins. L1 is used for the UART external clock. L2 and L3 are used for the UART transmit and receive. L4 and L5 are used for the timer input functions T2A and T2B. L6 and L7 are used for the timer input functions T3A and T3B.

The Port L has the following alternate features:

- L0 MIWU
- L1 MIWU or CKX
- L2 MIWU or TDX
- L3 MIWU or RDX
- L4 MIWU or T2A
- L5 MIWU or T2B
- L6 MIWU or T3A
- L7 MIWU or T3B

Port G is an 8-bit port with 5 I/O pins (G0, G2–G5), an input pin (G6), and two dedicated output pins (G1 and G7). Pins G0 and G2–G6 all have Schmitt Triggers on their inputs. Pin G1 serves as the dedicated WDOUT WATCHDOG output, while pin G7 is either input or output depending on the oscillator mask option selected. With the crystal oscillator option selected, G7 serves as the dedicated output pin for the CKO clock output. With the single-pin R/C oscillator mask option selected, G7 serves as a general purpose input pin but is also used to bring the device out of HALT mode with a low to high transition on G7. There are two registers associated with the G Port, a data register and a configuration register. Therefore, each of the 5 I/O bits (G0, G2–G5) can be individually configured under software control.



TL/DD/11214–6

FIGURE 3. I/O Port Configurations

5

Pin Descriptions (Continued)

Since G6 is an input only pin and G7 is the dedicated CKO clock output pin (crystal clock option) or general purpose input (R/C clock option), the associated bits in the data and configuration registers for G6 and G7 are used for special purpose functions as outlined below. Reading the G6 and G7 data bits will return zeros.

Note that the chip will be placed in the HALT mode by writing a "1" to bit 7 of the Port G Data Register. Similarly the chip will be placed in the IDLE mode by writing a "1" to bit 6 of the Port G Data Register.

Writing a "1" to bit 6 of the Port G Configuration Register enables the MICROWIRE/PLUS to operate with the alternate phase of the SK clock. The G7 configuration bit, if set high, enables the clock start up delay after HALT when the R/C clock configuration is used.

	Config Reg.	Data Reg.
G7	CLKDLY	HALT
G6	Alternate SK	IDLE

Port G has the following alternate features:

- G0 INTR (External Interrupt Input)
- G2 T1B (Timer T1 Capture Input)
- G3 T1A (Timer T1 I/O)
- G4 SO (MICROWIRE™ Serial Data Output)
- G5 SK (MICROWIRE Serial Clock)
- G6 SI (MICROWIRE Serial Data Input)

Port G has the following dedicated functions:

- G1 WDOUT WATCHDOG and/or Clock Monitor dedicated output
- G7 CKO Oscillator dedicated output or general purpose input

Port C is an 8-bit I/O port. The 40-pin device does not have a full complement of Port C pins. The unavailable pins are not terminated. A read operation for these unterminated pins will return unpredicatable values.

PORT I is an eight-bit Hi-Z input port. The 28-pin device does not have a full complement of Port I pins. The unavailable pins are not terminated i.e., they are floating. A read operation for these unterminated pins will return unpredictable values. The user must ensure that the software takes this into account by either masking or restricting the accesses to bit operations. The unterminated Port I pins will draw power only when addressed.

Port I1–I3 are used for Comparator 1. Port I4–I6 are used for Comparator 2.

The Port I has the following alternate features.

- I1 COMP1 − IN (Comparator 1 Negative Input)
- I2 COMP1 + IN (Comparator 1 Positive Input)
- I3 COMP1OUT (Comparator 1 Output)
- I4 COMP2 − IN (Comparator 2 Negative Input)
- I5 COMP2 + IN (Comparator 2 Positive Input)
- I6 COMP2OUT (Comparator 2 Output)

Port D is an 8-bit output port that is preset high when $\overline{\text{RESET}}$ goes low. The user can tie two or more D port outputs (except D2) together in order to get a higher drive.

Note: Care must be exercised with the D2 pin operation. At RESET, the external loads on this pin must ensure that the output voltages stay above 0.8 V_{CC} to prevent the chip from entering special modes. Also keep the external loading on D2 to less than 1000 pF.

Functional Description

The architecture of the device is modified Harvard architecture. With the Harvard architecture, the control store program memory (ROM) is separated from the data store memory (RAM). Both ROM and RAM have their own separate addressing space with separate address buses. The architecture, though based on Harvard architecture, permits transfer of data from ROM to RAM.

CPU REGISTERS

The CPU can do an 8-bit addition, subtraction, logical or shift operation in one instruction (t_c) cycle time.

There are six CPU registers:

A is the 8-bit Accumulator Register

PC is the 15-bit Program Counter Register

PU is the upper 7 bits of the program counter (PC)
PL is the lower 8 bits of the program counter (PC)

B is an 8-bit RAM address pointer, which can be optionally post auto incremented or decremented.

X is an 8-bit alternate RAM address pointer, which can be optionally post auto incremented or decremented.

SP is the 8-bit stack pointer, which points to the subroutine/interrupt stack (in RAM). The SP is initialized to RAM address 06F with reset.

S is the 8-bit Data Segment Address Register used to extend the lower half of the address range (00 to 7F) into 256 data segments of 128 bytes each.

All the CPU registers are memory mapped with the exception of the Accumulator (A) and the Program Counter (PC).

PROGRAM MEMORY

The program memory consists of 8192 bytes of ROM. These bytes may hold program instructions or constant data (data tables for the LAID instruction, jump vectors for the JID instruction, and interrupt vectors for the VIS instruction). The program memory is addressed by the 15-bit program counter (PC). All interrupts in the devices vector to program memory location 0FF Hex.

DATA MEMORY

The data memory address space includes the on-chip RAM and data registers, the I/O registers (Configuration, Data and Pin), the control registers, the MICROWIRE/PLUS SIO shift register, and the various registers, and counters associated with the timers (with the exception of the IDLE timer). Data memory is addressed directly by the instruction or indirectly by the B, X, SP pointers and S register.

The data memory consists of 256 bytes of RAM. Sixteen bytes of RAM are mapped as "registers" at addresses 0F0 to 0FF Hex. These registers can be loaded immediately, and also decremented and tested with the DRSZ (decrement register and skip if zero) instruction. The memory pointer registers X, SP, B and S are memory mapped into this space at address locations 0FC to 0FF Hex respectively, with the other registers being available for general usage.

The instruction set permits any bit in memory to be set, reset or tested. All I/O and registers (except A and PC) are memory mapped; therefore, I/O bits and register bits can be directly and individually set, reset and tested. The accumulator (A) bits can also be directly and individually tested.

Note: RAM contents are undefined upon power-up.

Data Memory Segment RAM Extension

Data memory address 0FF is used as a memory mapped location for the Data Segment Address Register (S).

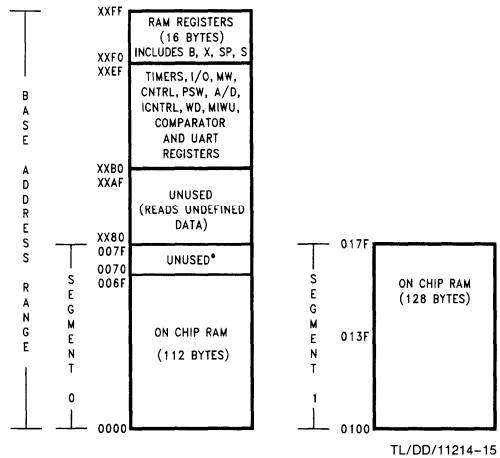
The data store memory is either addressed directly by a single byte address within the instruction, or indirectly relative to the reference of the B, X, or SP pointers (each contains a single-byte address). This single-byte address allows an addressing range of 256 locations from 00 to FF hex. The upper bit of this single-byte address divides the data store memory into two separate sections as outlined previously. With the exception of the RAM register memory from address locations 00F0 to 00FF, all RAM memory is memory mapped with the upper bit of the single-byte address being equal to zero. This allows the upper bit of the single-byte address to determine whether or not the base address range (from 0000 to 00FF) is extended. If this upper bit equals one (representing address range 0080 to 00FF), then address extension does not take place. Alternatively, if this upper bit equals zero, then the data segment extension register S is used to extend the base address range (from 0000 to 007F) from XX00 to XX7F, where XX represents the 8 bits from the S register. Thus the 128-byte data segment extensions are located from addresses 0100 to 017F for data segment 1, 0200 to 027F for data segment 2, etc., up to FF00 to FF7F for data segment 255. The base address range from 0000 to 007F represents data segment 0.

Figure 4 illustrates how the S register data memory extension is used in extending the lower half of the base address range (00 to 7F hex) into 256 data segments of 128 bytes each, with a total addressing range of 32 kbytes from XX00 to XX7F. This organization allows a total of 256 data segments of 128 bytes each with an additional upper base segment of 128 bytes. Furthermore, all addressing modes are available for all data segments. The S register must be changed under program control to move from one data segment (128 bytes) to another. However, the upper base segment (containing the 16 memory registers, I/O registers, control registers, etc.) is always available regardless of the contents of the S register, since the upper base segment (address range 0080 to 00FF) is independent of data segment extension.

The instructions that utilize the stack pointer (SP) always reference the stack as part of the base segment (Segment 0), regardless of the contents of the S register. The S register is not changed by these instructions. Consequently, the stack (used with subroutine linkage and interrupts) is always located in the base segment. The stack pointer will be initialized to point at data memory location 006F as a result of reset.

The 128 bytes of RAM contained in the base segment are split between the lower and upper base segments. The first 112 bytes of RAM are resident from address 0000 to 006F in the lower base segment, while the remaining 16 bytes of RAM represent the 16 data memory registers located at addresses 00F0 to 00FF of the upper base segment. No RAM is located at the upper sixteen addresses (0070 to 007F) of the lower base segment.

Additional RAM beyond these initial 128 bytes, however, will always be memory mapped in groups of 128 bytes (or less) at the data segment address extensions (XX00 to XX7F) of the lower base segment. The additional 128 bytes of RAM are memory mapped at address locations 0100 to 017F hex.



TL/DD/11214–15

*Reads as all ones.

FIGURE 4. RAM Organization

Reset

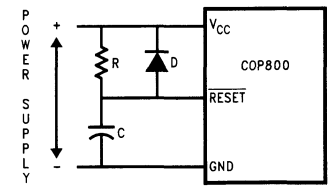
The $\overline{\text{RESET}}$ input when pulled low initializes the microcontroller. Initialization will occur whenever the $\overline{\text{RESET}}$ input is pulled low. Upon initialization, the data and configuration registers for ports L, G and C are cleared, resulting in these Ports being initialized to the TRI-STATE mode. Pin G1 of the G Port is an exception (as noted below) since pin G1 is dedicated as the WATCHDOG and/or Clock Monitor error output pin. Port D is set high. The PC, PSW, ICNTRL, CNTRL, T2CNTRL and T3CNTRL control registers are cleared. The UART registers PSR, ENU (except that TBMT bit is set), ENUR and ENUI are cleared. The Comparator Select Register is cleared. The S register is initialized to zero. The Multi-Input Wakeup registers WKEN, WKEDG and WKPND are cleared. The stack pointer, SP, is initialized to 6F Hex.

The device comes out of reset with both the WATCHDOG logic and the Clock Monitor detector armed, with the WATCHDOG service window bits set and the Clock Monitor bit set. The WATCHDOG and Clock Monitor circuits are inhibited during reset. The WATCHDOG service window bits being initialized high default to the maximum WATCHDOG service window of 64k t_C clock cycles. The Clock Monitor bit being initialized high will cause a Clock Monitor error following reset if the clock has not reached the minimum specified frequency at the termination of reset. A Clock Monitor error will cause an active low error output on pin G1. This error output will continue until 16 t_C–32 t_C clock cycles following the clock frequency reaching the minimum specified value, at which time the G1 output will enter the TRI-STATE mode.

The external RC network shown in *Figure 5* should be used to ensure that the $\overline{\text{RESET}}$ pin is held low until the power supply to the chip stabilizes.

5

Reset (Continued)



TL/DD/11214–16

RC > 5 × Power Supply Rise Time

FIGURE 5. Recommended Reset Circuit

Oscillator Circuits

The chip can be driven by a clock input on the CKI input pin which can be between DC and 10 MHz. The CKO output clock is on pin G7 (crystal configuration). The CKI input frequency is divided down by 10 to produce the instruction cycle clock ($1/t_c$).

Figure 6 shows the Crystal and R/C oscillator diagrams.

CRYSTAL OSCILLATOR

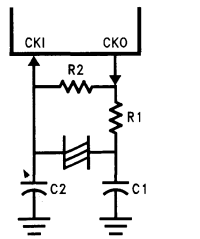
CKI and CKO can be connected to make a closed loop crystal (or resonator) controlled oscillator.

Table A shows the component values required for various standard crystal values.

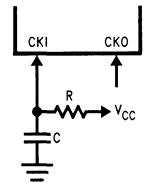
R/C OSCILLATOR

By selecting CKI as a single pin oscillator input, a single pin R/C oscillator circuit can be connected to it. CKO is available as a general purpose input, and/or HALT restart input.

Table B shows the variation in the oscillator frequencies as functions of the component (R and C) values.



TL/DD/11214–17



TL/DD/11214–18

FIGURE 6. Crystal and R/C Oscillator Diagrams

TABLE A. Crystal Oscillator Configuration, T_A = 25°C

R1 (kΩ)	R2 (MΩ)	C1 (pF)	C2 (pF)	CKI Freq (MHz)	Conditions
0	1	30	30–36	10	V_{CC} = 5V
0	1	30	30–36	4	V_{CC} = 5V
0	1	200	100–150	0.455	V_{CC} = 5V

TABLE B. RC Oscillator Configuration, T_A = 25°C

R (kΩ)	C (pF)	CKI Freq (MHz)	Instr. Cycle (μs)	Conditions
3.3	82	2.2 to 2.7	3.7 to 4.6	V_{CC} = 5V
5.6	100	1.1 to 1.3	7.4 to 9.0	V_{CC} = 5V
6.8	100	0.9 to 1.1	8.8 to 10.8	V_{CC} = 5V

Note: 3k ≤ R ≤ 200k

50 pF ≤ C ≤ 200 pF

Control Registers

CNTRL Register (Address X'00EE)

The Timer1 (T1) and MICROWIRE/PLUS control register contains the following bits:

SL1 & SL0	Select the MICROWIRE/PLUS clock divide by (00 = 2, 01 = 4, 1x = 8)
IEDG	External interrupt edge polarity select (0 = Rising edge, 1 = Falling edge)
MSEL	Selects G5 and G4 as MICROWIRE/PLUS signals SK and SO respectively
T1C0	Timer T1 Start/Stop control in timer modes 1 and 2
	Timer T1 Underflow Interrupt Pending Flag in timer mode 3
T1C1	Timer T1 mode control bit
T1C2	Timer T1 mode control bit
T1C3	Timer T1 mode control bit

T1C3	T1C2	T1C1	T1C0	MSEL	IEDG	SL1	SL0

Bit 7 Bit 0

Control Registers (Continued)

PSW Register (Address X'00EF)

The PSW register contains the following select bits:

- GIE Global interrupt enable (enables interrupts)
- EXEN Enable external interrupt
- BUSY MICROWIRE/PLUS busy shifting flag
- EXPND External interrupt pending
- T1ENA Timer T1 Interrupt Enable for Timer Underflow or T1A Input capture edge
- T1PNDA Timer T1 Interrupt Pending Flag (Autoreload RA in mode 1, T1 Underflow in Mode 2, T1A capture edge in mode 3)
- C Carry Flag
- HC Half Carry Flag

HC	C	T1PNDA	T1ENA	EXPND	BUSY	EXEN	GIE

Bit 7 Bit 0

The Half-Carry bit is also affected by all the instructions that affect the Carry flag. The SC (Set Carry) and RC (Reset Carry) instructions will respectively set or clear both the carry flags. In addition to the SC and RC instructions, ADC, SUBC, RRC and RLC instructions affect the carry and Half Carry flags.

ICNTRL Register (Address X'00E8)

The ICNTRL register contains the following bits:

- T1ENB Timer T1 Interrupt Enable for T1B Input capture edge
- T1PNDB Timer T1 Interrupt Pending Flag for T1B capture edge
- μWEN Enable MICROWIRE/PLUS interrupt
- μWPND MICROWIRE/PLUS interrupt pending
- T0EN Timer T0 Interrupt Enable (Bit 12 toggle)
- T0PND Timer T0 Interrupt pending
- LPEN L Port Interrupt Enable (Multi-Input Wakeup/Interrupt)

  Bit 7 could be used as a flag

Unused	LPEN	T0PND	T0EN	μWPND	μWEN	T1PNDB	T1ENB

Bit 7 Bit 0

T2CNTRL Register (Address X'00C6)

The T2CNTRL register contains the following bits:

- T2ENB Timer T2 Interrupt Enable for T2B Input capture edge
- T2PNDB Timer T2 Interrupt Pending Flag for T2B capture edge
- T2ENA Timer T2 Interrupt Enable for Timer Underflow or T2A Input capture edge
- T2PNDA Timer T2 Interrupt Pending Flag (Autoreload RA in mode 1, T2 Underflow in mode 2, T2A capture edge in mode 3)
- T2C0 Timer T2 Start/Stop control in timer modes 1 and 2 Timer T2 Underflow Interrupt Pending Flag in timer mode 3
- T2C1 Timer T2 mode control bit
- T2C2 Timer T2 mode control bit
- T2C3 Timer T2 mode control bit

T2C3	T2C2	T2C1	T2C0	T2PNDA	T2ENA	T2PNDB	T2ENB

Bit 7 Bit 0

T3CNTRL Register (Address X'00B6)

The T3CNTRL register contains the following bits:

- T3ENB Timer T3 Interrupt Enable for T3B
- T3PNDB Timer T3 Interrupt Pending Flag for T3B pin (T3B capture edge)
- T3ENA Timer T3 Interrupt Enable for Timer Underflow or T3A pin
- T3PNDA Timer T3 Interrupt Pending Flag (Autoload RA in mode 1, T3 Underflow in mode 2, T3a capture edge in mode 3)
- T3C0 Timer T3 Start/Stop control in timer modes 1 and 2

  Timer T3 Underflow Interrupt Pending Flag in timer mode 3
- T3C1 Timer T3 mode control bit
- T3C2 Timer T3 mode control bit
- T3C3 Timer T3 mode control bit

T3C3	T3C2	T3C1	T3C0	T3PNDA	T3ENA	T3PNDB	T3ENB

Bit 7 Bit 0

Timers

The device contains a very versatile set of timers (T0, T1, T2, T3). All timers and associated autoreload/capture registers power up containing random data.

TIMER T0 (IDLE TIMER)

The device supports applications that require maintaining real time and low power with the IDLE mode. This IDLE mode support is furnished by the IDLE timer T0, which is a 16-bit timer. The Timer T0 runs continuously at the fixed rate of the instruction cycle clock, t_C. The user cannot read or write to the IDLE Timer T0, which is a count down timer.

The Timer T0 supports the following functions:

Exit out of the Idle Mode (See Idle Mode description)
WATCHDOG logic (See WATCHDOG description)
Start up delay out of the HALT mode

The IDLE Timer T0 can generate an interrupt when the thirteenth bit toggles. This toggle is latched into the T0PND pending flag, and will occur every 4 ms at the maximum clock frequency (t_C = 1 μs). A control flag T0EN allows the interrupt from the thirteenth bit of Timer T0 to be enabled or disabled. Setting T0EN will enable the interrupt, while resetting it will disable the interrupt.

5

Timers (Continued)

TIMER T1, TIMER T2 AND TIMER T3

The device has a set of three powerful timer/counter blocks, T1, T2 and T3. The associated features and functioning of a timer block are described by referring to the timer block Tx. Since the three timer blocks, T1, T2 and T3 are identical, all comments are equally applicable to any of the three timer blocks.

Each timer block consists of a 16-bit timer, Tx, and two supporting 16-bit autoreload/capture registers, RxA and RxB. Each timer block has two pins associated with it, TxA and TxB. The pin TxA supports I/O required by the timer block, while the pin TxB is an input to the timer block. The powerful and flexible timer block allows the device to easily perform all timer functions with minimal software overhead. The timer block has three operating modes: Processor Independent PWM mode, External Event Counter mode, and Input Capture mode.

The control bits TxC3, TxC2, and TxC1 allow selection of the different modes of operation.

Mode 1. Processor Independent PWM Mode

As the name suggests, this mode allows the device to generate a PWM signal with very minimal user intervention. The user only has to define the parameters of the PWM signal (ON time and OFF time). Once begun, the timer block will continuously generate the PWM signal completely independent of the microcontroller. The user software services the timer block only when the PWM parameters require updating.

In this mode the timer Tx counts down at a fixed rate of t_C. Upon every underflow the timer is alternately reloaded with the contents of supporting registers, RxA and RxB. The very first underflow of the timer causes the timer to reload from the register RxA. Subsequent underflows cause the timer to be reloaded from the registers alternately beginning with the register RxB.

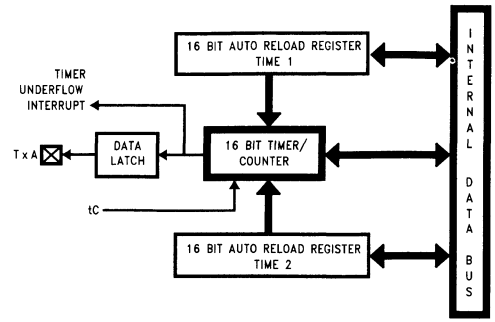
The Tx Timer control bits, TxC3, TxC2 and TxC1 set up the timer for PWM mode operation.

Figure 7 shows a block diagram of the timer in PWM mode.

The underflows can be programmed to toggle the TxA output pin. The underflows can also be programmed to generate interrupts.

Underflows from the timer are alternately latched into two pending flags, TxPNDA and TxPNDB. The user must reset these pending flags under software control. Two control enable flags, TxENA and TxENB, allow the interrupts from the timer underflow to be enabled or disabled. Setting the timer enable flag TxENA will cause an interrupt when a timer underflow causes the RxA register to be reloaded into the timer. Setting the timer enable flag TxENB will cause an interrupt when a timer underflow causes the RxB register to be reloaded into the timer. Resetting the timer enable flags will disable the associated interrupts.

Either or both of the timer underflow interrupts may be enabled. This gives the user the flexibility of interrupting once per PWM period on either the rising or falling edge of the PWM output. Alternatively, the user may choose to interrupt on both edges of the PWM output.



TL/DD/11214–19

FIGURE 7. Timer in PWM Mode

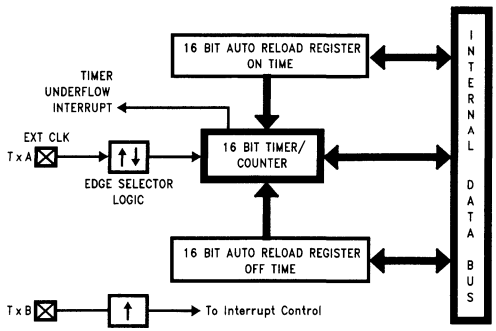
Mode 2. External Event Counter Mode

This mode is quite similar to the processor independent PWM mode described above. The main difference is that the timer, Tx, is clocked by the input signal from the TxA pin. The Tx timer control bits, TxC3, TxC2 and TxC1 allow the timer to be clocked either on a positive or negative edge from the TxA pin. Underflows from the timer are latched into the TxPNDA pending flag. Setting the TxENA control flag will cause an interrupt when the timer underflows.

In this mode the input pin TxB can be used as an independent positive edge sensitive interrupt input if the TxENB control flag is set. The occurrence of a positive edge on the TxB input pin is latched into the TxPNDB flag.

Figure 8 shows a block diagram of the timer in External Event Counter mode.

Note: The PWM output is not available in this mode since the TxA pin is being used as the counter input clock.



TL/DD/11214–20

FIGURE 8. Timer in External Event Counter Mode

Mode 3. Input Capture Mode

The device can precisely measure external frequencies or time external events by placing the timer block, Tx, in the input capture mode.

In this mode, the timer Tx is constantly running at the fixed t_C rate. The two registers, RxA and RxB, act as capture registers. Each register acts in conjunction with a pin. The register RxA acts in conjunction with the TxA pin and the register RxB acts in conjunction with the TxB pin.

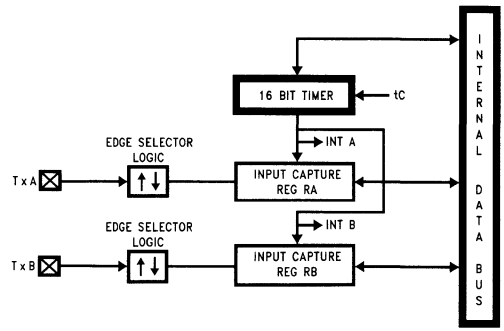
Timers (Continued)

The timer value gets copied over into the register when a trigger event occurs on its corresponding pin. Control bits, TxC3, TxC2 and TxC1, allow the trigger events to be specified either as a positive or a negative edge. The trigger condition for each input pin can be specified independently.

The trigger conditions can also be programmed to generate interrupts. The occurrence of the specified trigger condition on the TxA and TxB pins will be respectively latched into the pending flags, TxPNDA and TxPNDB. The control flag TxENA allows the interrupt on TxA to be either enabled or disabled. Setting the TxENA flag enables interrupts to be generated when the selected trigger condition occurs on the TxA pin. Similarly, the flag TxENB controls the interrupts from the TxB pin.

Underflows from the timer can also be programmed to generate interrupts. Underflows are latched into the timer TxC0 pending flag (the TxC0 control bit serves as the timer underflow interrupt pending flag in the Input Capture mode). Consequently, the TxC0 control bit should be reset when entering the Input Capture mode. The timer underflow interrupt is enabled with the TxENA control flag. When a TxA interrupt occurs in the Input Capture mode, the user must check both the TxPNDA and TxC0 pending flags in order to determine whether a TxA input capture or a timer underflow (or both) caused the interrupt.

Figure 9 shows a block diagram of the timer in Input Capture mode.



TL/DD/11214–21

FIGURE 9. Timer in Input Capture Mode

TIMER CONTROL FLAGS

The timers T1, T2 and T3 have indentical control structures. The control bits and their functions are summarized below.

TxC0	Timer Start/Stop control in Modes 1 and 2 (Processor Independent PWM and External Event Counter), where 1 = Start, 0 = Stop Timer Underflow Interrupt Pending Flag in Mode 3 (Input Capture)
TxPNDA	Timer Interrupt Pending Flag
TxPNDB	Timer Interrupt Pending Flag
TxENA	Timer Interrupt Enable Flag
TxENB	Timer Interrupt Enable Flag 1 = Timer Interrupt Enabled 0 = Timer Interrupt Disabled
TxC3	Timer mode control
TxC2	Timer mode control
TxC1	Timer mode control

Timers (Continued)

The timer mode control bits (TxC3, TxC2 and TxC1) are detailed below:

TxC3	TxC2	TxC1	Timer Mode	Interrupt A Source	Interrupt B Source	Timer Counts On
0	0	0	MODE 2 (External Event Counter)	Timer Underflow	Pos. TxB Edge	TxA Pos. Edge
0	0	1	MODE 2 (External Event Counter)	Timer Underflow	Pos. TxB Edge	TxA Neg. Edge
1	0	1	MODE 1 (PWM) TxA Toggle	Autoreload RA	Autoreload RB	t_C
1	0	0	MODE 1 (PWM) No TxA Toggle	Autoreload RA	Autoreload RB	t_C
0	1	0	MODE 3 (Capture) Captures: TxA Pos. Edge TxB Pos. Edge	Pos. TxA Edge or Timer Underflow	Pos. TxB Edge	t_C
1	1	0	MODE 3 (Capture) Captures: TxA Pos. Edge TxB Neg. Edge	Pos. TxA Edge or Timer Underflow	Neg. TxB Edge	t_C
0	1	1	MODE 3 (Capture) Captures: TxA Neg. Edge TxB Pos. Edge	Neg. TxA Edge or Timer Underflow	Pos. TxB Edge	t_C
1	1	1	MODE 3 (Capture) Captures: TxA Neg. Edge TxB Neg. Edge	Neg. TxA Edge or Timer Underflow	Neg. TxB Edge	t_C

Power Save Modes

The device offers the user two power save modes of operation: HALT and IDLE. In the HALT mode, all microcontroller activities are stopped. In the IDLE mode, the on-board oscillator circuitry the WATCHDOG logic, the Clock Monitor and timer T0 are active but all other microcontroller activities are stopped. In either mode, all on-board RAM, registers, I/O states, and timers (with the exception of T0) are unaltered.

HALT MODE

The device can be placed in the HALT mode by writing a "1" to the HALT flag (G7 data bit). All microcontroller activities, including the clock and timers, are stopped. The WATCHDOG logic is disabled during the HALT mode. However, the clock monitor circuitry if enabled remains active and will cause the WATCHDOG output pin (WDOUT) to go low. If the HALT mode is used and the user does not want to activate the WDOUT pin, the Clock Monitor should be disabled after the device comes out of reset (resetting the Clock Monitor control bit with the first write to the WDSVR register). In the HALT mode, the power requirements of the device are minimal and the applied voltage (V_{CC}) may be decreased to V_r (V_r = 2.0V) without altering the state of the machine.

The device supports three different ways of exiting the HALT mode. The first method of exiting the HALT mode is with the Multi-Input Wakeup feature on the L port. The second method is with a low to high transition on the CKO (G7) pin. This method precludes the use of the crystal clock configuration (since CKO becomes a dedicated output), and so may be used with an RC clock configuration. The third method of exiting the HALT mode is by pulling the $\overline{\text{RESET}}$ pin low.

Since a crystal or ceramic resonator may be selected as the oscillator, the Wakeup signal is not allowed to start the chip running immediately since crystal oscillators and ceramic resonators have a delayed start up time to reach full amplitude and frequency stability. The IDLE timer is used to generate a fixed delay to ensure that the oscillator has indeed stabilized before allowing instruction execution. In this case, upon detecting a valid Wakeup signal, only the oscillator circuitry is enabled. The IDLE timer is loaded with a value of 256 and is clocked with the t_C instruction cycle clock. The t_C clock is derived by dividing the oscillator clock down by a factor of 10. The Schmitt trigger following the CKI inverter on the chip ensures that the IDLE timer is clocked only when the oscillator has a sufficiently large amplitude to meet the Schmitt trigger specifications. This Schmitt trigger is not part of the oscillator closed loop. The startup timeout from the IDLE timer enables the clock signals to be routed to the rest of the chip.

If an RC clock option is being used, the fixed delay is introduced optionally. A control bit, CLKDLY, mapped as configuration bit G7, controls whether the delay is to be introduced or not. The delay is included if CLKDLY is set, and excluded if CLKDLY is reset. The CLKDLY bit is cleared on reset.

Power Save Modes (Continued)

The device has two mask options associated with the HALT mode. The first mask option enables the HALT mode feature, while the second mask option disables the HALT mode. With the HALT mode enable mask option, the device will enter and exit the HALT mode as described above. With the HALT disable mask option, the device cannot be placed in the HALT mode (writing a "1" to the HALT flag will have no effect).

The WATCHDOG detector circuit is inhibited during the HALT mode. However, the clock monitor circuit if enabled remains active during HALT mode in order to ensure a clock monitor error if the device inadvertently enters the HALT mode as a result of a runaway program or power glitch.

IDLE MODE

The device is placed in the IDLE mode by writing a "1" to the IDLE flag (G6 data bit). In this mode, all activities, except the associated on-board oscillator circuitry, the WATCHDOG logic, the clock monitor and the IDLE Timer T0, are stopped.

As with the HALT mode, the device can be returned to normal operation with a reset, or with a Multi-Input Wakeup from the L Port. Alternately, the microcontroller resumes normal operation from the IDLE mode when the thirteenth bit (representing 4.096 ms at internal clock frequency of 1 MHz, t_c = 1 μs) of the IDLE Timer toggles.

This toggle condition of the thirteenth bit of the IDLE Timer T0 is latched into the T0PND pending flag.

The user has the option of being interrupted with a transition on the thirteenth bit of the IDLE Timer T0. The interrupt can be enabled or disabled via the T0EN control bit. Setting the T0EN flag enables the interrupt and vice versa.

The user can enter the IDLE mode with the Timer T0 interrupt enabled. In this case, when the T0PND bit gets set, the device will first execute the Timer T0 interrupt service routine and then return to the instruction following the "Enter Idle Mode" instruction.

Alternatively, the user can enter the IDLE mode with the IDLE Timer T0 interrupt disabled. In this case, the device will resume normal operation with the instruction immediately following the "Enter IDLE Mode" instruction.

Note: It is necessary to program two NOP instructions following both the set HALT mode and set IDLE mode instructions. These NOP instructions are necessary to allow clock resynchronization following the HALT or IDLE modes.

Multi-Input Wakeup

The Multi-Input Wakeup feature is ued to return (wakeup) the device from either the HALT or IDLE modes. Alternately Multi-Input Wakeup/Interrupt feature may also be used to generate up to 8 edge selectable external interrupts.

Figure 10 shows the Multi-Input Wakeup logic.

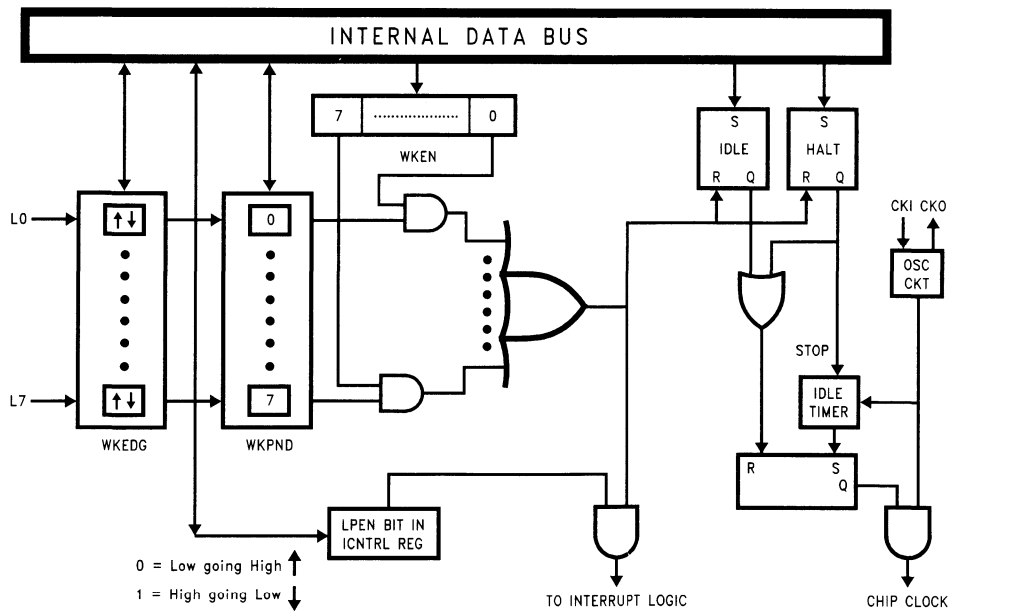


FIGURE 10. Multi-Input Wake Up Logic

Multi-Input Wakeup (Continued)

The Multi-Input Wakeup feature utilizes the L Port. The user selects which particular L port bit (or combination of L Port bits) will cause the device to exit the HALT or IDLE modes. The selection is done through the Reg: WKEN. The Reg: WKEN is an 8-bit read/write register, which contains a control bit for every L port bit. Setting a particular WKEN bit enables a Wakeup from the associated L port pin.

The user can select whether the trigger condition on the selected L Port pin is going to be either a positive edge (low to high transition) or a negative edge (high to low transition). This selection is made via the Reg: WKEDG, which is an 8-bit control register with a bit assigned to each L Port pin. Setting the control bit will select the trigger condition to be a negative edge on that particular L Port pin. Resetting the bit selects the trigger condition to be a positive edge. Changing an edge select entails several steps in order to avoid a pseudo Wakeup condition as a result of the edge change. First, the associated WKEN bit should be reset, followed by the edge select change in WKEDG. Next, the associated WKPND bit should be cleared, followed by the associated WKEN bit being re-enabled.

An example may serve to clarify this procedure. Suppose we wish to change the edge select from positive (low going high) to negative (high going low) for L Port bit 5, where bit 5 has previously been enabled for an input interrupt. The program would be as follows:

```
RBIT  5, WKEN
SBIT  5, WKEDG
RBIT  5, WKPND
SBIT  5, WKEN
```

If the L port bits have been used as outputs and then changed to inputs with Multi-Input Wakeup/Interrupt, a safety procedure should also be followed to avoid inherited pseudo wakeup conditions. After the selected L port bits have been changed from output to input but before the associated WKEN bits are enabled, the associated edge select bits in WKEDG should be set or reset for the desired edge selects, followed by the associated WKPND bits being cleared.

This same procedure should be used following reset, since the L port inputs are left floating as a result of reset.

The occurrence of the selected trigger condition for Multi-Input Wakeup is latched into a pending register called WKPND. The respective bits of the WKPND register will be set on the occurrence of the selected trigger edge on the corresponding Port L pin. The user has the responsibility of clearing these pending flags. Since WKPND is a pending register for the occurrence of selected wakeup conditions, the device will not enter the HALT mode if any Wakeup bit is both enabled and pending. Consequently, the user has the responsibility of clearing the pending flags before attempting to enter the HALT mode.

WKEN, WKPND and WKEDG are all read/write registers, and are cleared at reset.

PORT L INTERRUPTS

Port L provides the user with an additional eight fully selectable, edge sensitive interrupts which are all vectored into the same service subroutine.

The interrupt from Port L shares logic with the wake up circuitry. The register WKEN allows interrupts from Port L to be individually enabled or disabled. The register WKEDG specifies the trigger condition to be either a positive or a negative edge. Finally, the register WKPND latches in the pending trigger conditions.

The GIE (Global Interrupt Enable) bit enables the interrupt function.

A control flag, LPEN, functions as a global interrupt enable for Port L interrupts. Setting the LPEN flag will enable interrupts and vice versa. A separate global pending flag is not needed since the register WKPND is adequate.

Since Port L is also used for waking the device out of the HALT or IDLE modes, the user can elect to exit the HALT or IDLE modes either with or without the interrupt enabled. If he elects to disable the interrupt, then the device will restart execution from the instruction immediately following the instruction that placed the microcontroller in the HALT or IDLE modes. In the other case, the device will first execute the interrupt service routine and then revert to normal operation.

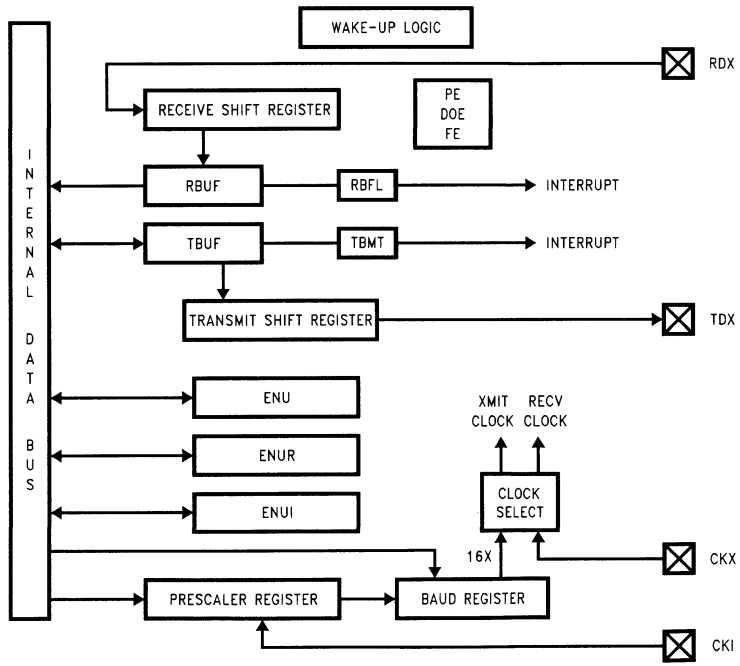
The Wakeup signal will not start the chip running immediately since crystal oscillators or ceramic resonators have a finite start up time. The IDLE Timer (T0) generates a fixed delay to ensure that the oscillator has indeed stabilized before allowing the device to execute instructions. In this case, upon detecting a valid Wakeup signal, only the oscillator circuitry and the IDLE Timer T0 are enabled. The IDLE Timer is loaded with a value of 256 and is clocked from the t_C instruction cycle clock. The t_C clock is derived by dividing down the oscillator clock by a factor of 10. A Schmitt trigger following the CKI on-chip inverter ensures that the IDLE timer is clocked only when the oscillator has a sufficiently large amplitude to meet the Schmitt trigger specifications. This Schmitt trigger is not part of the oscillator closed loop. The startup timeout from the IDLE timer enables the clock signals to be routed to the rest of the chip.

If the RC clock option is used, the fixed delay is under software control. A control flag, CLKDLY, in the G7 configuration bit allows the clock start up delay to be optionally inserted. Setting CLKDLY flag high will cause clock start up delay to be inserted and resetting it will exclude the clock start up delay. The CLKDLY flag is cleared during reset, so the clock start up delay is not present following reset with the RC clock options.

UART

The device contains a full-duplex software programmable UART. The UART *(Figure 11)* consists of a transmit shift register, a receiver shift register and seven addressable registers, as follows: a transmit buffer register (TBUF), a receiver buffer register (RBUF), a UART control and status register (ENU), a UART receive control and status register (ENUR), a UART interrupt and clock source register (ENUI), a prescaler select register (PSR) and baud (BAUD) register. The ENU register contains flags for transmit and receive functions; this register also determines the length of the data frame (7, 8 or 9 bits), the value of the ninth bit in transmission, and parity selection bits. The ENUR register flags framming, data overrun and parity errors while the UART is receiving.

Other functions of the ENUR register include saving the ninth bit received in the data frame, enabling or disabling the UART's attention mode of operation and providing additional receiver/transmitter status information via RCVG and XMTG bits. The determination of an internal or external clock source is done by the ENUI register, as well as selecting the number of stop bits and enabling or disabling transmit and receive interrupts. A control flag in this register can also select the UART mode of operation: asynchronous or synchronous.



TL/DD/11214–23

FIGURE 11. UART Block Diagram

UART (Continued)

UART CONTROL AND STATUS REGISTERS

The operation of the UART is programmed through three registers: ENU, ENUR and ENUI. The function of the individual bits in these registers is as follows:

ENU-UART Control and Status Register (Address at 0BA)

PEN	PSEL1	XBIT9/ PSEL0	CHL1	CHL0	ERR	RBFL	TBMT
0RW	0RW	0RW	0RW	0RW	0R	0R	1R

Bit 7 Bit 0

ENUR-UART Receive Control and Status Register (Address at 0BB)

DOE	FE	PE	SPARE	RBIT9	ATTN	XMTG	RCVG
0RD	0RD	0RD	0RW*	0R	0RW	0R	0R

Bit7 Bit0

ENUI-UART Interrupt and Clock Source Register (Address at 0BC)

STP2	STP78	ETDX	SSEL	XRCLK	XTCLK	ERI	ETI
0RW	0RW	0RW	0RW	0RW	0RW	0RW	0RW

Bit7 Bit0

*Bit is not used.

0 Bit is cleared on reset.

1 Bit is set to one on reset.

R Bit is read-only; it cannot be written by software.

RW Bit is read/write.

D Bit is cleared on read; when read by software as a one, it is cleared automatically. Writing to the bit does not affect its state.

DESCRIPTION OF UART REGISTER BITS

ENU—UART CONTROL AND STATUS REGISTER

TBMT: This bit is set when the UART transfers a byte of data from the TBUF register into the TSFT register for transmission. It is automatically reset when software writes into the TBUF register.

RBFL: This bit is set when the UART has received a complete character and has copied it into the RBUF register. It is automatically reset when software reads the character from RBUF.

ERR: This bit is a global UART error flag which gets set if any or a combination of the errors (DOE, FE, PE) occur.

CHL1, CHL0: These bits select the character frame format. Parity is not included and is generated/verified by hardware.

CHL1 = 0, CHL0 = 0 The frame contains eight data bits.
CHL1 = 0, CHL0 = 1 The frame contains seven data bits.
CHL1 = 1, CHL0 = 0 The frame contains nine data bits.
CHL1 = 1, CHL0 = 1 Loopback Mode selected. Transmitter output internally looped back to receiver input. Nine bit framing format is used.

XBIT9/PSEL0: Programs the ninth bit for transmission when the UART is operating with nine data bits per frame. For seven or eight data bits per frame, this bit in conjunction with PSEL1 selects parity.

PSEL1, PSEL0: Parity select bits.

PSEL1 = 0, PSEL0 = 0 Odd Parity (if Parity enabled)
PSEL1 = 0, PSEL0 = 1 Even Parity (if Parity enabled)
PSEL1 = 1, PSEL0 = 0 Mark(1) (if Parity enabled)
PSEL1 = 1, PSEL0 = 1 Space(0) (if Parity enabled)

PEN: This bit enables/disables Parity (7- and 8-bit modes only).

PEN = 0 Parity disabled.
PEN = 1 Parity enabled.

ENUR—UART RECEIVE CONTROL AND STATUS REGISTER

RCVG: This bit is set high whenever a framing error occurs and goes low when RDX goes high.

XMTG: This bit is set to indicate that the UART is transmitting. It gets reset at the end of the last frame (end of last Stop bit).

ATTN: ATTENTION Mode is enabled while this bit is set. This bit is cleared automatically on receiving a character with data bit nine set.

RBIT9: Contains the ninth data bit received when the UART is operating with nine data bits per frame.

SPARE: Reserved for future use.

PE: Flags a Parity Error.

PE = 0 Indicates no Parity Error has been detected since the last time the ENUR register was read.
PE = 1 Indicates the occurrence of a Parity Error.

FE: Flags a Framing Error.

FE = 0 Indicates no Framing Error has been detected since the last time the ENUR register was read.
FE = 1 Indicates the occurrence of a Framing Error.

DOE: Flags a Data Overrun Error.

DOE = 0 Indicates no Data Overrun Error has been detected since the last time the ENUR register was read.
DOE = 1 Indicates the occurrence of a Data Overrun Error.

ENUI—UART INTERRUPT AND CLOCK SOURCE REGISTER

ETI: This bit enables/disables interrupt from the transmitter section.

ETI = 0 Interrupt from the transmitter is disabled.
ETI = 1 Interrupt from the transmitter is enabled.

ERI: This bit enables/disables interrupt from the receiver section.

ERI = 0 Interrupt from the receiver is disabled.
ERI = 1 Interrupt from the receiver is enabled.

XTCLK: This bit selects the clock source for the transmitter-section.

XTCLK = 0 The clock source is selected through the PSR and BAUD registers.
XTCLK = 1 Signal on CKX (L1) pin is used as the clock.

XRCLK: This bit selects the clock source for the receiver section.

XRCLK = 0 The clock source is selected through the PSR and BAUD registers.
XRCLK = 1 Signal on CKX (L1) pin is used as the clock.

SSEL: UART mode select.

SSEL = 0 Asynchronous Mode.
SSEL = 1 Synchronous Mode.

UART (Continued)

ETDX: TDX (UART Transmit Pin) is the alternate function assigned to Port L pin L2; it is selected by setting ETDX bit. To simulate line break generation, software should reset ETDX bit and output logic zero to TDX pin through Port L data and configuration registers.

STP78: This bit is set to program the last Stop bit to be 7/8th of a bit in length.

STP2: This bit programs the number of Stop bits to be transmitted.

STP2 = 0 One Stop bit transmitted.
STP2 = 1 Two Stop bits transmitted.

Associated I/O Pins

Data is transmitted on the TDX pin and received on the RDX pin. TDX is the alternate function assigned to Port L pin L2; it is selected by setting ETDX (in the ENUI register) to one. RDX is an inherent function of Port L pin L3, requiring no setup.

The baud rate clock for the UART can be generated on-chip, or can be taken from an external source. Port L pin L1 (CKX) is the external clock I/O pin. The CKX pin can be either an input or an output, as determined by Port L Configuration and Data registers (Bit 1). As an input, it accepts a clock signal which may be selected to drive the transmitter and/or receiver. As an output, it presents the internal Baud Rate Generator output.

UART Operation

The UART has two modes of operation: asynchronous mode and synchronous mode.

ASYNCHRONOUS MODE

This mode is selected by resetting the SSEL (in the ENUI register) bit to zero. The input frequency to the UART is 16 times the baud rate.

The TSFT and TBUF registers double-buffer data for transmission. While TSFT is shifting out the current character on the TDX pin, the TBUF register may be loaded by software with the next byte to be transmitted. When TSFT finishes transmitting the current character the contents of TBUF are transferred to the TSFT register and the Transmit Buffer Empty Flag (TBMT in the ENU register) is set. The TBMT flag is automatically reset by the UART when software loads a new character into the TBUF register. There is also the XMTG bit which is set to indicate that the UART is transmitting. This bit gets reset at the end of the last frame (end of last Stop bit). TBUF is a read/write register.

The RSFT and RBUF registers double-buffer data being received. The UART receiver continually monitors the signal on the RDX pin for a low level to detect the beginning of a Start bit. Upon sensing this low level, it waits for half a bit time and samples again. If the RDX pin is still low, the receiver considers this to be a valid Start bit, and the remaining bits in the character frame are each sampled a single time, at the mid-bit position. Serial data input on the RDX pin is shifted into the RSFT register. Upon receiving the complete character, the contents of the RSFT register are copied into the RBUF register and the Received Buffer Full Flag (RBFL) is set. RBFL is automatically reset when software reads the character from the RBUF register. RBUF is a read only register. There is also the RCVG bit which is set high when a framing error occurs and goes low once RDX goes high. TBMT, XMTG, RBFL and RCVG are read only bits.

SYNCHRONOUS MODE

In this mode data is transferred synchronously with the clock. Data is transmitted on the rising edge and received on the falling edge of the synchronous clock.

This mode is selected by setting SSEL bit in the ENUI register. The input frequency to the UART is the same as the baud rate.

When an external clock input is selected at the CKX pin, data transmit and receive are performed synchronously with this clock through TDX/RDX pins.

If data transmit and receive are selected with the CKX pin as clock output, the device generates the synchronous clock output at the CKX pin. The internal baud rate generator is used to produce the synchronous clock. Data transmit and receive are performed synchronously with this clock.

FRAMING FORMATS

The UART supports several serial framing formats *(Figure 12).* The format is selected using control bits in the ENU, ENUR and ENUI registers.

The first format (1, 1a, 1b, 1c) for data transmission (CHL0 = 1, CHL1 = 0) consists of Start bit, seven Data bits (excluding parity) and 7/8, one or two Stop bits. In applications using parity, the parity bit is generated and verified by hardware.

The second format (CHL0 = 0, CHL1 = 0) consists of one Start bit, eight Data bits (excluding parity) and 7/8, one or two Stop bits. Parity bit is generated and verified by hardware.

The third format for transmission (CHL0 = 0, CHL1 = 1) consists of one Start bit, nine Data bits and 7/8, one or two Stop bits. This format also supports the UART "ATTENTION" feature. When operating in this format, all eight bits of TBUF and RBUF are used for data. The ninth data bit is transmitted and received using two bits in the ENU and ENUR registers, called XBIT9 and RBIT9. RBIT9 is a read only bit. Parity is not generated or verified in this mode.

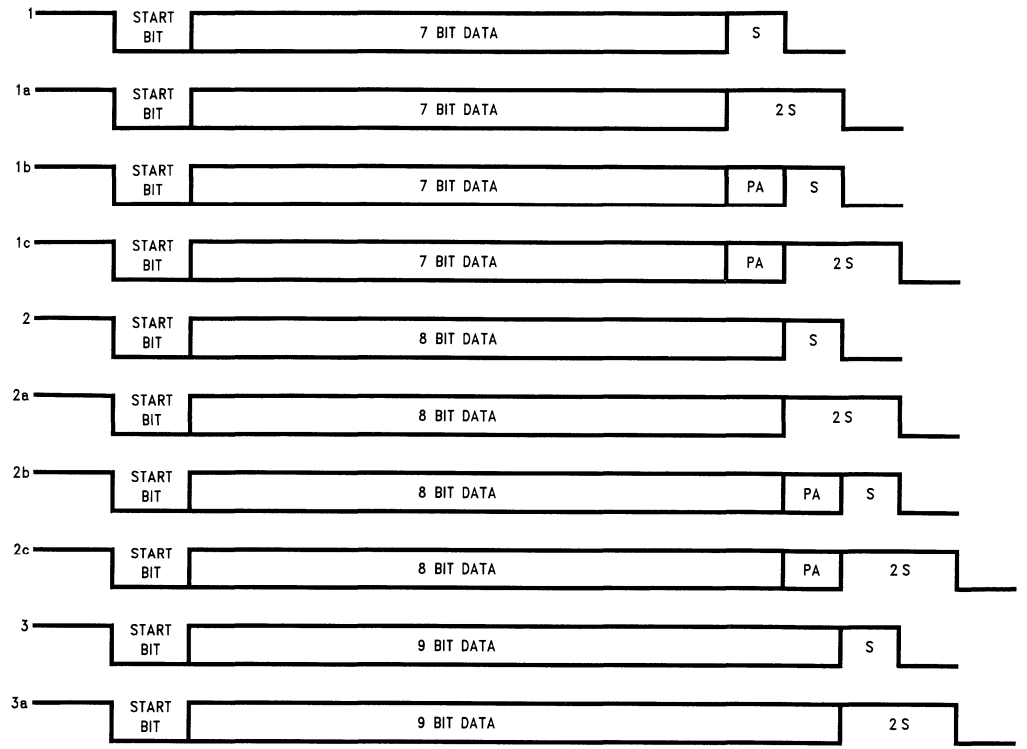
For any of the above framing formats, the last Stop bit can be programmed to be 7/8th of a bit in length. If two Stop bits are selected and the 7/8th bit is set (selected), the second Stop bit will be 7/8th of a bit in length.

The parity is enabled/disabled by PEN bit located in the ENU register. Parity is selected for 7- and 8-bit modes only. If parity is enabled (PEN = 1), the parity selection is then performed by PSEL0 and PSEL1 bits located in the ENU register.

Note that the XBIT9/PSEL0 bit located in the ENU register serves two mutually exclusive functions. This bit programs the ninth bit for transmission when the UART is operating with nine data bits per frame. There is no parity selection in this framing format. For other framing formats XBIT9 is not needed and the bit is PSEL0 used in conjunction with PSEL1 to select parity.

The frame formats for the receiver differ from the transmitter in the number of Stop bits required. The receiver only requires one Stop bit in a frame, regardless of the setting of the Stop bit selection bits in the control register. Note that an implicit assumption is made for full duplex UART operation that the framing formats are the same for the transmitter and receiver.

UART Operation (Continued)



TL/DD/11214–24

FIGURE 12. Framing Formats

UART INTERRUPTS

The UART is capable of generating interrupts. Interrupts are generated on Receive Buffer Full and Transmit Buffer Empty. Both interrupts have individual interrupt vectors. Two bytes of program memory space are reserved for each interrupt vector. The two vectors are located at addresses 0xEC to 0xEF Hex in the program memory space. The interrupts can be individually enabled or disabled using Enable Transmit Interrupt (ETI) and Enable Receive Interrupt (ERI) bits in the ENUI register.

The interrupt from the Transmitter is set pending, and remains pending, as long as both the TBMT and ETI bits are set. To remove this interrupt, software must either clear the ETI bit or write to the TBUF register (thus clearing the TBMT bit).

The interrupt from the receiver is set pending, and remains pending, as long as both the RBFL and ERI bits are set. To remove this interrupt, software must either clear the ERI bit or read from the RBUF register (thus clearing the RBFL bit).

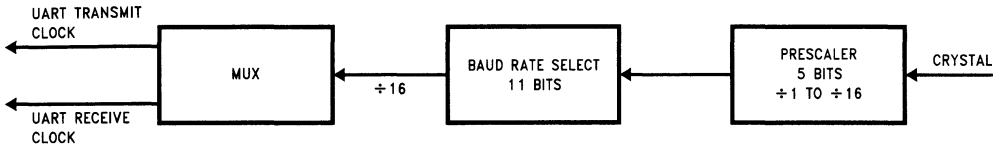
Baud Clock Generation

The clock inputs to the transmitter and receiver sections of the UART can be individually selected to come either from an external source at the CKX pin (port L, pin L1) or from a source selected in the PSR and BAUD registers. Internally, the basic baud clock is created from the oscillator frequency through a two-stage divider chain consisting of a 1–16 (increments of 0.5) prescaler and an 11-bit binary counter. *(Figure 13)* The divide factors are specified through two read/write registers shown in *Figure 14*. Note that the 11-bit Baud Rate Divisor spills over into the Prescaler Select Register (PSR). PSR is cleared upon reset.

As shown in Table I, a Prescaler Factor of 0 corresponds to NO CLOCK. NO CLOCK condition is the UART power down mode where the UART clock is turned off for power saving purpose. The user must also turn the UART clock off when a different baud rate is chosen.

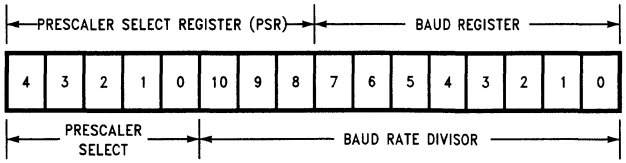
The correspondences between the 5-bit Prescaler Select and Prescaler factors are shown in Table I. Therer are many ways to calculate the two divisor factors, but one particularly effective method would be to achieve a 1.8432 MHz frequency coming out of the first stage. The 1.8432 MHz prescaler output is then used to drive the software programmable baud rate counter to create a x16 clock for the following baud rates: 110, 134.5, 150, 300, 600, 1200, 1800, 2400, 3600, 4800, 7200, 9600, 19200 and 38400 (Table II). Other baud rates may be created by using appropriate divisors. The x16 clock is then divided by 16 to provide the rate for the serial shift registers of the transmitter and receiver.

Baud Clock Generation (Continued)



TL/DD/11214–25

FIGURE 13. UART BAUD Clock Generation



TL/DD/11214–26

FIGURE 14. UART BAUD Clock Divisor Registers

TABLE I. Prescaler Factors

Prescaler Select	Prescaler Factor
00000	NO CLOCK
00001	1
00010	1.5
00011	2
00100	2.5
00101	3
00110	3.5
00111	4
01000	4.5
01001	5
01010	5.5
01011	6
01100	6.5
01101	7
01110	7.5
01111	8
10000	8.5
10001	9
10010	9.5
10011	10
10100	10.5
10101	11
10110	11.5
10111	12
11000	12.5
11001	13
11010	13.5
11011	14
11100	14.5
11101	15
11110	15.5
11111	16

TABLE II. Baud Rate Divisors (1.8432 MHz Prescaler Output)

Baud Rate	Baud Rate Divisor − 1 (N-1)
110 (110.03)	1046
134.5 (134.58)	855
150	767
300	383
600	191
1200	95
1800	63
2400	47
3600	31
4800	23
7200	15
9600	11
19200	5
38400	2

The entries in Table II assume a prescaler output of 1.8432 MHz. In the asynchronous mode the baud rate could be as high as 625k.

As an example, considering the Asynchronous Mode and a CKI clock of 4.608 MHz, the prescaler factor selected is:

$$4.608/1.8432 = 2.5$$

The 2.5 entry is available in Table I. The 1.8432 MHz prescaler output is then used with proper Baud Rate Divisor (Table II) to obtain different baud rates. For a baud rate of 19200 e.g., the entry in Table II is 5.

$N - 1 = 5$ ($N - 1$ is the value from Table II)

$N = 6$ (N is the Baud Rate Divisor)

$$\text{Baud Rate} = 1.8432\ \text{MHz}/(16 \times 6) = 19200$$

The divide by 16 is performed because in the asynchronous mode, the input frequency to the UART is 16 times the baud rate. The equation to calculate baud rates is given below.

The actual Baud Rate may be found from:

$$BR = Fc/(16 \times N \times P)$$

5

Baud Clock Generation (Continued)

Where:

BR is the Baud Rate

Fc is the CKI frequency

N is the Baud Rate Divisor (Table II).

P is the Prescaler Divide Factor selected by the value in the Prescaler Select Register (Table I)

Note: In the Synchronous Mode, the divisor 16 is replaced by two.

Example:

Asynchronous Mode:

Crystal Frequency = 5 MHz

Desired baud rate = 9600

Using the above equation N × P can be calculated first.

$$N \times P = (5 \times 10^6)/(16 \times 9600) = 32.552$$

Now 32.552 is divided by each Prescaler Factor (Table II) to obtain a value closest to an integer. This factor happens to be 6.5 (P = 6.5).

$$N = 32.552/6.5 = 5.008 \ (N = 5)$$

The programmed value (from Table II) should be 4 (N − 1).

Using the above values calculated for N and P:

$$BR = (5 \times 10^6)/(16 \times 5 \times 6.5) = 9615.384$$

$$\% \text{ error} = (9615.385 - 9600)/9600 = 0.16$$

Effect of HALT/IDLE

The UART logic is reinitialized when either the HALT or IDLE modes are entered. This reinitialization sets the TBMT flag and resets all read only bits in the UART control and status registers. Read/Write bits remain unchanged. The Transmit Buffer (TBUF) is not affected, but the Transmit Shift register (TSFT) bits are set to one. The receiver registers RBUF and RSFT are not affected.

The device will exit from the HALT/IDLE modes when the Start bit of a character is detected at the RDX (L3) pin. This feature is obtained by using the Multi-Input Wakeup scheme provided on the device.

Before entering the HALT or IDLE modes the user program must select the Wakeup source to be on the RDX pin. This selection is done by setting bit 3 of WKEN (Wakeup Enable) register. The Wakeup trigger condition is then selected to be high to low transition. This is done via the WKEDG register (bit 3 is one).

If the device is halted and crystal oscillator is used, the Wakeup signal will not start the chip running immediately because of the finite start up time requirement of the crystal oscillator. The idle timer (T0) generates a fixed (256 t_C) delay to ensure that the oscillator has indeed stabilized before allowing the device to execute code. The user has to consider this delay when data transfer is expected immediately after exiting the HALT mode.

Diagnostic

Bits CHARL0 and CHARL1 in the ENU register provide a loopback feature for diagnostic testing of the UART. When these bits are set to one, the following occur: The receiver input pin (RDX) is internally connected to the transmitter output pin (TDX); the output of the Transmitter Shift Register is "looped back" into the Receive Shift Register input. In this mode, data that is transmitted is immediately received. This feature allows the processor to verify the transmit and receive data paths of the UART.

Note that the framing format for this mode is the nine bit format; one Start bit, nine data bits, and 7/8, one or two Stop bits. Parity is not generated or verified in this mode.

Attention Mode

The UART Receiver section supports an alternate mode of operation, referred to as ATTENTION Mode. This mode of operation is selected by the ATTN bit in the ENUR register. The data format for transmission must also be selected as having nine Data bits and either 7/8, one or two Stop bits.

The ATTENTION mode of operation is intended for use in networking the device with other processors. Typically in such environments the messages consists of device addresses, indicating which of several destinations should receive them, and the actual data. This Mode supports a scheme in which addresses are flagged by having the ninth bit of the data field set to a 1. If the ninth bit is reset to a zero the byte is a Data byte.

While in ATTENTION mode, the UART monitors the communication flow, but ignores all characters until an address character is received. Upon receiving an address character, the UART signals that the character is ready by setting the RBFL flag, which in turn interrupts the processor if UART Receiver interrupts are enabled. The ATTN bit is also cleared automatically at this point, so that data characters as well as address characters are recognized. Software examines the contents of the RBUF and responds by deciding either to accept the subsequent data stream (by leaving the ATTN bit reset) or to wait until the next address character is seen (by setting the ATTN bit again).

Operation of the UART Transmitter is not affected by selection of this Mode. The value of the ninth bit to be transmitted is programmed by setting XBIT9 appropriately. The value of the ninth bit received is obtained by reading RBIT9. Since this bit is located in ENUR register where the error flags reside, a bit operation on it will reset the error flags.

Comparators

The device contains two differential comparators, each with a pair of inputs (positive and negative) and an output. Ports I1–I3 and I4–I6 are used for the comparators. The following is the Port I assignment:

- I1 Comparator1 negative input
- I2 Comparator1 positive input
- I3 Comparator1 output
- I4 Comparator2 negative input
- I5 Comparator2 positive input
- I6 Comparator2 output

A Comparator Select Register (CMPSL) is used to enable the comparators, read the outputs of the comparators internally, and enable the outputs of the comparators to the pins. Two control bits (enable and output enable) and one result bit are associated with each comparator. The comparator result bits (CMP1RD and CMP2RD) are read only bits which will read as zero if the associated comparator is not enabled. The Comparator Select Register is cleared with reset, resulting in the comparators being disabled. The comparators should also be disabled before entering either the HALT or IDLE modes in order to save power. The configuration of the CMPSL register is as follows:

Comparators (Continued)

CMPSL REGISTER (ADDRESS X'00B7)

The CMPSL register contains the following bits:

CMP1EN	Enable comparator 1
CMP1RD	Comparator 1 result (this is a read only bit, which will read as 0 if the comparator is not enabled)
CMP10E	Selects pin I3 as comparator 1 output provided that CMPIEN is set to enable the comparator
CMP2EN	Enable comparator 2
CMP2RD	Comparator 2 result (this is a read only bit, which will read as 0 if the comparator is not enabled)
CMP20E	Selects pin I6 as comparator 2 output provided that CMP2EN is set to enable the comparator

Unused	CMP20E	CMP2RD	CMP2EN	CMP10E	CMP1RD	CMP1EN	Unused
Bit 7							Bit 0

Note that the two unused bits of CMPSL may be used as software flags.

Comparator outputs have the same spec as Ports L and G except that the rise and fall times are symmetrical.

Interrupts

The device supports a vectored interrupt scheme. It supports a total of fourteen interrupt sources. The following table lists all the possible interrupt sources, their arbitration ranking and the memory locations reserved for the interrupt vector for each source.

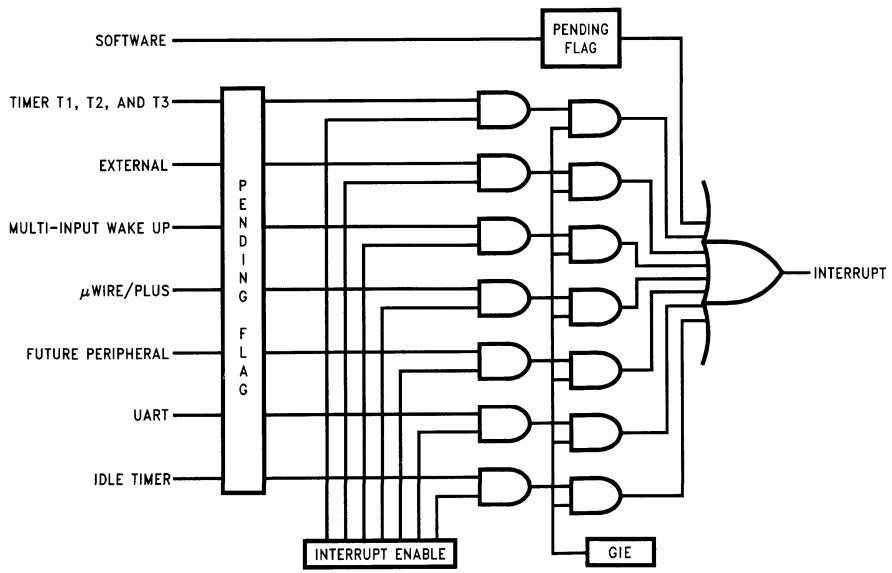
Two bytes of program memory space are reserved for each interrupt source. All interrupt sources except the software interrupt are maskable. Each of the maskable interrupts have an Enable bit and a Pending bit. A maskable interrupt is active if its associated enable and pending bits are set. If GIE = 1 and an interrupt is active, then the processor will be interrupted as soon as it is ready to start executing an instruction except if the above conditions happen during the Software Trap service routine. This exception is described in the Software Trap sub-section.

The interruption process is accomplished with the INTR instruction (opcode 00), which is jammed inside the Instruction Register and replaces the opcode about to be executed. The following steps are performed for every interrupt:

1. The GIE (Global Interrupt Enable) bit is reset.
2. The address of the instruction about to be executed is pushed into the stack.
3. The PC (Program Counter) branches to address 00FF. This procedure takes 7 t_C cycles to execute.

At this time, since GIE = 0, other maskable interrupts are disabled. The user is now free to do whatever context switching is required by saving the context of the machine in the stack with PUSH instructions. The user would then program a VIS (Vector Interrupt Select) instruction in order to branch to the interrupt service routine of the highest priority interrupt enabled and pending at the time of the VIS. Note that this is not necessarily the interrupt that caused the branch to address location 00FF Hex prior to the context switching.

Thus, if an interrupt with a higher rank than the one which caused the interruption becomes active before the decision of which interrupt to service is made by the VIS, then the interrupt with the higher rank will override any lower ones and will be acknowledged. The lower priority interrupt(s) are still pending, however, and will cause another interrupt immediately following the completion of the interrupt service routine associated with the higher priority interrupt just serviced. This lower priority interrupt will occur immediately following the RETI (Return from Interrupt) instruction at the end of the interrupt service routine just completed.



TL/DD/11214–27

FIGURE 15. Interrupt Block Diagram

Interrupts (Continued)

Arbitration Ranking	Source	Description	Vector Address Hi-Low Byte
(1) Highest	Software	INTR Instruction	0yFE–0yFF
	Reserved	for Future Use	0yFC–0yFD
(2)	External	Pin G0 Edge	0yFA–0yFB
(3)	Timer T0	Underflow	0yF8–0yF9
(4)	Timer T1	T1A/Underflow	0yF6–0yF7
(5)	Timer T1	T1B	0yF4–0yF5
(6)	MICROWIRE/PLUS	BUSY Goes Low	0yF2–0yF3
	Reserved	for Future Use	0yF0–0yF1
(7)	UART	Receive	0yEE–0yEF
(8)	UART	Transmit	0yEC–0yED
(9)	Timer T2	T2A/Underflow	0yEA–0yEB
(10)	Timer T2	T2B	0yE8–0yE9
(11)	Timer T3	T3A/Underflow	0yE6–0yE7
(12)	Timer T3	T3B	0yE4–0yE5
(13)	Port L/Wakeup	Port L Edge	0yE2–0yE3
(14) Lowest	Default	VIS Instr. Execution without Any Interrupts	0yE0–0yE1

y is VIS page, $y \neq 0$.

Inside the interrupt service routine, the associated pending bit has to be cleared by software. The RETI (Return from Interrupt) instruction at the end of the interrupt service routine will set the GIE (Global Interrupt Enable) bit, allowing the processor to be interrupted again if another interrupt is active and pending.

The VIS instruction looks at all the active interrupts at the time it is executed and performs an indirect jump to the beginning of the service routine of the one with the highest rank.

The addresses of the different interrupt service routines, called vectors, are chosen by the user and stored in ROM in a table starting at 01E0 (assuming that VIS is located between 00FF and 01DF). The vectors are 15-bit wide and therefore occupy 2 ROM locations.

VIS and the vector table must be located in the same 256-byte block (0y00 to 0yFF) except if VIS is located at the last address of a block. In this case, the table must be in the next block. The vector table cannot be inserted in the first 256-byte block ($y \neq 0$).

The vector of the maskable interrupt with the lowest rank is located at 0yE0 (Hi-Order byte) and 0yE1 (Lo-Order byte) and so forth in increasing rank number. The vector of the maskable interrupt with the highest rank is located at 0yFA (Hi-Order byte) and 0yFB (Lo-Order byte).

The Software Trap has the highest rank and its vector is located at 0yFE and 0yFF.

If, by accident, a VIS gets executed and no interrupt is active, then the PC (Program Counter) will branch to a vector located at 0yE0–0yE1.

WARNING

A Default VIS interrupt handler routine must be present. As a minimum, this handler should confirm that the GIE bit is cleared (this indicates that the interrupt sequence has been taken), take care of any required housekeeping, restore context and return. Some sort of Warm Restart procedure should be implemented. These events can occur without any error on the part of the system designer or programmer.

Note: There is always the possibility of an interrupt occurring during an instruction which is attempting to reset the GIE bit or any other interrupt enable bit. If this occurs when a single cycle instruction is being used to reset the interrupt enable bit, the interrupt enable bit will be reset but an interrupt may still occur. This is because interrupt processing is started at the same time as the interrupt bit is being reset. To avoid this scenario, the user should always use a two-, three-, or four-cycle instruction to reset interrupt enable bits.

Figure 15 shows the Interrupt block diagram.

SOFTWARE TRAP

The Software Trap ($\overline{ST}$) is a special kind of non-maskable interrupt which occurs when the INTR instruction (used to acknowledge interrupts) is fetched from ROM and placed inside the instruction register. This may happen when the PC is pointing beyond the available ROM address space or when the stack is over-popped.

Interrupts (Continued)

When an ST occurs, the user can re-initialize the stack pointer and do a recovery procedure (similar to reset, but not necessarily containing all of the same initialization procedures) before restarting.

The occurrence of an ST is latched into the ST pending bit. The GIE bit is not affected and the ST pending bit **(not accessible by the user)** is used to inhibit other interrupts and to direct the program to the ST service routine with the VIS instruction. The RPND instruction is used to clear the software interrupt pending bit. This pending bit is also cleared on reset.

The ST has the highest rank among all interrupts.

Nothing (except another ST) can interrupt an ST being serviced.

WATCHDOG

The device contains a WATCHDOG and clock monitor. The WATCHDOG is designed to detect the user program getting stuck in infinite loops resulting in loss of program control or "runaway" programs. The Clock Monitor is used to detect the absence of a clock or a very slow clock below a specified rate on the CKI pin.

The WATCHDOG consists of two independent logic blocks: WD UPPER and WD LOWER. WD UPPER establishes the upper limit on the service window and WD LOWER defines the lower limit of the service window.

Servicing the WATCHDOG consists of writing a specific value to a WATCHDOG Service Register named WDSVR which is memory mapped in the RAM. This value is composed of three fields, consisting of a 2-bit Window Select, a 5-bit Key Data field, and the 1-bit Clock Monitor Select field. Table III shows the WDSVR register.

The lower limit of the service window is fixed at 2048 instruction cycles. Bits 7 and 6 of the WDSVR register allow the user to pick an upper limit of the service window.

Table IV shows the four possible combinations of lower and upper limits for the WATCHDOG service window. This flexibility in choosing the WATCHDOG service window prevents any undue burden on the user software.

Bits 5, 4, 3, 2 and 1 of the WDSVR register represent the 5-bit Key Data field. The key data is fixed at 01100. Bit 0 of the WDSVR Register is the Clock Monitor Select bit.

TABLE III. WATCHDOG Service Register (WDSVR)

Window Select		Key Data					Clock Monitor
X	X	0	1	1	0	0	Y
7	6	5	4	3	2	1	0

TABLE IV. WATCHDOG Service Window Select

WDSVR Bit 7	WDSVR Bit 6	Service Window (Lower-Upper Limits)
0	0	2k–8k t_C Cycles
0	1	2k–16k t_C Cycles
1	0	2k–32k t_C Cycles
1	1	2k–64k t_C Cycles

Clock Monitor

The Clock Monitor aboard the device can be selected or deselected under program control. The Clock Monitor is guaranteed not to reject the clock if the instruction cycle clock ($1/t_C$) is greater or equal to 10 kHz. This equates to a clock input rate on CKI of greater or equal to 100 kHz.

WATCHDOG Operation

The WATCHDOG and Clock Monitor are disabled during reset. The device comes out of reset with the WATCHDOG armed, the WATCHDOG Window Select bits (bits 6, 7 of the WDSVR Register) set, and the Clock Monitor bit (bit 0 of the WDSVR Register) enabled. Thus, a Clock Monitor error will occur after coming out of reset, if the instruction cycle clock frequency has not reached a minimum specified value, including the case where the oscillator fails to start.

The WDSVR register can be written to only once after reset and the key data (bits 5 through 1 of the WDSVR Register) must match to be a valid write. This write to the WDSVR register involves two irrevocable choices: (i) the selection of the WATCHDOG service window (ii) enabling or disabling of the Clock Monitor. Hence, the first write to WDSVR Register involves selecting or deselecting the Clock Monitor, select the WATCHDOG service window and match the WATCHDOG key data. Subsequent writes to the WDSVR register will compare the value being written by the user to the WATCHDOG service window value and the key data (bits 7 through 1) in the WDSVR Register. Table V shows the sequence of events that can occur.

The user must service the WATCHDOG at least once before the upper limit of the service window expires. The WATCHDOG may not be serviced more than once in every lower limit of the service window. The user may service the WATCHDOG as many times as wished in the time period between the lower and upper limits of the service window. The first write to the WDSVR Register is also counted as a WATCHDOG service.

The WATCHDOG has an output pin associated with it. This is the WDOUT pin, on pin 1 of the port G. WDOUT is active low. The WDOUT pin is in the high impedance state in the inactive state. Upon triggering the WATCHDOG, the logic will pull the WDOUT (G1) pin low for an additional 16 t_C–32 t_C cycles after the signal level on WDOUT pin goes below the lower Schmitt trigger threshold. After this delay, the device will stop forcing the WDOUT output low.

The WATCHDOG service window will restart when the WDOUT pin goes high. It is recommended that the user tie the WDOUT pin back to V_{CC} through a resistor in order to pull WDOUT high.

A WATCHDOG service while the WDOUT signal is active will be ignored. The state of the WDOUT pin is not guaranteed on reset, but if it powers up low then the WATCHDOG will time out and WDOUT will enter high impedance state.

The Clock Monitor forces the G1 pin low upon detecting a clock frequency error. The Clock Monitor error will continue until the clock frequency has reached the minimum specified value, after which the G1 output will enter the high impedance TRI-STATE mode following 16 t_C–32 t_C clock cycles. The Clock Monitor generates a continual Clock Monitor error if the oscillator fails to start, or fails to reach the minimum specified frequency. The specification for the Clock Monitor is as follows:

$1/t_C > 10$ kHz—No clock rejection.

$1/t_C < 10$ Hz—Guaranteed clock rejection.

WATCHDOG Operation (Continued)

WATCHDOG AND CLOCK MONITOR SUMMARY

The following salient points regarding the WATCHDOG and CLOCK MONITOR should be noted:

- Both the WATCHDOG and CLOCK MONITOR detector circuits are inhibited during RESET.
- Following RESET, the WATCHDOG and CLOCK MONITOR are both enabled, with the WATCHDOG having he maximum service window selected.
- The WATCHDOG service window and CLOCK MONITOR enable/disable option can only be changed once, during the initial WATCHDOG service following RESET.
- The initial WATCHDOG service must match the key data value in the WATCHDOG Service register WDSVR in order to avoid a WATCHDOG error.
- Subsequent WATCHDOG services must match all three data fields in WDSVR in order to avoid WATCHDOG errors.
- The correct key data value cannot be read from the WATCHDOG Service register WDSVR. Any attempt to read this key data value of 01100 from WDSVR will read as key data value of all 0's.
- The WATCHDOG detector circuit is inhibited during both the HALT and IDLE modes.
- The CLOCK MONITOR detector circuit is active during both the HALT and IDLE modes. Consequently, the device inadvertently entering the HALT mode will be detected as a CLOCK MONITOR error (provided that the CLOCK MONITOR enable option has been selected by the program).
- With the single-pin R/C oscillator mask option selected and the CLKDLY bit reset, the WATCHDOG service window will resume following HALT mode from where it left off before entering the HALT mode.
- With the crystal oscillator mask option selected, or with the single-pin R/C oscillator mask option selected and the CLKDLY bit set, the WATCHDOG service window will be set to its selected value from WDSVR following HALT. Consequently, the WATCHDOG should not be serviced for at least 2048 instruction cycles following HALT, but must be serviced within the selected window to avoid a WATCHDOG error.
- The IDLE timer T0 is not initialized with RESET.
- The user can sync in to the IDLE counter cycle with an IDLE counter (T0) interrupt or by monitoring the T0PND flag. The T0PND flag is set whenever the thirteenth bit of the IDLE counter toggles (every 4096 instruction cycles). The user is responsible for resetting the T0PND flag.
- A hardware WATCHDOG service occurs just as the device exits the IDLE mode. Consequently, the WATCHDOG should not be serviced for at least 2048 instruction cycles following IDLE, but must be serviced within the selected window to avoid a WATCHDOG error.
- Following RESET, the initial WATCHDOG service (where the service window and the CLOCK MONITOR enable/disable must be selected) may be programmed anywhere within the maximum service window (65,536 instruction cycles) initialized by RESET. Note that this initial WATCHDOG service may be programmed within the initial 2048 instruction cycles without causing a WATCHDOG error.

Detection of Illegal Conditions

The device can detect various illegal conditions resulting from coding errors, transient noise, power supply voltage drops, runaway programs, etc.

Reading of undefined ROM gets zeros. The opcode for software interrupt is zero. If the program fetches instructions from undefined ROM, this will force a software interrupt, thus signaling that an illegal condition has occurred.

The subroutine stack grows down for each call (jump to subroutine), interrupt, or PUSH, and grows up for each return or POP. The stack pointer is initialized to RAM location 06F Hex during reset. Consequently, if there are more returns than calls, the stack pointer will point to addresses 070 and 071 Hex (which are undefined RAM). Undefined RAM from addresses 070 to 07F (Segment 0), 140 to 17F (Segment 1), and all other segments (i.e., Segments 2 . . . etc.) is read as all 1's, which in turn will cause the program to return to address 7FFF Hex. This is an undefined ROM location and the instruction fetched (all 0's) from this location will generate a software interrupt signaling an illegal condition.

Thus, the chip can detect the following illegal conditions:

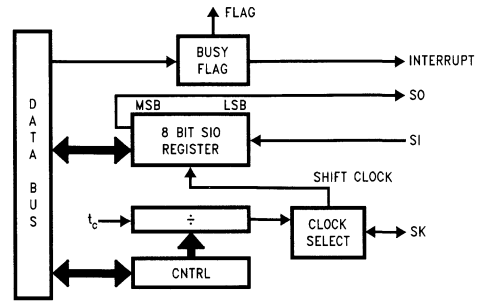
a. Executing from undefined ROM

b. Over "POP"ing the stack by having more returns than calls.

When the software interrupt occurs, the user can re-initialize the stack pointer and do a recovery procedure before restarting (this recovery program is probably similar to that following reset, but might not contain the same program initialization procedures). The recovery program should reset the software interrupt pending bit using the RPND instruction.

MICROWIRE/PLUS

MICROWIRE/PLUS is a serial synchronous communications interface. The MICROWIRE/PLUS capability enables the device to interface with any of National Semiconductor's MICROWIRE peripherals (i.e. A/D converters, display drivers, E²PROMs etc.) and with other microcontrollers which support the MICROWIRE interface. It consists of an 8-bit serial shift register (SIO) with serial data input (SI), serial data output (SO) and serial shift clock (SK). *Figure 12* shows a block diagram of the MICROWIRE/PLUS logic.



TL/DD/11214–28

FIGURE 16. MICROWIRE/PLUS Block Diagram

The shift clock can be selected from either an internal source or an external source. Operating the MICROWIRE/PLUS arrangement with the internal clock source is called the Master mode of operation. Similarly, operating the MICROWIRE/PLUS arrangement with an external shift clock is called the Slave mode of operation.

The CNTRL register is used to configure and control the MICROWIRE/PLUS mode. To use the MICROWIRE/PLUS, the MSEL bit in the CNTRL register is set to one. In the master mode, the SK clock rate is selected by the two bits, SL0 and SL1, in the CNTRL register. Table VI details the different clock rates that may be selected.

TABLE V. WATCHDOG Service Actions

Key Data	Window Data	Clock Monitor	Action
Match	Match	Match	Valid Service: Restart Service Window
Don't Care	Mismatch	Don't Care	Error: Generate WATCHDOG Output
Mismatch	Don't Care	Don't Care	Error: Generate WATCHDOG Output
Don't Care	Don't Care	Mismatch	Error: Generate WATCHDOG Output

TABLE VI. MICROWIRE/PLUS Master Mode Clock Select

SL1	SL0	SK
0	0	$2 \times t_c$
0	1	$4 \times t_c$
1	x	$8 \times t_c$

Where t_c is the instruction cycle clock

5

MICROWIRE/PLUS (Continued)

MICROWIRE/PLUS OPERATION

Setting the BUSY bit in the PSW register causes the MICROWIRE/PLUS to start shifting the data. It gets reset when eight data bits have been shifted. The user may reset the BUSY bit by software to allow less than 8 bits to shift. If enabled, an interrupt is generated when eight data bits have been shifted. The device may enter the MICROWIRE/PLUS mode either as a Master or as a Slave. *Figure 13* shows how two devices, microcontrollers and several peripherals may be interconnected using the MICROWIRE/PLUS arrangements.

Warning:

The SIO register should only be loaded when the SK clock is low. Loading the SIO register while the SK clock is high will result in undefined data in the SIO register. SK clock is normally low when not shifting.

Setting the BUSY flag when the input SK clock is high in the MICROWIRE/PLUS slave mode may cause the current SK clock for the SIO shift register to be narrow. For safety, the BUSY flag should only be set when the input SK clock is low.

MICROWIRE/PLUS Master Mode Operation

In the MICROWIRE/PLUS Master mode of operation the shift clock (SK) is generated internally by the device. The MICROWIRE Master always initiates all data exchanges. The MSEL bit in the CNTRL register must be set to enable the SO and SK functions onto the G Port. The SO and SK pins must also be selected as outputs by setting appropriate bits in the Port G configuration register. Table VII summarizes the bit settings required for Master mode of operation.

MICROWIRE/PLUS Slave Mode Operation

In the MICROWIRE/PLUS Slave mode of operation the SK clock is generated by an external source. Setting the MSEL bit in the CNTRL register enables the SO and SK functions onto the G Port. The SK pin must be selected as an input and the SO pin is selected as an output pin by setting and resetting the appropriate bits in the Port G configuration register. Table VII summarizes the settings required to enter the Slave mode of operation.

The user must set the BUSY flag immediately upon entering the Slave mode. This will ensure that all data bits sent by the Master will be shifted properly. After eight clock pulses the BUSY flag will be cleared and the sequence may be repeated.

Alternate SK Phase Operation

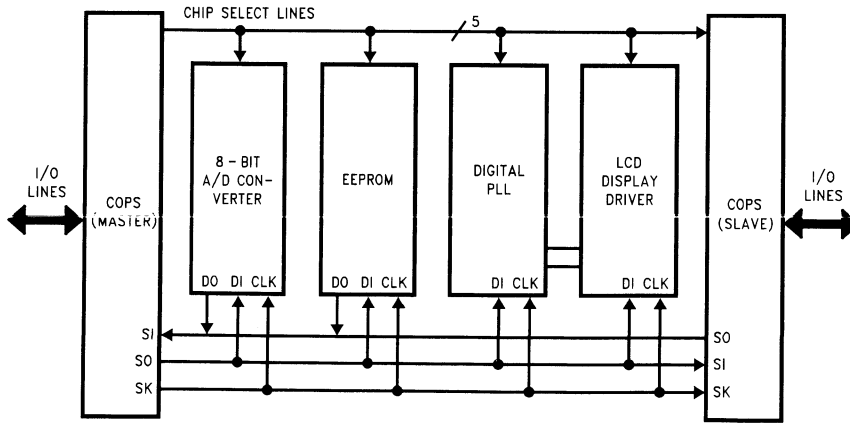
The device allows either the normal SK clock or an alternate phase SK clock to shift data in and out of the SIO register. In both the modes the SK is normally low. In the normal mode data is shifted in on the rising edge of the SK clock and the data is shifted out on the falling edge of the SK clock. The SIO register is shifted on each falling edge of the SK clock. In the alternate SK phase operation, data is shifted in on the falling edge of the SK clock and shifted out on the rising edge of the SK clock.

A control flag, SKSEL, allows either the normal SK clock or the alternate SK clock to be selected. Resetting SKSEL causes the MICROWIRE/PLUS logic to be clocked from the normal SK signal. Setting the SKSEL flag selects the alternate SK clock. The SKSEL is mapped into the G6 configuration bit. The SKSEL flag will power up in the reset condition, selecting the normal SK signal.

TABLE VII

This table assumes that the control flag MSEL is set.

G4 (SO) Config. Bit	G5 (SK) Config. Bit	G4 Fun.	G5 Fun.	Operation
1	1	SO	Int. SK	MICROWIRE/PLUS Master
0	1	TRI-STATE	Int. SK	MICROWIRE/PLUS Master
1	0	SO	Ext. SK	MICROWIRE/PLUS Slave
0	0	TRI-STATE	Ext. SK	MICROWIRE/PLUS Slave



TL/DD/11214–32

FIGURE 17. MICROWIRE/PLUS Application

Memory Map

All RAM, ports and registers (except A and PC) are mapped into data memory address space.

Address S/ADD REG	Contents
0000 to 006F	On-Chip RAM bytes (112 bytes)
0070 to 007F	Unused RAM Address Space (Reads As All Ones)
xx80 to xxAF	Unused RAM Address Space (Reads Undefined Data)
xxB0	Timer T3 Lower Byte
XXB1	Timer T3 Upper Byte
xxB2	Timer T3 Autoload Register T3RA Lower Byte
xxB3	Timer T3 Autoload Register T3RA Upper Byte
xxB4	Timer T3 Autoload Register T3RB Lower Byte
xxB5	Timer T3 Autoload Register T3RB Upper Byte
xxB6	Timer T3 Control Register
xxB7	Comparator Select Register (CMPSL)
xxB8	UART Transmit Buffer (TBUF)
xxB9	UART Receive Buffer (RBUF)
xxBA	UART Control and Status Register (ENU)
xxBB	UART Receive Control and Status Register (ENUR)
xxBC	UART Interrupt and Clock Source Register (ENUI)
xxBD	UART Baud Register (BAUD)
xxBE	UART Prescale Select Register (PSR)
xxBF	Reserved for UART
xxC0	Timer T2 Lower Byte
xxC1	Timer T2 Upper Byte
xxC2	Timer T2 Autoload Register T2RA Lower Byte
xxC3	Timer T2 Autoload Register T2RA Upper Byte
xxC4	Timer T2 Autoload Register T2RB Lower Byte
xxC5	Timer T2 Autoload Register T2RB Upper Byte
xxC6	Timer T2 Control Register
xxC7	WATCHDOG Service Register (Reg:WDSVR)
xxC8	MIWU Edge Select Register (Reg:WKEDG)
xxC9	MIWU Enable Register (Reg:WKEN)
xxCA	MIWU Pending Register (Reg:WKPND)
xxCB	Reserved
xxCC	Reserved
xxCD to xxCF	Reserved

Address S/ADD REG	Contents
xxD0	Port L Data Register
xxD1	Port L Configuration Register
xxD2	Port L Input Pins (Read Only)
xxD3	Reserved for Port L
xxD4	Port G Data Register
xxD5	Port G Configuration Register
xxD6	Port G Input Pins (Read Only)
xxD7	Port I Input Pins (Read Only)
xxD8	Port C Data Register
xxD9	Port C Configuration Register
xxDA	Port C Input Pins (Read Only)
xxDB	Reserved for Port C
xxDC	Port D
xxDD to DF	Reserved for Port D
xxE0 to xxE5	Reserved for EE Control Registers
xxE6	Timer T1 Autoload Register T1RB Lower Byte
xxE7	Timer T1 Autoload Register T1RB Upper Byte
xxE8	ICNTRL Register
xxE9	MICROWIRE/PLUS Shift Register
xxEA	Timer T1 Lower Byte
xxEB	Timer T1 Upper Byte
xxEC	Timer T1 Autoload Register T1RA Lower Byte
xxED	Timer T1 Autoload Register T1RA Upper Byte
xxEE	CNTRL Control Register
xxEF	PSW Register
xxF0 to FB	On-Chip RAM Mapped as Registers
xxFC	X Register
xxFD	SP Register
xxFE	B Register
xxFF	S Register
0100–017F	On-Chip 128 RAM Bytes

Reading memory locations 0070H–007FH (Segment 0) will return all ones. Reading unused memory locations 0080H–00AFH (Segment 0) will return undefined data. Reading memory locations from other unused Segments (i.e., Segment 2, Segment 3, ... etc.) will return all ones.

Addressing Modes

There are ten addressing modes, six for operand addressing and four for transfer of control.

OPERAND ADDRESSING MODES

Register Indirect

This is the "normal" addressing mode. The operand is the data memory addressed by the B pointer or X pointer.

Register Indirect (with auto post increment or decrement of pointer)

This addressing mode is used with the LD and X instructions. The operand is the data memory addressed by the B pointer or X pointer. This is a register indirect mode that automatically post increments or decrements the B or X register after executing the instruction.

Direct

The instruction contains an 8-bit address field that directly points to the data memory for the operand.

Immediate

The instruction contains an 8-bit immediate field as the operand.

Short Immediate

This addressing mode is used with the Load B Immediate instruction. The instruction contains a 4-bit immediate field as the operand.

Indirect

This addressing mode is used with the LAID instruction. The contents of the accumulator are used as a partial address (lower 8 bits of PC) for accessing a data operand from the program memory.

TRANSFER OF CONTROL ADDRESSING MODES

Relative

This mode is used for the JP instruction, with the instruction field being added to the program counter to get the new program location. JP has a range from −31 to +32 to allow a 1-byte relative jump (JP + 1 is implemented by a NOP instruction). There are no "pages" when using JP, since all 15 bits of PC are used.

Absolute

This mode is used with the JMP and JSR instructions, with the instruction field of 12 bits replacing the lower 12 bits of the program counter (PC). This allows jumping to any location in the current 4k program memory segment.

Absolute Long

This mode is used with the JMPL and JSRL instructions, with the instruction field of 15 bits replacing the entire 15 bits of the program counter (PC). This allows jumping to any location up to 32k in the program memory space.

Indirect

This mode is used with the JID instruction. The contents of the accumulator are used as a partial address (lower 8 bits of PC) for accessing a location in the program memory. The contents of this program memory location serve as a partial address (lower 8 bits of PC) for the jump to the next instruction.

Note: The VIS is a special case of the Indirect Transfer of Control addressing mode, where the double byte vector associated with the interrupt is transferred from adjacent addresses in the program memory into the program counter (PC) in order to jump to the associated interrupt service routine.

Instruction Set

Register and Symbol Definition

Registers	
A	8-Bit Accumulator Register
B	8-Bit Address Register
X	8-Bit Address Register
SP	8-Bit Stack Pointer Register
PC	15-Bit Program Counter Register
PU	Upper 7 Bits of PC
PL	Lower 8 Bits of PC
C	1 Bit of PSW Register for Carry
HC	1 Bit of PSW Register for Half Carry
GIE	1 Bit of PSW Register for Global Interrupt Enable
VU	Interrupt Vector Upper Byte
VL	Interrupt Vector Lower Byte

Symbols	
[B]	Memory Indirectly Addressed by B Register
[X]	Memory Indirectly Addressed by X Register
MD	Direct Addressed Memory
Mem	Direct Addressed Memory or [B]
Meml	Direct Addressed Memory or [B] or Immediate Data
Imm	8-Bit Immediate Data
Reg	Register Memory: Addresses F0 to FF (Includes B, X and SP)
Bit	Bit Number (0 to 7)
←	Loaded with
↔	Exchanged with

Instruction Set (Continued)

INSTRUCTION SET

ADD	A,Meml	ADD	A ← A + Meml
ADC	A,Meml	ADD with Carry	A ← A + Meml + C, C ← Carry HC ← Half Carry
SUBC	A,Meml	Subtract with Carry	A ← A − $\overline{\text{Meml}}$ + C, C ← Carry HC ← Half Carry
AND	A,Meml	Logical AND	A ← A and Meml
ANDSZ	A,Imm	Logical AND Immed., Skip if Zero	Skip next if (A and Imm) = 0
OR	A,Meml	Logical OR	A ← A or Meml
XOR	A,Meml	Logical EXclusive OR	A ← A xor Meml
IFEQ	MD,Imm	IF EQual	Compare MD and Imm, Do next if MD = Imm
IFEQ	A,Meml	IF EQual	Compare A and Meml, Do next if A = Meml
IFNE	A,Meml	IF Not Equal	Compare A and Meml, Do next if A ≠ Meml
IFGT	A,Meml	IF Greater Than	Compare A and Meml, Do next if A > Meml
IFBNE	#	If B Not Equal	Do next if lower 4 bits of B ≠ Imm
DRSZ	Reg	Decrement Reg., Skip if Zero	Reg ← Reg − 1, Skip if Reg = 0
SBIT	#,Mem	Set BIT	1 to bit, Mem (bit = 0 to 7 immediate)
RBIT	#,Mem	Reset BIT	0 to bit, Mem
IFBIT	#,Mem	IF BIT	If bit in A or Mem is true do next instruction
RPND		Reset PeNDing Flag	Reset Software Interrupt Pending Flag
X	A,Mem	EXchange A with Memory	A ⟷ Mem
X	A,[X]	EXchange A with Memory [X]	A ⟷ [X]
LD	A,Meml	LoaD A with Memory	A ← Meml
LD	A,[X]	LoaD A with Memory [X]	A ← [X]
LD	B,Imm	LoaD B with Immed.	B ← Imm
LD	Mem,Imm	LoaD Memory Immed	Mem ← Imm
LD	Reg,Imm	LoaD Register Memory Immed.	Reg ← Imm
X	A, [B ±]	EXchange A with Memory [B]	A ⟷ [B], (B ← B ±1)
X	A, [X ±]	EXchange A with Memory [X]	A ⟷ [X], (X ← ±1)
LD	A, [B±]	LoaD A with Memory [B]	A ← [B], (B ← B ±1)
LD	A, [X±]	LoaD A with Memory [X]	A ← [X], (X ← X±1)
LD	[B±],Imm	LoaD Memory [B] Immed.	[B] ← Imm, (B ← B±1)
CLR	A	CLeaR A	A ← 0
INC	A	INCrement A	A ← A + 1
DEC	A	DECrementA	A ← A − 1
LAID		Load A InDirect from ROM	A ← ROM (PU,A)
DCOR	A	Decimal CORrect A	A ← BCD correction of A (follows ADC, SUBC)
RRC	A	Rotate A Right thru C	C → A7 → . . . → A0 → C
RLC	A	Rotate A Left thru C	C ← A7 ← . . . ← A0 ← C
SWAP	A	SWAP nibbles of A	A7 . . . A4 ⟷ A3 . . . A0
SC		Set C	C ← 1, HC ← 1
RC		Reset C	C ← 0, HC ← 0
IFC		IF C	IF C is true, do next instruction
IFNC		IF Not C	If C is not true, do next instruction
POP	A	POP the stack into A	SP ← SP + 1, A ← [SP]
PUSH	A	PUSH A onto the stack	[SP] ← A, SP ← SP − 1
VIS		Vector to Interrupt Service Routine	PU ← [VU], PL ← [VL]
JMPL	Addr.	Jump absolute Long	PC ← ii (ii = 15 bits, 0 to 32k)
JMP	Addr.	Jump absolute	PC9 . . . 0 ← i (i = 12 bits)
JP	Disp.	Jump relative short	PC ← PC + r (r is −31 to +32, except 1)
JSRL	Addr.	Jump SubRoutine Long	[SP] ← PL, [SP−1] ← PU,SP−2, PC ← ii
JSR	Addr	Jump SubRoutine	[SP] ← PL, [SP−1] ← PU,SP−2, PC9 . . . 0 ← i
JID		Jump InDirect	PL ← ROM (PU,A)
RET		RETurn from subroutine	SP + 2, PL ← [SP], PU ← [SP−1]
RETSK		RETurn and SKip	SP + 2, PL ← [SP],PU ← [SP−1]
RETI		RETurn from Interrupt	SP + 2, PL ← [SP],PU ← [SP−1],GIE ← 1
INTR		Generate an Interrupt	[SP] ← PL, [SP−1] ← PU, SP−2, PC ← 0FF
NOP		No OPeration	PC ← PC + 1

Instruction Execution Time

Most instructions are single byte (with immediate addressing mode instructions taking two bytes).

Most single byte instructions take one cycle time to execute.

Skipped instructions require x number of cycles to be skipped, where x equals the number of bytes in the skipped instruction opcode.

See the BYTES and CYCLES per INSTRUCTION table for details.

Bytes and Cycles per Instruction

The following table shows the number of bytes and cycles for each instruction in the format of byte/cycle.

Arithmetic and Logic Instructions

	[B]	Direct	Immed.
ADD	1/1	3/4	2/2
ADC	1/1	3/4	2/2
SUBC	1/1	3/4	2/2
AND	1/1	3/4	2/2
OR	1/1	3/4	2/2
XOR	1/1	3/4	2/2
IFEQ	1/1	3/4	2/2
IFNE	1/1	3/4	2/2
IFGT	1/1	3/4	2/2
IFBNE	1/1		
DRSZ		1/3	
SBIT	1/1	3/4	
RBIT	1/1	3/4	
IFBIT	1/1	3/4	

RPND	1/1

Instructions Using A & C

CLRA	1/1
INCA	1/1
DECA	1/1
LAID	1/3
DCOR	1/1
RRCA	1/1
RLCA	1/1
SWAPA	1/1
SC	1/1
RC	1/1
IFC	1/1
IFNC	1/1
PUSHA	1/3
POPA	1/3
ANDSZ	2/2

Transfer of Control Instructions

JMPL	3/4
JMP	2/3
JP	1/3
JSRL	3/5
JSR	2/5
JID	1/3
VIS	1/5
RET	1/5
RETSK	1/5
RETI	1/5
INTR	1/7
NOP	1/1

Memory Transfer Instructions

	Register Indirect		Direct	Immed.	Register Indirect Auto Incr. & Decr.		
	[B]	[X]			[B+, B−]	[X+, X−]	
X A,*	1/1	1/3	2/3		1/2	1/3	
LD A,*	1/1	1/3	2/3	2/2	1/2	1/3	
LD B, Imm				1/1			(IF B < 16)
LD B, Imm				2/2			(IF B > 15)
LD Mem, Imm	2/2		3/3		2/2		
LD Reg, Imm			2/3				
IFEQ MD, Imm			3/3				

* = > Memory location addressed by B or X or directly.

Opcode Table

Upper Nibble Along X-Axis
Lower Nibble Along Y-Axis

F	E	D	C	B	A	9	8	
JP −15	JP −31	LD 0F0, # i	DRSZ 0F0	RRCA	RC	ADC A,#i	ADC A,[B]	0
JP −14	JP −30	LD 0F1, # i	DRSZ 0F1	*	SC	SUBC A, #i	SUB A,[B]	1
JP −13	JP −29	LD 0F2, # i	DRSZ 0F2	X A, [X+]	X A,[B+]	IFEQ A,#i	IFEQ A,[B]	2
JP −12	JP −28	LD 0F3, # i	DRSZ 0F3	X A, [X−]	X A,[B−]	IFGT A,#i	IFGT A,[B]	3
JP −11	JP −27	LD 0F4, # i	DRSZ 0F4	VIS	LAID	ADD A,#i	ADD A,[B]	4
JP −10	JP −26	LD 0F5, # i	DRSZ 0F5	RPND	JID	AND A,#i	AND A,[B]	5
JP −9	JP −25	LD 0F6, # i	DRSZ 0F6	X A,[X]	X A,[B]	XOR A,#i	XOR A,[B]	6
JP −8	JP −24	LD 0F7, # i	DRSZ 0F7	*	*	OR A,#i	OR A,[B]	7
JP −7	JP −23	LD 0F8, # i	DRSZ 0F8	NOP	RLCA	LD A,#i	IFC	8
JP −6	JP −22	LD 0F9, # i	DRSZ 0F9	IFNE A,[B]	IFEQ Md,#i	IFNE A,#i	IFNC	9
JP −5	JP −21	LD 0FA, # i	DRSZ 0FA	LD A,[X+]	LD A,[B+]	LD [B+],#i	INCA	A
JP −4	JP −20	LD 0FB, # i	DRSZ 0FB	LD A,[X−]	LD A,[B−]	LD [B−],#i	DECA	B
JP −3	JP −19	LD 0FC, # i	DRSZ 0FC	LD Md,#i	JMPL	X A,Md	POPA	C
JP −2	JP −18	LD 0FD, # i	DRSZ 0FD	DIR	JSRL	LD A,Md	RETSK	D
JP −1	JP −17	LD 0FE, # i	DRSZ 0FE	LD A,[X]	LD A,[B]	LD [B],#i	RET	E
JP −0	JP −16	LD 0FF, # i	DRSZ 0FF	*	*	LD B,#i	RETI	F

Opcode Table (Continued)

Upper Nibble Along X-Axis

Lower Nibble Along Y-Axis

7	6	5	4	3	2	1	0	
IFBIT 0,[B]	ANDSZ A, #i	LD B,#0F	IFBNE 0	JSR x000–x0FF	JMP x000–x0FF	JP +17	INTR	0
IFBIT 1,[B]	*	LD B,#0E	IFBNE 1	JSR x100–x1FF	JMP x100–x1FF	JP +18	JP + 2	1
IFBIT 2,[B]	*	LD B,#0D	IFBNE 2	JSR x200–x2FF	JMP x200–x2FF	JP +19	JP + 3	2
IFBIT 3,[B]	*	LD B,#0C	IFBNE 3	JSR x300–x3FF	JMP x300–x3FF	JP +20	JP + 4	3
IFBIT 4,[B]	CLRA	LD B,#0B	IFBNE 4	JSR x400–x4FF	JMP x400–x4FF	JP +21	JP + 5	4
IFBIT 5,[B]	SWAPA	LD B,#0A	IFBNE 5	JSR x500–x5FF	JMP x500–x5FF	JP +22	JP + 6	5
IFBIT 6,[B]	DCORA	LD B,#09	IFBNE 6	JSR x600–x6FF	JMP x600–x6FF	JP +23	JP + 7	6
IFBIT 7,[B]	PUSHA	LD B,#08	IFBNE 7	JSR x700–x7FF	JMP x700–x7FF	JP +24	JP + 8	7
SBIT 0,[B]	RBIT 0,[B]	LD B,#07	IFBNE 8	JSR x800–x8FF	JMP x800–x8FF	JP +25	JP + 9	8
SBIT 1,[B]	RBIT 1,[B]	LD B,#06	IFBNE 9	JSR x900–x9FF	JMP x900–x9FF	JP +26	JP + 10	9
SBIT 2,[B]	RBIT 2,[B]	LD B,#05	IFBNE 0A	JSR xA00–xAFF	JMP xA00–xAFF	JP +27	JP + 11	A
SBIT 3,[B]	RBIT 3,[B]	LD B,#04	IFBNE 0B	JSR xB00–xBFF	JMP xB00–xBFF	JP +28	JP + 12	B
SBIT 4,[B]	RBIT 4,[B]	LD B,#03	IFBNE 0C	JSR xC00–xCFF	JMP xC00–xCFF	JP +29	JP + 13	C
SBIT 5,[B]	RBIT 5,[B]	LD B,#02	IFBNE 0D	JSR xD00–xDFF	JMP xD00–xDFF	JP +30	JP + 14	D
SBIT 6,[B]	RBIT 6,[B]	LD B,#01	IFBNE 0E	JSR xE00–xEFF	JMP xE00–xEFF	JP +31	JP + 15	E
SBIT 7,[B]	RBIT 7,[B]	LD B,#00	IFBNE 0F	JSR xF00–xFFF	JMP xF00–xFFF	JP +32	JP + 16	F

Where,

i is the immediate data
Md is a directly addressed memory location
* is an unused opcode

Note: The opcode 60 Hex is also the opcode for IFBIT #i,A

Mask Options

The mask programmable options are shown below. The options are programmed at the same time as the ROM pattern submission.

```
OPTION 1: CLOCK CONFIGURATION
  = 1     Crystal Oscillator (CKI/lO)
              G7 (CKO) is clock generator
              output to crystal/resonator
              CKI is the clock input
  = 2     Single-pin RC controlled
              oscillator (CKI/lO)
              G7 is available as a HALT
              restart and/or general purpose
              input

OPTION 2: HALT
  = 1     Enable HALT mode
  = 2     Disable HALT mode
OPTION 3: BONDING OPTIONS
  = 1     44-Pin PLCC
  = 2     40-Pin DIP
  = 3     N/A
  = 4     28-Pin DIP
  = 5     28-Pin SO
```

Development Support

SUMMARY

- iceMASTER: IM-COP8/400—Full feature in-circuit emulation for all COP8 products. A full set of COP8 Basic and Feature Family device and package specific probes are available.
- COP8 Debug Module: Moderate cost in-circuit emulation and development programming unit.
- COP8 Evaluation and Programming Unit: EPU-COP888GG—low cost in-circuit simulation and development programming unit.
- Assembler: COP8-DEV-IBMA. A DOS installable cross development Assembler, Linker, Librarian and Utility Software Development Tool Kit.
- C Compiler: COP8C. A DOS installable cross development Software Tool Kit.
- OTP/EPROM Programmer Support: Covering needs from engineering prototype, pilot production to full production environments.

IceMASTER (IM) IN-CIRCUIT EMULATION

The iceMASTER IM-COP8/400 is a full feature, PC-based, in-circuit emulation tool developed and marketed by MetaLink Corporation to support the whole COP8 family of products. National is a resale vendor for these products.

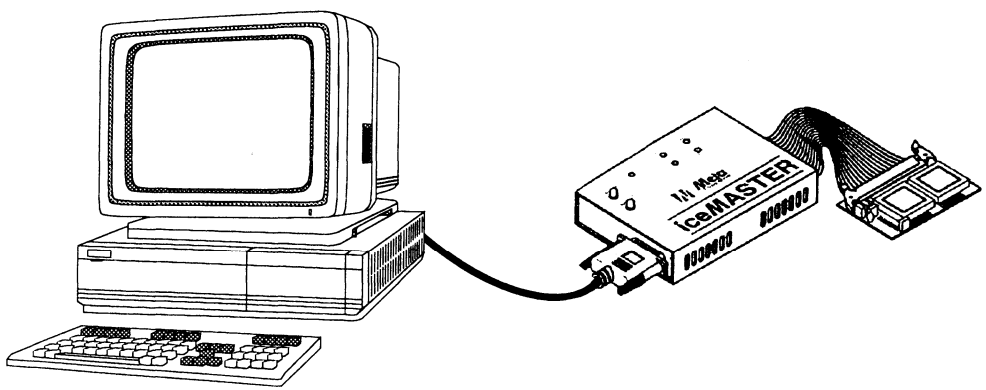
See *Figure 19* for configuration.

The iceMASTER IM-COP8/400 with its device specific COP8 Probe provides a rich feature set for developing, testing and maintaining product:

- Real-time in-circuit emulation; full 2.4V–5.5V operation range, full DC-10 MHz clock. Chip options are programmable or jumper selectable.
- Direct connection to application board by package compatible socket or surface mount assembly.
- Full 32 kbytes of loadable programming space that overlays (replaces) the on-chip ROM or EPROM. On-chip RAM and I/O blocks are used directly or recreated on the probe as necessary.
- Full 4k frame synchronous trace memory. Address, instruction, and 8 unspecified, circuit connectable trace lines. Display can be HLL source (e.g., C source), assembly or mixed.
- A full 64k hardware configurable break, trace on, trace off control, and pass count increment events.
- Tool set integrated interactive symbolic debugger—supports both assembler (COFF) and C Compiler (.COD) linked object formats.
- Real time peformance profiling analysis; selectable bucket definition.
- Watch windows, content updated automatically at each execution break.
- Instruction by instruction memory/register changes displayed on source window when in single step operation.
- Single base unit & debugger software reconfigurable to support the entire COP8 family; only the probe personality needs to change. Debugger software is processor customized, and reconfigured from a master model file.
- Processor specific symbolic display of registers and bit level assignments, configured from master model file.
- Halt/Idle mode notification.
- On-line HELP customized to specific processor using master model file.
- Includes a copy of COP8-DEV-IBMA assembler & linker SDK.

IM Order Information

Base Unit	
IM-COP8/400-1	iceMASTER base unit, 110V power supply
IM-COP8/400-2	iceMASTER base unit, 220V power supply
iceMASTER Probe	
MHW-884EG28DWPC MHW-888EG40DWPC MHW-888EG44PWPC	28 DIP 40 DIP 44 PLCC
28 DIP to 28 SO Adapter	
MHW-SOIC28	28 SO



TL/DD/11214–33

FIGURE 19. COP8 iceMASTER Environment

Development Support (Continued)

IceMASTER DEBUG MODULE (DM)

The iceMASTER Debug Module is a PC based, combination in-circuit emulation tool and COP8 based OTP/EPROM programming tool developed and marketed by MetaLink Corporation to support the whole COP8 family of products. National is a resale vendor for these products.

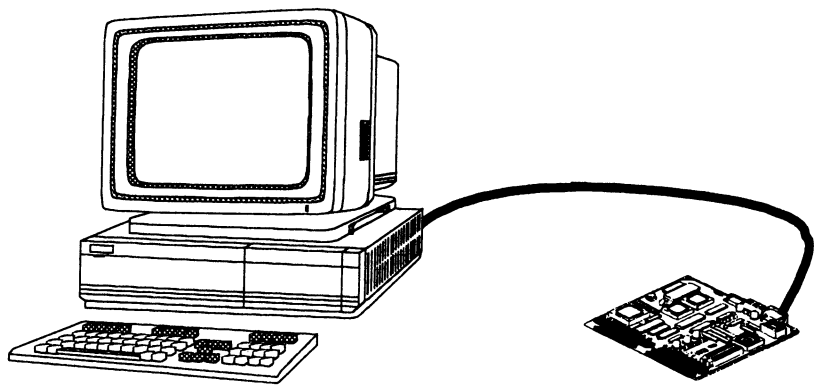
See *Figure 20* for configuration.

The iceMASTER Debug Module is a moderate cost development tool. It has the capability of in-circuit emulation for a specific COP8 microcontroller and in addition serves as a programming tool for COP8 OTP and EPROM product families. Summary of features is as follows:

- Real-time in-circuit emulation; full operating voltage range operation, full DC-10 MHz clock.
- All processor I/O pins can be cabled to an application development board with package compatible cable to socket and surface mount assembly.
- Full 32 kbytes of loadable programming space that overlays (replaces) the on-chip ROM or EPROM. On-chip RAM and I/O blocks are used directly or recreated as necessary.
- 100 frames of synchronous trace memory. The display can be HLL source (C source), assembly or mixed. The most recent history prior to a break is available in the trace memory.
- Configured break points; uses INTR instruction which is modestly intrusive.
- Software—only supported features are selectable.
- Tool set integrated interactive symbolic debugger—supports both assembler (COFF) and C Compiler (.COD) SDK linked object formats.
- Instruction by instruction memory/register changes displayed when in single step operation.
- Debugger software is processor customized, and reconfigured from a master model file.
- Processor specific symbolic display of registers and bit level assignments, configured from master model file.
- Halt/Idle mode notification.
- Programming menu supports full product line of programmable OTP and EPROM COP8 products. Program data is taken directly from the overlay RAM.
- Programming of 44 PLCC and 68 PLCC parts requires external programming adapters.
- Includes wallmount power supply.
- On-board V_{PP} generator from 5V input or connection to external supply supported. Requires V_{PP} level adjustment per the family programming specification (correct level is provided on an on-screen pop-down display).
- On-line HELP customized to specific processor using master model file.
- Includes a copy of COP8-DEV-IBMA assembler and linker SDK.

DM Order Information

Debug Module Unit	
COP8-DM/888EG	
Cable Adapters	
DM-COP8/28D	28 DIP
DM-COP8/40D	40 DIP
DM-COP8/44P	44 PLCC
28 DIP to 28 SO Adapter	
DM-COP8/28D-SO	28 SO



TL/DD/11214-34

FIGURE 20. COP8-DM Environment

Development Support (Continued)

IceMASTER EVALUATION PROGRAMMING UNIT (EPU)

The iceMASTER EPU-COP888GG is a PC-based, in-circuit simulation tool to support the feature family COP8 products.

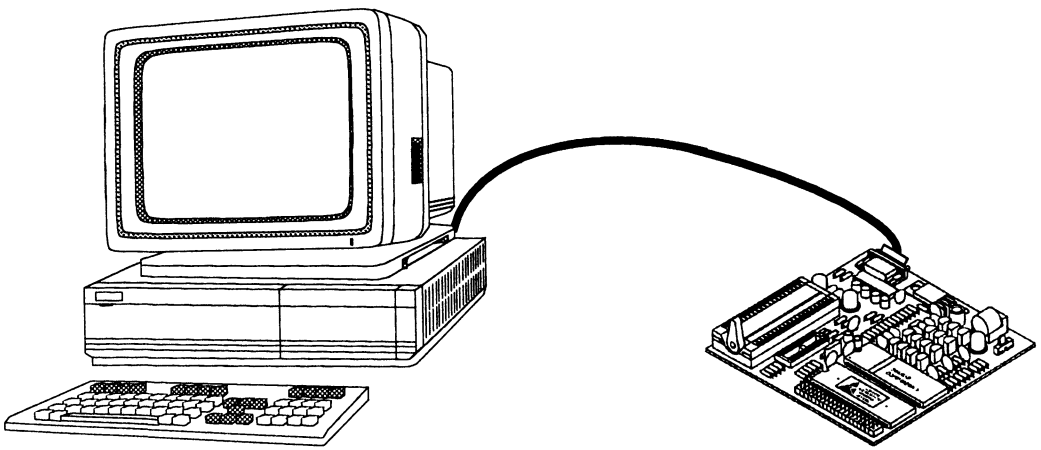
See *Figure 21* for configuration.

The simulation capability is a very low cost means of evaluating the general COP8 architecture. In addition, the EPU has programming capability, with added adapters, for programming the whole COP8 product family of OTP and EPROM products. The product includes the following features:

- Non-real-time in-circuit simulation. Program overlay memory is PC resident; instructions are downloaded over RS-232 as executed. Approximate performance is 20 kHz.
- Includes a 40-pin DIP cable adapter. Other target packages are not supported. All processor I/O pins are cabled to the application development environment.
- Full 32 kbytes of loadable programming space that overlays (replaces) the on-chip ROM or EPROM. On-chip RAM and I/O blocks are used directly or recreated as necessary.
- On-chip timer and watch-dog execution are not well synchronized to the instruction simulation.
- 100 frames of synchronous trace memory. The display can be HLL source (e.g., C source), assembly or mixed. The most recent history prior to a break is available in the trace memory.
- Up to eight software configured break points; uses INTR instruction which is modestly intrusive.
- Common look-feel debugger software across all MetaLink products-only supported features are selectable.
- Tool set integrated interactive symbolic debugger—supports both assembler (COFF) and C Compiler (.COD) SDK linked object formats.
- Instruction by instruction memory/register changes displayed when in single step operation.
- Processor specific symbolic display of registers and bit level assignments, configured from master model file.
- Halt/Idle mode notification. Restart requires special handling.
- Programming menu supports full product line of programmable OTP and EPROM COP8 products. Only a 40 ZIF socket is available on the EPU unit. Adapters are available for other part package configurations.
- Integral wall mount power supply provides 5V and develops the required V_{PP} to program parts.
- Includes a copy of COP8-DEV-IBMA assembler, linker SDK.

EPU Order Information

Evaluation Programming Unit	
EPU-COP888GG	Evaluation Programming Unit with debugger & programmer control software and 40 ZIF programming socket
General Programming Adapters	
COP8-PGMA-DS	28 & 20 DIP and SOIC adapter
COP8-PGMA-DS44P	28 & 20 DIP and SOIC plus 44 PLCC adapter



TL/DD/11214-35

FIGURE 21. EPU-COP8 Tool Environment

5

Development Support (Continued)

COP8 ASSEMBLER/LINKER SOFTWARE DEVELOPMENT TOOL KIT

National Semiconductor offers a relocatable COP8 macro cross assembler, linker, librarian and utility software development tool kit. Features are summarized as follows:

- Basic and Feature Family instruction set by "device" type.
- Nested macro capability.
- Extensive set of assembler directives.
- Supported on PC/DOS platform.
- Generates National standard COFF output files.
- Integrated Linker & Librarian.
- Integrated utilities to generate ROM code file outputs.
- DUMPCOFF utility.

This product is integrated as a part of MetaLink tools as a development kit, fully supported by the MetaLink debugger. It may be ordered separately or it is bundled with the MetaLink products at no additional cost.

Order Information

Assembler SDK	
COP8-DEV-IBMA	Assembler SDK on installable 3.5" PC/DOS Floppy Disk Drive format. Periodic upgrades and most recent version is available on National's BBS and Internet.

COP8 C COMPILER

A C Compiler is developed and marketed by Byte Craft Limited. The COP8C compiler is a fully integrated development tool specifically designed to support the compact embedded configuration of the COP8 family of products.

Features are summarized as follows:

- ANSI C with some restrictions and extensions that optimize development for the COP8 embedded application.
- BITS data type extension. Register declaration #pragma with direct bit level definitions.
- C language support for interrupt routines.
- Expert system, rule based code generation and optimization.
- Performs consistency checks against the architectural definitions of the target COP8 device.
- Generates program memory code.
- Supports linking of compiled object or COP8 assembled object formats.
- Global optimization of linked code.
- Symbolic debug load format fully source level supported by the MetaLink debugger.

Approved List

Manufacturer	North America	Europe	Asia
BP Microsystems	(800) 225-2102 (713) 688-4600 Fax: (713) 688-0920	+49-8152-4183 +49-8856-932616	+852-234-16611 +852-2710-8121
Data I/O	(800) 426-1045 (206) 881-6444 Fax: (206) 882-1043	+44-0734-440011	Call North America
HI-LO	(510) 623-8860	Call Asia	+886-2-764-0215 Fax: +886-2-756-6403
ICE Technology	(800) 624-8949 (919) 430-7915	+44-1226-767404 Fax: 0-1226-370-434	
MetaLink	(800) 638-2423 (602) 926-0797 Fax: (602) 693-0681	+49-80 9156 96-0 Fax: +49-80 9123 86	+852-737-1800
Systems General	(408) 263-6667	+41-1-9450300	+886-2-9173005 Fax: +886-2-911-1283
Needhams	(916) 924-8037 Fax: (916) 924-8065		

Development Support (Continued)

SINGLE CHIP OTP/EMULATOR SUPPORT

The COP8 family is supported by single chip OTP emulators. For detailed information refer to the emulator specific datasheet and the emulator selection table below:

OTP Emulator Ordering Information

Device Number	Clock Option	Package	Emulates
COP87L88EGV-XE	Crystal/ HALT En	44 PLCC	COP888EG
COP87L88EGN-XE	Crystal/ HALT En	40 DIP	COP888EG
COP87L84EGN-XE	Crystal/ HALT En	28 DIP	COP884EG
COP87L84EGM-XE	Crystal/ HALT En	28 SO	COP884EG

INDUSTRY WIDE OTP/EPROM PROGRAMMING SUPPORT

Programming support, in addition to the MetaLink development tools, is provided by a full range of independent approved vendors to meet the needs from the engineering laboratory to full production.

AVAILABLE LITERATURE

For more information, please see the COP8 Basic Family User's Manual, Literature Number 620895, COP8 Feature Family User's Manual, Literature Number 620897 and National's Family of 8-bit Microcontrollers COP8 Selection Guide, Literature Number 630009.

DIAL-A-HELPER SERVICE

Dial-A-Helper is a service provided by the Microcontroller Applications group. The Dial-A-Helper is an Electronic Information System that may be accessed as a Bulletin Board System (BBS) via data modem, as an FTP site on the Internet via standard FTP client application or as an FTP site on the Internet using a standard Internet browser such as Netscape or Mosaic.

The Dial-A-Helper system provides access to an automated information storage and retrieval system. The system capabilities include a MESSAGE SECTION (electronic mail, when accessed as a BBS) for communications to and from the Microcontroller Applications Group and a FILE SECTION which consists of several file areas where valuable application software and utilities could be found.

DIAL-A-HELPER BBS via a Standard Modem

Modem: CANADA/U.S.: (800) NSC-MICRO
(800) 672-6427
EUROPE: (+49) 0-8141-351332
Baud: 14.4k
Set-Up: Length: 8-Bit
Parity: None
Stop Bit: 1
Operation: 24 Hours, 7 Days

DIAL-A-HELPER via FTP

ftp nscmicro.nsc.com
user: anonymous
password: username@yourhost.site.domain

DIAL-A-HELPER via a WorldWide Web Browser

ftp://nscmicro. nsc.com

National Semiconductor on the WorldWide Web

See us on the WorldWide Web at: http://www.national.com

CUSTOMER RESPONSE CENTER

Complete product information and technical support is available from National's customer response centers.

CANADA/ U.S.:	Tel:		(800) 272-9959
	email:		support@tevm2.nsc.com
EUROPE:	email:		europe.support@nsc.com
	Deutsch	Tel:	+49 (0) 180-530 85 85
	English	Tel:	+49 (0) 180-532 78 32
	Français	Tel:	+49 (0) 180-532 93 58
	Italiano	Tel:	+49 (0) 180-534 16 80
JAPAN:	Tel:		+81-043-299-2309
S.E. ASIA:	Beijing	Tel:	(+86) 10-6856-8601
	Shanghai	Tel:	(+86) 21-6415-4092
	Hong Kong	Tel:	(+852) 2737-1600
	Korea	Tel:	(+82) 2-3771-6909
	Malaysia	Tel:	(+60-4) 644-9061
	Singapore	Tel:	(+65) 255-2226
	Taiwan	Tel:	+886-2-521-3288
AUSTRALIA:	Tel:		(+61) 3-9558-9999
INDIA:	Tel:		(+91) 80-559-9467

National Semiconductor

PRELIMINARY

COP688FH/COP684FH/COP888FH/COP884FH/ COP988FH/COP984FH
8-Bit Microcontroller with UART, Three Multi-Function Timers and Multiply/Divide Block

General Description

The COP888 family of microcontrollers uses an 8-bit single chip core architecture fabricated with National Semiconductor's M²CMOS™ process technology. The COP888FH is a member of this expandable 8-bit core processor family of microcontrollers. (Continued)

Key Features

- Multiply/Divide Functions
- Full duplex UART
- Three 16-bit timers, each with two 16-bit registers supporting:
  - — Processor Independent PWM mode
  - — External Event counter mode
  - — Input Capture mode
- Quiet design (low radiated emissions)
- 12 kbytes on-board ROM
- 512 bytes on-board RAM

Additional Peripheral Features

- Idle Timer
- Multi-Input Wakeup (MIWU) with optional interrupts (8)
- Two analog comparators
- WATCHDOG™ and Clock Monitor logic
- MICROWIRE/PLUS™ serial I/O

I/O Features

- Memory mapped I/O
- Software selectable I/O options (TRI-STATE®, Push-Pull, Weak Pull-Up, and High Impedance)
- Schmitt trigger inputs on ports G and L
- Packages:
  - — 40 DIP with 36 I/O pins
  - — 44 PLCC with 40 I/O pins
  - — 28 SO with 24 I/O pins
  - — 28 DIP with 24 I/O pins

CPU/Instruction Set Features

- 1 μs instruction cycle time
- Fourteen multi-source vectored interrupts servicing
  - — External Interrupt
  - — Idle Timer T0
  - — Three Timers (Each with 2 Interrupts)
  - — MICROWIRE/PLUS
  - — Multi-Input Wake Up
  - — Software Trap
  - — UART (2)
  - — Default VIS (default interrupt)
- Versatile and easy to use instruction set
- 8-bit Stack Pointer (SP)—stack in RAM
- Two 8-bit Register Indirect Data Memory Pointers (B and X)

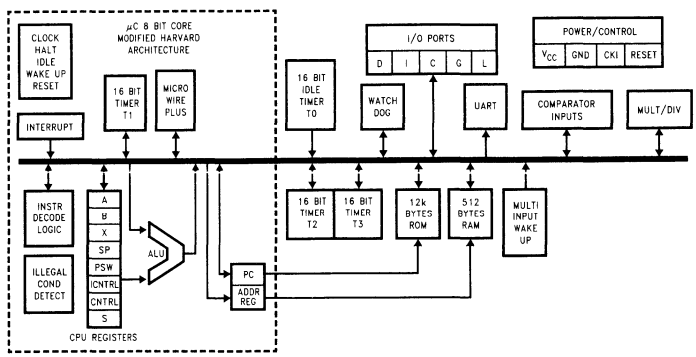
Fully Static CMOS

- Low current drain (typically < 5 μA)
- Two power saving modes: HALT and IDLE
- Single supply operation: 2.5V–5.5V
- Temperature ranges: 0°C to +70°C,
  −40°C to +85°C
  −55°C to +125°C

Development Support

- Emulation and OTP devices
- Real time emulation and full program debug offered by MetaLink Development System

Block Diagram



TL/DD/12602–1

FIGURE 1. COP888FH Block Diagram

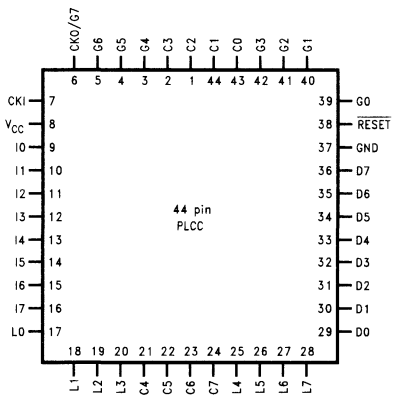
General Description (Continued)

It is a fully static part, fabricated using double-metal silicon gate microCMOS technology. Features include an 8-bit memory mapped architecture, MICROWIRE/PLUS serial I/O, three 16-bit timer/counters supporting three modes (Processor Independent PWM generation, External Event counter, and Input Capture mode capabilities), full duplex UART, two comparators, and two power savings modes (HALT and IDLE), both with a multi-sourced wakeup/interrupt capability. This multi-sourced interrupt capability may also be used independent of the HALT or IDLE modes. Each I/O pin has software selectable configurations. The device operates over a voltage range of 2.5V to 5.5V. High throughput is achieved with an efficient, regular instruction set operating at a maximum of 1 μs per instruction rate.

Low radiated emissions are achieved by gradual turn-on output drivers and internal I_{CC} smoothing filters on the chip logic and crystal oscillator.

Connection Diagrams

Plastic Chip Carrier

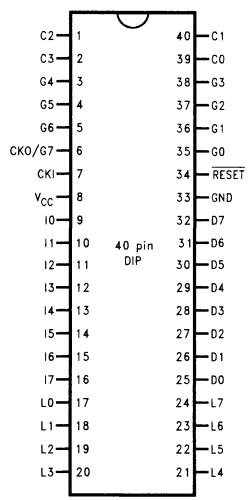


TL/DD/12602-2

Top View

Order Number COP688FH-XXX/V, COP888FH-XXX/V or COP988FH-XXX/V
See NS Plastic Chip Package Number V44A

Dual-In-Line Package

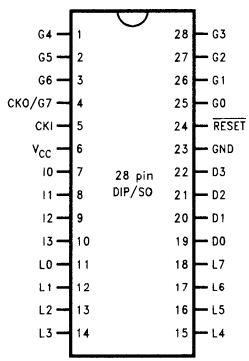


TL/DD/12602-3

Top View

Order Number COP688FH-XXX/N, COP888FH-XXX/N or COP988FH-XXX/N
See NS Molded Package Number N40A

Dual-In-Line Package



TL/DD/12602-4

Top View

Order Number COP684FH-XXX/M, COP884FH-XXX/M, COP984FH-XXX/M, COP684FH-XXX/N, COP884FH-XXX/N or COP984FH-XXX/N
See NS Molded Package Number M28B or N28A

FIGURE 2. Connection Diagrams

Connection Diagrams (Continued)

Pinouts for 28-, 40- and 44-Pin Packages

Port	Type	Alt. Fun	Alt. Fun	28-Pin Pack.	40-Pin Pack.	44-Pin Pack.
L0	I/O	MIWU		11	17	17
L1	I/O	MIWU	CKX	12	18	18
L2	I/O	MIWU	TDX	13	19	19
L3	I/O	MIWU	RDX	14	20	20
L4	I/O	MIWU	T2A	15	21	25
L5	I/O	MIWU	T2B	16	22	26
L6	I/O	MIWU	T3A	17	23	27
L7	I/O	MIWU	T3B	18	24	28
G0	I/O	INT		25	35	39
G1	WDOUT			26	36	40
G2	I/O	T1B		27	37	41
G3	I/O	T1A		28	38	42
G4	I/O	SO		1	3	3
G5	I/O	SK		2	4	4
G6	I	SI		3	5	5
G7	I/CKO	HALT Restart		4	6	6
D0	O			19	25	29
D1	O			20	26	30
D2	O			21	27	31
D3	O			22	28	32
D4	O				29	33
D5	O				30	34
D6	O				31	35
D7	O				32	36
I0	I			7	9	9
I1	I	COMP1IN−		8	10	10
I2	I	COMP1IN+		9	11	11
I3	I	COMP1OUT		10	12	12
I4	I	COMP2IN−			13	13
I5	I	COMP2IN+			14	14
I6	I	COMP2OUT			15	15
I7	I				16	16
C0	I/O				39	43
C1	I/O				40	44
C2	I/O				1	1
C3	I/O				2	2
C4	I/O					21
C5	I/O					22
C6	I/O					23
C7	I/O					24
V_{CC}				6	8	8
GND				23	33	37
CKI				5	7	7
$\overline{\text{RESET}}$				24	34	38

COP98XFH

Absolute Maximum Ratings

If Military/Aerospace specified devices are required, please contact the National Semiconductor Sales Office/Distributors for availability and specifications.

Supply Voltage (V_{CC})	7V
Voltage at Any Pin	−0.3V to V_{CC} + 0.3V
Total Current into V_{CC} Pin (Source)	100 mA
Total Current out of GND Pin (Sink)	110 mA
Storage Temperature Range	−65°C to +140°C

Note: *Absolute maximum ratings indicate limits beyond which damage to the device may occur. DC and AC electrical specifications are not ensured when operating the device at absolute maximum ratings.*

DC Electrical Characteristics COP98XFH: 0°C ≤ T_A ≤ +70°C unless otherwise specified

Parameter	Conditions	Min	Typ	Max	Units
Operating Voltage COP98XCS		2.5		4.0	V
COP98XCSH		4.0		6.0	V
Power Supply Ripple (Note 1)	Peak-to-Peak			0.1 V_{CC}	V
Supply Current (Note 2)					
CKI = 10 MHz	V_{CC} = 5.5V, t_c = 1 μs			12.5	mA
CKI = 4 MHz	V_{CC} = 5.5V, t_c = 2.5 μs			5.5	mA
CKI = 4 MHz	V_{CC} = 4V, t_c = 2.5 μs			2.5	mA
CKI = 1 MHz	V_{CC} = 4V, t_c = 10 μs			1.4	mA
HALT Current (Note 3)	V_{CC} = 5.5V, CKI = 0 MHz		<5	8	μA
	V_{CC} = 4V, CKI = 0 MHz		<3	4	μA
IDLE Current					
CKI = 10 MHz	V_{CC} = 5.5V, t_c = 1 μs			3.5	mA
CKI = 4 MHz	V_{CC} = 5.5V, t_c = 2.5 μs			2.5	mA
CKI = 1 MHz	V_{CC} = 4V, t_c = 10 μs			0.7	mA
Input Levels					
$\overline{\text{RESET}}$					
Logic High		0.8 V_{CC}			V
Logic Low				0.2 V_{CC}	V
CKI (External and Crystal Osc. Modes)					
Logic High		0.7 V_{CC}			V
Logic Low				0.2 V_{CC}	V
All Other Inputs					
Logic High		0.7 V_{CC}			V
Logic Low				0.2 V_{CC}	V
Hi-Z Input Leakage	V_{CC} = 5.5V, V_{IN} = 0V	−1		+1	μA
Input Pullup Current	V_{CC} = 5.5V, V_{IN} = 0V	−40		−250	μA
G and L Port Input Hysteresis				0.35 V_{CC}	V
Output Current Levels					
D Outputs					
Source	V_{CC} = 4V, V_{OH} = 3.3V	−0.4			mA
	V_{CC} = 2.5V, V_{OH} = 1.8V	−0.2			mA
Sink	V_{CC} = 4V, V_{OL} = 1V	10			mA
	V_{CC} = 2.5V, V_{OL} = 0.4V	2.0			mA
All Others					
Source (Weak Pull-Up Mode)	V_{CC} = 4V, V_{OH} = 2.7V	−10		−100	μA
	V_{CC} = 2.5V, V_{OH} = 1.8V	−2.5		−33	μA
Source (Push-Pull Mode)	V_{CC} = 4V, V_{OH} = 3.3V	−0.4			mA
	V_{CC} = 2.5V, V_{OH} = 1.8V	−0.2			mA
Sink (Push-Pull Mode)	V_{CC} = 4V, V_{OL} = 0.4V	1.6			mA
	V_{CC} = 2.5V, V_{OL} = 0.4V	0.7			mA
TRI-STATE Leakage	V_{CC} = 5.5V	−1		+1	μA

COP98XFH

DC Electrical Characteristics COP98XFH:

$0°C \leq T_A \leq +70°C$ unless otherwise specified (Continued)

Parameter	Conditions	Min	Typ	Max	Units
Allowable Sink/Source Current per Pin					
D Outputs (Sink)				15	mA
All Others				3	mA
Maximum Input Current without Latchup (Note 4)	$T_A = 25°C$			±100	mA
RAM Retention Voltage, V_r	500 ns Rise and Fall Time (Min)	2			V
Input Capacitance				7	pF
Load Capacitance on D2				1000	pF

AC Electrical Characteristics COP98XFH: $0°C \leq T_A \leq +70°C$ unless otherwise specified

Parameter	Conditions	Min	Typ	Max	Units
Instruction Cycle Time (t_c)	$4V \leq V_{CC} \leq 5.5V$	1		DC	μs
Crystal Resonator or External	$2.5V \leq V_{CC} < 4V$	2.5		DC	μs
R/C Oscillator	$4V \leq V_{CC} \leq 5.5V$	3		DC	μs
	$2.5V \leq V_{CC} < 4V$	7.5		DC	μs
CKI Clock Duty Cycle (Note 5)	f = Max	45		55	%
Rise Time (Note 5)	f = 10 MHz Ext Clock			5	μs
Fall Time (Note 5)	f = 10 MHz Ext Clock			5	μs
Inputs					
t_{SETUP}	$4V \leq V_{CC} \leq 5.5V$	200			ns
	$2.5V \leq V_{CC} < 4V$	500			ns
t_{HOLD}	$4V \leq V_{CC} \leq 5.5V$	60			ns
	$2.5V \leq V_{CC} < 4V$	150			ns
Output Propagation Delay (Note 5)	$R_L = 2.2k, C_L = 100$ pF				
t_{PD1}, t_{PD0}					
SO, SK	$4V \leq V_{CC} \leq 5.5V$			0.7	μs
	$2.5V \leq V_{CC} < 4V$			1.75	μs
All Others	$4V \leq V_{CC} \leq 5.5V$			1	μs
	$2.5V \leq V_{CC} < 4V$			2.5	μs
MICROWIRE™ Setup Time (t_{UWS})		20			ns
MICROWIRE Hold Time (t_{UWH})		56			ns
MICROWIRE Output Propagation Delay (t_{UPD})				220	ns
Input Pulse Width					
Interrupt Input High Time		1			t_c
Interrupt Input Low Time		1			t_c
Timer Input High Time		1			t_c
Timer Input Low Time		1			t_c
Reset Pulse Width		1			μs

Note 1: Rate of voltage change must be less than 0.5V/ms.

Note 2: Supply current is measured after running 2000 cycles with a square wave CKI input, CKO open, inputs at rails and outputs open.

Note 3: The HALT mode will stop CKI from oscillating in the RC and the Crystal configurations. Test conditions: All inputs tied to V_{CC}, L and G ports in the TRI-STATE mode and tied to ground, all outputs low and tied to ground. The clock monitor and the comparator are disabled.

Note 4: Pins G6 and $\overline{\text{RESET}}$ are designed with a high voltage input network for factory testing. These pins allow input voltages greater than V_{CC} and the pins will have sink current to V_{CC} when biased at voltages greater than V_{CC} (the pins do not have source current when biased at a voltage below V_{CC}). The effective resistance to V_{CC} is 750Ω (typical). These two pins will not latch up. The voltage at the pins must be limited to less than 14V.

Note 5: The output propagation delay is referenced to the end of the instruction cycle where the output change occurs.

Absolute Maximum Ratings

If Military/Aerospace specified devices are required, please contact the National Semiconductor Sales Office/Distributors for availability and specifications.

Supply Voltage (V_{CC})	7V
Voltage at Any Pin	−0.3V to V_{CC} + 0.3V
Total Current into V_{CC} Pin (Source)	100 mA
Total Current out of GND Pin (Sink)	110 mA
Storage Temperature Range	−65°C to +140°C

Note: *Absolute maximum ratings indicate limits beyond which damage to the device may occur. DC and AC electrical specifications are not ensured when operating the device at absolute maximum ratings.*

DC Electrical Characteristics COP88XFH: $-40°C \le T_A \le +85°C$ unless otherwise specified

Parameter	Conditions	Min	Typ	Max	Units
Operating Voltage		2.5		5.5	V
Power Supply Ripple (Note 1)	Peak-to-Peak			0.1 V_{CC}	V
Supply Current (Note 2)					
CKI = 10 MHz	V_{CC} = 5.5V, t_c = 1 μs			12.5	mA
CKI = 4 MHz	V_{CC} = 5.5V, t_c = 2.5 μs			5.5	mA
CKI = 4 MHz	V_{CC} = 4.0V, t_c = 2.5 μs			2.5	mA
CKI = 1 MHz	V_{CC} = 4.0V, t_c = 10 μs			1.4	mA
HALT Current (Note 3)	V_{CC} = 5.5V, CKI = 0 MHz		<5	10	μA
	V_{CC} = 4.0V, CKI = 0 MHz		<3	6	μA
IDLE Current					
CKI = 10 MHz	V_{CC} = 5.5V, t_c = 1 μs			3.5	mA
CKI = 4 MHz	V_{CC} = 5.5V, t_c = 2.5 μs			2.5	mA
CKI = 1 MHz	V_{CC} = 4.0V, t_c = 10 μs			0.7	mA
Input Levels					
$\overline{\text{RESET}}$					
Logic High		0.8 V_{CC}			V
Logic Low				0.2 V_{CC}	V
CKI (All Other Inputs)					
Logic High		0.7 V_{CC}			V
Logic Low				0.2 V_{CC}	V
Hi-Z Input Leakage	V_{CC} = 5.5V, V_{IN} = 0V	−2		+2	μA
Input Pullup Current	V_{CC} = 5.5V, V_{IN} = 0V	−40		−250	μA
G and L Port Input Hysteresis	(Note 5)			0.35 V_{CC}	V
Output Current Levels					
D Outputs					
Source	V_{CC} = 4V, V_{OH} = 3.3V	−0.4			mA
	V_{CC} = 2.5V, V_{OH} = 1.8V	−0.2			mA
Sink	V_{CC} = 4V, V_{OL} = 1V	10			mA
	V_{CC} = 2.5V, V_{OL} = 0.4V	2.0			mA
All Others					
Source (Weak Pull-Up Mode)	V_{CC} = 4V, V_{OH} = 2.7V	−10		−100	μA
	V_{CC} = 2.5V, V_{OH} = 1.8V	−2.5		−33	μA
Source (Push-Pull Mode)	V_{CC} = 4V, V_{OH} = 3.3V	−0.4			mA
	V_{CC} = 2.5V, V_{OH} = 1.8V	−0.2			mA
Sink (Push-Pull Mode)	V_{CC} = 4V, V_{OL} = 0.4V	1.6			mA
	V_{CC} = 2.5V, V_{OL} = 0.4V	0.7			mA
TRI-STATE Leakage	V_{CC} = 5.5V	−2		+2	μA

COP88XFH

DC Electrical Characteristics COP88XFH:

$-40°C \leq T_A \leq +85°C$ unless otherwise specified (Continued)

Parameter	Conditions	Min	Typ	Max	Units
Allowable Sink/Source Current per Pin					
D Outputs (Sink)				15	mA
All others				3	mA
Maximum Input Current without Latchup (Notes 4, 5)	Room Temp			±100	mA
RAM Retention Voltage, V_r	500 ns Rise and Fall Time (Min)	2			V
Input Capacitance	(Note 5)			7	pF
Load Capacitance on D2	(Note 5)			1000	pF

AC Electrical Characteristics COP88XFH: $-40°C \leq T_A \leq +85°C$ unless otherwise specified

Parameter	Conditions	Min	Typ	Max	Units
Instruction Cycle Time (t_c)					
Crystal Resonator or External	$2.5V \leq V_{CC} \leq 4.0V$	2.5		DC	μs
	$4.0V \leq V_{CC} \leq 5.5V$	1.0		DC	μs
R/C Oscillator	$2.5V \leq V_{CC} < 4.0V$	7.5		DC	μs
	$4.0V \leq V_{CC} \leq 5.5V$	3.0		DC	μs
CKI Clock Duty Cycle (Note 5)	f = Max	45		55	%
Rise Time (Note 5)	f = 10 MHz Ext Clock			5	μs
Fall Time (Note 5)	f = 10 MHz Ext Clock			5	μs
Inputs					
t_{SETUP}	$4.0V \leq V_{CC} \leq 5.5V$	200			ns
	$2.5V \leq V_{CC} < 4.0V$	500			ns
t_{HOLD}	$4.0V \leq V_{CC} \leq 5.5V$	60			ns
	$2.5V \leq V_{CC} < 4.0V$	150			ns
Output Propagation Delay	$R_L = 2.2k, C_L = 100$ pF				
t_{PD1}, t_{PD0}					
SO, SK	$4.0V \leq V_{CC} \leq 5.5V$			0.7	μs
	$2.5V \leq V_{CC} < 4.0V$			1.75	μs
All Others	$4.0V \leq V_{CC} \leq 5.5V$			1	μs
	$2.5V \leq V_{CC} < 4.0V$			2.5	μs
MICROWIRE Setup Time (t_{UWS}) (Note 5)	$V_{CC} \geq 4.0V$	20			ns
MICROWIRE Hold Time (t_{UWH}) (Note 5)	$V_{CC} \geq 4.0V$	56			ns
MICROWIRE Output Propagation Delay (t_{UPD})	$V_{CC} \geq 4.0V$			220	ns
Input Pulse Width (Note 6)					
Interrupt Input High Time		1			t_c
Interrupt Input Low Time		1			t_c
Timer 1, 2, 3 Input High Time		1			t_c
Timer 1, 2, 3 Input Low Time		1			t_c
Reset Pulse Width		1			μs

Note 1: Maximum rate of voltage change must be less than 0.5V/ms.

Note 2: Supply current is measured after running 2000 cycles with a square wave CKI input, CKO open, inputs at rails and outputs open.

Note 3: The HALT mode will stop CKI from oscillating in the RC and the Crystal configurations. Measurement of I_{DD} HALT is done with device neither sourcing or sinking current; with L, C, and G0–G5 programmed as low outputs and not driving a load; all outputs programmed low and not driving a load; all inputs tied to V_{CC}; clock monitor and comparators disabled. Parameter refers to HALT mode entered via setting bit 7 of the G Port data register. Part will pull up CKI during HALT in crystal clock mode.

Note 4: Pins G6 and $\overline{\text{RESET}}$ are designed with a high voltage input network. These pins allow input voltages greater than V_{CC} and the pins will have sink current to V_{CC} when biased at voltages greater than V_{CC} (the pins do not have source current when biased at a voltage below V_{CC}). The effective resistance to V_{CC} is 750Ω (typical). These two pins will not latch up. The voltage at the pins must be limited to less than 14V. **WARNING: Voltages in excess of 14V will cause damage to the pins. This warning excludes ESD transients.**

Note 5: Parameter characterized but not tested.

Note 6: t_c = Instruction cycle time.

COP68XFH

Absolute Maximum Ratings

If Military/Aerospace specified devices are required, please contact the National Semiconductor Sales Office/Distributors for availability and specifications.

Supply Voltage (V_{CC})	7V
Voltage at Any Pin	−0.3V to V_{CC} + 0.3V
Total Current into V_{CC} Pin (Source)	100 mA
Total Current out of GND Pin (Sink)	110 mA
Storage Temperature Range	−65°C to +140°C

Note: *Absolute maximum ratings indicate limits beyond which damage to the device may occur. DC and AC electrical specifications are not ensured when operating the device at absolute maximum ratings.*

DC Electrical Characteristics COP68XFH: $-55°C \le T_A \le +125°C$ unless otherwise specified

Parameter	Conditions	Min	Typ	Max	Units
Operating Voltage		4.5		5.5	V
Power Supply Ripple (Note 1)	Peak-to-Peak			0.1 V_{CC}	V
Supply Current (Note 2)					
CKI = 10 MHz	V_{CC} = 5.5V, t_c = 1 μs			12.5	mA
CKI = 4 MHz	V_{CC} = 5.5V, t_c = 2.5 μs			5.5	mA
HALT Current (Note 3)	V_{CC} = 5.5V, CKI = 0 MHz		<10	30	μA
IDLE Current					
CKI = 10 MHz	V_{CC} = 5.5V, t_c = 1 μs			3.5	mA
CKI = 4 MHz	V_{CC} = 5.5V, t_c = 2.5 μs			2.5	mA
Input Levels					
$\overline{\text{RESET}}$					
Logic High		0.8 V_{CC}			V
Logic Low				0.2 V_{CC}	V
CKI (All Other Inputs)					
Logic High		0.7 V_{CC}			V
Logic Low				0.2 V_{CC}	V
Hi-Z Input Leakage	V_{CC} = 5.5V, V_{IN} = 0V	−5		+5	μA
Input Pullup Current	V_{CC} = 5.5V, V_{IN} = 0V	−35		−400	μA
G and L Port Input Hysteresis	(Note 5)			0.35 V_{CC}	V
Output Current Levels					
D Outputs					
Source	V_{CC} = 4.5V, V_{OH} = 3.3V	−0.4			mA
Sink	V_{CC} = 4.5V, V_{OL} = 1V	9			mA
All Others					
Source (Weak Pull-Up Mode)	V_{CC} = 4.5V, V_{OH} = 2.7V	−9		−140	μA
Source (Push-Pull Mode)	V_{CC} = 4.5V, V_{OH} = 3.3V	−0.4			mA
Sink (Push-Pull Mode)	V_{CC} = 4.5V, V_{OL} = 0.4V	1.4			mA
TRI-STATE Leakage	V_{CC} = 5.5V	−5		+5	μA

COP68XFH

DC Electrical Characteristics COP68XFH: $-55°C \le T_A \le +125°C$ unless otherwise specified (Continued)

Parameter	Conditions	Min	Typ	Max	Units
Allowable Sink/Source Current per Pin					
D Outputs (Sink)				12	mA
All others				2.5	mA
Maximum Input Current without Latchup (Note 4, 5)	Room Temp			±100	mA
RAM Retention Voltage, V_r	500 ns Rise and Fall Time (Min)	2			V
Input Capacitance	(Note 5)			7	pF
Load Capacitance on D2	(Note 5)			1000	pF

AC Electrical Characteristics COP68XFH: $-55°C \le T_A \le +125°C$ unless otherwise specified

Parameter	Conditions	Min	Typ	Max	Units
Instruction Cycle Time (t_c)					
Crystal Resonator or External	$V_{CC} \ge 4.5V$	1.0		DC	μs
CKI Clock Duty Cycle (Note 5)	f = Max	45		55	%
Rise Time (Note 5)	f = 10 MHz Ext Clock			5	μs
Fall Time (Note 5)	f = 10 MHz Ext Clock			5	μs
Inputs					
t_{SETUP}	$V_{CC} \ge 4.5V$	200			ns
t_{HOLD}	$V_{CC} \ge 4.5V$	60			ns
Output Propagation Delay (Note 6)	R_L = 2.2k, C_L = 100 pF				
t_{PD1}, t_{PD0}					
SO, SK	$V_{CC} \ge 4.5V$			0.7	μs
All Others	$V_{CC} \ge 4.5V$			1	μs
MICROWIRE™ Setup Time (t_{UWS}) (Note 5)		20			ns
MICROWIRE Hold Time (t_{UWH}) (Note 5)		56			ns
MICROWIRE Output Propagation Delay (t_{UPD})				220	ns
Input Pulse Width (Note 6)					
Interrupt Input High Time		1			t_c
Interrupt Input Low Time		1			t_c
Timer 1, 2, 3 Input High Time		1			t_c
Timer 1, 2, 3 Input Low Time		1			t_c
Reset Pulse Width		1			μs

Note 1: Maximum rate of voltage change must be less than 0.5V/ms.

Note 2: Supply current is measured after running 2000 cycles with a square wave CKI input, CKO open, inputs at rails and outputs open.

Note 3: The HALT mode will stop CKI from oscillating in the RC and the Crystal configurations. Measurement of I_{DD} HALT is done with device neither sourcing or sinking current; with L, C, and G0–G5 programmed as low outputs and not driving a load; all outputs programmed low and not driving a load; all inputs tied to V_{CC}; clock monitor and comparators disabled. Parameter refers to HALT mode entered via setting bit 7 of the G Port data register. Part will pull up CKI during HALT in crystal clock mode.

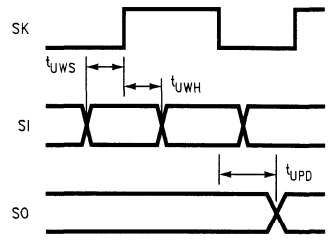
Note 4: Pins G6 and $\overline{\text{RESET}}$ are designed with a high voltage input network. These pins allow input voltages greater than V_{CC} and the pins will have sink current to V_{CC} when biased at voltages greater than V_{CC} (the pins do not have source current when biased at a voltage below V_{CC}). The effective resistance to V_{CC} is 750Ω (typical). These two pins will not latch up. The voltage at the pins must be limited to less than 14V. **WARNING: Voltages in excess of 14V will cause damage to the pins. This warning excludes ESD transients.**

Note 5: Parameter characterized but not tested.

Note 6: t_c = Instruction cycle time.

Comparators AC and DC Characteristics $V_{CC} = 5V, -40°C \leq T_A \leq +85°C$

Parameter	Conditions	Min	Typ	Max	Units
Input Offset Voltage	$0.4V \leq V_{IN} \leq V_{CC} - 1.5V$		±10	±25	mV
Input Common Mode Voltage Range		0.4		$V_{CC} - 1.5$	V
Low Level Output Current	$V_{OL} = 0.4V$	1.6			mA
High Level Output Current	$V_{OH} = 4.6V$	1.6			mA
DC Supply Current Per Comparator (When Enabled)				250	μA
Response Time	100 mV Overdrive, 100 pF Load			1	μs



TL/DD/12602-5

FIGURE 3. MICROWIRE/PLUS Timing

Pin Descriptions

V_{CC} and GND are the power supply pins. All V_{CC} and GND pins must be connected.

CKI is the clock input. This can come from an R/C generated oscillator, or a crystal oscillator (in conjunction with CKO). See Oscillator Description section.

$\overline{\text{RESET}}$ is the master reset input. See Reset Description section.

The device contains three bidirectional 8-bit I/O ports (C, G and L), where each individual bit may be independently configured as an input (Schmitt trigger inputs on ports L and G), output or TRI-STATE under program control. Three data memory address locations are allocated for each of these I/O ports. Each I/O port has two associated 8-bit memory mapped registers, the CONFIGURATION register and the output DATA register. A memory mapped address is also reserved for the input pins of each I/O port. (See the memory map for the various addresses associated with the I/O ports.) *Figure 4* shows the I/O port configurations. The DATA and CONFIGURATION registers allow for each port bit to be individually configured under software control as shown below:

CONFIGURATION Register	DATA Register	Port Set-Up
0	0	Hi-Z Input (TRI-STATE Output)
0	1	Input with Weak Pull-Up
1	0	Push-Pull Zero Output
1	1	Push-Pull One Output

PORT L is an 8-bit I/O port. All L-pins have Schmitt triggers on the inputs.

The Port L supports Multi-Input Wake Up on all eight pins. L1 is used for the UART external clock. L2 and L3 are used for the UART transmit and receive. L4 and L5 are used for the timer input functions T2A and T2B. L6 and L7 are used for the timer input functions T3A and T3B.

The Port L has the following alternate features:

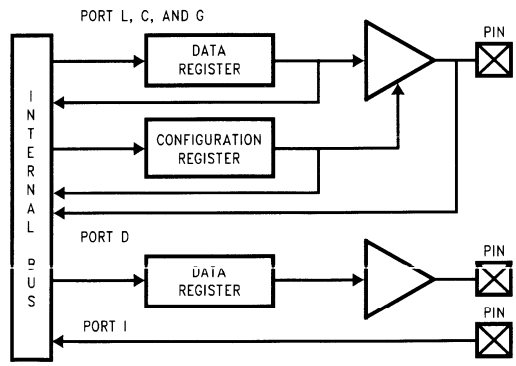
- L0 MIWU
- L1 MIWU or CKX
- L2 MIWU or TDX
- L3 MIWU or RDX
- L4 MIWU or T2A
- L5 MIWU or T2B
- L6 MIWU or T3A
- L7 MIWU or T3B

Port G is an 8-bit port with 5 I/O pins (G0, G2–G5), an input pin (G6), and two dedicated output pins (G1 and G7). Pins G0 and G2–G6 all have Schmitt Triggers on their inputs. Pin G1 serves as the dedicated WDOUT WATCHDOG output, while pin G7 is either input or output depending on the oscillator mask option selected. With the crystal oscillator option selected, G7 serves as the dedicated output pin for the CKO clock output. With the single-pin R/C oscillator mask option selected, G7 serves as a general purpose input pin but is also used to bring the device out of HALT mode with a low to high transition on G7. There are two registers associated with the G Port, a data register and a configuration register. Therefore, each of the 5 I/O bits (G0, G2–G5) can be individually configured under software control.

Since G6 is an input only pin and G7 is the dedicated CKO clock output pin (crystal clock option) or general purpose input (R/C clock option), the associated bits in the data and configuration registers for G6 and G7 are used for special purpose functions as outlined below. Reading the G6 and G7 data bits will return zeros.

Note that the chip will be placed in the HALT mode by writing a "1" to bit 7 of the Port G Data Register. Similarly the chip will be placed in the IDLE mode by writing a "1" to bit 6 of the Port G Data Register.

Writing a "1" to bit 6 of the Port G Configuration Register enables the MICROWIRE/PLUS to operate with the alternate phase of the SK clock. The G7 configuration bit, if set high, enables the clock start up delay after HALT when the R/C clock configuration is used.



TL/DD/12602–6

FIGURE 4. I/O Port Configurations

Pin Descriptions (Continued)

	Config Reg.	Data Reg.
G7	CLKDLY	HALT
G6	Alternate SK	IDLE

Port G has the following alternate features:

- G0 INTR (External Interrupt Input)
- G2 T1B (Timer T1 Capture Input)
- G3 T1A (Timer T1 I/O)
- G4 SO (MICROWIRE™ Serial Data Output)
- G5 SK (MICROWIRE Serial Clock)
- G6 SI (MICROWIRE Serial Data Input)

Port G has the following dedicated functions:

- G1 WDOUT WATCHDOG and/or Clock Monitor dedicated output
- G7 CKO Oscillator dedicated output or general purpose input

Port C is an 8-bit I/O port. The 40-pin device does not have a full complement of Port C pins. The unavailable pins are not terminated. A read operation for these unterminated pins will return unpredicatable values.

PORT I is an eight-bit Hi-Z input port. The 28-pin device does not have a full complement of Port I pins. The unavailable pins are not terminated i.e., they are floating. A read operation for these unterminated pins will return unpredictable values. The user must ensure that the software takes this into account by either masking or restricting the accesses to bit operations. The unterminated Port I pins will draw power only when addressed.

Port I1–I3 are used for Comparator 1. Port I4–I6 are used for Comparator 2.

The Port I has the following alternate features.

- I1 COMP1 −IN (Comparator 1 Negative Input)
- I2 COMP1 +IN (Comparator 1 Positive Input)
- I3 COMP1OUT (Comparator 1 Output)
- I4 COMP2 −IN (Comparator 2 Negative Input)
- I5 COMP2 +IN (Comparator 2 Positive Input)
- I6 COMP2OUT (Comparator 2 Output)

Port D is an 8-bit output port that is preset high when $\overline{\text{RESET}}$ goes low. The user can tie two or more D port outputs (except D2) together in order to get a higher drive.

Note: Care must be exercised with the D2 pin operation. At RESET, the external loads on this pin must ensure that the output voltages stay above 0.8 V_{CC} to prevent the chip from entering special modes. Also keep the external loading on D2 to less than 1000 pF.

Functional Description

The architecture of the device is modified Harvard architecture. With the Harvard architecture, the control store program memory (ROM) is separated from the data store memory (RAM). Both ROM and RAM have their own separate addressing space with separate address buses. The architecture, though based on Harvard architecture, permits transfer of data from ROM to RAM.

CPU REGISTERS

The CPU can do an 8-bit addition, subtraction, logical or shift operation in one instruction (t_C) cycle time.

There are six CPU registers:

A is the 8-bit Accumulator Register

PC is the 15-bit Program Counter Register

PU is the upper 7 bits of the program counter (PC)
PL is the lower 8 bits of the program counter (PC)

B is an 8-bit RAM address pointer, which can be optionally post auto incremented or decremented.

X is an 8-bit alternate RAM address pointer, which can be optionally post auto incremented or decremented.

SP is the 8-bit stack pointer, which points to the subroutine/interrupt stack (in RAM). The SP is initialized to RAM address 06F with reset.

S is the 8-bit Data Segment Address Register used to extend the lower half of the address range (00 to 7F) into 256 data segments of 128 bytes each.

All the CPU registers are memory mapped with the exception of the Accumulator (A) and the Program Counter (PC).

PROGRAM MEMORY

The program memory consists of 12288 bytes of ROM. These bytes may hold program instructions or constant data (data tables for the LAID instruction, jump vectors for the JID instruction, and interrupt vectors for the VIS instruction). The program memory is addressed by the 15-bit program counter (PC). All interrupts in the devices vector to program memory location 0FF Hex.

DATA MEMORY

The data memory address space includes the on-chip RAM and data registers, the I/O registers (Configuration, Data and Pin), the control registers, the MICROWIRE/PLUS SIO shift register, and the various registers, and counters associated with the timers (with the exception of the IDLE timer). Data memory is addressed directly by the instruction or indirectly by the B, X, SP pointers and S register.

The data memory consists of 512 bytes of RAM. Sixteen bytes of RAM are mapped as "registers" at addresses 0F0 to 0FF Hex. These registers can be loaded immediately, and also decremented and tested with the DRSZ (decrement register and skip if zero) instruction. The memory pointer registers X, SP, B and S are memory mapped into this space at address locations 0FC to 0FF Hex respectively, with the other registers being available for general usage.

The instruction set permits any bit in memory to be set, reset or tested. All I/O and registers (except A and PC) are memory mapped; therefore, I/O bits and register bits can be directly and individually set, reset and tested. The accumulator (A) bits can also be directly and individually tested.

Note: RAM contents are undefined upon power-up.

Data Memory Segment RAM Extension

Data memory address 0FF is used as a memory mapped location for the Data Segment Address Register (S).

The data store memory is either addressed directly by a single byte address within the instruction, or indirectly relative to the reference of the B, X, or SP pointers (each contains a single-byte address). This single-byte address allows an addressing range of 256 locations from 00 to FF hex. The upper bit of this single-byte address divides the data store memory into two separate sections as outlined previously. With the exception of the RAM register memory from address locations 00F0 to 00FF, all RAM memory is memory mapped with the upper bit of the single-byte address being equal to zero. This allows the upper bit of the single-byte

Data Memory Segment RAM Extension (Continued)

address to determine whether or not the base address range (from 0000 to 00FF) is extended. If this upper bit equals one (representing address range 0080 to 00FF), then address extension does not take place. Alternatively, if this upper bit equals zero, then the data segment extension register S is used to extend the base address range (from 0000 to 007F) from XX00 to XX7F, where XX represents the 8 bits from the S register. Thus the 128-byte data segment extensions are located from addresses 0100 to 017F for data segment 1, 0200 to 027F for data segment 2, etc., up to FF00 to FF7F for data segment 255. The base address range from 0000 to 007F represents data segment 0.

Figure 5 illustrates how the S register data memory extension is used in extending the lower half of the base address range (00 to 7F hex) into 256 data segments of 128 bytes each, with a total addressing range of 32 kbytes from XX00 to XX7F. This organization allows a total of 256 data segments of 128 bytes each with an additional upper base segment of 128 bytes. Furthermore, all addressing modes are available for all data segments. The S register must be changed under program control to move from one data segment (128 bytes) to another. However, the upper base segment (containing the 16 memory registers, I/O registers, control registers, etc.) is always available regardless of the contents of the S register, since the upper base segment (address range 0080 to 00FF) is independent of data segment extension.

The instructions that utilize the stack pointer (SP) always reference the stack as part of the base segment (Segment 0), regardless of the contents of the S register. The S register is not changed by these instructions. Consequently, the stack (used with subroutine linkage and interrupts) is always located in the base segment. The stack pointer will be initialized to point at data memory location 006F as a result of reset.

The 128 bytes of RAM contained in the base segment are split between the lower and upper base segments. The first 112 bytes of RAM are resident from address 0000 to 006F in the lower base segment, while the remaining 16 bytes of RAM represent the 16 data memory registers located at addresses 00F0 to 00FF of the upper base segment. No RAM is located at the upper sixteen addresses (0070 to 007F) of the lower base segment.

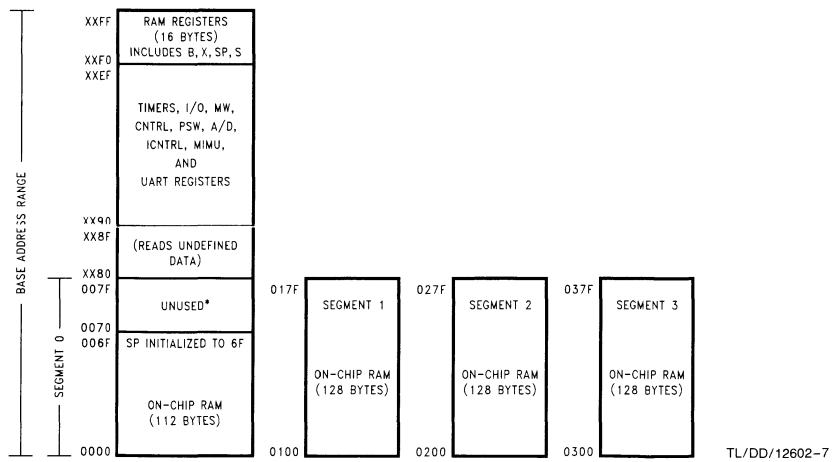
Additional RAM beyond these initial 128 bytes, however, will always be memory mapped in groups of 128 bytes (or less) at the data segment address extensions (XX00 to XX7F) of the lower base segment. The additional 128 bytes of RAM are memory mapped at address locations 0100 to 017F hex.

Reset

The $\overline{\text{RESET}}$ input when pulled low initializes the microcontroller. Initialization will occur whenever the $\overline{\text{RESET}}$ input is pulled low. Upon initialization, the data and configuration registers for ports L, G and C are cleared, resulting in these Ports being initialized to the TRI-STATE mode. Pin G1 of the G Port is an exception (as noted below) since pin G1 is dedicated as the WATCHDOG and/or Clock Monitor error output pin. Port D is set high. The PC, PSW, ICNTRL, CNTRL, T2CNTRL and T3CNTRL control registers are cleared. The UART registers PSR, ENU (except that TBMT bit is set), ENUR and ENUI are cleared. The Comparator Select Register is cleared. The S register is initialized to zero. The Multi-Input Wakeup registers WKEN, WKEDG and WKPND are cleared. (Wakeup register WKPND is unknown.) The stack pointer, SP, is initialized to 6F Hex.

The device comes out of reset with both the WATCHDOG logic and the Clock Monitor detector armed, with the WATCHDOG service window bits set and the Clock Monitor bit set. The WATCHDOG and Clock Monitor circuits are inhibited during reset. The WATCHDOG service window bits being initialized high default to the maximum WATCHDOG service window of 64k t_C clock cycles. The Clock Monitor bit being initialized high will cause a Clock Monitor error following reset if the clock has not reached the minimum specified frequency at the termination of reset. A Clock Monitor error will cause an active low error output on pin G1. This error output will continue until 16 t_C–32 t_C clock cycles following the clock frequency reaching the minimum specified value, at which time the G1 output will enter the TRI-STATE mode.

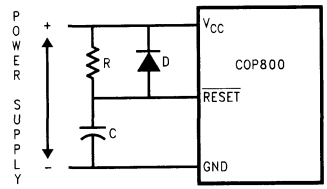
The external RC network shown in *Figure 6* should be used to ensure that the $\overline{\text{RESET}}$ pin is held low until the power supply to the chip stabilizes.



*Reads as all ones.

FIGURE 5. RAM Organization

Reset (Continued)



TL/DD/12602-8

RC > 5 × Power Supply Rise Time

FIGURE 6. Recommended Reset Circuit

Oscillator Circuits

The chip can be driven by a clock input on the CKI input pin which can be between DC and 10 MHz. The CKO output clock is on pin G7 (crystal configuration). The CKI input frequency is divided down by 10 to produce the instruction cycle clock (t_C).

Figure 7 shows the Crystal and R/C oscillator diagrams.

CRYSTAL OSCILLATOR

CKI and CKO can be connected to make a closed loop crystal (or resonator) controlled oscillator.

Table A shows the component values required for various standard crystal values.

R/C OSCILLATOR

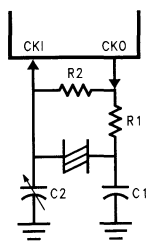
By selecting CKI as a single pin oscillator input, a single pin R/C oscillator circuit can be connected to it. CKO is available as a general purpose input, and/or HALT restart input.

Table B shows the variation in the oscillator frequencies as functions of the component (R and C) values.

EXTERNAL OSCILLATOR

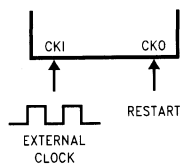
CKI can be driven by an external clock signal. CKO is available as a general purpose input and/or HALT restart control.

Crystal Oscillator



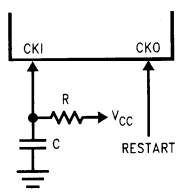
TL/DD/12602-9

External Oscillator



TL/DD/12602-10

R/C Oscillator



TL/DD/12602-11

FIGURE 7. Crystal R/C, and External Oscillator Diagrams

TABLE A. Crystal Oscillator Configuration, T_A = 25°C

R1 (kΩ)	R2 (MΩ)	C1 (pF)	C2 (pF)	CKI Freq (MHz)	Conditions
0	1	30	30-36	10	V_{CC} = 5V
0	1	30	30-36	4	V_{CC} = 5V
0	1	200	100-150	0.455	V_{CC} = 5V

TABLE B. RC Oscillator Configuration, T_A = 25°C

R (kΩ)	C (pF)	CKI Freq (MHz)	Instr. Cycle (μs)	Conditions
3.3	82	2.2 to 2.7	3.7 to 4.6	V_{CC} = 5V
5.6	100	1.1 to 1.3	7.4 to 9.0	V_{CC} = 5V
6.8	100	0.9 to 1.1	8.8 to 10.8	V_{CC} = 5V

Note: 3k ≤ R ≤ 200k

50 pF ≤ C ≤ 200 pF

Current Drain

The total current drain of the chip depends on:

1. Oscillator operation mode—I1
2. Internal switching current—I2
3. Internal leakage current—I3
4. Output source current—I4
5. DC current caused by external input not at V_{CC} or GND—I5
6. Comparator DC supply current when enabled—I6
7. Clock Monitor current when enabled—I7

Thus the total current drain, It, is given as

$$It = I1 + I2 + I3 + I4 + I5 + I6 + I7$$

To reduce the total current drain, each of the above components must be minimum.

The chip will draw more current as the CKI input frequency increases up to the maximum 10 MHz value. Operating with a crystal network will draw more current than an external square-wave. Switching current, governed by the equation below, can be reduced by lowering voltage and frequency. Leakage current can be reduced by lowering voltage and temperature. The other two items can be reduced by carefully designing the end-user's system.

$$I2 = C \times V \times f$$

where C = equivalent capacitance of the chip
V = operating voltage
f = CKI frequency

5

Control Registers

CNTRL Register (Address X'00EE)

The Timer1 (T1) and MICROWIRE/PLUS control register contains the following bits:

SL1 & SL0 Select the MICROWIRE/PLUS clock divide by (00 = 2, 01 = 4, 1x = 8)

IEDG External interrupt edge polarity select (0 = Rising edge, 1 = Falling edge)

MSEL Selects G5 and G4 as MICROWIRE/PLUS signals SK and SO respectively

T1C0 Timer T1 Start/Stop control in timer modes 1 and 2

Timer T1 Underflow Interrupt Pending Flag in timer mode 3

T1C1 Timer T1 mode control bit

T1C2 Timer T1 mode control bit

T1C3 Timer T1 mode control bit

T1C3	T1C2	T1C1	T1C0	MSEL	IEDG	SL1	SL0

Bit 7 Bit 0

PSW Register (Address X'00EF)

The PSW register contains the following select bits:

GIE Global interrupt enable (enables interrupts)

EXEN Enable external interrupt

BUSY MICROWIRE/PLUS busy shifting flag

EXPND External interrupt pending

T1ENA Timer T1 Interrupt Enable for Timer Underflow or T1A Input capture edge

T1PNDA Timer T1 Interrupt Pending Flag (Autoreload RA in mode 1, T1 Underflow in Mode 2, T1A capture edge in mode 3)

C Carry Flag

HC Half Carry Flag

HC	C	T1PNDA	T1ENA	EXPND	BUSY	EXEN	GIE

Bit 7 Bit 0

The Half-Carry bit is also affected by all the instructions that affect the Carry flag. The SC (Set Carry) and RC (Reset Carry) instructions will respectively set or clear both the carry flags. In addition to the SC and RC instructions, ADC, SUBC, RRC and RLC instructions affect the carry and Half Carry flags.

ICNTRL Register (Address X'00E8)

The ICNTRL register contains the following bits:

T1ENB Timer T1 Interrupt Enable for T1B Input capture edge

T1PNDB Timer T1 Interrupt Pending Flag for T1B capture edge

μWEN Enable MICROWIRE/PLUS interrupt

μWPND MICROWIRE/PLUS interrupt pending

T0EN Timer T0 Interrupt Enable (Bit 12 toggle)

T0PND Timer T0 Interrupt pending

LPEN L Port Interrupt Enable (Multi-Input Wakeup/Interrupt)

Bit 7 could be used as a flag

Unused	LPEN	T0PND	T0EN	μWPND	μWEN	T1PNDB	T1ENB

Bit 7 Bit 0

T2CNTRL Register (Address X'00C6)

The T2CNTRL register contains the following bits:

T2ENB Timer T2 Interrupt Enable for T2B Input capture edge

T2PNDB Timer T2 Interrupt Pending Flag for T2B capture edge

T2ENA Timer T2 Interrupt Enable for Timer Underflow or T2A Input capture edge

T2PNDA Timer T2 Interrupt Pending Flag (Autoreload RA in mode 1, T2 Underflow in mode 2, T2A capture edge in mode 3)

T2C0 Timer T2 Start/Stop control in timer modes 1 and 2 Timer T2 Underflow Interrupt Pending Flag in timer mode 3

T2C1 Timer T2 mode control bit

T2C2 Timer T2 mode control bit

T2C3 Timer T2 mode control bit

T2C3	T2C2	T2C1	T2C0	T2PNDA	T2ENA	T2PNDB	T2ENB

Bit 7 Bit 0

T3CNTRL Register (Address X'00B6)

The T3CNTRL register contains the following bits:

T3ENB Timer T3 Interrupt Enable for T3B Input capture edge

T3PNDB Timer T3 Interrupt Pending Flag for T3B pin (T3B capture edge)

T3ENA Timer T3 Interrupt Enable for Timer Underflow or T3A Input capture edge

T3PNDA Timer T3 Interrupt Pending Flag (Autoload RA in mode 1, T3 Underflow in mode 2, T3A capture edge in mode 3)

T3C0 Timer T3 Start/Stop control in timer modes 1 and 2

Timer T3 Underflow Interrupt Pending Flag in timer mode 3

T3C1 Timer T3 mode control bit

T3C2 Timer T3 mode control bit

T3C3 Timer T3 mode control bit

T3C3	T3C2	T3C1	T3C0	T3PNDA	T3ENA	T3PNDB	T3ENB

Bit 7 Bit 0

Timers

The device contains a very versatile set of timers (T0, T1, T2, T3). All timers and associated autoreload/capture registers power up containing random data.

TIMER T0 (IDLE TIMER)

The device supports applications that require maintaining real time and low power with the IDLE mode. This IDLE

Timers (Continued)

mode support is furnished by the IDLE timer T0, which is a 16-bit timer. The Timer T0 runs continuously at the fixed rate of the instruction cycle clock, t_C. The user cannot read or write to the IDLE Timer T0, which is a count down timer.

The Timer T0 supports the following functions:

- Exit out of the Idle Mode (See Idle Mode description)
- WATCHDOG logic (See WATCHDOG description)
- Start up delay out of the HALT mode

The IDLE Timer T0 can generate an interrupt when the thirteenth bit toggles. This toggle is latched into the T0PND pending flag, and will occur every 4 ms at the maximum clock frequency (t_C = 1 μs). A control flag T0EN allows the interrupt from the thirteenth bit of Timer T0 to be enabled or disabled. Setting T0EN will enable the interrupt, while resetting it will disable the interrupt.

TIMER T1, TIMER T2 AND TIMER T3

The device has a set of three powerful timer/counter blocks, T1, T2 and T3. The associated features and functioning of a timer block are described by referring to the timer block Tx. Since the three timer blocks, T1, T2 and T3 are identical, all comments are equally applicable to any of the three timer blocks.

Each timer block consists of a 16-bit timer, Tx, and two supporting 16-bit autoreload/capture registers, RxA and RxB. Each timer block has two pins associated with it, TxA and TxB. The pin TxA supports I/O required by the timer block, while the pin TxB is an input to the timer block. The powerful and flexible timer block allows the device to easily perform all timer functions with minimal software overhead. The timer block has three operating modes: Processor Independent PWM mode, External Event Counter mode, and Input Capture mode.

The control bits TxC3, TxC2, and TxC1 allow selection of the different modes of operation.

Mode 1. Processor Independent PWM Mode

As the name suggests, this mode allows the device to generate a PWM signal with very minimal user intervention. The user only has to define the parameters of the PWM signal (ON time and OFF time). Once begun, the timer block will continuously generate the PWM signal completely independent of the microcontroller. The user software services the timer block only when the PWM parameters require updating.

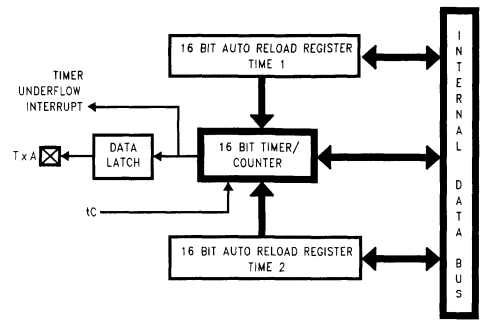
In this mode the timer Tx counts down at a fixed rate of t_C. Upon every underflow the timer is alternately reloaded with the contents of supporting registers, RxA and RxB. The very first underflow of the timer causes the timer to reload from the register RxA. Subsequent underflows cause the timer to be reloaded from the registers alternately beginning with the register RxB.

The Tx Timer control bits, TxC3, TxC2 and TxC1 set up the timer for PWM mode operation.

Figure 8 shows a block diagram of the timer in PWM mode. The underflows can be programmed to toggle the TxA output pin. The underflows can also be programmed to generate interrupts.

Underflows from the timer are alternately latched into two pending flags, TxPNDA and TxPNDB. The user must reset these pending flags under software control. Two control enable flags, TxENA and TxENB, allow the interrupts from the timer underflow to be enabled or disabled. Setting the timer enable flag TxENA will cause an interrupt when a timer underflow causes the RxA register to be reloaded into the timer. Setting the timer enable flag TxENB will cause an interrupt when a timer underflow causes the RxB register to be reloaded into the timer. Resetting the timer enable flags will disable the associated interrupts.

Either or both of the timer underflow interrupts may be enabled. This gives the user the flexibility of interrupting once per PWM period on either the rising or falling edge of the PWM output. Alternatively, the user may choose to interrupt on both edges of the PWM output.



TL/DD/12602-12

FIGURE 8. Timer in PWM Mode

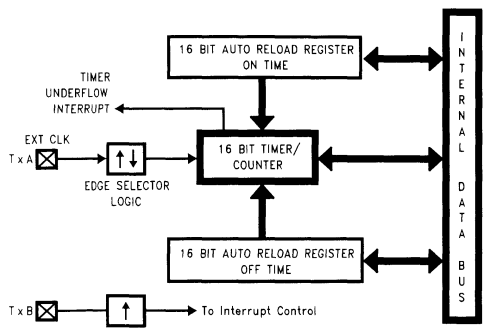
Mode 2. External Event Counter Mode

This mode is quite similar to the processor independent PWM mode described above. The main difference is that the timer, Tx, is clocked by the input signal from the TxA pin. The Tx timer control bits, TxC3, TxC2 and TxC1 allow the timer to be clocked either on a positive or negative edge from the TxA pin. Underflows from the timer are latched into the TxPNDA pending flag. Setting the TxENA control flag will cause an interrupt when the timer underflows.

In this mode the input pin TxB can be used as an independent positive edge sensitive interrupt input if the TxENB control flag is set. The occurrence of a positive edge on the TxB input pin is latched into the TxPNDB flag.

Figure 9 shows a block diagram of the timer in External Event Counter mode.

Note: The PWM output is not available in this mode since the TxA pin is being used as the counter input clock.



TL/DD/12602-13

FIGURE 9. Timer in External Event Counter Mode

Timers (Continued)

Mode 3. Input Capture Mode

The device can precisely measure external frequencies or time external events by placing the timer block, Tx, in the input capture mode.

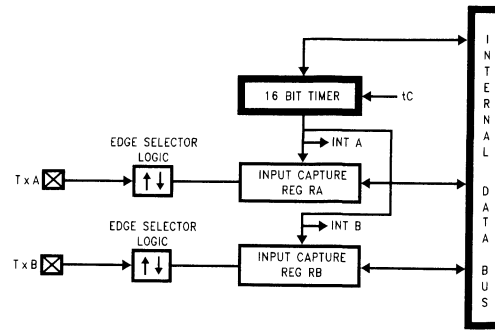
In this mode, the timer Tx is constantly running at the fixed t_C rate. The two registers, RxA and RxB, act as capture registers. Each register acts in conjunction with a pin. The register RxA acts in conjunction with the TxA pin and the register RxB acts in conjunction with the TxB pin.

The timer value gets copied over into the register when a trigger event occurs on its corresponding pin. Control bits, TxC3, TxC2 and TxC1, allow the trigger events to be specified either as a positive or a negative edge. The trigger condition for each input pin can be specified independently.

The trigger conditions can also be programmed to generate interrupts. The occurrence of the specified trigger condition on the TxA and TxB pins will be respectively latched into the pending flags, TxPNDA and TxPNDB. The control flag TxENA allows the interrupt on TxA to be either enabled or disabled. Setting the TxENA flag enables interrupts to be generated when the selected trigger condition occurs on the TxA pin. Similarly, the flag TxENB controls the interrupts from the TxB pin.

Underflows from the timer can also be programmed to generate interrupts. Underflows are latched into the timer TxC0 pending flag (the TxC0 control bit serves as the timer underflow interrupt pending flag in the Input Capture mode). Consequently, the TxC0 control bit should be reset when entering the Input Capture mode. The timer underflow interrupt is enabled with the TxENA control flag. When a TxA interrupt occurs in the Input Capture mode, the user must check both the TxPNDA and TxC0 pending flags in order to determine whether a TxA input capture or a timer underflow (or both) caused the interrupt.

Figure 10 shows a block diagram of the timer in Input Capture mode.



TL/DD/12602–14

FIGURE 10. Timer in Input Capture Mode

TIMER CONTROL FLAGS

The timers T1, T2 and T3 have indentical control structures. The control bits and their functions are summarized below.

TxC0	Timer Start/Stop control in Modes 1 and 2 (Processor Independent PWM and External Event Counter), where 1 = Start, 0 = Stop Timer Underflow Interrupt Pending Flag in Mode 3 (Input Capture)
TxPNDA	Timer Interrupt Pending Flag
TxPNDB	Timer Interrupt Pending Flag
TxENA	Timer Interrupt Enable Flag
TxENB	Timer Interrupt Enable Flag 1 = Timer Interrupt Enabled 0 = Timer Interrupt Disabled
TxC3	Timer mode control
TxC2	Timer mode control
TxC1	Timer mode control

Timers (Continued)

The timer mode control bits (TxC3, TxC2 and TxC1) are detailed below:

TxC3	TxC2	TxC1	Timer Mode	Interrupt A Source	Interrupt B Source	Timer Counts On
0	0	0	MODE 2 (External Event Counter)	Timer Underflow	Pos. TxB Edge	TxA Pos. Edge
0	0	1	MODE 2 (External Event Counter)	Timer Underflow	Pos. TxB Edge	TxA Neg. Edge
1	0	1	MODE 1 (PWM) TxA Toggle	Autoreload RA	Autoreload RB	t_C
1	0	0	MODE 1 (PWM) No TxA Toggle	Autoreload RA	Autoreload RB	t_C
0	1	0	MODE 3 (Capture) Captures: TxA Pos. Edge TxB Pos. Edge	Pos. TxA Edge or Timer Underflow	Pos. TxB Edge	t_C
1	1	0	MODE 3 (Capture) Captures: TxA Pos. Edge TxB Neg. Edge	Pos. TxA Edge or Timer Underflow	Neg. TxB Edge	t_C
0	1	1	MODE 3 (Capture) Captures: TxA Neg. Edge TxB Pos. Edge	Neg. TxA Edge or Timer Underflow	Pos. TxB Edge	t_C
1	1	1	MODE 3 (Capture) Captures: TxA Neg. Edge TxB Neg. Edge	Neg. TxA Edge or Timer Underflow	Neg. TxB Edge	t_C

Power Save Modes

The device offers the user two power save modes of operation: HALT and IDLE. In the HALT mode, all microcontroller activities are stopped. In the IDLE mode, the on-board oscillator circuitry and timer T0 are active but all other microcontroller activities are stopped. In either mode, all on-board RAM, registers, I/O states, and timers (with the exception of T0) are unaltered.

HALT MODE

The device can be placed in the HALT mode by writing a "1" to the HALT flag (G7 data bit). All microcontroller activities, including the clock and timers, are stopped. The WATCHDOG logic on the device is disabled during the HALT mode. However, the clock monitor circuitry if enabled remains active and will cause the WATCHDOG output pin (WDOUT) to go low. If the HALT mode is used and the user does not want to activate the WDOUT pin, the Clock Monitor should be disabled after the device comes out of reset (resetting the Clock Monitor control bit with the first write to the WDSVR register). In the HALT mode, the power requirements of the device are minimal and the applied voltage (V_{CC}) may be decreased to V_r (V_r = 2.0V) without altering the state of the machine.

The device supports three different ways of exiting the HALT mode. The first method of exiting the HALT mode is with the Multi-Input Wakeup feature on the L port. The second method is with a low to high transition on the CKO (G7) pin. This method precludes the use of the crystal clock configuration (since CKO becomes a dedicated output), and so may be used with an RC clock configuration. The third method of exiting the HALT mode is by pulling the $\overline{\text{RESET}}$ pin low.

Since a crystal or ceramic resonator may be selected as the oscillator, the Wakeup signal is not allowed to start the chip running immediately since crystal oscillators and ceramic resonators have a delayed start up time to reach full amplitude and frequency stability. The IDLE timer is used to generate a fixed delay to ensure that the oscillator has indeed stabilized before allowing instruction execution. In this case, upon detecting a valid Wakeup signal, only the oscillator circuitry is enabled. The IDLE timer is loaded with a value of 256 and is clocked with the t_C instruction cycle clock. The t_C clock is derived by dividing the oscillator clock down by a factor of 10. The Schmitt trigger following the CKI inverter on the chip ensures that the IDLE timer is clocked only when the oscillator has a sufficiently large amplitude to meet the Schmitt trigger specifications. This Schmitt trigger is not part of the oscillator closed loop. The startup timeout from the IDLE timer enables the clock signals to be routed to the rest of the chip.

If an RC clock option is being used, the fixed delay is introduced optionally. A control bit, CLKDLY, mapped as configuration bit G7, controls whether the delay is to be introduced or not. The delay is included if CLKDLY is set, and excluded if CLKDLY is reset. The CLKDLY bit is cleared on reset.

Power Save Modes (Continued)

The device has two mask options associated with the HALT mode. The first mask option enables the HALT mode feature, while the second mask option disables the HALT mode. With the HALT mode enable mask option, the device will enter and exit the HALT mode as described above. With the HALT disable mask option, the device cannot be placed in the HALT mode (writing a "1" to the HALT flag will have no effect, the HALT flag will remain "0").

IDLE MODE

The device is placed in the IDLE mode by writing a "1" to the IDLE flag (G6 data bit). In this mode, all activities, except the associated on-board oscillator circuitry, and the IDLE Timer T0, are stopped.

The power supply requirements of the microcontroller in this mode of operation are typically around 30% of normal power requirement of the microcontroller.

As with the HALT mode, the device can be returned to normal operation with a reset, or with a Multi-Input Wakeup from the L Port. Alternately, the microcontroller resumes normal operation from the IDLE mode when the thirteenth bit (representing 4.096 ms at internal clock frequency of 1 MHz, $t_C = 1\ \mu s$) of the IDLE Timer toggles.

This toggle condition of the thirteenth bit of the IDLE Timer T0 is latched into the T0PND pending flag.

The user has the option of being interrupted with a transition on the thirteenth bit of the IDLE Timer T0. The interrupt can be enabled or disabled via the T0EN control bit. Setting the T0EN flag enables the interrupt and vice versa.

The user can enter the IDLE mode with the Timer T0 interrupt enabled. In this case, when the T0PND bit gets set, the device will first execute the Timer T0 interrupt service routine and then return to the instruction following the "Enter Idle Mode" instruction.

Alternatively, the user can enter the IDLE mode with the IDLE Timer T0 interrupt disabled. In this case, the device will resume normal operation with the instruction immediately following the "Enter IDLE Mode" instruction.

Note: It is necessary to program two NOP instructions following both the set HALT mode and set IDLE mode instructions. These NOP instructions are necessary to allow clock resynchronization following the HALT or IDLE modes.

Multi-Input Wakeup

The Multi-Input Wakeup feature is ued to return (wakeup) the device from either the HALT or IDLE modes. Alternately Multi-Input Wakeup/Interrupt feature may also be used to generate up to 8 edge selectable external interrupts.

Figure 11 shows the Multi-Input Wakeup logic.

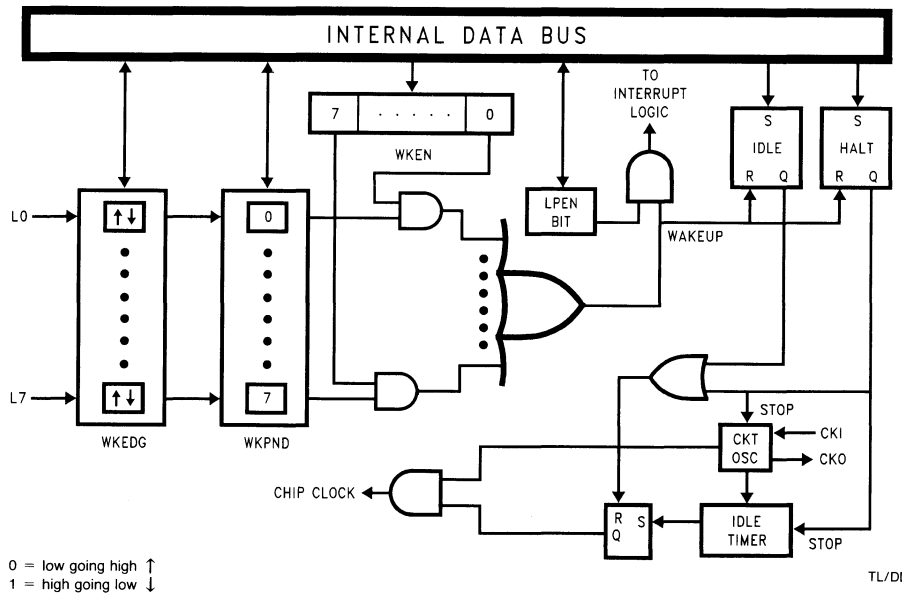


FIGURE 11. Multi-Input Wake Up Logic

Multi-Input Wakeup (Continued)

The Multi-Input Wakeup feature utilizes the L Port. The user selects which particular L port bit (or combination of L Port bits) will cause the device to exit the HALT or IDLE modes. The selection is done through the Reg: WKEN. The Reg: WKEN is an 8-bit read/write register, which contains a control bit for every L port bit. Setting a particular WKEN bit enables a Wakeup from the associated L port pin.

The user can select whether the trigger condition on the selected L Port pin is going to be either a positive edge (low to high transition) or a negative edge (high to low transition). This selection is made via the Reg: WKEDG, which is an 8-bit control register with a bit assigned to each L Port pin. Setting the control bit will select the trigger condition to be a negative edge on that particular L Port pin. Resetting the bit selects the trigger condition to be a positive edge. Changing an edge select entails several steps in order to avoid a pseudo Wakeup condition as a result of the edge change. First, the associated WKEN bit should be reset, followed by the edge select change in WKEDG. Next, the associated WKPND bit should be cleared, followed by the associated WKEN bit being re-enabled.

An example may serve to clarify this procedure. Suppose we wish to change the edge select from positive (low going high) to negative (high going low) for L Port bit 5, where bit 5 has previously been enabled for an input interrupt. The program would be as follows:

```
RBIT  5, WKEN
SBIT  5, WKEDG
RBIT  5, WKPND
SBIT  5, WKEN
```

If the L port bits have been used as outputs and then changed to inputs with Multi-Input Wakeup/Interrupt, a safety procedure should also be followed to avoid inherited pseudo wakeup conditions. After the selected L port bits have been changed from output to input but before the associated WKEN bits are enabled, the associated edge select bits in WKEDG should be set or reset for the desired edge selects, followed by the associated WKPND bits being cleared.

This same procedure should be used following reset, since the L port inputs are left floating as a result of reset.

The occurrence of the selected trigger condition for Multi-Input Wakeup is latched into a pending register called WKPND. The respective bits of the WKPND register will be set on the occurrence of the selected trigger edge on the corresponding Port L pin. The user has the responsibility of clearing these pending flags. Since WKPND is a pending register for the occurrence of selected wakeup conditions, the device will not enter the HALT mode if any Wakeup bit is both enabled and pending. Consequently, the user has the responsibility of clearing the pending flags before attempting to enter the HALT mode.

WKEN, WKPND and WKEDG are all read/write registers, and are cleared at reset.

PORT L INTERRUPTS

Port L provides the user with an additional eight fully selectable, edge sensitive interrupts which are all vectored into the same service subroutine.

The interrupt from Port L shares logic with the wake up circuitry. The register WKEN allows interrupts from Port L to be individually enabled or disabled. The register WKEDG specifies the trigger condition to be either a positive or a negative edge. Finally, the register WKPND latches in the pending trigger conditions.

The GIE (Global Interrupt Enable) bit enables the interrupt function.

A control flag, LPEN, functions as a global interrupt enable for Port L interrupts. Setting the LPEN flag will enable interrupts and vice versa. A separate global pending flag is not needed since the register WKPND is adequate.

Since Port L is also used for waking the device out of the HALT or IDLE modes, the user can elect to exit the HALT or IDLE modes either with or without the interrupt enabled. If he elects to disable the interrupt, then the device will restart execution from the instruction immediately following the instruction that placed the microcontroller in the HALT or IDLE modes. In the other case, the device will first execute the interrupt service routine and then revert to normal operation. (See HALT mode for clock option wakeup information.)

Note: There is always the possibility of an interrupt occurring during an instruction which is attempting to reset the GIE bit or any other interrupt enable bit. If this occurs when a single cycle instruction is being used to reset the interrupt enable bit, the interrupt enable bit will be reset but an interrupt may still occur. This is because interrupt processing is started at the same time as the interrupt bit is being reset. To avoid this scenario, the user should always use a two, three, or four cycle instruction to reset interrupt enable bits.

UART

The device contains a full-duplex software programmable UART. The UART *(Figure 12)* consists of a transmit shift register, a receiver shift register and seven addressable registers, as follows: a transmit buffer register (TBUF), a receiver buffer register (RBUF), a UART control and status register (ENU), a UART receive control and status register (ENUR), a UART interrupt and clock source register (ENUI), a prescaler select register (PSR) and baud (BAUD) register. The ENU register contains flags for transmit and receive functions; this register also determines the length of the data frame (7, 8 or 9 bits), the value of the ninth bit in transmission, and parity selection bits. The ENUR register flags framming, data overrun and parity errors while the UART is receiving.

Other functions of the ENUR register include saving the ninth bit received in the data frame, enabling or disabling the UART's attention mode of operation and providing additional receiver/transmitter status information via RCVG and XMTG bits. The determination of an internal or external clock source is done by the ENUI register, as well as selecting the number of stop bits and enabling or disabling transmit and receive interrupts. A control flag in this register can also select the UART mode of operation: asynchronous or synchronous.

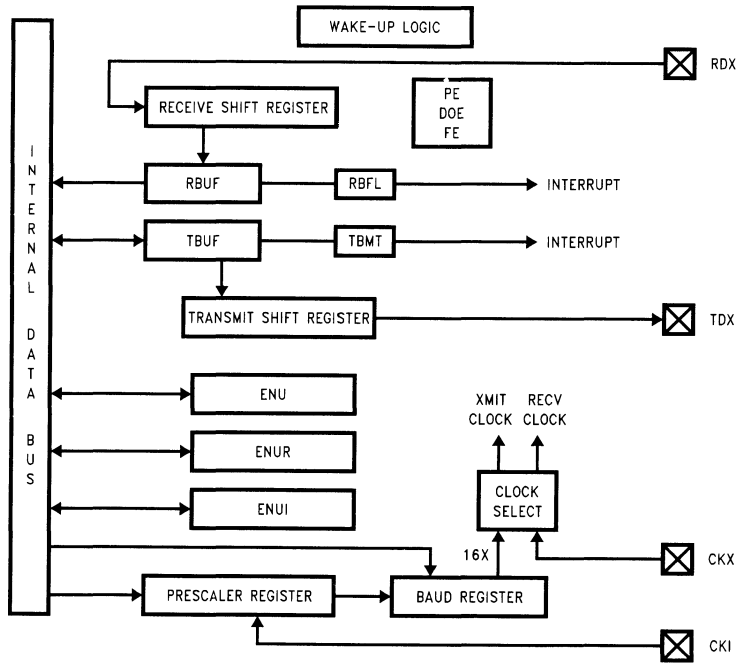


FIGURE 12. UART Block Diagram

UART (Continued)

UART CONTROL AND STATUS REGISTERS

The operation of the UART is programmed through three registers: ENU, ENUR and ENUI. The function of the individual bits in these registers is as follows:

ENU-UART Control and Status Register (Address at 0BA)

PEN	PSEL1	XBIT9/ PSEL0	CHL1	CHL0	ERR	RBFL	TBMT
0RW	0RW	0RW	0RW	0RW	0R	0R	1R

Bit 7 Bit 0

ENUR-UART Receive Control and Status Register (Address at 0BB)

DOE	FE	PE	SPARE	RBIT9	ATTN	XMTG	RCVG
0RD	0RD	0RD	0RW*	0R	0RW	0R	0R

Bit 7 Bit 0

ENUI-UART Interrupt and Clock Source Register (Address at 0BC)

STP2	STP78	ETDX	SSEL	XRCLK	XTCLK	ERI	ETI
0RW	0RW	0RW	0RW	0RW	0RW	0RW	0RW

Bit 7 Bit 0

*Bit is not used.

0 Bit is cleared on reset.

1 Bit is set to one on reset.

R Bit is read-only; it cannot be written by software.

RW Bit is read/write.

D Bit is cleared on read; when read by software as a one, it is cleared automatically. Writing to the bit does not affect its state.

DESCRIPTION OF UART REGISTER BITS

ENU—UART CONTROL AND STATUS REGISTER

TBMT: This bit is set when the UART transfers a byte of data from the TBUF register into the TSFT register for transmission. It is automatically reset when software writes into the TBUF register.

RBFL: This bit is set when the UART has received a complete character and has copied it into the RBUF register. It is automatically reset when software reads the character from RBUF.

ERR: This bit is a global UART error flag which gets set if any or a combination of the errors (DOE, FE, PE) occur.

CHL1, CHL0: These bits select the character frame format. Parity is not included and is generated/verified by hardware.

CHL1 = 0, CHL0 = 0 The frame contains eight data bits.
CHL1 = 0, CHL0 = 1 The frame contains seven data bits.
CHL1 = 1, CHL0 = 0 The frame contains nine data bits.
CHL1 = 1, CHL0 = 1 Loopback Mode selected. Transmitter output internally looped back to receiver input. Nine bit framing format is used.

XBIT9/PSEL0: Programs the ninth bit for transmission when the UART is operating with nine data bits per frame. For seven or eight data bits per frame, this bit in conjunction with PSEL1 selects parity.

PSEL1, PSEL0: Parity select bits.

PSEL1 = 0, PSEL0 = 0 Odd Parity (if Parity enabled)
PSEL1 = 0, PSEL0 = 1 Even Parity (if Parity enabled)
PSEL1 = 1, PSEL0 = 0 Mark(1) (if Parity enabled)
PSEL1 = 1, PSEL0 = 1 Space(0) (if Parity enabled)

PEN: This bit enables/disables Parity (7- and 8-bit modes only).

PEN = 0 Parity disabled.
PEN = 1 Parity enabled.

ENUR—UART RECEIVE CONTROL AND STATUS REGISTER

RCVG: This bit is set high whenever a framing error occurs and goes low when RDX goes high.

XMTG: This bit is set to indicate that the UART is transmitting. It gets reset at the end of the last frame (end of last Stop bit).

ATTN: ATTENTION Mode is enabled while this bit is set. This bit is cleared automatically on receiving a character with data bit nine set.

RBIT9: Contains the ninth data bit received when the UART is operating with nine data bits per frame.

SPARE: Reserved for future use.

PE: Flags a Parity Error.

PE = 0 Indicates no Parity Error has been detected since the last time the ENUR register was read.
PE = 1 Indicates the occurrence of a Parity Error.

FE: Flags a Framing Error.

FE = 0 Indicates no Framing Error has been detected since the last time the ENUR register was read.
FE = 1 Indicates the occurrence of a Framing Error.

DOE: Flags a Data Overrun Error.

DOE = 0 Indicates no Data Overrun Error has been detected since the last time the ENUR register was read.
DOE = 1 Indicates the occurrence of a Data Overrun Error.

ENUI—UART INTERRUPT AND CLOCK SOURCE REGISTER

ETI: This bit enables/disables interrupt from the transmitter section.

ETI = 0 Interrupt from the transmitter is disabled.
ETI = 1 Interrupt from the transmitter is enabled.

ERI: This bit enables/disables interrupt from the receiver section.

ERI = 0 Interrupt from the receiver is disabled.
ERI = 1 Interrupt from the receiver is enabled.

XTCLK: This bit selects the clock source for the transmitter-section.

XTCLK = 0 The clock source is selected through the PSR and BAUD registers.
XTCLK = 1 Signal on CKX (L1) pin is used as the clock.

XRCLK: This bit selects the clock source for the receiver section.

XRCLK = 0 The clock source is selected through the PSR and BAUD registers.
XRCLK = 1 Signal on CKX (L1) pin is used as the clock.

SSEL: UART mode select.

SSEL = 0 Asynchronous Mode.
SSEL = 1 Synchronous Mode.

UART (Continued)

ETDX: TDX (UART Transmit Pin) is the alternate function assigned to Port L pin L2; it is selected by setting ETDX bit. To simulate line break generation, software should reset ETDX bit and output logic zero to TDX pin through Port L data and configuration registers.

STP78: This bit is set to program the last Stop bit to be 7/8th of a bit in length.

STP2: This bit programs the number of Stop bits to be transmitted.

STP2 = 0 One Stop bit transmitted.
STP2 = 1 Two Stop bits transmitted.

Associated I/O Pins

Data is transmitted on the TDX pin and received on the RDX pin. TDX is the alternate function assigned to Port L pin L2; it is selected by setting ETDX (in the ENUI register) to one. RDX is an inherent function of Port L pin L3, requiring no setup.

The baud rate clock for the UART can be generated on-chip, or can be taken from an external source. Port L pin L1 (CKX) is the external clock I/O pin. The CKX pin can be either an input or an output, as determined by Port L Configuration and Data registers (Bit 1). As an input, it accepts a clock signal which may be selected to drive the transmitter and/or receiver. As an output, it presents the internal Baud Rate Generator output.

UART Operation

The UART has two modes of operation: asynchronous mode and synchronous mode.

ASYNCHRONOUS MODE

This mode is selected by resetting the SSEL (in the ENUI register) bit to zero. The input frequency to the UART is 16 times the baud rate.

The TSFT and TBUF registers double-buffer data for transmission. While TSFT is shifting out the current character on the TDX pin, the TBUF register may be loaded by software with the next byte to be transmitted. When TSFT finishes transmitting the current character the contents of TBUF are transferred to the TSFT register and the Transmit Buffer Empty Flag (TBMT in the ENU register) is set. The TBMT flag is automatically reset by the UART when software loads a new character into the TBUF register. There is also the XMTG bit which is set to indicate that the UART is transmitting. This bit gets reset at the end of the last frame (end of last Stop bit). TBUF is a read/write register.

The RSFT and RBUF registers double-buffer data being received. The UART receiver continually monitors the signal on the RDX pin for a low level to detect the beginning of a Start bit. Upon sensing this low level, it waits for half a bit time and samples again. If the RDX pin is still low, the receiver considers this to be a valid Start bit, and the remaining bits in the character frame are each sampled a single time, at the mid-bit position. Serial data input on the RDX pin is shifted into the RSFT register. Upon receiving the complete character, the contents of the RSFT register are copied into the RBUF register and the Received Buffer Full Flag (RBFL) is set. RBFL is automatically reset when software reads the character from the RBUF register. RBUF is a read only register. There is also the RCVG bit which is set high when a framing error occurs and goes low once RDX goes high. TBMT, XMTG, RBFL and RCVG are read only bits.

SYNCHRONOUS MODE

In this mode data is transferred synchronously with the clock. Data is transmitted on the rising edge and received on the falling edge of the synchronous clock.

This mode is selected by setting SSEL bit in the ENUI register. The input frequency to the UART is the same as the baud rate.

When an external clock input is selected at the CKX pin, data transmit and receive are performed synchronously with this clock through TDX/RDX pins.

If data transmit and receive are selected with the CKX pin as clock output, the device generates the synchronous clock output at the CKX pin. The internal baud rate generator is used to produce the synchronous clock. Data transmit and receive are performed synchronously with this clock.

FRAMING FORMATS

The UART supports several serial framing formats *(Figure 13).* The format is selected using control bits in the ENU, ENUR and ENUI registers.

The first format (1, 1a, 1b, 1c) for data transmission (CHL0 = 1, CHL1 = 0) consists of Start bit, seven Data bits (excluding parity) and 7/8, one or two Stop bits. In applications using parity, the parity bit is generated and verified by hardware.

The second format (CHL0 = 0, CHL1 = 0) consists of one Start bit, eight Data bits (excluding parity) and 7/8, one or two Stop bits. Parity bit is generated and verified by hardware.

The third format for transmission (CHL0 = 0, CHL1 = 1) consists of one Start bit, nine Data bits and 7/8, one or two Stop bits. This format also supports the UART "ATTENTION" feature. When operating in this format, all eight bits of TBUF and RBUF are used for data. The ninth data bit is transmitted and received using two bits in the ENU and ENUR registers, called XBIT9 and RBIT9. RBIT9 is a read only bit. Parity is not generated or verified in this mode.

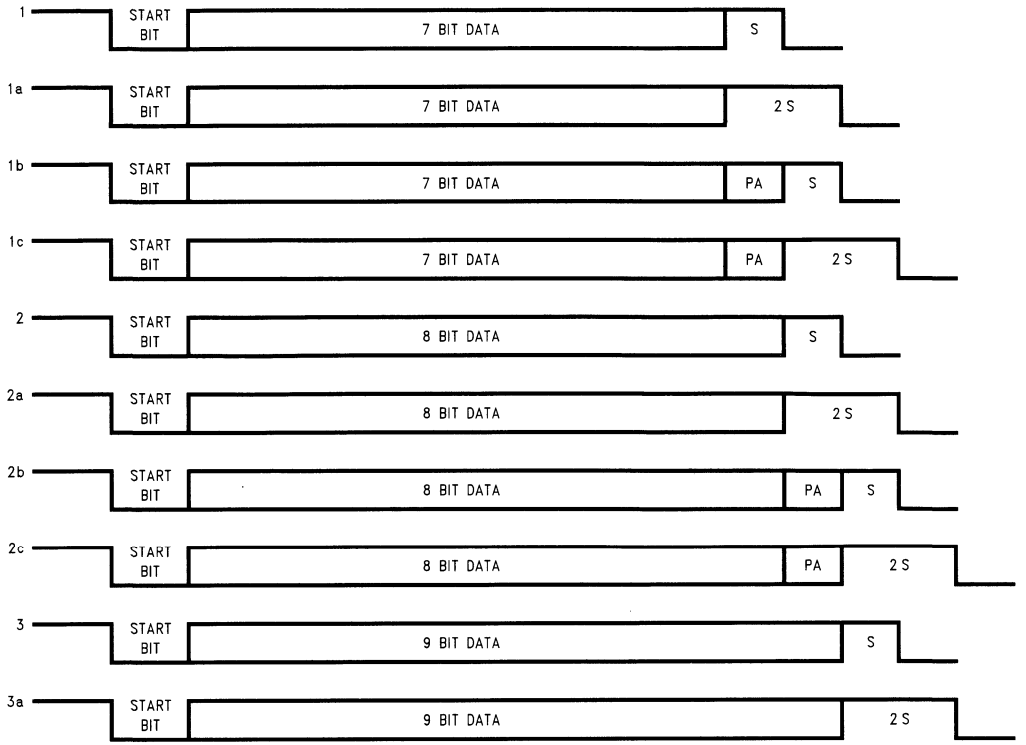
For any of the above framing formats, the last Stop bit can be programmed to be 7/8th of a bit in length. If two Stop bits are selected and the 7/8th bit is set (selected), the second Stop bit will be 7/8th of a bit in length.

The parity is enabled/disabled by PEN bit located in the ENU register. Parity is selected for 7- and 8-bit modes only. If parity is enabled (PEN = 1), the parity selection is then performed by PSEL0 and PSEL1 bits located in the ENU register.

Note that the XBIT9/PSEL0 bit located in the ENU register serves two mutually exclusive functions. This bit programs the ninth bit for transmission when the UART is operating with nine data bits per frame. There is no parity selection in this framing format. For other framing formats XBIT9 is not needed and the bit is PSEL0 used in conjunction with PSEL1 to select parity.

The frame formats for the receiver differ from the transmitter in the number of Stop bits required. The receiver only requires one Stop bit in a frame, regardless of the setting of the Stop bit selection bits in the control register. Note that an implicit assumption is made for full duplex UART operation that the framing formats are the same for the transmitter and receiver.

UART Operation (Continued)



TL/DD/12602–17

FIGURE 13. Framing Formats

UART INTERRUPTS

The UART is capable of generating interrupts. Interrupts are generated on Receive Buffer Full and Transmit Buffer Empty. Both interrupts have individual interrupt vectors. Two bytes of program memory space are reserved for each interrupt vector. The two vectors are located at addresses 0xEC to 0xEF Hex in the program memory space. The interrupts can be individually enabled or disabled using Enable Transmit Interrupt (ETI) and Enable Receive Interrupt (ERI) bits in the ENUI register.

The interrupt from the Transmitter is set pending, and remains pending, as long as both the TBMT and ETI bits are set. To remove this interrupt, software must either clear the ETI bit or write to the TBUF register (thus clearing the TBMT bit).

The interrupt from the receiver is set pending, and remains pending, as long as both the RBFL and ERI bits are set. To remove this interrupt, software must either clear the ERI bit or read from the RBUF register (thus clearing the RBFL bit).

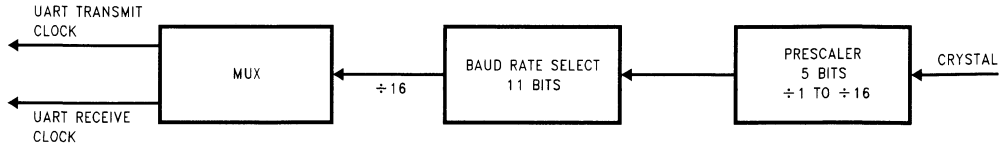
Baud Clock Generation

The clock inputs to the transmitter and receiver sections of the UART can be individually selected to come either from an external source at the CKX pin (port L, pin L1) or from a source selected in the PSR and BAUD registers. Internally, the basic baud clock is created from the oscillator frequency through a two-stage divider chain consisting of a 1–16 (increments of 0.5) prescaler and an 11-bit binary counter. *(Figure 14)* The divide factors are specified through two read/write registers shown in *Figure 15*. Note that the 11-bit Baud Rate Divisor spills over into the Prescaler Select Register (PSR). PSR is cleared upon reset.

As shown in Table I, a Prescaler Factor of 0 corresponds to NO CLOCK. NO CLOCK condition is the UART power down mode where the UART clock is turned off for power saving purpose. The user must also turn the UART clock off when a different baud rate is chosen.

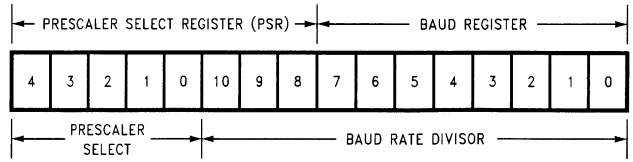
The correspondences between the 5-bit Prescaler Select and Prescaler factors are shown in Table I. Therer are many ways to calculate the two divisor factors, but one particularly effective method would be to achieve a 1.8432 MHz frequency coming out of the first stage. The 1.8432 MHz prescaler output is then used to drive the software programmable baud rate counter to create a 16x clock for the following baud rates: 110, 134.5, 150, 300, 600, 1200, 1800, 2400, 3600, 4800, 7200, 9600, 19200 and 38400 (Table II). Other baud rates may be created by using appropriate divisors. The 16x clock is then divided by 16 to provide the rate for the serial shift registers of the transmitter and receiver.

Baud Clock Generation (Continued)



TL/DD/12602–18

FIGURE 14. UART BAUD Clock Generation



TL/DD/12602–19

FIGURE 15. UART BAUD Clock Divisor Registers

TABLE I. Prescaler Factors

Prescaler Select	Prescaler Factor
00000	NO CLOCK
00001	1
00010	1.5
00011	2
00100	2.5
00101	3
00110	3.5
00111	4
01000	4.5
01001	5
01010	5.5
01011	6
01100	6.5
01101	7
01110	7.5
01111	8
10000	8.5
10001	9
10010	9.5
10011	10
10100	10.5
10101	11
10110	11.5
10111	12
11000	12.5
11001	13
11010	13.5
11011	14
11100	14.5
11101	15
11110	15.5
11111	16

TABLE II. Baud Rate Divisors (1.8432 MHz Prescaler Output)

Baud Rate	Baud Rate Divisor − 1 (N-1)
110 (110.03)	1046
134.5 (134.58)	855
150	767
300	383
600	191
1200	95
1800	63
2400	47
3600	31
4800	23
7200	15
9600	11
19200	5
38400	2

The entries in Table II assume a prescaler output of 1.8432 MHz. In the asynchronous mode the baud rate could be as high as 625k.

As an example, considering the Asynchronous Mode and a CKI clock of 4.608 MHz, the prescaler factor selected is:

$$4.608/1.8432 = 2.5$$

The 2.5 entry is available in Table I. The 1.8432 MHz prescaler output is then used with proper Baud Rate Divisor (Table II) to obtain different baud rates. For a baud rate of 19200 e.g., the entry in Table II is 5.

$N - 1 = 5$ (N − 1 is the value from Table II)

$N = 6$ (N is the Baud Rate Divisor)

$$\text{Baud Rate} = 1.8432\ \text{MHz}/(16 \times 6) = 19200$$

The divide by 16 is performed because in the asynchronous mode, the input frequency to the UART is 16 times the baud rate. The equation to calculate baud rates is given below.

The actual Baud Rate may be found from:

$$BR = Fc/(16 \times N \times P)$$

Baud Clock Generation (Continued)

Where:

BR is the Baud Rate

Fc is the CKI frequency

N is the Baud Rate Divisor (Table II).

P is the Prescaler Divide Factor selected by the value in the Prescaler Select Register (Table I)

Note: In the Synchronous Mode, the divisor 16 is replaced by two.

Example:

Asynchronous Mode:

Crystal Frequency = 5 MHz

Desired baud rate = 9600

Using the above equation N × P can be calculated first.

$$N \times P = (5 \times 10^6)/(16 \times 9600) = 32.552$$

Now 32.552 is divided by each Prescaler Factor (Table II) to obtain a value closest to an integer. This factor happens to be 6.5 (P = 6.5).

$$N = 32.552/6.5 = 5.008 \ (N = 5)$$

The programmed value (from Table II) should be 4 (N − 1).

Using the above values calculated for N and P:

$$BR = (5 \times 10^6)/(16 \times 5 \times 6.5) = 9615.384$$

$$\% \text{ error} = (9615.385 - 9600)/9600 = 0.16$$

Effect of HALT/IDLE

The UART logic is reinitialized when either the HALT or IDLE modes are entered. This reinitialization sets the TBMT flag and resets all read only bits in the UART control and status registers. Read/Write bits remain unchanged. The Transmit Buffer (TBUF) is not affected, but the Transmit Shift register (TSFT) bits are set to one. The receiver registers RBUF and RSFT are not affected.

The device will exit from the HALT/IDLE modes when the Start bit of a character is detected at the RDX (L3) pin. This feature is obtained by using the Multi-Input Wakeup scheme provided on the device.

Before entering the HALT or IDLE modes the user program must select the Wakeup source to be on the RDX pin. This selection is done by setting bit 3 of WKEN (Wakeup Enable) register. The Wakeup trigger condition is then selected to be high to low transition. This is done via the WKEDG register (Bit 3 is one.)

If the device is halted and crystal oscillator is used, the Wakeup signal will not start the chip running immediately because of the finite start up time requirement of the crystal oscillator. The idle timer (T0) generates a fixed (256 t_C) delay to ensure that the oscillator has indeed stabilized before allowing the device to execute code. The user has to consider this delay when data transfer is expected immediately after exiting the HALT mode.

Diagnostic

Bits CHARL0 and CHARL1 in the ENU register provide a loopback feature for diagnostic testing of the UART. When these bits are set to one, the following occur: The receiver input pin (RDX) is internally connected to the transmitter output pin (TDX); the output of the Transmitter Shift Register is "looped back" into the Receive Shift Register input. In this mode, data that is transmitted is immediately received. This feature allows the processor to verify the transmit and receive data paths of the UART.

Note that the framing format for this mode is the nine bit format; one Start bit, nine data bits, and 7/8, one or two Stop bits. Parity is not generated or verified in this mode.

Attention Mode

The UART Receiver section supports an alternate mode of operation, referred to as ATTENTION Mode. This mode of operation is selected by the ATTN bit in the ENUR register. The data format for transmission must also be selected as having nine Data bits and either 7/8, one or two Stop bits.

The ATTENTION mode of operation is intended for use in networking the device with other processors. Typically in such environments the messages consists of device addresses, indicating which of several destinations should receive them, and the actual data. This Mode supports a scheme in which addresses are flagged by having the ninth bit of the data field set to a 1. If the ninth bit is reset to a zero the byte is a Data byte.

While in ATTENTION mode, the UART monitors the communication flow, but ignores all characters until an address character is received. Upon receiving an address character, the UART signals that the character is ready by setting the RBFL flag, which in turn interrupts the processor if UART Receiver interrupts are enabled. The ATTN bit is also cleared automatically at this point, so that data characters as well as address characters are recognized. Software examines the contents of the RBUF and responds by deciding either to accept the subsequent data stream (by leaving the ATTN bit reset) or to wait until the next address character is seen (by setting the ATTN bit again).

Operation of the UART Transmitter is not affected by selection of this Mode. The value of the ninth bit to be transmitted is programmed by setting XBIT9 appropriately. The value of the ninth bit received is obtained by reading RBIT9. Since this bit is located in ENUR register where the error flags reside, a bit operation on it will reset the error flags.

Comparators

The device contains two differential comparators, each with a pair of inputs (positive and negative) and an output. Ports I1–I3 and I4–I6 are used for the comparators. The following is the Port I assignment:

- I1 Comparator1 negative input
- I2 Comparator1 positive input
- I3 Comparator1 output
- I4 Comparator2 negative input
- I5 Comparator2 positive input
- I6 Comparator2 output

A Comparator Select Register (CMPSL) is used to enable the comparators, read the outputs of the comparators internally, and enable the outputs of the comparators to the pins. Two control bits (enable and output enable) and one result bit are associated with each comparator. The comparator result bits (CMP1RD and CMP2RD) are read only bits which will read as zero if the associated comparator is not enabled. The Comparator Select Register is cleared with

Comparators (Continued)

reset, resulting in the comparators being disabled. The comparators should also be disabled before entering either the HALT or IDLE modes in order to save power. The configuration of the CMPSL register is as follows:

CMPSL REGISTER (ADDRESS X'00B7)

The CMPSL register contains the following bits:

CMP1EN Enable comparator 1

CMP1RD Comparator 1 result (this is a read only bit, which will read as 0 if the comparator is not enabled)

CMP10E Selects pin I3 as comparator 1 output provided that CMPIEN is set to enable the comparator

CMP2EN Enable comparator 2

CMP2RD Comparator 2 result (this is a read only bit, which will read as 0 if the comparator is not enabled)

CMP20E Selects pin I6 as comparator 2 output provided that CMP2EN is set to enable the comparator

Unused	CMP20E	CMP2RD	CMP2EN	CMP10E	CMP1RD	CMP1EN	Unused

Bit 7 Bit 0

Note that the two unused bits of CMPSL may be used as software flags.

Comparator outputs have the same spec as Ports L and G except that the rise and fall times are symmetrical.

Multiply/Divide

This device contains a multiply/divide block. This block supports a 1 byte x 2 bytes (3 bytes result) multiply or a 3 bytes/2 bytes (2 bytes result) divide operation. The multiply or divide operation is executed by setting control bits located in the multiply/divide control register. The multiply or divide operands must be placed into the appropriate memory mapped locations before the operation is initiated.

CONTROL REGISTER BITS

The Multiply/Divide control register (MDCR) is located at address xx9D. It has the following bit assignments:

MULT Start Multiplication Operation (1 = start)

DIV Start Division Operation (1 = start)

DIVOVF Division Overflow (if the result of a division is greater than 16 bits or the user attempted to divide by zero; 1 = error)

Rsvd	Rsvd	Rsvd	Rsvd	Rsvd	DIV OVF	DIV	MULT

Bit 7 Bit 0

After the appropriate MDR registers are loaded, the MULT and DIV start bits are set by the user to start a multiply or divide operation. The division operation has priority, if both bits are set simultaneously. The MULT and DIV bits are BOTH automatically cleared by hardware at the end of a divide or multiply operation. Each division operation causes the DIVOVF flag to be set/reset as appropriate. The DIVOVF flag is cleared following a multiplication operation. DIVOVF is a read-only bit. The MULT and DIV bits are read/writable. Bits 3–7 in MDCR should not be used, as the MULT and DIV operations will change their values.

MULTIPLY/DIVIDE OPERATION

For the multiply operation, the multiplicand is placed at addresses xx9B and xx9C. The multiplier is placed at address xx99. For the divide operation, the dividend is placed at addresses xx98 to xx9A and the divisor is placed at addresses xx9B to xx9C. In both operations, all operands are interpreted as unsigned values. The divide or multiply operation is started by setting the appropriate MDCR bit. If both the MULT and DIV bits are set, the microcontroller performs a divide operation. (The user is not required to read or clear the DIVOVF error bit prior to beginning a new multiply/divide operation. This bit is ignored during subsequent operations. However, the next divide operation will overwrite the error flag as appropriate, and the next multiply operation will clear it.)

The multiply operation requires 1 instruction cycle to complete. The divide operation requires 2 instruction cycles to complete. A divide by zero or a division which produces an overflow requires only 1 instruction cycle to execute. The MDR1 through MDR5 registers and the MDCR register can not be read from or written to during a multiply or divide operation. Any attempt to write in to these registers will be ignored. Any attempt to read these registers will return undefined data.

The result of a multiply is placed in addresses xx99–xx9B. The result of a divide is placed in a ddresses xx98–xx99. If a division by zero is attempted or if the resulting quotient of a divide operation is more than 16 bits long, then the DIVOVF bit is set in the multiply/divide control register. The dividend and the divisor are left unchanged. The divide operation always causes the DIVOVF flag to be set or reset as appropriate. The DIVOVF flag is cleared following a multiply operation.

RESET STATE

A reset signal applied to the device during normal operation has the following affects:

MDCR is cleared, and any operation in progress is stopped.

MDR1 through MDR5 are undefined.

TABLE III. Multiply/Divide Registers

Register Name (Address)	Multiplication Assignment		Division Assignment	
	Before Operation	After Operation	Before Operation	After Operation
MDR1 (xx98)	Unused	Unchanged	Low Byte of Dividend	Low Byte of Result
MDR2 (xx99)	Multiplier	Low Byte of Result	Middle Byte of Dividend	High Byte of Result
MDR3 (xx9A)		Middle Byte of Result	High Byte of Dividend	Undefined
MDR4 (xx9B)	Low Byte of Multiplicand	High Byte of Result	Low Byte of Divisor	Low Byte of Divisor
MDR5 (xx9C)	High Byte of Multiplicand	Unchanged	High Byte of Divisor	High Byte of Divisor

Interrupts

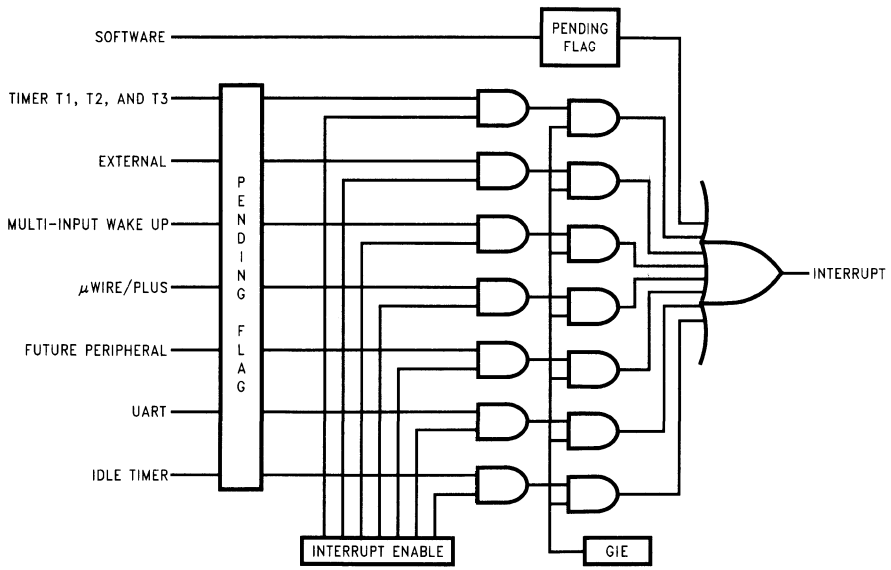
The device supports a vectored interrupt scheme. It supports a total of fourteen interrupt sources. The following table lists all the possible interrupt sources, their arbitration ranking and the memory locations reserved for the interrupt vector for each source.

Two bytes of program memory space are reserved for each interrupt source. All interrupt sources except the software interrupt are maskable. Each of the maskable interrupts have an Enable bit and a Pending bit. A maskable interrupt is active if its associated enable and pending bits are set. If GIE = 1 and an interrupt is active, then the processor will be interrupted as soon as it is ready to start executing an instruction except if the above conditions happen during the Software Trap service routine. This exception is described in the Software Trap sub-section.

The interruption process is accomplished with the INTR instruction (opcode 00), which is jammed inside the Instruction Register and replaces the opcode about to be executed. The following steps are performed for every interrupt:

1. The GIE (Global Interrupt Enable) bit is reset.
2. The address of the instruction about to be executed is pushed into the stack.
3. The PC (Program Counter) branches to address 00FF. This procedure takes 7 t_C cycles to execute.

At this time, since GIE = 0, other maskable interrupts are disabled. The user is now free to do whatever context switching is required by saving the context of the machine in the stack with PUSH instructions. The user would then program a VIS (Vector Interrupt Select) instruction in order to branch to the interrupt service routine of the highest priority interrupt enabled and pending at the time of the VIS. Note that this is not necessarily the interrupt that caused the branch to address location 00FF Hex prior to the context switching.



TL/DD/12602-20

FIGURE 16. Interrupt Block Diagram

Interrupts (Continued)

Arbitration Ranking	Source	Description	Vector Address Hi-Low Byte
(1) Highest	Software	INTR Instruction	0yFE–0yFF
(2)	Reserved		0yFC–0yFD
(3)	External	Pin G0 Edge	0yFA–0yFB
(4)	Timer T0	Underflow	0yF8–0yF9
(5)	Timer T1	T1A/Underflow	0yF6–0yF7
(6)	Timer T1	T1B	0yF4–0yF5
(7)	MICROWIRE/PLUS	BUSY Goes Low	0yF2–0yF3
(8)	Reserved		0yF0–0yF1
(9)	UART	Receive	0yEE–0yEF
(10)	UART	Transmit	0yEC–0yED
(11)	Timer T2	T2A/Underflow	0yEA–0yEB
(12)	Timer T2	T2B	0yE8–0yE9
(13)	Timer T3	T3A/Underflow	0yE6–0yE7
(14)	Timer T3	T3B	0yE4–0yE5
(15)	Port L/Wakeup	Port L Edge	0yE2–0yE3
(16) Lowest	Default VIS	Reserved	0yE0–0yE1

*y is a variable which represents the VIS block. VIS and the vector table must be located in the same 256-byte block except if VIS is located at the last address of a block. In this case, the table must be in the next block.

Thus, if an interrupt with a higher rank than the one which caused the interruption becomes active before the decision of which interrupt to service is made by the VIS, then the interrupt with the higher rank will override any lower ones and will be acknowledged. The lower priority interrupt(s) are still pending, however, and will cause another interrupt immediately following the completion of the interrupt service routine associated with the higher priority interrupt just serviced. This lower priority interrupt will occur immediately following the RETI (Return from Interrupt) instruction at the end of the interrupt service routine just completed.

Inside the interrupt service routine, the associated pending bit has to be cleared by software. The RETI (Return from Interrupt) instruction at the end of the interrupt service routine will set the GIE (Global Interrupt Enable) bit, allowing the processor to be interrupted again if another interrupt is active and pending.

The VIS instruction looks at all the active interrupts at the time it is executed and performs an indirect jump to the beginning of the service routine of the one with the highest rank.

The addresses of the different interrupt service routines, called vectors, are chosen by the user and stored in ROM in a table starting at 01E0 (assuming that VIS is located between 00FF and 01DF). The vectors are 15-bit wide and therefore occupy 2 ROM locations.

VIS and the vector table must be located in the same 256-byte block (0y00 to 0yFF) except if VIS is located at the last address of a block. In this case, the table must be in the next block. The vector table cannot be inserted in the first 256-byte block (y $\neq$ 0).

The vector of the maskable interrupt with the lowest rank is located at 0yE0 (Hi-Order byte) and 0yE1 (Lo-Order byte) and so forth in increasing rank number. The vector of the maskable interrupt with the highest rank is located at 0yFA (Hi-Order byte) and 0yFB (Lo-Order byte).

The Software Trap has the highest rank and its vector is located at 0yFE and 0yFF.

If, by accident, a VIS gets executed and no interrupt is active, then the PC (Program Counter) will branch to a vector located at 0yE0–0yE1.

WARNING

A Default VIS interrupt handler routine must be present. As a minimum, this handler should confirm that the GIE bit is cleared (this indicates that the interrupt sequence has been taken), take care of any required housekeeping, restore context and return. Some sort of Warm Restart procedure should be implemented. These events can occur without any error on the part of the system designer or programmer.

Note: There is always the possibility of an interrupt occurring during an instruction which is attempting to reset the GIE bit or any other interrupt enable bit. If this occurs when a single cycle instruction is being used to reset the interrupt enable bit, the interrupt enable bit will be reset but an interrupt may still occur. This is because interrupt processing is started at the same time as the interrupt bit is being reset. To avoid this scenario, the user should always use a two, three, or four cycle instruction to reset interrupt enable bits.

Figure 16 shows the Interrupt block diagram.

Interrupts (Continued)

SOFTWARE TRAP

The Software Trap (ST) is a special kind of non-maskable interrupt which occurs when the INTR instruction (used to acknowledge interrupts) is fetched from ROM and placed inside the instruction register. This may happen when the PC is pointing beyond the available ROM address space or when the stack is over-popped.

When an ST occurs, the user can re-initialize the stack pointer and do a recovery procedure (similar to reset, but not necessarily containing all of the same initialization procedures) before restarting.

The occurrence of an ST is latched into the ST pending bit. The GIE bit is not affected and the ST pending bit **(not accessible by the user)** is used to inhibit other interrupts and to direct the program to the ST service routine with the VIS instruction. The RPND instruction is used to clear the software interrupt pending bit. This pending bit is also cleared on reset.

The ST has the highest rank among all interrupts.

Nothing (except another ST) can interrupt an ST being serviced.

WATCHDOG

The device contains a WATCHDOG and clock monitor. The WATCHDOG is designed to detect the user program getting stuck in infinite loops resulting in loss of program control or "runaway" programs. The Clock Monitor is used to detect the absence of a clock or a very slow clock below a specified rate on the CKI pin.

The WATCHDOG consists of two independent logic blocks: WD UPPER and WD LOWER. WD UPPER establishes the upper limit on the service window and WD LOWER defines the lower limit of the service window.

Servicing the WATCHDOG consists of writing a specific value to a WATCHDOG Service Register named WDSVR which is memory mapped in the RAM. This value is composed of three fields, consisting of a 2-bit Window Select, a 5-bit Key Data field, and the 1-bit Clock Monitor Select field. Table III shows the WDSVR register.

The lower limit of the service window is fixed at 2048 instruction cycles. Bits 7 and 6 of the WDSVR register allow the user to pick an upper limit of the service window.

Table V shows the four possible combinations of lower and upper limits for the WATCHDOG service window. This flexibility in choosing the WATCHDOG service window prevents any undue burden on the user software.

Bits 5, 4, 3, 2 and 1 of the WDSVR register represent the 5-bit Key Data field. The key data is fixed at 01100. Bit 0 of the WDSVR Register is the Clock Monitor Select bit.

TABLE IV. WATCHDOG Service Register (WDSVR)

Window Select		Key Data					Clock Monitor
X	X	0	1	1	0	0	Y
7	6	5	4	3	2	1	0

TABLE V. WATCHDOG Service Window Select

WDSVR Bit 7	WDSVR Bit 6	Service Window (Lower-Upper Limits)
0	0	2k–8k t_C Cycles
0	1	2k–16k t_C Cycles
1	0	2k–32k t_C Cycles
1	1	2k–64k t_C Cycles

Clock Monitor

The Clock Monitor aboard the device can be selected or deselected under program control. The Clock Monitor is guaranteed not to reject the clock if the instruction cycle clock ($1/t_C$) is greater or equal to 10 kHz. This equates to a clock input rate on CKI of greater or equal to 100 kHz.

WATCHDOG Operation

The WATCHDOG and Clock Monitor are disabled during reset. The device comes out of reset with the WATCHDOG armed, the WATCHDOG Window Select bits (bits 6, 7 of the WDSVR Register) set, and the Clock Monitor bit (bit 0 of the WDSVR Register) enabled. Thus, a Clock Monitor error will occur after coming out of reset, if the instruction cycle clock frequency has not reached a minimum specified value, including the case where the oscillator fails to start.

The WDSVR register can be written to only once after reset and the key data (bits 5 through 1 of the WDSVR Register) must match to be a valid write. This write to the WDSVR register involves two irrevocable choices: (i) the selection of the WATCHDOG service window (ii) enabling or disabling of the Clock Monitor. Hence, the first write to WDSVR Register involves selecting or deselecting the Clock Monitor, select the WATCHDOG service window and match the WATCHDOG key data. Subsequent writes to the WDSVR register will compare the value being written by the user to the WATCHDOG service window value and the key data (bits 7 through 1) in the WDSVR Register. Table VI shows the sequence of events that can occur.

The user must service the WATCHDOG at least once before the upper limit of the service window expires. The WATCHDOG may not be serviced more than once in every lower limit of the service window. The user may service the WATCHDOG as many times as wished in the time period between the lower and upper limits of the service window. The first write to the WDSVR Register is also counted as a WATCHDOG service.

The WATCHDOG has an output pin associated with it. This is the WDOUT pin, on pin 1 of the port G. WDOUT is active low. The WDOUT pin is in the high impedance state in the inactive state. Upon triggering the WATCHDOG, the logic will pull the WDOUT (G1) pin low for an additional 16 t_C–32 t_C cycles after the signal level on WDOUT pin goes below the lower Schmitt trigger threshold. After this delay, the device will stop forcing the WDOUT output low.

The WATCHDOG service window will restart when the WDOUT pin goes high. It is recommended that the user tie the WDOUT pin back to V_{CC} through a resistor in order to pull WDOUT high.

A WATCHDOG service while the WDOUT signal is active will be ignored. The state of the WDOUT pin is not guaranteed on reset, but if it powers up low then the WATCHDOG will time out and WDOUT will enter high impedance state.

WATCHDOG Operation (Continued)

The Clock Monitor forces the G1 pin low upon detecting a clock frequency error. The Clock Monitor error will continue until the clock frequency has reached the minimum specified value, after which the G1 output will enter the high impedance TRI-STATE mode following 16 t_C–32 t_C clock cycles. The Clock Monitor generates a continual Clock Monitor error if the oscillator fails to start, or fails to reach the minimum specified frequency. The specification for the Clock Monitor is as follows:

$1/t_C > 10$ kHz—No clock rejection.

$1/t_C < 10$ Hz—Guaranteed clock rejection.

WATCHDOG AND CLOCK MONITOR SUMMARY

The following salient points regarding the WATCHDOG and CLOCK MONITOR should be noted:

- Both the WATCHDOG and CLOCK MONITOR detector circuits are inhibited during RESET.
- Following RESET, the WATCHDOG and CLOCK MONITOR are both enabled, with the WATCHDOG having he maximum service window selected.
- The WATCHDOG service window and CLOCK MONITOR enable/disable option can only be changed once, during the initial WATCHDOG service following RESET.
- The initial WATCHDOG service must match the key data value in the WATCHDOG Service register WDSVR in order to avoid a WATCHDOG error.
- Subsequent WATCHDOG services must match all three data fields in WDSVR in order to avoid WATCHDOG errors.
- The correct key data value cannot be read from the WATCHDOG Service register WDSVR. Any attempt to read this key data value of 01100 from WDSVR will read as key data value of all 0's.
- The WATCHDOG detector circuit is inhibited during both the HALT and IDLE modes.
- The CLOCK MONITOR detector circuit is active during both the HALT and IDLE modes. Consequently, the device inadvertently entering the HALT mode will be detected as a CLOCK MONITOR error (provided that the CLOCK MONITOR enable option has been selected by the program).
- With the single-pin R/C oscillator mask option selected and the CLKDLY bit reset, the WATCHDOG service window will resume following HALT mode from where it left off before entering the HALT mode.
- With the crystal oscillator mask option selected, or with the single-pin R/C oscillator mask option selected and the CLKDLY bit set, the WATCHDOG service window will be set to its selected value from WDSVR following HALT. Consequently, the WATCHDOG should not be serviced for at least 2048 instruction cycles following HALT, but must be serviced within the selected window to avoid a WATCHDOG error.
- The IDLE timer T0 is not initialized with RESET.
- The user can sync in to the IDLE counter cycle with an IDLE counter (T0) interrupt or by monitoring the T0PND flag. The T0PND flag is set whenever the thirteenth bit of the IDLE counter toggles (every 4096 instruction cycles). The user is responsible for resetting the T0PND flag.
- A hardware WATCHDOG service occurs just as the device exits the IDLE mode. Consequently, the WATCHDOG should not be serviced for at least 2048 instruction cycles following IDLE, but must be serviced within the selected window to avoid a WATCHDOG error.
- Following RESET, the initial WATCHDOG service (where the service window and the CLOCK MONITOR enable/disable must be selected) may be programmed anywhere within the maximum service window (65,536 instruction cycles) initialized by RESET. Note that this initial WATCHDOG service may be programmed within the initial 2048 instruction cycles without causing a WATCHDOG error.

Detection of Illegal Conditions

The device can detect various illegal conditions resulting from coding errors, transient noise, power supply voltage drops, runaway programs, etc.

Reading of undefined ROM gets zeros. The opcode for software interrupt is zero. If the program fetches instructions from undefined ROM, this will force a software interrupt, thus signaling that an illegal condition has occurred.

The subroutine stack grows down for each call (jump to subroutine), interrupt, or PUSH, and grows up for each return or POP. The stack pointer is initialized to RAM location 06F Hex during reset. Consequently, if there are more returns than calls, the stack pointer will point to addresses 070 and 071 Hex (which are undefined RAM). Undefined RAM from addresses 070 to 07F (Segment 0), 140 to 17F (Segment 1), and all other segments (i.e., Segments 2 . . . etc.) is read as all 1's, which in turn will cause the program to return to address 7FFF Hex. This is an undefined ROM location and the instruction fetched (all 0's) from this location will generate a software interrupt signaling an illegal condition.

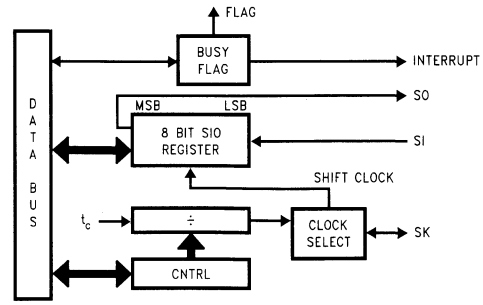
Thus, the chip can detect the following illegal conditions:

a. Executing from undefined ROM
b. Over "POP"ing the stack by having more returns than calls.

When the software interrupt occurs, the user can re-initialize the stack pointer and do a recovery procedure before restarting (this recovery program is probably similar to that following reset, but might not contain the same program initialization procedures). The recovery program should reset the software interrupt pending bit using the RPND instruction.

MICROWIRE/PLUS

MICROWIRE/PLUS is a serial synchronous communications interface. The MICROWIRE/PLUS capability enables the device to interface with any of National Semiconductor's MICROWIRE peripherals (i.e. A/D converters, display drivers, E^2PROMs etc.) and with other microcontrollers which support the MICROWIRE interface. It consists of an 8-bit serial shift register (SIO) with serial data input (SI), serial data output (SO) and serial shift clock (SK). *Figure 17* shows a block diagram of the MICROWIRE/PLUS logic.



TL/DD/12602–22

FIGURE 17. MICROWIRE/PLUS Block Diagram

The shift clock can be selected from either an internal source or an external source. Operating the MICROWIRE/PLUS arrangement with the internal clock source is called the Master mode of operation. Similarly, operating the MICROWIRE/PLUS arrangement with an external shift clock is called the Slave mode of operation.

The CNTRL register is used to configure and control the MICROWIRE/PLUS mode. To use the MICROWIRE/PLUS, the MSEL bit in the CNTRL register is set to one. In the master mode, the SK clock rate is selected by the two bits, SL0 and SL1, in the CNTRL register. Table VII details the different clock rates that may be selected.

TABLE VI. WATCHDOG Service Actions

Key Data	Window Data	Clock Monitor	Action
Match	Match	Match	Valid Service: Restart Service Window
Don't Care	Mismatch	Don't Care	Error: Generate WATCHDOG Output
Mismatch	Don't Care	Don't Care	Error: Generate WATCHDOG Output
Don't Care	Don't Care	Mismatch	Error: Generate WATCHDOG Output

TABLE VII. MICROWIRE/PLUS Master Mode Clock Select

SL1	SL0	SK
0	0	$2 \times t_c$
0	1	$4 \times t_c$
1	x	$8 \times t_c$

Where t_c is the instruction cycle clock

5

MICROWIRE/PLUS (Continued)

MICROWIRE/PLUS OPERATION

Setting the BUSY bit in the PSW register causes the MICROWIRE/PLUS to start shifting the data. It gets reset when eight data bits have been shifted. The user may reset the BUSY bit by software to allow less than 8 bits to shift. If enabled, an interrupt is generated when eight data bits have been shifted. The device may enter the MICROWIRE/PLUS mode either as a Master or as a Slave. *Figure 18* shows how two devices, microcontrollers and several peripherals may be interconnected using the MICROWIRE/PLUS arrangements.

Warning:

The SIO register should only be loaded when the SK clock is low. Loading the SIO register while the SK clock is high will result in undefined data in the SIO register. SK clock is normally low when not shifting.

Setting the BUSY flag when the input SK clock is high in the MICROWIRE/PLUS slave mode may cause the current SK clock for the SIO shift register to be narrow. For safety, the BUSY flag should only be set when the input SK clock is low.

MICROWIRE/PLUS Master Mode Operation

In the MICROWIRE/PLUS Master mode of operation the shift clock (SK) is generated internally by the device. The MICROWIRE Master always initiates all data exchanges. The MSEL bit in the CNTRL register must be set to enable the SO and SK functions onto the G Port. The SO and SK pins must also be selected as outputs by setting appropriate bits in the Port G configuration register. Table VII summarizes the bit settings required for Master mode of operation.

MICROWIRE/PLUS Slave Mode Operation

In the MICROWIRE/PLUS Slave mode of operation the SK clock is generated by an external source. Setting the MSEL bit in the CNTRL register enables the SO and SK functions onto the G Port. The SK pin must be selected as an input and the SO pin is selected as an output pin by setting and resetting the appropriate bits in the Port G configuration register. Table VII summarizes the settings required to enter the Slave mode of operation.

The user must set the BUSY flag immediately upon entering the Slave mode. This will ensure that all data bits sent by the Master will be shifted properly. After eight clock pulses the BUSY flag will be cleared and the sequence may be repeated.

Alternate SK Phase Operation

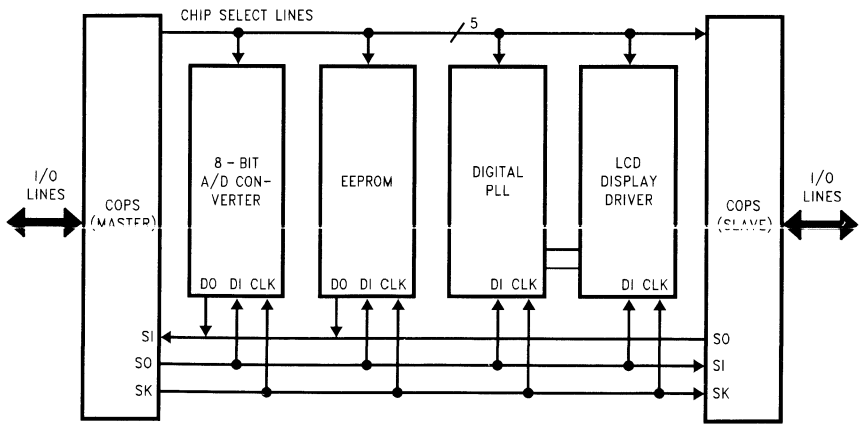
The device allows either the normal SK clock or an alternate phase SK clock to shift data in and out of the SIO register. In both the modes the SK is normally low. In the normal mode data is shifted in on the rising edge of the SK clock and the data is shifted out on the falling edge of the SK clock. The SIO register is shifted on each falling edge of the SK clock. In the alternate SK phase operation, data is shifted in on the falling edge of the SK clock and shifted out on the rising edge of the SK clock.

A control flag, SKSEL, allows either the normal SK clock or the alternate SK clock to be selected. Resetting SKSEL causes the MICROWIRE/PLUS logic to be clocked from the normal SK signal. Setting the SKSEL flag selects the alternate SK clock. The SKSEL is mapped into the G6 configuration bit. The SKSEL flag will power up in the reset condition, selecting the normal SK signal.

TABLE VIII

This table assumes that the control flag MSEL is set.

G4 (SO) Config. Bit	G5 (SK) Config. Bit	G4 Fun.	G5 Fun.	Operation
1	1	SO	Int. SK	MICROWIRE/PLUS Master
0	1	TRI-STATE	Int. SK	MICROWIRE/PLUS Master
1	0	SO	Ext. SK	MICROWIRE/PLUS Slave
0	0	TRI-STATE	Ext. SK	MICROWIRE/PLUS Slave



TL/DD/12602–23

FIGURE 18. MICROWIRE/PLUS Application

Memory Map

All RAM, ports and registers (except A and PC) are mapped into data memory address space.

Address S/ADD REG	Contents
0000 to 006F	On-Chip RAM bytes (112 bytes)
0070 to 007F	Unused RAM Address Space (Reads As All Ones)
xx80 to xxAF	Unused RAM Address Space (Reads Undefined Data)
0098	Dividend or Result Byte (MDR1)
0099	Dividend/Multiplier or Result Byte (MDR2)
009A	Dividend/Result Byte or Undefined (MDR3)
009B	Dividend/Multiplicand or Result Byte (MDR4)
009C	Divisor or Multiplicand Byte (MDR5) Multiply/Divide Control Register (MDCR)
xxB0	Timer T3 Lower Byte
XXB1	Timer T3 Upper Byte
xxB2	Timer T3 Autoload Register T3RA Lower Byte
xxB3	Timer T3 Autoload Register T3RA Upper Byte
xxB4	Timer T3 Autoload Register T3RB Lower Byte
xxB5	Timer T3 Autoload Register T3RB Upper Byte
xxB6	Timer T3 Control Register
xxB7	Comparator Select Register (CMPSL)
xxB8	UART Transmit Buffer (TBUF)
xxB9	UART Receive Buffer (RBUF)
xxBA	UART Control and Status Register (ENU)
xxBB	UART Receive Control and Status Register (ENUR)
xxBC	UART Interrupt and Clock Source Register (ENUI)
xxBD	UART Baud Register (BAUD)
xxBE	UART Prescale Select Register (PSR)
xxBF	Reserved for UART
xxC0	Timer T2 Lower Byte
xxC1	Timer T2 Upper Byte
xxC2	Timer T2 Autoload Register T2RA Lower Byte
xxC3	Timer T2 Autoload Register T2RA Upper Byte
xxC4	Timer T2 Autoload Register T2RB Lower Byte
xxC5	Timer T2 Autoload Register T2RB Upper Byte
xxC6	Timer T2 Control Register
xxC7	WATCHDOG Service Register (Reg:WDSVR)
xxC8	MIWU Edge Select Register (Reg:WKEDG)
xxC9	MIWU Enable Register (Reg:WKEN)
xxCA	MIWU Pending Register (Reg:WKPND)
xxCB	Reserved
xxCC	Reserved
xxCD to xxCF	Reserved

Address S/ADD REG	Contents
xxD0	Port L Data Register
xxD1	Port L Configuration Register
xxD2	Port L Input Pins (Read Only)
xxD3	Reserved for Port L
xxD4	Port G Data Register
xxD5	Port G Configuration Register
xxD6	Port G Input Pins (Read Only)
xxD7	Port I Input Pins (Read Only)
xxD8	Port C Data Register
xxD9	Port C Configuration Register
xxDA	Port C Input Pins (Read Only)
xxDB	Reserved for Port C
xxDC	Port D
xxDD to DF	Reserved for Port D
xxE0 to xxE5	Reserved for EE Control Registers
xxE6	Timer T1 Autoload Register T1RB Lower Byte
xxE7	Timer T1 Autoload Register T1RB Upper Byte
xxE8	ICNTRL Register
xxE9	MICROWIRE/PLUS Shift Register
xxEA	Timer T1 Lower Byte
xxEB	Timer T1 Upper Byte
xxEC	Timer T1 Autoload Register T1RA Lower Byte
xxED	Timer T1 Autoload Register T1RA Upper Byte
xxEE	CNTRL Control Register
xxEF	PSW Register
xxF0 to FB	On-Chip RAM Mapped as Registers
xxFC	X Register
xxFD	SP Register
xxFE	B Register
xxFF	S Register
0100 to 017F	On-Chip 128 RAM Bytes
0200 to 027F	On-Chip 128 RAM Bytes
0300 to 037F	On-Chip 128 RAM Bytes

Reading memory locations 0070H–007FH (Segment 0) will return all ones. Reading unused memory locations 0080H–00AFH (Segment 0) will return undefined data. Reading memory locations from other unused Segments (i.e., Segment 4, Segment 5, ... etc.) will return all ones.

Addressing Modes

There are ten addressing modes, six for operand addressing and four for transfer of control.

OPERAND ADDRESSING MODES

Register Indirect

This is the "normal" addressing mode. The operand is the data memory addressed by the B pointer or X pointer.

Register Indirect (with auto post increment or decrement of pointer)

This addressing mode is used with the LD and X instructions. The operand is the data memory addressed by the B pointer or X pointer. This is a register indirect mode that automatically post increments or decrements the B or X register after executing the instruction.

Direct

The instruction contains an 8-bit address field that directly points to the data memory for the operand.

Immediate

The instruction contains an 8-bit immediate field as the operand.

Short Immediate

This addressing mode is used with the Load B Immediate instruction. The instruction contains a 4-bit immediate field as the operand.

Indirect

This addressing mode is used with the LAID instruction. The contents of the accumulator are used as a partial address (lower 8 bits of PC) for accessing a data operand from the program memory.

TRANSFER OF CONTROL ADDRESSING MODES

Relative

This mode is used for the JP instruction, with the instruction field being added to the program counter to get the new program location. JP has a range from −31 to +32 to allow a 1-byte relative jump (JP + 1 is implemented by a NOP instruction). There are no "pages" when using JP, since all 15 bits of PC are used.

Absolute

This mode is used with the JMP and JSR instructions, with the instruction field of 12 bits replacing the lower 12 bits of the program counter (PC). This allows jumping to any location in the current 4k program memory segment.

Absolute Long

This mode is used with the JMPL and JSRL instructions, with the instruction field of 15 bits replacing the entire 15 bits of the program counter (PC). This allows jumping to any location up to 32k program memory space.

Indirect

This mode is used with the JID instruction. The contents of the accumulator are used as a partial address (lower 8 bits of PC) for accessing a location in the program memory. The contents of this program memory location serve as a partial address (lower 8 bits of PC) for the jump to the next instruction.

Note: The VIS is a special case of the Indirect Transfer of Control addressing mode, where the double byte vector associated with the interrupt is transferred from adjacent addresses in the program memory into the program counter (PC) in order to jump to the associated interrupt service routine.

Instruction Set

Register and Symbol Definition

Registers	
A	8-Bit Accumulator Register
B	8-Bit Address Register
X	8-Bit Address Register
SP	8-Bit Stack Pointer Register
PC	15-Bit Program Counter Register
PU	Upper 7 Bits of PC
PL	Lower 8 Bits of PC
C	1 Bit of PSW Register for Carry
HC	1 Bit of PSW Register for Half Carry
GIE	1 Bit of PSW Register for Global Interrupt Enable
VU	Interrupt Vector Upper Byte
VL	Interrupt Vector Lower Byte

Symbols	
[B]	Memory Indirectly Addressed by B Register
[X]	Memory Indirectly Addressed by X Register
MD	Direct Addressed Memory
Mem	Direct Addressed Memory or [B]
Meml	Direct Addressed Memory or [B] or Immediate Data
Imm	8-Bit Immediate Data
Reg	Register Memory: Addresses F0 to FF (Includes B, X and SP)
Bit	Bit Number (0 to 7)
←	Loaded with
⟷	Exchanged with

Instruction Set (Continued)

INSTRUCTION SET

ADD	A,Meml	ADD	A ← A + Meml
ADC	A,Meml	ADD with Carry	A ← A + Meml + C, C ← Carry HC ← Half Carry
SUBC	A,Meml	Subtract with Carry	A ← A − $\overline{\text{Meml}}$ + C, C ← Carry HC ← Half Carry
AND	A,Meml	Logical AND	A ← A and Meml
ANDSZ	A,Imm	Logical AND Immed., Skip if Zero	Skip next if (A and Imm) = 0
OR	A,Meml	Logical OR	A ← A or Meml
XOR	A,Meml	Logical EXclusive OR	A ← A xor Meml
IFEQ	MD,Imm	IF EQual	Compare MD and Imm, Do next if MD = Imm
IFEQ	A,Meml	IF EQual	Compare A and Meml, Do next if A = Meml
IFNE	A,Meml	IF Not Equal	Compare A and Meml, Do next if A ≠ Meml
IFGT	A,Meml	IF Greater Than	Compare A and Meml, Do next if A > Meml
IFBNE	#	If B Not Equal	Do next if lower 4 bits of B ≠ Imm
DRSZ	Reg	Decrement Reg., Skip if Zero	Reg ← Reg − 1, Skip if Reg = 0
SBIT	#,Mem	Set BIT	1 to bit, Mem (bit = 0 to 7 immediate)
RBIT	#,Mem	Reset BIT	0 to bit, Mem
IFBIT	#,Mem	IF BIT	If bit in A or Mem is true do next instruction
RPND		Reset PeNDing Flag	Reset Software Interrupt Pending Flag
X	A,Mem	EXchange A with Memory	A ⟷ Mem
X	A,[X]	EXchange A with Memory [X]	A ⟷ [X]
LD	A,Meml	LoaD A with Memory	A ← Meml
LD	A,[X]	LoaD A with Memory [X]	A ← [X]
LD	B,Imm	LoaD B with Immed.	B ← Imm
LD	Mem,Imm	LoaD Memory Immed	Mem ← Imm
LD	Reg,Imm	LoaD Register Memory Immed.	Reg ← Imm
X	A, [B ±]	EXchange A with Memory [B]	A ⟷ [B], (B ← B ±1)
X	A, [X ±]	EXchange A with Memory [X]	A ⟷ [X], (X ← X ±1)
LD	A, [B±]	LoaD A with Memory [B]	A ← [B], (B ← B ±1)
LD	A, [X±]	LoaD A with Memory [X]	A ← [X], (X ← X±1)
LD	[B±],Imm	LoaD Memory [B] Immed.	[B] ← Imm, (B ← B±1)
CLR	A	CLeaR A	A ← 0
INC	A	INCrement A	A ← A + 1
DEC	A	DECrementA	A ← A − 1
LAID		Load A InDirect from ROM	A ← ROM (PU,A)
DCOR	A	Decimal CORrect A	A ← BCD correction of A (follows ADC, SUBC)
RRC	A	Rotate A Right thru C	C → A7 → . . . → A0 → C
RLC	A	Rotate A Left thru C	C ← A7 ← . . . ← A0 ← C
SWAP	A	SWAP nibbles of A	A7 . . . A4 ⟷ A3 . . . A0
SC		Set C	C ← 1, HC ← 1
RC		Reset C	C ← 0, HC ← 0
IFC		IF C	IF C is true, do next instruction
IFNC		IF Not C	If C is not true, do next instruction
POP	A	POP the stack into A	SP ← SP + 1, A ← [SP]
PUSH	A	PUSH A onto the stack	[SP] ← A, SP ← SP − 1
VIS		Vector to Interrupt Service Routine	PU ← [VU], PL ← [VL]
JMPL	Addr.	Jump absolute Long	PC ← ii (ii = 15 bits, 0 to 32k)
JMP	Addr.	Jump absolute	PC9 . . . 0 ← i (i = 12 bits)
JP	Disp.	Jump relative short	PC ← PC + r (r is −31 to +32, except 1)
JSRL	Addr.	Jump SubRoutine Long	[SP] ← PL, [SP−1] ← PU,SP−2, PC ← ii
JSR	Addr	Jump SubRoutine	[SP] ← PL, [SP−1] ← PU,SP−2, PC9 . . . 0 ← i
JID		Jump InDirect	PL ← ROM (PU,A)
RET		RETurn from subroutine	SP + 2, PL ← [SP], PU ← [SP−1]
RETSK		RETurn and SKip	SP + 2, PL ← [SP],PU ← [SP−1], Skip Next Instruction
RETI		RETurn from Interrupt	SP + 2, PL ← [SP],PU ← [SP−1],GIE ← 1
INTR		Generate an Interrupt	[SP] ← PL, [SP−1] ← PU, SP−2, PC ← 0FF
NOP		No OPeration	PC ← PC + 1

Instruction Execution Time

Most instructions are single byte (with immediate addressing mode instructions taking two bytes).

Most single byte instructions take one cycle time to execute.

Skipped instructions require x number of cycles to be skipped, where x equals the number of bytes in the skipped instruction opcode.

See the BYTES and CYCLES per INSTRUCTION table for details.

Bytes and Cycles per Instruction

The following table shows the number of bytes and cycles for each instruction in the format of byte/cycle.

Arithmetic and Logic Instructions

	[B]	Direct	Immed.
ADD	1/1	3/4	2/2
ADC	1/1	3/4	2/2
SUBC	1/1	3/4	2/2
AND	1/1	3/4	2/2
OR	1/1	3/4	2/2
XOR	1/1	3/4	2/2
IFEQ	1/1	3/4	2/2
IFNE	1/1	3/4	2/2
IFGT	1/1	3/4	2/2
IFBNE	1/1		
DRSZ		1/3	
SBIT	1/1	3/4	
RBIT	1/1	3/4	
IFBIT	1/1	3/4	

RPND	1/1

Instructions Using A & C

CLRA	1/1
INCA	1/1
DECA	1/1
LAID	1/3
DCOR	1/1
RRCA	1/1
RLCA	1/1
SWAPA	1/1
SC	1/1
RC	1/1
IFC	1/1
IFNC	1/1
PUSHA	1/3
POPA	1/3
ANDSZ	2/2

Transfer of Control Instructions

JMPL	3/4
JMP	2/3
JP	1/3
JSRL	3/5
JSR	2/5
JID	1/3
VIS	1/5
RET	1/5
RETSK	1/5
RETI	1/5
INTR	1/7
NOP	1/1

Memory Transfer Instructions

	Register Indirect		Direct	Immed.	Register Indirect Auto Incr. & Decr.		
	[B]	[X]			[B+, B−]	[X+, X−]	
X A,*	1/1	1/3	2/3		1/2	1/3	
LD A,*	1/1	1/3	2/3	2/2	1/2	1/3	
LD B, Imm				1/1			(IF B < 16)
LD B, Imm				2/2			(IF B > 15)
LD Mem, Imm	2/2		3/3		2/2		
LD Reg, Imm			2/3				
IFEQ MD, Imm			3/3				

* = > Memory location addressed by B or X or directly.

Opcode Table

Upper Nibble Along X-Axis
Lower Nibble Along Y-Axis

F	E	D	C	B	A	9	8	
JP −15	JP −31	LD 0F0, # i	DRSZ 0F0	RRCA	RC	ADC A, #i	ADC A,[B]	0
JP −14	JP −30	LD 0F1, # i	DRSZ 0F1	*	SC	SUBC A, #i	SUB A,[B]	1
JP −13	JP −29	LD 0F2, # i	DRSZ 0F2	X A, [X+]	X A,[B+]	IFEQ A, #i	IFEQ A,[B]	2
JP −12	JP −28	LD 0F3, # i	DRSZ 0F3	X A, [X−]	X A,[B−]	IFGT A, #i	IFGT A,[B]	3
JP −11	JP −27	LD 0F4, # i	DRSZ 0F4	VIS	LAID	ADD A, #i	ADD A,[B]	4
JP −10	JP −26	LD 0F5, # i	DRSZ 0F5	RPND	JID	AND A, #i	AND A,[B]	5
JP −9	JP −25	LD 0F6, # i	DRSZ 0F6	X A,[X]	X A,[B]	XOR A, #i	XOR A,[B]	6
JP −8	JP −24	LD 0F7, # i	DRSZ 0F7	*	*	OR A, #i	OR A,[B]	7
JP −7	JP −23	LD 0F8, # i	DRSZ 0F8	NOP	RLCA	LD A, #i	IFC	8
JP −6	JP −22	LD 0F9, # i	DRSZ 0F9	IFNE A,[B]	IFEQ Md, #i	IFNE A, #i	IFNC	9
JP −5	JP −21	LD 0FA, # i	DRSZ 0FA	LD A,[X+]	LD A,[B+]	LD [B+], #i	INCA	A
JP −4	JP −20	LD 0FB, # i	DRSZ 0FB	LD A,[X−]	LD A,[B−]	LD [B−], #i	DECA	B
JP −3	JP −19	LD 0FC, # i	DRSZ 0FC	LD Md, #i	JMPL	X A,Md	POPA	C
JP −2	JP −18	LD 0FD, # i	DRSZ 0FD	DIR	JSRL	LD A,Md	RETSK	D
JP −1	JP −17	LD 0FE, # i	DRSZ 0FE	LD A,[X]	LD A,[B]	LD [B], #i	RET	E
JP −0	JP −16	LD 0FF, # i	DRSZ 0FF	*	*	LD B, #i	RETI	F

Opcode Table (Continued)

Upper Nibble Along X-Axis

Lower Nibble Along Y-Axis

7	6	5	4	3	2	1	0	
IFBIT 0,[B]	ANDSZ A, #i	LD B,#0F	IFBNE 0	JSR x000–x0FF	JMP x000–x0FF	JP +17	INTR	0
IFBIT 1,[B]	*	LD B,#0E	IFBNE 1	JSR x100–x1FF	JMP x100–x1FF	JP +18	JP + 2	1
IFBIT 2,[B]	*	LD B,#0D	IFBNE 2	JSR x200–x2FF	JMP x200–x2FF	JP +19	JP + 3	2
IFBIT 3,[B]	*	LD B,#0C	IFBNE 3	JSR x300–x3FF	JMP x300–x3FF	JP +20	JP + 4	3
IFBIT 4,[B]	CLRA	LD B,#0B	IFBNE 4	JSR x400–x4FF	JMP x400–x4FF	JP +21	JP + 5	4
IFBIT 5,[B]	SWAPA	LD B,#0A	IFBNE 5	JSR x500–x5FF	JMP x500–x5FF	JP +22	JP + 6	5
IFBIT 6,[B]	DCORA	LD B,#09	IFBNE 6	JSR x600–x6FF	JMP x600–x6FF	JP +23	JP + 7	6
IFBIT 7,[B]	PUSHA	LD B,#08	IFBNE 7	JSR x700–x7FF	JMP x700–x7FF	JP +24	JP + 8	7
SBIT 0,[B]	RBIT 0,[B]	LD B,#07	IFBNE 8	JSR x800–x8FF	JMP x800–x8FF	JP +25	JP + 9	8
SBIT 1,[B]	RBIT 1,[B]	LD B,#06	IFBNE 9	JSR x900–x9FF	JMP x900–x9FF	JP +26	JP + 10	9
SBIT 2,[B]	RBIT 2,[B]	LD B,#05	IFBNE 0A	JSR xA00–xAFF	JMP xA00–xAFF	JP +27	JP + 11	A
SBIT 3,[B]	RBIT 3,[B]	LD B,#04	IFBNE 0B	JSR xB00–xBFF	JMP xB00–xBFF	JP +28	JP + 12	B
SBIT 4,[B]	RBIT 4,[B]	LD B,#03	IFBNE 0C	JSR xC00–xCFF	JMP xC00–xCFF	JP +29	JP + 13	C
SBIT 5,[B]	RBIT 5,[B]	LD B,#02	IFBNE 0D	JSR xD00–xDFF	JMP xD00–xDFF	JP +30	JP + 14	D
SBIT 6,[B]	RBIT 6,[B]	LD B,#01	IFBNE 0E	JSR xE00–xEFF	JMP xE00–xEFF	JP +31	JP + 15	E
SBIT 7,[B]	RBIT 7,[B]	LD B,#00	IFBNE 0F	JSR xF00–xFFF	JMP xF00–xFFF	JP +32	JP + 16	F

Where,

i is the immediate data

Md is a directly addressed memory location

* is an unused opcode

Note: The opcode 60 Hex is also the opcode for IFBIT #i,A

Mask Options

The mask programmable options are shown below. The options are programmed at the same time as the ROM pattern submission.

```
OPTION 1: CLOCK CONFIGURATION
  = 1     Crystal Oscillator (CKI/IO)
              G7 (CKO) is clock generator
              output to crystal/resonator
              CKI is the clock input
  = 2     Extend (CKI/IO) CKO available as
          G7 input
Crystal Oscillator (CKI/IO)
  = 3     Single-pin RC controlled
              oscillator (CKI/IO)
              G7 is available as a HALT
              restart and/or general purpose
              input
OPTION 2: HALT
  = 1     Enable HALT mode
  = 2     Disable HALT mode
OPTION 3: BONDING OPTIONS
  = 1     44-Pin PLCC
  = 2     40-Pin DIP
  = 3     N/A
  = 4     28-Pin DIP
  = 5     28-Pin SO
```

The chip can be driven by a clock input on the CKI input pin which can be between DC and 10 MHz. The CKO output clock is on pin G7 (if clock option = 1 has been selected). The CKI input frequency is divided down by 10 to produce the instruction cycle clock ($1/t_C$).

Development Support

Summary

- iceMASTER™: IM-COP8/400—Full feature in-circuit emulation for all COP8 products. A full set of COP8 Basic and Feature Family device and package specific probes are available.
- COP8 Debug Module: Moderate cost in-circuit emulation and development programming unit.
- COP8 Evaluation and Programming Unit: EPU-COP888GG—low cost In-circuit simulation and development programming unit.
- Assembler: COP-8-DEV-IBMA. A DOS installable cross development Assembler, Linker, Librarian and Utility Software Development Tool Kit.
- C Compiler: COP8C. A DOS installable cross development Software Tool Kit.
- OTP/EPROM Programmer Support: Covering needs from engineering prototype, pilot production to full production environments.

IceMASTER (IM) IN-CIRCUIT EMULATION

The iceMASTER IM-COP8/400 is a full feature, PC based, in-circuit emulation tool developed and marketed by Meta-Link Corporation to support the whole COP8 family of products. National is a resale vendor for these products.

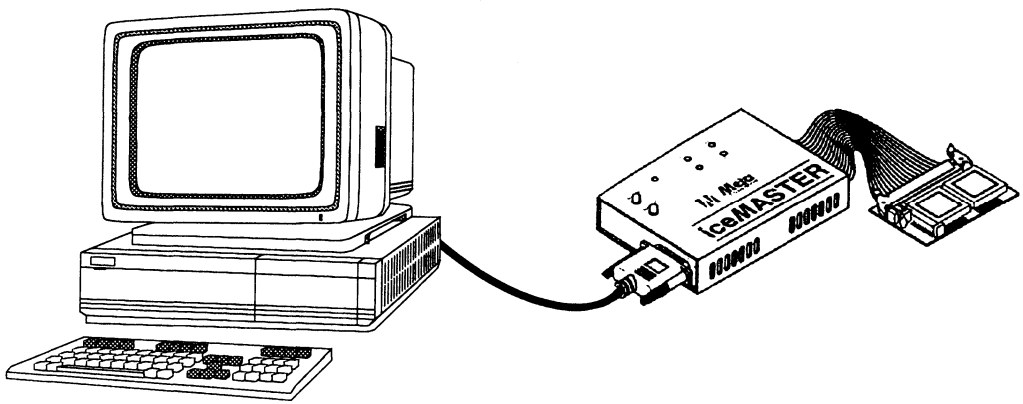
See *Figure 19* for configuration.

The iceMASTER IM-COP8/400 with its device specific COP8 Probe provides a rich feature set for developing, testing and maintaining product:

- Real-time in-circuit emulation; full 2.4V–5.5V operation range, full DC-10 MHz clock. Chip options are programmable or jumper selectable.
- Direct connection to application board by package compatible socket or surface mount assembly.
- Full 32k byte of loadable programming space that overlays (replaces) the on-chip ROM or EPROM. On-chip RAM and I/O blocks are used directly or recreated on the probe as necessary.
- Full 4k frame synchronous trace memory. Address, instruction, and 8 unspecified, circuit connectable trace lines. Display can be HLL source (e.g., C source), assembly or mixed.
- A full 64k hardware configurable break, trace on, trace off control, and pass count increment events.
- Tool set integrated interactive symbolic debugger—supports both assembler (COFF) and C Compiler (.COD) linked object formats.
- Real time performance profiling analysis; selectable bucket definition.
- Watch windows, content updated automatically at each execution break.
- Instruction by instruction memory/register changes displayed on source window when in single step operation.
- Single base unit and debugger software reconfigurable to support the entire COP8 family; only the probe personality needs to change. Debugger software is processor customized, and reconfigured from a master model file.
- Processor specific symbolic display of registers and bit level assignments, configured from master model file.
- Halt/Idle mode notification.
- On-line HELP customized to specific processor using master model file.
- Includes a copy of COP8-DEV-IBMA assembler and linker SDK.

IM Order Information

Base Unit	
IM-COP8/400-1	iceMASTER base unit, 110V power supply
IM-COP8/400-2	iceMASTER base unit, 220V power supply
iceMASTER Probe	
MHW-884FH28DWPC	28 DIP
MHW-888FH40DWPC	40 DIP
MHW-888FH44PWPC	44 PLCC
28 DIP to 28 SO Adapter	
MHW-SOIC28	28 SO



TL/DD/12602–24

FIGURE 19. COP8 iceMASTER Environment

5

Development Support (Continued)

IceMASTER DEBUG MODULE (DM)

The iceMASTER Debug Module is a PC based, combination in-circuit emulation tool and COP8 based OTP/EPROM programming tool developed and marketed by MetaLink Corporation to support the whole COP8 family of products. National is a resale vendor for these products.

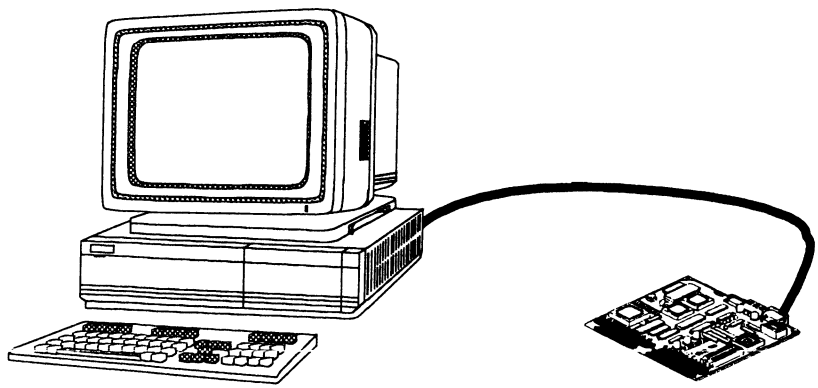
See *Figure 20* for configuration.

The iceMASTER Debug Module is a moderate cost development tool. It has the capability of in-circuit emulation for a specific COP8 microcontroller and in addition serves as a programming tool for COP8 OTP and EPROM product families. Summary of features is as follows:

- Real-time in-circuit emulation; full operating voltage range operation, full DC-10 MHz clock.
- All processor I/O pins can be cabled to an application development board with package compatible cable to socket and surface mount assembly.
- Full 32k byte of loadable programming space that overlays (replaces) the on-chip ROM or EPROM. On-chip RAM and I/O blocks are used directly or recreated as necessary.
- 100 frames of synchronous trace memory. The display can be HLL source (C source), assembly or mixed. The most recent history prior to a break is available in the trace memory.
- Configured break points; uses INTR instruction which is modestly intrusive.
- Software—only supported features are selectable.
- Tool set integrated interactive symbolic debugger—supports both assembler (COFF) and C Compiler (.COD) SDK linked object formats.
- Instruction by instruction memory/register changes displayed when in single step operation.
- Debugger software is processor customized, and reconfigured from a master model file.
- Processor specific symbolic display of registers and bit level assignments, configured from master model file.
- Halt/Idle mode notification.
- Programming menu supports full product line of programmable OTP and EPROM COP8 products. Program data is taken directly from the overlay RAM.
- Programming of 44PLCC 68PLCC parts requires external programming adapters.
- Power supply. (Includes wallmount)
- On-board V_{PP} generator from 5V input or connection to external supply supported. Requires V_{PP} level adjustment per the family programming specification (correct level is provided on an on-screen pop-down display). On-line HELP customized to specific processor using master model file.
- Includes a copy of COP8-DEV-IBMA assembler and linker SDK.

IM Order Information

Debug Module Unit	
COP8-DM/888FH	
Cable Adapters	
DM-COP8/28D	28 DIP
DM-COP8/40D	40 DIP
DM-COP8/44P	44 PLCC
28 DIP to 28 SO Adapter	
MHW-SOIC28	28 SO



TL/DD/12602–25

FIGURE 20. COP8-DM Environment

Development Support (Continued)

COP8 ASSEMBLER/LINKER SOFTWARE DEVELOPMENT TOOL KIT

National Semiconductor offers a relocateable COP8 macro cross assembler, linker, librarian and utility software development tool kit. Features are summarized as follows:

- Basic and Feature Family instruction set by "device" type.
- Nested macro capability.
- Extensive set of assembler directives.
- Supported on PC/DOS platform.
- Generates National standard COFF output files.
- Integrated Linker and Librarian.
- Integrated utilities to generate ROM code file outputs.
- DUMPCOFF utility.

This product is integrated as a part of MetaLink tools as a development kit, fully supported by the MetaLink debugger. It may be ordered separately or it is bundled with the MetaLink products at no additional cost.

Order Information

Assembler SDK	
COP8-DEV-IBMA	Assembler SDK on installable 3.5" PC/DOS Floppy Disk Drive format. Periodic upgrades and most recent version is available on National's BBS and Internet.

COP8 C COMPILER

A C Compiler is developed and marketed by Byte Craft Limited. The COP8C compiler is a fully integrated development tool specifically designed to support the compact embedded configuration of the COP8 family of products.

Features are summarized as follows:

- ANSI C with some restrictions and extensions that optimize development for the COP8 embedded application.
- BITS data type extension. Register declaration #pragma with direct bit level definitions.
- C language support for interrupt routines.
- Expert system, rule based code geration and optimization.
- Performs consistency checks against the architectural definitions of the target COP8 device.
- Generates program memory code.
- Supports linking of compiled object or COP8 assembled object formats.
- Global optimization of linked code.
- Symbolic debug load format fully source level supported by the MetaLink debugger.

SINGLE CHIP OTP/EMULATOR SUPPORT

The COP8 family is supported by single chip OTP emulators. For detailed information refer to the emulator specific datasheet and the emulator selection table below:

OTP Emulator Ordering Information

Device Number	Clock Option	Package	Emulates
COP87L84FHN-XE	Crystal	28N	COP884FH
COP87L84FHN-XE	External	28N	COP884FH
COP87L84FHM-XE	Crystal	28 SO	COP884FH
COP87L84FHM-XE	External	28 SO	COP884FH
COP87L88FHN-XE	Crystal	40N	COP888FH
COP87L88FHN-TE	External	40N	COP888FH
COP87L88FHV-XE	Crystal	44V	COP888FH
COP87L88FHV-TE	External	44V	COP888FH

Approved List

Manufacturer	North America	Europe	Asia
BP Microsystems	(800) 225-2102 (713) 688-4600 Fax: (713) 688-0920	+49-8152-4183 +49-8856-932616	+852-234-16611 +852-2710-8121
Data I/O	(800) 426-1045 (206) 881-6444 Fax: (206) 882-1043	+44-0734-440011	Call North America
HI-LO	(510) 623-8860	Call Asia	+886-2-764-0215 Fax: +886-2-756-6403
ICE Technology	(800) 624-8949 (919) 430-7915	+44-1226-767404 Fax: 0-1226-370-434	
MetaLink	(800) 638-2423 (602) 926-0797 Fax: (602) 693-0681	+49-80 9156 96-0 Fax: +49-80 9123 86	+852-737-1800
Systems General	(408) 263-6667	+41-1-9450300	+886-2-917-3005 Fax: +886-2-911-1283
Needhams	(916) 924-8037 Fax: (916) 924-8065		

Development Support (Continued)

INDUSTRY WIDE OTP/EPROM PROGRAMMING SUPPORT

Programming support, in addition to the MetaLink development tools, is provided by a full range of independent approved vendors to meet the needs from the engineering laboratory to full production.

AVAILABLE LITERATURE

For more information, please see the COP8 Basic Family User's Manual, Literature Number 620895, COP8 Feature Family User's Manual Literature Number 620897 and National's Family of 8-bit Microcontrollers COP8 Selection Guide, Literature Number 630009.

DIAL-A-HELPER SERVICE

Dial-A-Helper is a service provided by the Microcontroller Applications group. The Dial-A-Helper is an Electronic Information System that may be accessed as a Bulletin Board System (BBS) via data modem, as an FTP site on the Internet via standard FTP client application or as an FTP site on the Internet using a standard Internet browser such as Netscape or Mosaic.

The Dial-A-Helper system provides access to an automated information storage and retrieval system. The system capabilities include a MESSAGE SECTION (electronic mail, when accessed as a BBS) for communication to and from the Microcontroller Applications Group and a FILE SECTION which consists of several file areas where valuable application software and utilities could be found.

DIAL-A-HELPER BBS via a Standard Modem

Modem:	CANADA/U.S.:	(800) NSC-MICRO
		(800) 672-6427
	EUROPE:	(+49) 0-8141-351332
	Baud:	14.4k
	Set-up:	Length: 8-Bit
		Parity: None
		Stop Bit: 1
	Operation:	24 Hrs., 7 Days

DIAL-A-HELPER via FTP

ftp nscmicro.nsc.com

user: anonymous

password: username@yourhost.site.domain

DIAL-A-HELPER via a WorldWide Web Browser

ftp://nscmicro.nsc.com

National Semiconductor on the WorldWide Web

See us on the WorldWide Web at: http://www.national.com

CUSTOMER RESPONSE CENTER

Complete product information and technical support is available from National's customer response centers.

CANADA/US:	Tel:	(800) 272-9959
	email:	support @tevm2.nsc.com
EUROPE:	email:	europe.support@nsc.com
	Deutsch Tel:	+49 (0) 180-530 85 85
	English Tel:	+49 (0) 180-532 78 32
	Français Tel:	+49 (0) 180-532 93 58
	Italiano Tel:	+49 (0) 180-534 16 80
JAPAN:	Tel:	+81-043-299-2309
S.E. ASIA:	Beijing Tel:	(+86) 106-856-8601
	Shanghai Tel:	(+86) 21-6415-4092
	Hong Kong Tel:	(+852) 2737-1600
	Korea Tel:	(+82) 2-3771-6909
	Malaysia Tel:	(+60-4) 644-9061
	Singapore Tel:	(+65) 255-2226
	Taiwan Tel:	+886-2-521-3288
AUSTRALIA:	Tel:	(+61) 3-9558-9999
INDIA:	Tel:	(+91) 80-559-9467

National Semiconductor

COP688GG/COP888GG
8-Bit Microcontroller with UART and Three Multi-Function Timers

General Description

The COP8™ feature family of microcontrollers uses an 8-bit single chip core architecture fabricated with National Semiconductor's M²CMOS™ process technology. The COP888GG is a member of this expandable 8-bit core processor family of microcontrollers. (Continued)

Key Features

- Full duplex UART
- Three 16-bit timers, each with two 16-bit registers supporting:
  - — Processor Independent PWM mode
  - — External Event counter mode
  - — Input Capture mode
- Quiet design (low radiated emissions)
- 16 kbytes on-board ROM
- 512 bytes on-board RAM

Additional Peripheral Features

- Idle Timer
- Multi-Input Wake-Up (MIWU) with optional interrupts (8)
- Two analog comparators
- WATCHDOG™ and Clock Monitor logic
- MICROWIRE/PLUS™ serial I/O

I/O Features

- Memory mapped I/O
- Software selectable I/O options (TRI-STATE® Output, Push-Pull Output, Weak Pull-Up Input, High Impedance Input)
- Schmitt trigger inputs on ports G and L
- Packages:
  - — 40 DIP with 36 I/O pins
  - — 44 PQFP with 40 I/O pins
  - — 44 PLCC with 40 I/O pins

CPU/Instruction Set Features

- 1 μs instruction cycle time
- Fourteen multi-source vectored interrupts servicing
  - — External Interrupt with selectable edge
  - — Idle Timer T0
  - — Three Timers (Each with 2 Interrupts)
  - — MICROWIRE/PLUS™
  - — Multi-Input Wake-Up
  - — Software Trap
  - — UART (2)
  - — Default VIS (default interrupt)
- Versatile and easy-to-use instruction set
- 8-bit Stack Pointer (SP) — stack in RAM
- Two 8-bit Register Indirect Data Memory Pointers (B and X)

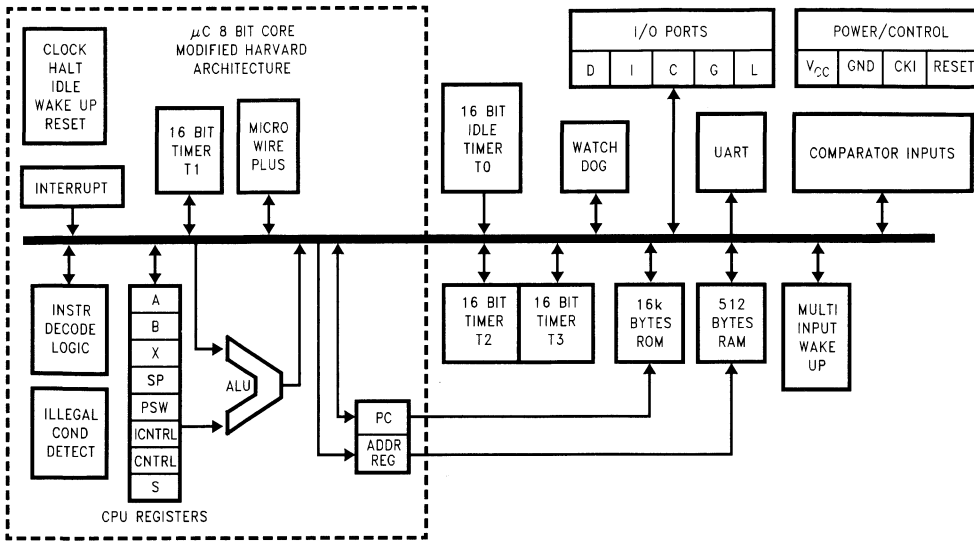
Fully Static CMOS

- Two power saving modes: HALT and IDLE
- Low current drain (typically <1 μA)
- Single supply operation: 2.5V to 5.5V
- Temperature ranges:
  $-40°C$ to $+85°C$, $-55°C$ to $+125°C$

Development Support

- Emulation and OTP devices
- Real time emulation and full program debug offered by MetaLink Development System

Block Diagram



TL/DD/12823–1

FIGURE 1. Block Diagram

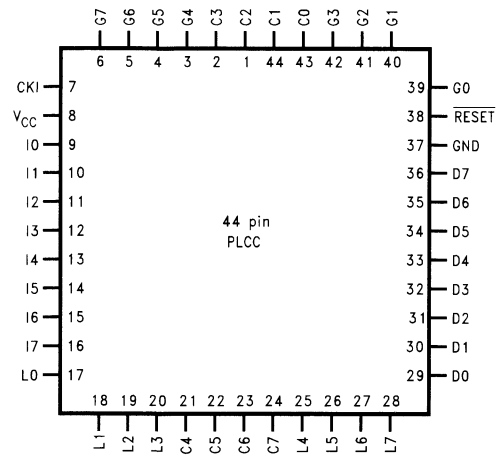
General Description (Continued)

It is a fully static part, fabricated using double-metal silicon gate microCMOS technology. Features include an 8-bit memory mapped architecture, MICROWIRE/PLUS serial I/O, three 16-bit timer/counters supporting three modes (Processor Independent PWM generation, External Event counter, and Input Capture mode capabilities), full duplex UART, two comparators, and two power saving modes (HALT and IDLE), both with a multi-sourced wakeup/interrupt capability. This multi-sourced interrupt capability may also be used independent of the HALT or IDLE modes. Each I/O pin has software selectable configurations. The devices operate over a voltage range of 2.5V to 5.5V. High throughput is achieved with an efficient, regular instruction set operating at a maximum rate of 1 μs per instruction.

Low radiated emissions are achieved by gradual turn-on output drivers and internal I_{CC} smoothing filters on the chip logic and crystal oscillator.

Connection Diagrams

Plastic Chip Carrier

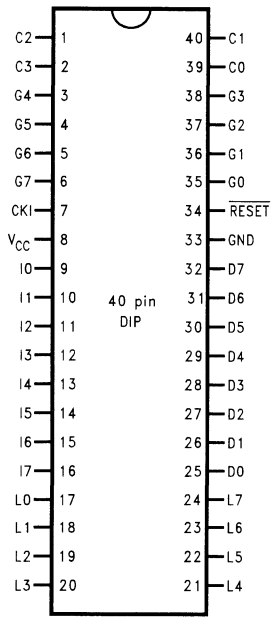


TL/DD/12823-2

Top View

Order Number COP688GG-XXX/V, COP888GG-XXX/V
See NS Package Number V44A

Dual-In-Line Package

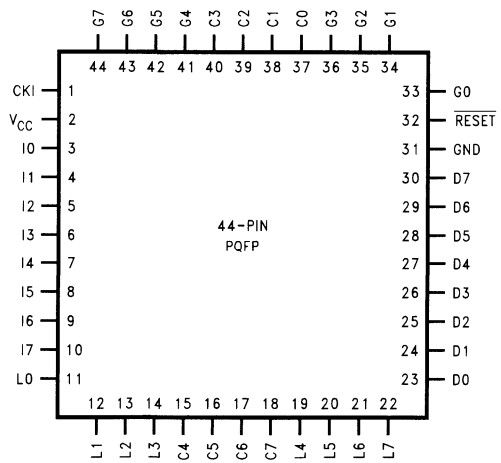


TL/DD/12823-3

Top View

Order Number COP688GG-XXX/N, COP888GG-XXX/N
See NS Package Number N40A

Molded Plastic Quad Flat Package



TL/DD/12823-24

Top View

Order Number COP688GG-XXX/VEJ,
COP888GG-XXX/VEJ
See NS Package Number VEJ44A

FIGURE 2. Connection Diagrams

Connection Diagrams (Continued)

Pinouts for 40- and 44-Pin Packages

Port	Type	Alt. Fun	Alt. Fun	40-Pin DIP	44-Pin PLCC	44-Pin PQFP
L0	I/O	MIWU		17	17	11
L1	I/O	MIWU	CKX	18	18	12
L2	I/O	MIWU	TDX	19	19	13
L3	I/O	MIWU	RDX	20	20	14
L4	I/O	MIWU	T2A	21	25	19
L5	I/O	MIWU	T2B	22	26	20
L6	I/O	MIWU	T3A	23	27	21
L7	I/O	MIWU	T3B	24	28	22
G0	I/O	INT		35	39	33
G1	WDOUT			36	40	34
G2	I/O	T1B		37	41	35
G3	I/O	T1A		38	42	36
G4	I/O	SO		3	3	41
G5	I/O	SK		4	4	42
G6	I	SI		5	5	43
G7	I/CKO	HALT Restart		6	6	44
D0	O			25	29	23
D1	O			26	30	24
D2	O			27	31	25
D3	O			28	32	26
D4	O			29	33	27
D5	O			30	34	28
D6	O			31	35	29
D7	O			32	36	30
I0	I			9	9	3
I1	I	COMP1IN−		10	10	4
I2	I	COMP1IN+		11	11	5
I3	I	COMP1OUT		12	12	6
I4	I	COMP2IN−		13	13	7
I5	I	COMP2IN+		14	14	8
I6	I	COMP2OUT		15	15	9
I7	I			16	16	10
C0	I/O			39	43	37
C1	I/O			40	44	38
C2	I/O			1	1	39
C3	I/O			2	2	40
C4	I/O				21	15
C5	I/O				22	16
C6	I/O				23	17
C7	I/O				24	18
V_{CC}				8	8	2
GND				33	37	31
CKI				7	7	1
$\overline{\text{RESET}}$				34	38	32

Absolute Maximum Ratings

Supply Voltage (V_{CC})	7V	Total Current out of GND Pin (Sink)	110 mA
Voltage at Any Pin	$-0.3V$ to $V_{CC} + 0.3V$	Storage Temperature Range	$-65°C$ to $+140°C$
Total Current into V_{CC} Pin (Source)	100 mA		

Note: *Absolute maximum ratings indicate limits beyond which damage to the device may occur. DC and AC electrical specifications are not ensured when operating the device at absolute maximum ratings.*

DC Electrical Characteristics

COP888GG: $-40°C \le T_A \le +85°C$ unless otherwise specified

Parameter	Conditions	Min	Typ	Max	Units
Operating Voltage		2.5		5.5	V
Power Supply Ripple (Note 1)	Peak-to-Peak			0.1 V_{CC}	V
Supply Current (Note 2)					
CKI = 10 MHz	$V_{CC} = 5.5V$, $t_c = 1\ \mu s$			12.5	mA
CKI = 4 MHz	$V_{CC} = 5.5V$, $t_c = 2.5\ \mu s$			5.5	mA
CKI = 4 MHz	$V_{CC} = 4.0V$, $t_c = 2.5\ \mu s$			2.5	mA
CKI = 1 MHz	$V_{CC} = 4.0V$, $t_c = 10\ \mu s$			1.4	mA
HALT Current (Note 3)	$V_{CC} = 5.5V$, CKI = 0 MHz		<1	12	μA
	$V_{CC} = 4.0V$, CKI = 0 MHz		<0.5	8	μA
IDLE Current					
CKI = 10 MHz	$V_{CC} = 5.5V$, $t_c = 1\ \mu s$			3.5	mA
CKI = 4 MHz	$V_{CC} = 5.5V$, $t_c = 2.5\ \mu s$			2.5	mA
CKI = 1 MHz	$V_{CC} = 4.0V$, $t_c = 10\ \mu s$			0.7	mA
Input Levels (V_{IH}, V_{IL})					
$\overline{RESET}$					
Logic High		0.8 V_{CC}			V
Logic Low				0.2 V_{CC}	V
All Other Inputs					
Logic High		0.7 V_{CC}			V
Logic Low				0.2 V_{CC}	V
Hi-Z Input Leakage	$V_{CC} = 5.5V$, $V_{IN} = 0V$	−2		+2	μA
Input Pullup Current	$V_{CC} = 5.5V$, $V_{IN} = 0V$	−40		−250	μA
G and L Port Input Hysteresis (Note 7)				0.35 V_{CC}	V
Output Current Levels					
D Outputs					
Source	$V_{CC} = 4.0V$, $V_{OH} = 3.3V$	−0.4			mA
	$V_{CC} = 2.5V$, $V_{OH} = 1.8V$	−0.2			mA
Sink (Note 4)	$V_{CC} = 4.0V$, $V_{OL} = 1V$	10			mA
	$V_{CC} = 2.5V$, $V_{OL} = 0.4V$	2.0			mA
All Others					
Source (Weak Pull-Up Mode)	$V_{CC} = 4.0V$, $V_{OH} = 2.7V$	−10		−100	μA
	$V_{CC} = 2.5V$, $V_{OH} = 1.8V$	−2.5		−33	μA
Source (Push-Pull Mode)	$V_{CC} = 4.0V$, $V_{OH} = 3.3V$	−0.4			mA
	$V_{CC} = 2.5V$, $V_{OH} = 1.8V$	−0.2			mA
Sink (Push-Pull Mode)	$V_{CC} = 4.0V$, $V_{OL} = 0.4V$	1.6			mA
	$V_{CC} = 2.5V$, $V_{OL} = 0.4V$	0.7			mA
TRI-STATE Leakage	$V_{CC} = 5.5V$	−2		+2	μA
Allowable Sink/Source Current per Pin (Note 6)					
D Outputs (Sink)				15	mA
All others				3	mA
Maximum Input Current without Latchup (Notes 5, 7)				±100	mA
RAM Retention Voltage, V_r	500 ns Rise and Fall Time (min)	2			V
Input Capacitance (Note 7)				7	pF
Load Capacitance on D2 (Note 7)				1000	pF

5

AC Electrical Characteristics COP888GG: $-40°C \leq T_A \leq +85°C$ unless otherwise specified

Parameter	Conditions	Min	Typ	Max	Units
Instruction Cycle Time (t_c)					
Crystal, Resonator	$2.5V \leq V_{CC} \leq 4.0V$	2.5		DC	μs
	$4.0V \leq V_{CC} \leq 5.5V$	1		DC	μs
R/C Oscillator	$2.5V \leq V_{CC} \leq 4.0V$	7.5		DC	μs
	$4.0V \leq V_{CC} \leq 5.5V$	3		DC	μs
Inputs					
t_{SETUP}	$4.0V \leq V_{CC} \leq 5.5V$	200			ns
	$2.5V \leq V_{CC} \leq 4.0V$	500			ns
t_{HOLD}	$4.5V \leq V_{CC} \leq 5.5V$	60			ns
	$2.5V \leq V_{CC} \leq 4.0V$	150			ns
Output Propagation Delay	$R_L = 2.2k, C_L = 100$ pF				
t_{PD1}, t_{PD0}					
SO, SK	$4.0V \leq V_{CC} \leq 5.5V$			0.7	μs
	$2.5V \leq V_{CC} \leq 4.0V$			1.75	μs
All Others	$4.0V \leq V_{CC} \leq 5.5V$			1.0	μs
	$2.5V \leq V_{CC} \leq 4.0V$			2.5	μs
MICROWIRE Setup Time (t_{UWS}) (Note 7)	$V_{CC} \geq 4.0V$	20			ns
MICROWIRE Hold Time (t_{UWH}) (Note 7)	$V_{CC} \geq 4.0V$	56			ns
MICROWIRE Output Propagation Delay (t_{UPD})	$V_{CC} \geq 4.0V$			220	ns
Input Pulse Width					
Interrupt Input High Time		1.0			t_c
Interrupt Input Low Time		1.0			t_c
Timer 1, 2, 3 Input High Time		1.0			t_c
Timer 1, 2, 3 Input Low Time		1.0			t_c
Reset Pulse Width		1.0			μs

t_c = Instruction Cycle Time

Note 1: Maximum rate of voltage change must be less than 0.5 V/ms.

Note 2: Supply and IDLE currents are measured with CKI driven with a square wave Oscillator, CKO driven 180° out of phase with CKI, inputs connected to V_{CC} and outputs driven low but not connected to a load.

Note 3: The HALT mode will stop CKI from oscillating in the RC and the Crystal configurations by bringing CKI high. Test Conditions: All inputs tied to V_{CC}, L and G ports in the TRI-STATE mode and tied to ground, all outputs low and tied to ground. The clock monitor is disabled.

Note 4: The user must guarantee that D2 pin does not source more than 10 mA during RESET. If D2 sources more than 10 mA during reset, the device will go into programming mode.

Note 5: Pins G6 and $\overline{\text{RESET}}$ are designed with a high voltage input network. These pins allow input voltages $> V_{CC}$ and the pins will have sink current to V_{CC} when biased at voltages $> V_{CC}$ (the pins do not have source current when biased at a voltage below V_{CC}). The effective resistance to V_{CC} is 750Ω (typical). These two pins will not latch up. The voltage at the pins must be limited to $< 14V$. **WARNING: Voltages in excess of 14V will cause damage to the pins. This warning excludes ESD transients.**

Note 6: The output propagation delay is referenced to the end of the instruction cycle where the output change occurs.

Note 7: Parameter characterized but not tested.

Absolute Maximum Ratings

If Military/Aerospace specified devices are required, please contact the National Semiconductor Sales Office/Distributors for availability and specifications.

Supply Voltage (V_{CC})	7V
Voltage at Any Pin	−0.3V to V_{CC} + 0.3V
Total Current into V_{CC} Pin (Source)	100 mA
Total Current out of GND Pin (Sink)	110 mA
Storage Temperature Range	−65°C to +140°C

Note: *Absolute maximum ratings indicate limits beyond which damage to the device may occur. DC and AC electrical specifications are not ensured when operating the device at absolute maximum ratings.*

DC Electrical Characteristics

COP688GG: $-55°C \le T_A \le +125°C$ unless otherwise specified

Parameter	Conditions	Min	Typ	Max	Units
Operating Voltage		4.5		5.5	V
Power Supply Ripple (Note 1)	Peak-to-Peak			0.1 V_{CC}	V
Supply Current (Note 2)					
CKI = 10 MHz	V_{CC} = 5.5V, t_c = 1 μs			12.5	mA
CKI = 4 MHz	V_{CC} = 5.5V, t_c = 2.5 μs			5.5	mA
HALT Current (Note 3)	V_{CC} = 5.5V, CKI = 0 MHz		<10	30	μA
IDLE Current					
CKI = 10 MHz	V_{CC} = 5.5V, t_c = 1 μs			3.5	mA
CKI = 4 MHz	V_{CC} = 5.5V, t_c = 2.5 μs			2.5	mA
Input Levels (V_{IH}, V_{IL})					
$\overline{\text{RESET}}$					
Logic High		0.8 V_{CC}			V
Logic Low				0.2 V_{CC}	V
CKI, All Other Inputs					
Logic High		0.7 V_{CC}			V
Logic Low				0.2 V_{CC}	V
Hi-Z Input Leakage	V_{CC} = 5.5V, V_{IN} = 0V	−5		+5	μA
Input Pullup Current	V_{CC} = 5.5V, V_{IN} = 0V	−35		−400	μA
G and L Port Input Hysteresis	(Note 5)			0.35 V_{CC}	V
Output Current Levels					
D Outputs					
Source	V_{CC} = 4.5V, V_{OH} = 3.3V	0.4			mA
Sink	V_{CC} = 4.5V, V_{OL} = 1V	9			mA
All Others					
Source (Weak Pull-Up Mode)	V_{CC} = 4.5V, V_{OH} = 2.7V	−9		−140	μA
Source (Push-Pull Mode)	V_{CC} = 4.5V, V_{OH} = 3.3V	0.4			mA
Sink (Push-Pull Mode)	V_{CC} = 4.5V, V_{OL} = 0.4V	1.4			mA
TRI-STATE Leakage	V_{CC} = 5.5V	−5		+5	μA
Allowable Sink/Source Current per Pin					
D Outputs (Sink)				12	mA
All others				2.5	mA
Maximum Input Current without Latchup (Notes 4, 5)	Room Temperature			±100	mA
RAM Retention Voltage, V_r	500 ns Rise and Fall Time (min)	2			V
Input Capacitance	(Note 5)			7	pF
Load Capacitance on D2	(Note 5)			1000	pF

AC Electrical Characteristics COP688GG: $-55°C \leq T_A \leq +125°C$ unless otherwise specified

Parameter	Conditions	Min	Typ	Max	Units
Instruction Cycle Time (t_c)					
Crystal, Resonator	$V_{CC} \geq 4.5V$	1.0		DC	μs
Inputs					
t_{SETUP}	$V_{CC} \geq 4.5V$	200			ns
t_{HOLD}	$V_{CC} \geq 4.5V$	60			ns
Output Propagation Delay	$R_L = 2.2k, C_L = 100$ pF				
t_{PD1}, t_{PD0}					
SO, SK	$V_{CC} \geq 4.5V$			0.7	μs
All Others	$V_{CC} \geq 4.5V$			1	μs
MICROWIRE Setup Time (t_{UWS}) (Note 5)		20			ns
MICROWIRE Hold Time (t_{UWH}) (Note 5)		56			ns
MICROWIRE Output Propagation Delay (t_{UPD})				220	ns
Input Pulse Width (Note 6)					
Interrupt Input High Time		1			t_c
Interrupt Input Low Time		1			t_c
Timer 1, 2, 3 Input High Time		1			t_c
Timer 1, 2, 3 Input Low Time		1			t_c
Reset Pulse Width		1			μs

Note 1: Maximum rate of voltage change must be less than 0.5 V/ms.

Note 2: Supply current is measured after running 2000 cycles with a square wave, CKI input, CKO open, inputs at rails and outputs open.

Note 3: The HALT mode will stop CKI from oscillating in the RC and the Crystal configurations. Measurement of IDD HALT is done with device neither sourcing or sinking current; with L, C, and G0–G5 programmed as low outputs and not driving a load; all outputs programmed low and not driving a load; all inputs tied to V_{CC}; clock monitor and comparators disabled. Parameter refers to HALT mode entered via setting bit 7 of the G Port data register. Part will pull up CKI during HALT in crystal clock mode.

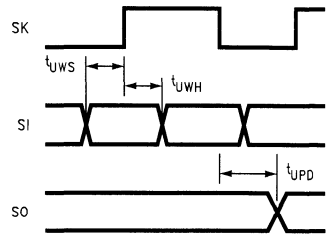
Note 4: Pins G6 and $\overline{\text{RESET}}$ are designed with a high voltage input network. These pins allow input voltages $> V_{CC}$ and the pins will have sink current to V_{CC} when biased at voltages $> V_{CC}$ (the pins do not have source current when biased at a voltage below V_{CC}). The effective resistance to V_{CC} is 750Ω (typical). These two pins will not latch up. The voltage at the pins must be limited to $< 14V$. **WARNING: Voltages in excess of 14V will cause damage to the pins. This warning excludes ESD transients.**

Note 5: Parameter characterized but not tested.

Note 6: t_c = Instruction Cycle Time.

Comparators AC and DC Characteristics $V_{CC} = 5V, -40°C \leq T_A \leq \pm 85°C$.

Parameter	Conditions	Min	Typ	Max	Units
Input Offset Voltage	$0.4V \leq V_{IN} \leq V_{CC} - 1.5V$		±10	±25	mV
Input Common Mode Voltage Range		0.4		$V_{CC} - 1.5$	V
Low Level Output Current	$V_{OL} = 0.4V$	1.6			mA
High Level Output Current	$V_{OH} = 4.6V$	1.6			mA
DC Supply Current per Comparator (When Enabled)				250	μA
Response Time	100 mV Overdrive, 100 pF Load			1	μs



TL/DD/12823-4

FIGURE 3. MICROWIRE/PLUS Timing

Pin Descriptions

V_{CC} and GND are the power supply pins. All V_{CC} and GND pins must be connected.

CKI is the clock input. This can come from an R/C generated oscillator, or a crystal oscillator (in conjunction with CKO). See Oscillator Description section.

$\overline{\text{RESET}}$ is the master reset input. See Reset Description section.

The device contains three bidirectional 8-bit I/O ports (C, G and L), where each individual bit may be independently configured as an input (Schmitt Trigger inputs on ports L and G), output or TRI-STATE under program control. Three data memory address locations are allocated for each of these I/O ports. Each I/O port has two associated 8-bit memory mapped registers, the CONFIGURATION register and the output DATA register. A memory mapped address is also reserved for the input pins of each I/O port. (See the memory map for the various addresses associated with the I/O ports.) *Figure 4* shows the I/O port configurations. The DATA and CONFIGURATION registers allow for each port bit to be individually configured under software control as shown below:

CONFIGURATION Register	DATA Register	Port Set-Up
0	0	Hi-Z Input (TRI-STATE Output)
0	1	Input with Weak Pull-Up
1	0	Push-Pull Zero Output
1	1	Push-Pull One Output

PORT L is an 8-bit I/O port. All L-pins have Schmitt triggers on the inputs.

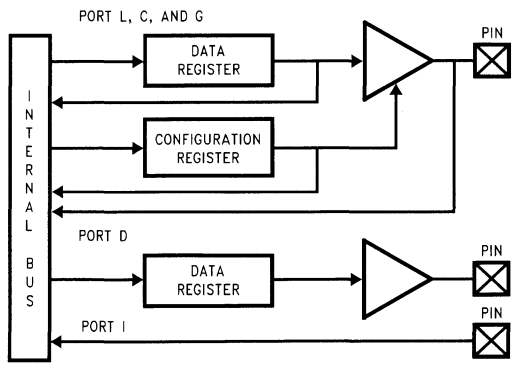
The Port L supports Multi-Input Wake Up on all eight pins. L1 is used for the UART external clock. L2 and L3 are used for the UART transmit and receive. L4 and L5 are used for the timer input functions T2A and T2B. L6 and L7 are used for the timer input functions T3A and T3B.

The Port L has the following alternate features:

- L0 MIWU
- L1 MIWU or CKX
- L2 MIWU or TDX
- L3 MIWU or RDX
- L4 MIWU or T2A
- L5 MIWU or T2B
- L6 MIWU or T3A
- L7 MIWU or T3B

Port G is an 8-bit port with 5 I/O pins (G0, G2–G5), an input pin (G6), and a dedicated output pin (G7). Pins G0 and G2–G6 all have Schmitt Triggers on their inputs. Pin G1 serves as the dedicated WDOUT WATCHDOG output, while pin G7 is either input or output depending on the oscillator mask option selected. With the crystal oscillator option selected, G7 serves as the dedicated output pin for the CKO clock output. With the single-pin R/C oscillator mask option selected, G7 serves as a general purpose input pin but is also used to bring the device out of HALT mode with a low to high transition on G7. There are two registers associated with the G Port, a data register and a configuration register. Therefore, each of the 5 I/O bits (G0, G2–G5) can be individually configured under software control.

Since G6 is an input only pin and G7 is the dedicated CKO clock output pin (crystal clock option) or general purpose input (R/C clock option), the associated bits in the data and configuration registers for G6 and G7 are used for special purpose functions as outlined on the next page. Reading the G6 and G7 data bits will return zeros.



TL/DD/12823–5

FIGURE 4. I/O Port Configurations

Pin Descriptions (Continued)

Note that the chip will be placed in the HALT mode by writing a "1" to bit 7 of the Port G Data Register. Similarly the chip will be placed in the IDLE mode by writing a "1" to bit 6 of the Port G Data Register.

Writing a "1" to bit 6 of the Port G Configuration Register enables the MICROWIRE/PLUS to operate with the alternate phase of the SK clock. The G7 configuration bit, if set high, enables the clock start up delay after HALT when the R/C clock configuration is used.

	Config Reg.	Data Reg.
G7	CLKDLY	HALT
G6	Alternate SK	IDLE

Port G has the following alternate features:

- G0 INTR (External Interrupt Input)
- G2 T1B (Timer T1 Capture Input)
- G3 T1A (Timer T1 I/O)
- G4 SO (MICROWIRE Serial Data Output)
- G5 SK (MICROWIRE Serial Clock)
- G6 SI (MICROWIRE Serial Data Input)

Port G has the following dedicated functions:

- G1 WDOUT WATCHDOG and/or Clock Monitor dedicated output
- G7 CKO Oscillator dedicated output or general purpose input

Port C is an 8-bit I/O port. The 40-pin device does not have a full complement of Port C pins. The unavailable pins are not terminated. A read operation for these unterminated pins will return unpredicatable values.

Port I is an eight-bit Hi-Z input port.

Port I1–I3 are used for Comparator 1. Port I4–I6 are used for Comparator 2.

The Port I has the following alternate features:

- I1 COMP1 − IN (Comparator 1 Negative Input)
- I2 COMP1 + IN (Comparator 1 Positive Input)
- I3 COMP1OUT (Comparator 1 Output)
- I4 COMP2 − IN (Comparator 2 Negative Input)
- I5 COMP2 + IN (Comparator 2 Positive Input)
- I6 COMP2OUT (Comparator 2 Output)

Port D is an 8-bit output port that is preset high when $\overline{\text{RESET}}$ goes low. The user can tie two or more D port outputs (except D2) together in order to get a higher drive.

Functional Description

The architecture of the device is modified Harvard architecture. With the Harvard architecture, the control store program memory (ROM) is separated from the data store memory (RAM). Both ROM and RAM have their own separate addressing space with separate address buses. The architecture, though based on Harvard architecture, permits transfer of data from ROM to RAM.

CPU REGISTERS

The CPU can do an 8-bit addition, subtraction, logical or shift operation in one instruction (t_C) cycle time.

There are six CPU registers:

A is the 8-bit Accumulator Register

PC is the 15-bit Program Counter Register

PU is the upper 7 bits of the program counter (PC)
PL is the lower 8 bits of the program counter (PC)

B is an 8-bit RAM address pointer, which can be optionally post auto incremented or decremented.

X is an 8-bit alternate RAM address pointer, which can be optionally post auto incremented or decremented.

SP is the 8-bit stack pointer, which points to the subroutine/interrupt stack (in RAM). The SP is initialized to RAM address 06F with reset.

S is the 8-bit Data Segment Address Register used to extend the lower half of the address range (00 to 7F) into 256 data segments of 128 bytes each.

All the CPU registers are memory mapped with the exception of the Accumulator (A) and the Program Counter (PC).

PROGRAM MEMORY

The program memory consists of 16 kbytes of ROM. These bytes may hold program instructions or constant data (data tables for the LAID instruction, jump vectors for the JID instruction, and interrupt vectors for the VIS instruction). The program memory is addressed by the 15-bit program counter (PC). All interrupts in the devices vector to program memory location 0FF Hex.

DATA MEMORY

The data memory address space includes the on-chip RAM and data registers, the I/O registers (Configuration, Data and Pin), the control registers, the MICROWIRE/PLUS SIO shift register, and the various registers, and counters associated with the timers (with the exception of the IDLE timer). Data memory is addressed directly by the instruction or indirectly by the B, X, SP pointers and S register.

The data memory consists of 512 bytes of RAM. Sixteen bytes of RAM are mapped as "registers" at addresses 0F0 to 0FF Hex. These registers can be loaded immediately, and also decremented and tested with the DRSZ (decrement register and skip if zero) instruction. The memory pointer registers X, SP, B and S are memory mapped into this space at address locations 0FC to 0FF Hex respectively, with the other registers being available for general usage.

The instruction set permits any bit in memory to be set, reset or tested. All I/O and registers (except A and PC) are memory mapped; therefore, I/O bits and register bits can be directly and individually set, reset and tested. The accumulator (A) bits can also be directly and individually tested.

Note: RAM contents are undefined upon power-up.

Data Memory Segment RAM Extension

Data memory address 0FF is used as a memory mapped location for the Data Segment Address Register (S).

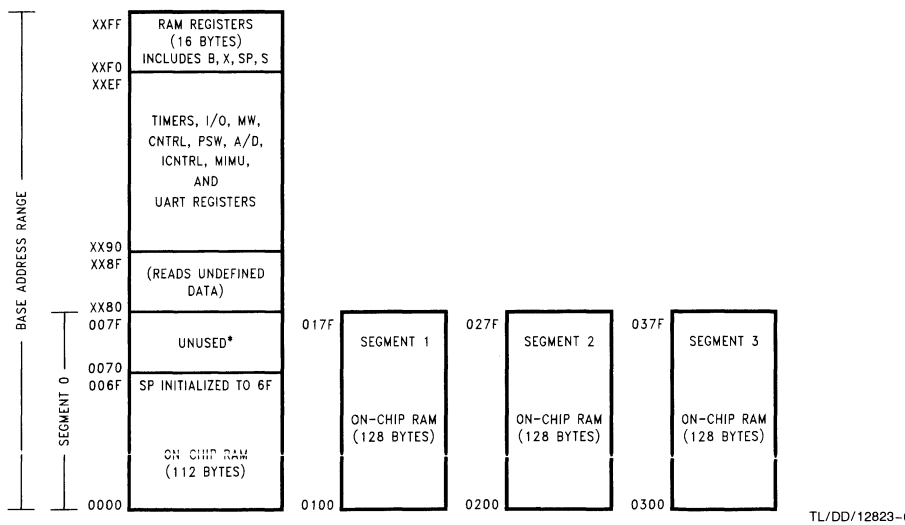
The data store memory is either addressed directly by a single byte address within the instruction, or indirectly relative to the reference of the B, X, or SP pointers (each contains a single-byte address). This single-byte address allows an addressing range of 256 locations from 00 to FF hex. The upper bit of this single-byte address divides the data store memory into two separate sections as outlined previously. With the exception of the RAM register memory from address locations 00F0 to 00FF, all RAM memory is memory mapped with the upper bit of the single-byte address being equal to zero. This allows the upper bit of the single-byte address to determine whether or not the base address range (from 0000 to 00FF) is extended. If this upper bit equals one (representing address range 0080 to 00FF), then address extension does not take place. Alternatively, if this upper bit equals zero, then the data segment extension register S is used to extend the base address range (from 0000 to 007F) from XX00 to XX7F, where XX represents the 8 bits from the S register. Thus the 128-byte data segment extensions are located from addresses 0100 to 017F for data segment 1, 0200 to 027F for data segment 2, etc., up to FF00 to FF7F for data segment 255. The base address range from 0000 to 007F represents data segment 0.

Figure 5 illustrates how the S register data memory extension is used in extending the lower half of the base address range (00 to 7F hex) into 256 data segments of 128 bytes each, with a total addressing range of 32 kbytes from XX00 to XX7F. This organization allows a total of 256 data segments of 128 bytes each with an additional upper base segment of 128 bytes. Furthermore, all addressing modes are available for all data segments. The S register must be changed under program control to move from one data segment (128 bytes) to another. However, the upper base segment (containing the 16 memory registers, I/O registers, control registers, etc.) is always available regardless of the contents of the S register, since the upper base segment (address range 0080 to 00FF) is independent of data segment extension.

The instructions that utilize the stack pointer (SP) always reference the stack as part of the base segment (Segment 0), regardless of the contents of the S register. The S register is not changed by these instructions. Consequently, the stack (used with subroutine linkage and interrupts) is always located in the base segment. The stack pointer will be initialized to point at data memory location 006F as a result of reset.

The 128 bytes of RAM contained in the base segment are split between the lower and upper base segments. The first 112 bytes of RAM are resident from address 0000 to 006F in the lower base segment, while the remaining 16 bytes of RAM represent the 16 data memory registers located at addresses 00F0 to 00FF of the upper base segment. No RAM is located at the upper sixteen addresses (0070 to 007F) of the lower base segment.

Additional RAM beyond these initial 128 bytes, however, will always be memory mapped in groups of 128 bytes (or less) at the data segment address extensions (XX00 to XX7F) of the lower base segment. The additional 128 bytes of RAM are memory mapped at address locations 0100 to 017F hex.



*Reads as all ones.

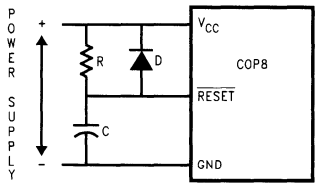
FIGURE 5. RAM Organization

Reset

The $\overline{\text{RESET}}$ input when pulled low initializes the microcontroller. Initialization will occur whenever the $\overline{\text{RESET}}$ input is pulled low. Upon initialization, the data and configuration registers for ports L, G and C are cleared, resulting in these Ports being initialized to the TRI-STATE mode. Pin G1 of the G Port is an exception (as noted below) since pin G1 is dedicated as the WATCHDOG and/or Clock Monitor error output pin. Port D is set high. The PC, PSW, ICNTRL, CNTRL, T2CNTRL and T3CNTRL control registers are cleared. The Comparator Select Register is cleared. The S register is initialized to zero. The Multi-Input Wakeup registers WKEN and WKEDG are cleared. Wakeup register WKPND is unknown. The stack pointer, SP, is initialized to 6F hex.

The device comes out of reset with both the WATCHDOG logic and the Clock Monitor detector armed, with the WATCHDOG service window bits set and the Clock Monitor bit set. The WATCHDOG and Clock Monitor circuits are inhibited during reset. The WATCHDOG service window bits being initialized high default to the maximum WATCHDOG service window of 64k t_C clock cycles. The Clock Monitor bit being initialized high will cause a Clock Monitor error following reset if the clock has not reached the minimum specified frequency at the termination of reset. A Clock Monitor error will cause an active low error output on pin G1. This error output will continue until 16 t_C–32 t_C clock cycles following the clock frequency reaching the minimum specified value, at which time the G1 output will enter the TRI-STATE mode.

The external RC network shown in *Figure 6* should be used to ensure that the $\overline{\text{RESET}}$ pin is held low until the power supply to the chip stabilizes.



TL/DD/12823-7

RC > 5 × Power Supply Rise Time

FIGURE 6. Recommended Reset Circuit

Oscillator Circuits

The chip can be driven by a clock input on the CKI input pin which can be between DC and 10 MHz. The CKO output clock is on pin G7 (crystal configuration). The CKI input frequency is divided down by 10 to produce the instruction cycle clock ($1/t_C$).

Figure 7 shows the Crystal and R/C oscillator diagrams.

CRYSTAL OSCILLATOR

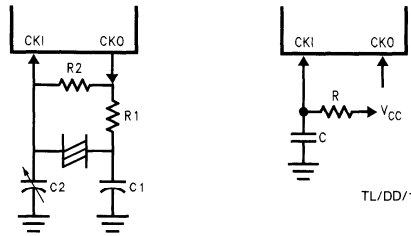
CKI and CKO can be connected to make a closed loop crystal (or resonator) controlled oscillator.

Table I shows the component values required for various standard crystal values.

R/C OSCILLATOR

By selecting CKI as a single pin oscillator input, a single pin R/C oscillator circuit can be connected to it. CKO is available as a general purpose input, and/or HALT restart input.

Table II shows the variation in the oscillator frequencies as functions of the component (R and C) values.



TL/DD/12823-8

TL/DD/12823-9

FIGURE 7. Crystal and R/C Oscillator Diagrams

TABLE I. Crystal Oscillator Configuration, T_A = 25°C

R1 (kΩ)	R2 (MΩ)	C1 (pF)	C2 (pF)	CKI Freq (MHz)	Conditions
0	1	30	30–36	10	V_{CC} = 5V
0	1	30	30–36	4	V_{CC} = 5V
0	1	200	100–150	0.455	V_{CC} = 5V

TABLE II. RC Oscillator Configuration, T_A = 25°C

R (kΩ)	C (pF)	CKI Freq (MHz)	Instr. Cycle (μs)	Conditions
3.3	82	2.2 to 2.7	3.7 to 4.6	V_{CC} = 5V
5.6	100	1.1 to 1.3	7.4 to 9.0	V_{CC} = 5V
6.8	100	0.9 to 1.1	8.8 to 10.8	V_{CC} = 5V

Note: 3k ≤ R ≤ 200k

50 pF ≤ C ≤ 200 pF

5

Control Registers

CNTRL Register (Address X'00EE)

The Timer1 (T1) and MICROWIRE/PLUS control register contains the following bits:

- SL1 & SL0 Select the MICROWIRE/PLUS clock divide by (00 = 2, 01 = 4, 1x = 8)
- IEDG External interrupt edge polarity select (0 = Rising edge, 1 = Falling edge)
- MSEL Selects G5 and G4 as MICROWIRE/PLUS signals SK and SO respectively
- T1C0 Timer T1 Start/Stop control in timer modes 1 and 2
  Timer T1 Underflow Interrupt Pending Flag in timer mode 3
- T1C1 Timer T1 mode control bit
- T1C2 Timer T1 mode control bit
- T1C3 Timer T1 mode control bit

T1C3	T1C2	T1C1	T1C0	MSEL	IEDG	SL1	SL0

Bit 7 Bit 0

PSW Register (Address X'00EF)

The PSW register contains the following select bits:

- GIE Global interrupt enable (enables interrupts)
- EXEN Enable external interrupt
- BUSY MICROWIRE/PLUS busy shifting flag
- EXPND External interrupt pending
- T1ENA Timer T1 Interrupt Enable for Timer Underflow or T1A Input capture edge
- T1PNDA Timer T1 Interrupt Pending Flag (Autoreload RA in mode 1, T1 Underflow in Mode 2, T1A capture edge in mode 3)
- C Carry Flag
- HC Half Carry Flag

HC	C	T1PNDA	T1ENA	EXPND	BUSY	EXEN	GIE

Bit 7 Bit 0

The Half-Carry bit is also affected by all the instructions that affect the Carry flag. The SC (Set Carry) and RC (Reset Carry) instructions will respectively set or clear both the carry flags. In addition to the SC and RC instructions, ADC, SUBC, RRC and RLC instructions affect the carry and Half Carry flags.

ICNTRL Register (Address X'00E8)

The ICNTRL register contains the following bits:

- T1ENB Timer T1 Interrupt Enable for T1B Input capture edge
- T1PNDB Timer T1 Interrupt Pending Flag for T1B capture edge
- μWEN Enable MICROWIRE/PLUS interrupt
- μWPND MICROWIRE/PLUS interrupt pending
- T0EN Timer T0 Interrupt Enable (Bit 12 toggle)
- T0PND Timer T0 Interrupt pending
- LPEN L Port Interrupt Enable (Multi-Input Wakeup/Interrupt)
  Bit 7 could be used as a flag

Unused	LPEN	T0PND	T0EN	μWPND	μWEN	T1PNDB	T1ENB

Bit 7 Bit 0

T2CNTRL Register (Address X'00C6)

The T2CNTRL register contains the following bits:

- T2ENB Timer T2 Interrupt Enable for T2B Input capture edge
- T2PNDB Timer T2 Interrupt Pending Flag for T2B capture edge
- T2ENA Timer T2 Interrupt Enable for Timer Underflow or T2A Input capture edge
- T2PNDA Timer T2 Interrupt Pending Flag (Autoreload RA in mode 1, T2 Underflow in mode 2, T2A capture edge in mode 3)
- T2C0 Timer T2 Start/Stop control in timer modes 1 and 2 Timer T2 Underflow Interrupt Pending Flag in timer mode 3
- T2C1 Timer T2 mode control bit
- T2C2 Timer T2 mode control bit
- T2C3 Timer T2 mode control bit

T2C3	T2C2	T2C1	T2C0	T2PNDA	T2ENA	T2PNDB	T2ENB

Bit 7 Bit 0

T3CNTRL Register (Address X'00B6)

The T3CNTRL register contains the following bits:

- T3ENB Timer T3 Interrupt Enable for T3B Input capture edge
- T3PNDB Timer T3 Interrupt Pending Flag for T3B capture edge
- T3ENA Timer T3 Interrupt Enable for Timer Underflow or T3A Input capture edge
- T3PNDA Timer T3 Interrupt Pending Flag (Autoload RA in mode 1, T3 Underflow in mode 2, T3A capture edge in mode 3)
- T3C0 Timer T3 Start/Stop control in timer modes 1 and 2
  Timer T3 Underflow Interrupt Pending Flag in timer mode 3
- T3C1 Timer T3 mode control bit
- T3C2 Timer T3 mode control bit
- T3C3 Timer T3 mode control bit

T3C3	T3C2	T3C1	T3C0	T3PNDA	T3ENA	T3PNDB	T3ENB

Bit 7 Bit 0

Timers

The device contains a very versatile set of timers (T0, T1, T2, T3). All timers and associated autoreload/capture registers power up containing random data.

TIMER T0 (IDLE TIMER)

The device supports applications that require maintaining real time and low power with the IDLE mode. This IDLE mode support is furnished by the IDLE timer T0, which is a 16-bit timer. The Timer T0 runs continuously at the fixed rate of the instruction cycle clock, t_C. The user cannot read or write to the IDLE Timer T0, which is a count down timer.

The Timer T0 supports the following functions:

- Exit out of the Idle Mode (See Idle Mode description)
- WATCHDOG logic (See WATCHDOG description)
- Start up delay out of the HALT mode

The IDLE Timer T0 can generate an interrupt when the thirteenth bit toggles. This toggle is latched into the T0PND pending flag, and will occur every 4 ms at the maximum clock frequency (t_C = 1 μs). A control flag T0EN allows the interrupt from the thirteenth bit of Timer T0 to be enabled or disabled. Setting T0EN will enable the interrupt, while resetting it will disable the interrupt.

TIMER T1, TIMER T2 AND TIMER T3

The device has a set of three powerful timer/counter blocks, T1, T2 and T3. The associated features and functioning of a timer block are described by referring to the timer block Tx. Since the three timer blocks, T1, T2 and T3 are identical, all comments are equally applicable to any of the three timer blocks.

Each timer block consists of a 16-bit timer, Tx, and two supporting 16-bit autoreload/capture registers, RxA and RxB. Each timer block has two pins associated with it, TxA and TxB. The pin TxA supports I/O required by the timer block, while the pin TxB is an input to the timer block. The powerful and flexible timer block allows the device to easily perform all timer functions with minimal software overhead. The timer block has three operating modes: Processor Independent PWM mode, External Event Counter mode, and Input Capture mode.

The control bits TxC3, TxC2, and TxC1 allow selection of the different modes of operation.

Mode 1. Processor Independent PWM Mode

As the name suggests, this mode allows the device to generate a PWM signal with very minimal user intervention. The user only has to define the parameters of the PWM signal (ON time and OFF time). Once begun, the timer block will continuously generate the PWM signal completely independent of the microcontroller. The user software services the timer block only when the PWM parameters require updating.

In this mode the timer Tx counts down at a fixed rate of t_C. Upon every underflow the timer is alternately reloaded with the contents of supporting registers, RxA and RxB. The very first underflow of the timer causes the timer to reload from the register RxA. Subsequent underflows cause the timer to be reloaded from the registers alternately beginning with the register RxB.

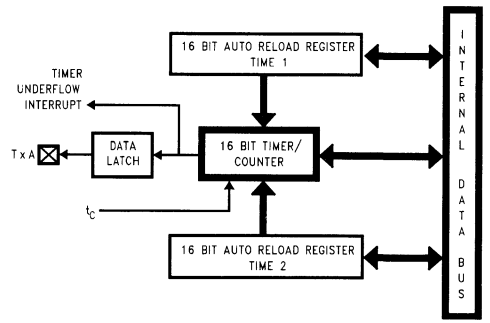
The Tx Timer control bits, TxC3, TxC2 and TxC1 set up the timer for PWM mode operation.

Figure 8 shows a block diagram of the timer in PWM mode.

The underflows can be programmed to toggle the TxA output pin. The underflows can also be programmed to generate interrupts.

Underflows from the timer are alternately latched into two pending flags, TxPNDA and TxPNDB. The user must reset these pending flags under software control. Two control enable flags, TxENA and TxENB, allow the interrupts from the timer underflow to be enabled or disabled. Setting the timer enable flag TxENA will cause an interrupt when a timer underflow causes the RxA register to be reloaded into the timer. Setting the timer enable flag TxENB will cause an interrupt when a timer underflow causes the RxB register to be reloaded into the timer. Resetting the timer enable flags will disable the associated interrupts.

Either or both of the timer underflow interrupts may be enabled. This gives the user the flexibility of interrupting once per PWM period on either the rising or falling edge of the PWM output. Alternatively, the user may choose to interrupt on both edges of the PWM output.



TL/DD/12823–10

FIGURE 8. Timer in PWM Mode

Mode 2. External Event Counter Mode

This mode is quite similar to the processor independent PWM mode described above. The main difference is that the timer, Tx, is clocked by the input signal from the TxA pin. The Tx timer control bits, TxC3, TxC2 and TxC1 allow the timer to be clocked either on a positive or negative edge from the TxA pin. Underflows from the timer are latched into the TxPNDA pending flag. Setting the TxENA control flag will cause an interrupt when the timer underflows.

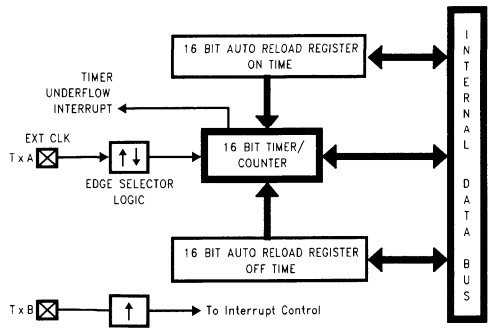
In this mode the input pin TxB can be used as an independent positive edge sensitive interrupt input if the TxENB control flag is set. The occurrence of a positive edge on the TxB input pin is latched into the TxPNDB flag.

5

Timers (Continued)

Figure 9 shows a block diagram of the timer in External Event Counter mode.

Note: The PWM output is not available in this mode since the TxA pin is being used as the counter input clock.



TL/DD/12823–11

FIGURE 9. Timer in External Event Counter Mode

Mode 3. Input Capture Mode

The device can precisely measure external frequencies or time external events by placing the timer block, Tx, in the input capture mode.

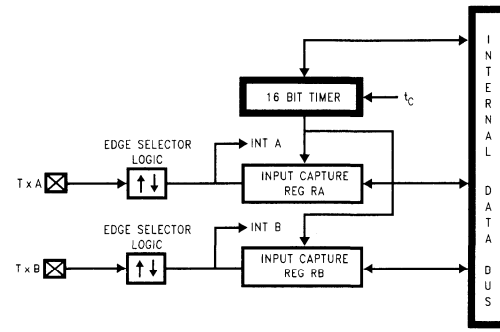
In this mode, the timer Tx is constantly running at the fixed t_C rate. The two registers, RxA and RxB, act as capture registers. Each register acts in conjunction with a pin. The register RxA acts in conjunction with the TxA pin and the register RxB acts in conjunction with the TxB pin.

The timer value gets copied over into the register when a trigger event occurs on its corresponding pin. Control bits, TxC3, TxC2 and TxC1, allow the trigger events to be specified either as a positive or a negative edge. The trigger condition for each input pin can be specified independently.

The trigger conditions can also be programmed to generate interrupts. The occurrence of the specified trigger condition on the TxA and TxB pins will be respectively latched into the pending flags, TxPNDA and TxPNDB. The control flag TxENA allows the interrupt on TxA to be either enabled or disabled. Setting the TxENA flag enables interrupts to be generated when the selected trigger condition occurs on the TxA pin. Similarly, the flag TxENB controls the interrupts from the TxB pin.

Underflows from the timer can also be programmed to generate interrupts. Underflows are latched into the timer TxC0 pending flag (the TxC0 control bit serves as the timer underflow interrupt pending flag in the Input Capture mode). Consequently, the TxC0 control bit should be reset when entering the Input Capture mode. The timer underflow interrupt is enabled with the TxENA control flag. When a TxA interrupt occurs in the Input Capture mode, the user must check both the TxPNDA and TxC0 pending flags in order to determine whether a TxA input capture or a timer underflow (or both) caused the interrupt.

Figure 10 shows a block diagram of the timer in Input Capture mode.



TL/DD/12823–12

FIGURE 10. Timer in Input Capture Mode

TIMER CONTROL FLAGS

The timers T1, T2 and T3 have indentical control structures. The control bits and their functions are summarized below.

TxC0	Timer Start/Stop control in Modes 1 and 2 (Processor Independent PWM and External Event Counter), where 1 = Start, 0 = Stop Timer Underflow Interrupt Pending Flag in Mode 3 (Input Capture)
TxPNDA	Timer Interrupt Pending Flag
TxPNDB	Timer Interrupt Pending Flag
TxENA	Timer Interrupt Enable Flag
TxENB	Timer Interrupt Enable Flag 1 = Timer Interrupt Enabled 0 = Timer Interrupt Disabled
TxC3	Timer mode control
TxC2	Timer mode control
TxC1	Timer mode control

Timers (Continued)

The timer mode control bits (TxC3, TxC2 and TxC1) are detailed below:

TxC3	TxC2	TxC1	Timer Mode	Interrupt A Source	Interrupt B Source	Timer Counts On
0	0	0	MODE 2 (External Event Counter)	Timer Underflow	Pos. TxB Edge	TxA Pos. Edge
0	0	1	MODE 2 (External Event Counter)	Timer Underflow	Pos. TxB Edge	TxA Neg. Edge
1	0	1	MODE 1 (PWM) TxA Toggle	Autoreload RA	Autoreload RB	t_C
1	0	0	MODE 1 (PWM) No TxA Toggle	Autoreload RA	Autoreload RB	t_C
0	1	0	MODE 3 (Capture) Captures: TxA Pos. Edge TxB Pos. Edge	Pos. TxA Edge or Timer Underflow	Pos. TxB Edge	t_C
1	1	0	MODE 3 (Capture) Captures: TxA Pos. Edge TxB Neg. Edge	Pos. TxA Edge or Timer Underflow	Neg. TxB Edge	t_C
0	1	1	MODE 3 (Capture) Captures: TxA Neg. Edge TxB Pos. Edge	Neg. TxA Edge or Timer Underflow	Pos. TxB Edge	t_C
1	1	1	MODE 3 (Capture) Captures: TxA Neg. Edge TxB Neg. Edge	Neg. TxA Edge or Timer Underflow	Neg. TxB Edge	t_C

Power Save Modes

The device offers the user two power save modes of operation: HALT and IDLE. In the HALT mode, all microcontroller activities are stopped. In the IDLE mode, the on-board oscillator circuitry the WATCHDOG logic, the Clock Monitor and timer T0 are active but all other microcontroller activities are stopped. In either mode, all on-board RAM, registers, I/O states, and timers (with the exception of T0) are unaltered.

HALT MODE

The device can be placed in the HALT mode by writing a "1" to the HALT flag (G7 data bit). All microcontroller activities, including the clock and timers, are stopped. The WATCHDOG logic is disabled during the HALT mode. However, the clock monitor circuitry if enabled remains active and will cause the WATCHDOG output pin (WDOUT) to go low. If the HALT mode is used and the user does not want to activate the WDOUT pin, the Clock Monitor should be disabled after the device comes out of reset (resetting the Clock Monitor control bit with the first write to the WDSVR register). In the HALT mode, the power requirements of the device are minimal and the applied voltage (V_{CC}) may be decreased to V_r (V_r = 2.0V) without altering the state of the machine.

The device supports three different ways of exiting the HALT mode. The first method of exiting the HALT mode is with the Multi-Input Wakeup feature on the L port. The second method is with a low to high transition on the CKO (G7) pin. This method precludes the use of the crystal clock configuration (since CKO becomes a dedicated output), and so may be used with an RC clock configuration. The third method of exiting the HALT mode is by pulling the $\overline{\text{RESET}}$ pin low.

Since a crystal or ceramic resonator may be selected as the oscillator, the Wakeup signal is not allowed to start the chip running immediately since crystal oscillators and ceramic resonators have a delayed start up time to reach full amplitude and frequency stability. The IDLE timer is used to generate a fixed delay to ensure that the oscillator has indeed stabilized before allowing instruction execution. In this case, upon detecting a valid Wakeup signal, only the oscillator circuitry is enabled. The IDLE timer is loaded with a value of 256 and is clocked with the t_C instruction cycle clock. The t_C clock is derived by dividing the oscillator clock down by a factor of 10. The Schmitt trigger following the CKI inverter on the chip ensures that the IDLE timer is clocked only when the oscillator has a sufficiently large amplitude to meet the Schmitt trigger specifications. This Schmitt trigger is not part of the oscillator closed loop. The startup timeout from the IDLE timer enables the clock signals to be routed to the rest of the chip.

If an RC clock option is being used, the fixed delay is introduced optionally. A control bit, CLKDLY, mapped as configuration bit G7, controls whether the delay is to be introduced or not. The delay is included if CLKDLY is set, and excluded if CLKDLY is reset. The CLKDLY bit is cleared on reset.

Power Save Modes (Continued)

The device has two mask options associated with the HALT mode. The first mask option enables the HALT mode feature, while the second mask option disables the HALT mode. With the HALT mode enable mask option, the device will enter and exit the HALT mode as described above. With the HALT disable mask option, the device cannot be placed in the HALT mode (writing a "1" to the HALT flag will have no effect, the HALT flag will remain "0").

IDLE MODE

The device is placed in the IDLE mode by writing a "1" to the IDLE flag (G6 data bit). In this mode, all activities, except the associated on-board oscillator circuitry and the IDLE Timer T0, are stopped. The power supply requirements of the micro-controller in this mode of operation are typically around 30% of normal power requirement of the microcontroller.

As with the HALT mode, the device can be returned to normal operation with a reset, or with a Multi-Input Wakeup from the L Port. Alternately, the microcontroller resumes normal operation from the IDLE mode when the thirteenth bit (representing 4.096 ms at internal clock frequency of 1 MHz, t_c = 1 μs) of the IDLE Timer toggles.

This toggle condition of the thirteenth bit of the IDLE Timer T0 is latched into the T0PND pending flag.

The user has the option of being interrupted with a transition on the thirteenth bit of the IDLE Timer T0. The interrupt can be enabled or disabled via the T0EN control bit. Setting the T0EN flag enables the interrupt and vice versa.

The user can enter the IDLE mode with the Timer T0 interrupt enabled. In this case, when the T0PND bit gets set, the device will first execute the Timer T0 interrupt service routine and then return to the instruction following the "Enter Idle Mode" instruction.

Alternatively, the user can enter the IDLE mode with the IDLE Timer T0 interrupt disabled. In this case, the device will resume normal operation with the instruction immediately following the "Enter IDLE Mode" instruction.

Note: It is necessary to program two NOP instructions following both the set HALT mode and set IDLE mode instructions. These NOP instructions are necessary to allow clock resynchronization following the HALT or IDLE modes.

Multi-Input Wakeup

The Multi-Input Wakeup feature is ued to return (wakeup) the device from either the HALT or IDLE modes. Alternately Multi-Input Wakeup/Interrupt feature may also be used to generate up to 8 edge selectable external interrupts.

Figure 11 shows the Multi-Input Wakeup logic.

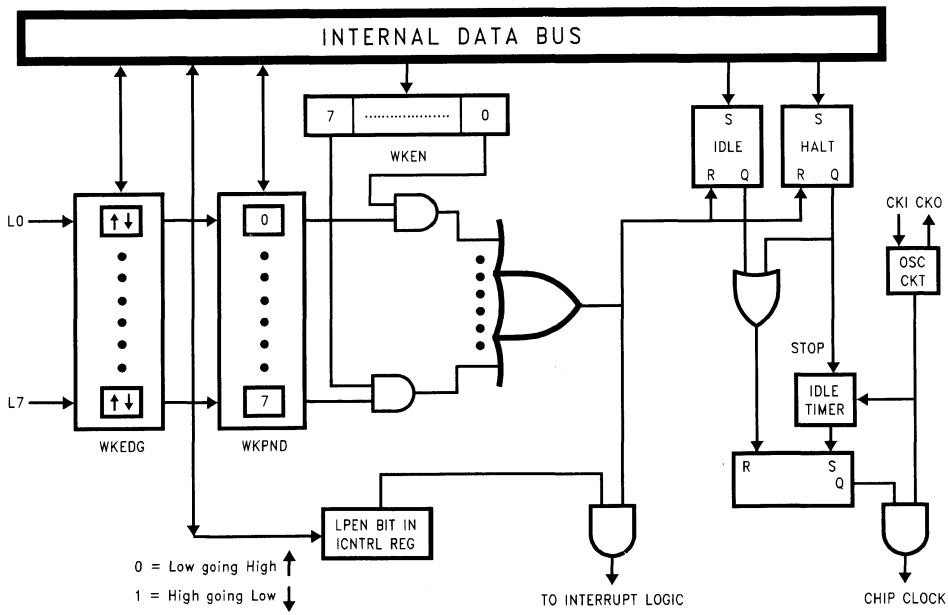


FIGURE 11. Multi-Input Wake Up Logic

Multi-Input Wakeup (Continued)

The Multi-Input Wakeup feature utilizes the L Port. The user selects which particular L port bit (or combination of L Port bits) will cause the device to exit the HALT or IDLE modes. The selection is done through the Reg: WKEN. The Reg: WKEN is an 8-bit read/write register, which contains a control bit for every L port bit. Setting a particular WKEN bit enables a Wakeup from the associated L port pin.

The user can select whether the trigger condition on the selected L Port pin is going to be either a positive edge (low to high transition) or a negative edge (high to low transition). This selection is made via the Reg: WKEDG, which is an 8-bit control register with a bit assigned to each L Port pin. Setting the control bit will select the trigger condition to be a negative edge on that particular L Port pin. Resetting the bit selects the trigger condition to be a positive edge. Changing an edge select entails several steps in order to avoid a pseudo Wakeup condition as a result of the edge change. First, the associated WKEN bit should be reset, followed by the edge select change in WKEDG. Next, the associated WKPND bit should be cleared, followed by the associated WKEN bit being re-enabled.

An example may serve to clarify this procedure. Suppose we wish to change the edge select from positive (low going high) to negative (high going low) for L Port bit 5, where bit 5 has previously been enabled for an input interrupt. The program would be as follows:

```
RBIT  5, WKEN      ; Disable Port bit 5 for
                   ; Wakeup
SBIT  5, WKEDG     ; Select neg going edge
                   ; sensitivity
RBIT  5, WKPND     ; Clear pending bit
SBIT  5, WKEN      ; Re-enable the bit for
                   ; Wakeup
```

If the L port bits have been used as outputs and then changed to inputs with Multi-Input Wakeup/Interrupt, a safety procedure should also be followed to avoid inherited pseudo wakeup conditions. After the selected L port bits have been changed from output to input but before the associated WKEN bits are enabled, the associated edge select bits in WKEDG should be set or reset for the desired edge selects, followed by the associated WKPND bits being cleared.

This same procedure should be used following reset, since the L port inputs are left floating as a result of reset.

The occurrence of the selected trigger condition for Multi-Input Wakeup is latched into a pending register called WKPND. The respective bits of the WKPND register will be set on the occurrence of the selected trigger edge on the corresponding Port L pin. The user has the responsibility of clearing these pending flags. Since WKPND is a pending register for the occurrence of selected wakeup conditions, the device will not enter the HALT mode if any Wakeup bit is both enabled and pending. Consequently, the user has the responsibility of clearing the pending flags before attempting to enter the HALT mode.

WKEN, WKPND and WKEDG are all read/write registers, and are cleared at reset.

PORT L INTERRUPTS

Port L provides the user with an additional eight fully selectable, edge sensitive interrupts which are all vectored into the same service subroutine.

The interrupt from Port L shares logic with the wake up circuitry. The register WKEN allows interrupts from Port L to be individually enabled or disabled. The register WKEDG specifies the trigger condition to be either a positive or a negative edge. Finally, the register WKPND latches in the pending trigger conditions.

The GIE (Global Interrupt Enable) bit enables the interrupt function.

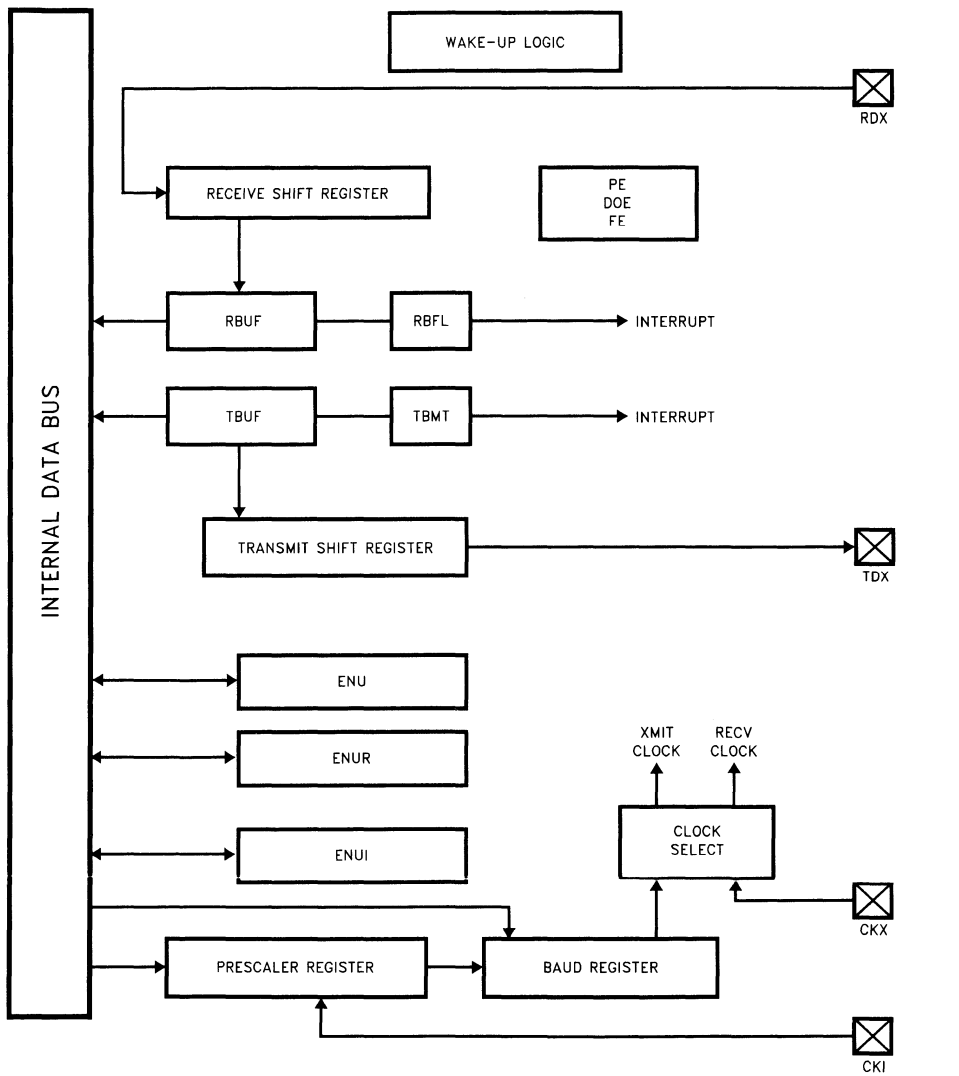
A control flag, LPEN, functions as a global interrupt enable for Port L interrupts. Setting the LPEN flag will enable interrupts and vice versa. A separate global pending flag is not needed since the register WKPND is adequate.

Since Port L is also used for waking the device out of the HALT or IDLE modes, the user can elect to exit the HALT or IDLE modes either with or without the interrupt enabled. If he elects to disable the interrupt, then the device will restart execution from the instruction immediately following the instruction that placed the microcontroller in the HALT or IDLE modes. In the other case, the device will first execute the interrupt service routine and then revert to normal operation. (See HALT MODE for clock option wakeup information.)

UART

The device contains a full-duplex software programmable UART. The UART *(Figure 12)* consists of a transmit shift register, a receive shift register and seven addressable registers, as follows: a transmit buffer register (TBUF), a receiver buffer register (RBUF), a UART control and status register (ENU), a UART receive control and status register (ENUR), a UART interrupt and clock source register (ENUI), a prescaler select register (PSR) and baud (BAUD) register. The ENU register contains flags for transmit and receive functions; this register also determines the length of the data frame (7, 8 or 9 bits), the value of the ninth bit in transmission, and parity selection bits. The ENUR register flags framing, data overrun and parity errors while the UART is receiving.

Other functions of the ENUR register include saving the ninth bit received in the data frame, enabling or disabling the UART's attention mode of operation and providing additional receiver/transmitter status information via RCVG and XMTG bits. The determination of an internal or external clock source is done by the ENUI register, as well as selecting the number of stop bits and enabling or disabling transmit and receive interrupts. A control flag in this register can also select the UART mode of operation: asynchronous or synchronous.



TL/DD/12823–14

FIGURE 12. UART Block Diagram

UART (Continued)

UART CONTROL AND STATUS REGISTERS

The operation of the UART is programmed through three registers: ENU, ENUR and ENUI. The function of the individual bits in these registers is as follows:

ENU-UART Control and Status Register (Address at 0BA)

PEN	PSEL1	XBIT9/ PSEL0	CHL1	CHL0	ERR	RBFL	TBMT
0RW	0RW	0RW	0RW	0RW	0R	0R	1R

Bit 7 Bit 0

ENUR-UART Receive Control and Status Register (Address at 0BB)

DOE	FE	PE	SPARE	RBIT9	ATTN	XMTG	RCVG
0RD	0RD	0RD	0RW*	0R	0RW	0R	0R

Bit 7 Bit 0

ENUI-UART Interrupt and Clock Source Register (Address at 0BC)

STP2	STP78	ETDX	SSEL	XRCLK	XTCLK	ERI	ETI
0RW	0RW	0RW	0RW	0RW	0RW	0RW	0RW

Bit 7 Bit 0

*Bit is not used.

0 Bit is cleared on reset.

1 Bit is set to one on reset.

R Bit is read-only; it cannot be written by software.

RW Bit is read/write.

D Bit is cleared on read; when read by software as a one, it is cleared automatically. Writing to the bit does not affect its state.

DESCRIPTION OF UART REGISTER BITS

ENU—UART CONTROL AND STATUS REGISTER

TBMT: This bit is set when the UART transfers a byte of data from the TBUF register into the TSFT register for transmission. It is automatically reset when software writes into the TBUF register.

RBFL: This bit is set when the UART has received a complete character and has copied it into the RBUF register. It is automatically reset when software reads the character from RBUF.

ERR: This bit is a global UART error flag which gets set if any or a combination of the errors (DOE, FE, PE) occur.

CHL1, CHL0: These bits select the character frame format. Parity is not included and is generated/verified by hardware.

CHL1 = 0, CHL0 = 0 The frame contains eight data bits.
CHL1 = 0, CHL0 = 1 The frame contains seven data bits.
CHL1 = 1, CHL0 = 0 The frame contains nine data bits.
CHL1 = 1, CHL0 = 1 Loopback Mode selected. Transmitter output internally looped back to receiver input. Nine bit framing format is used.

XBIT9/PSEL0: Programs the ninth bit for transmission when the UART is operating with nine data bits per frame. For seven or eight data bits per frame, this bit in conjunction with PSEL1 selects parity.

PSEL1, PSEL0: Parity select bits.

PSEL1 = 0, PSEL0 = 0 Odd Parity (if Parity enabled)
PSEL1 = 0, PSEL0 = 1 Odd Parity (if Parity enabled)
PSEL1 = 1, PSEL0 = 0 Mark(1) (if Parity enabled)
PSEL1 = 1, PSEL0 = 1 Space(0) (if Parity enabled)

PEN: This bit enables/disables Parity (7- and 8-bit modes only).

PEN = 0 Parity disabled.
PEN = 1 Parity enabled.

ENUR—UART RECEIVE CONTROL AND STATUS REGISTER

RCVG: This bit is set high whenever a framing error occurs and goes low when RDX goes high.

XMTG: This bit is set to indicate that the UART is transmitting. It gets reset at the end of the last frame (end of last Stop bit).

ATTN: ATTENTION Mode is enabled while this bit is set. This bit is cleared automatically on receiving a character with data bit nine set.

RBIT9: Contains the ninth data bit received when the UART is operating with nine data bits per frame.

SPARE: Reserved for future use.

PE: Flags a Parity Error.

PE = 0 Indicates no Parity Error has been detected since the last time the ENUR register was read.
PE = 1 Indicates the occurrence of a Parity Error.

FE: Flags a Framing Error.

FE = 0 Indicates no Framing Error has been detected since the last time the ENUR register was read.
FE = 1 Indicates the occurrence of a Framing Error.

DOE: Flags a Data Overrun Error.

DOE = 0 Indicates no Data Overrun Error has been detected since the last time the ENUR register was read.
DOE = 1 Indicates the occurrence of a Data Overrun Error.

ENUI—UART INTERRUPT AND CLOCK SOURCE REGISTER

ETI: This bit enables/disables interrupt from the transmitter section.

ETI = 0 Interrupt from the transmitter is disabled.
ETI = 1 Interrupt from the transmitter is enabled.

ERI: This bit enables/disables interrupt from the receiver section.

ERI = 0 Interrupt from the receiver is disabled.
ERI = 1 Interrupt from the receiver is enabled.

XTCLK: This bit selects the clock source for the transmitter section.

XTCLK = 0 The clock source is selected through the PSR and BAUD registers.
XTCLK = 1 Signal on CKX (L1) pin is used as the clock.

XRCLK: This bit selects the clock source for the receiver section.

XRCLK = 0 The clock source is selected through the PSR and BAUD registers.
XRCLK = 1 Signal on CKX (L1) pin is used as the clock.

SSEL: UART mode select.

SSEL = 0 Asynchronous Mode.
SSEL = 1 Synchronous Mode.

UART (Continued)

ETDX: TDX (UART Transmit Pin) is the alternate function assigned to Port L pin L2; it is selected by setting ETDX bit. To simulate line break generation, software should reset ETDX bit and output logic zero to TDX pin through Port L data and configuration registers.

STP78: This bit is set to program the last Stop bit to be 7/8th of a bit in length.

STP2: This bit programs the number of Stop bits to be transmitted.

STP2 = 0 One Stop bit transmitted.
STP2 = 1 Two Stop bits transmitted.

Associated I/O Pins

Data is transmitted on the TDX pin and received on the RDX pin. TDX is the alternate function assigned to Port L pin L2; it is selected by setting ETDX (in the ENUI register) to one. RDX is an inherent function of Port L pin L3, requiring no setup.

The baud rate clock for the UART can be generated on-chip, or can be taken from an external source. Port L pin L1 (CKX) is the external clock I/O pin. The CKX pin can be either an input or an output, as determined by Port L Configuration and Data registers (Bit 1). As an input, it accepts a clock signal which may be selected to drive the transmitter and/or receiver. As an output, it presents the internal Baud Rate Generator output.

UART Operation

The UART has two modes of operation: asynchronous mode and synchronous mode.

ASYNCHRONOUS MODE

This mode is selected by resetting the SSEL (in the ENUI register) bit to zero. The input frequency to the UART is 16 times the baud rate.

The TSFT and TBUF registers double-buffer data for transmission. While TSFT is shifting out the current character on the TDX pin, the TBUF register may be loaded by software with the next byte to be transmitted. When TSFT finishes transmitting the current character the contents of TBUF are transferred to the TSFT register and the Transmit Buffer Empty Flag (TBMT in the ENU register) is set. The TBMT flag is automatically reset by the UART when software loads a new character into the TBUF register. There is also the XMTG bit which is set to indicate that the UART is transmitting. This bit gets reset at the end of the last frame (end of last Stop bit). TBUF is a read/write register.

The RSFT and RBUF registers double-buffer data being received. The UART receiver continually monitors the signal on the RDX pin for a low level to detect the beginning of a Start bit. Upon sensing this low level, it waits for half a bit time and samples again. If the RDX pin is still low, the receiver considers this to be a valid Start bit, and the remaining bits in the character frame are each sampled a single time, at the mid-bit position. Serial data input on the RDX pin is shifted into the RSFT register. Upon receiving the complete character, the contents of the RSFT register are copied into the RBUF register and the Received Buffer Full Flag (RBFL) is set. RBFL is automatically reset when software reads the character from the RBUF register. RBUF is a read only register. There is also the RCVG bit which is set high when a framing error occurs and goes low once RDX goes high. TBMT, XMTG, RBFL and RCVG are read only bits.

SYNCHRONOUS MODE

In this mode data is transferred synchronously with the clock. Data is transmitted on the rising edge and received on the falling edge of the synchronous clock.

This mode is selected by setting SSEL bit in the ENUI register. The input frequency to the UART is the same as the baud rate.

When an external clock input is selected at the CKX pin, data transmit and receive are performed synchronously with this clock through TDX/RDX pins.

If data transmit and receive are selected with the CKX pin as clock output, the device generates the synchronous clock output at the CKX pin. The internal baud rate generator is used to produce the synchronous clock. Data transmit and receive are performed synchronously with this clock.

FRAMING FORMATS

The UART supports several serial framing formats *(Figure 13)*. The format is selected using control bits in the ENU, ENUR and ENUI registers.

The first format (1, 1a, 1b, 1c) for data transmission (CHL0 = 1, CHL1 = 0) consists of Start bit, seven Data bits (excluding parity) and 7/8, one or two Stop bits. In applications using parity, the parity bit is generated and verified by hardware.

The second format (CHL0 = 0, CHL1 = 0) consists of one Start bit, eight Data bits (excluding parity) and 7/8, one or two Stop bits. Parity bit is generated and verified by hardware.

The third format for transmission (CHL0 = 0, CHL1 = 1) consists of one Start bit, nine Data bits and 7/8, one or two Stop bits. This format also supports the UART "ATTENTION" feature. When operating in this format, all eight bits of TBUF and RBUF are used for data. The ninth data bit is transmitted and received using two bits in the ENU and ENUR registers, called XBIT9 and RBIT9. RBIT9 is a read only bit. Parity is not generated or verified in this mode.

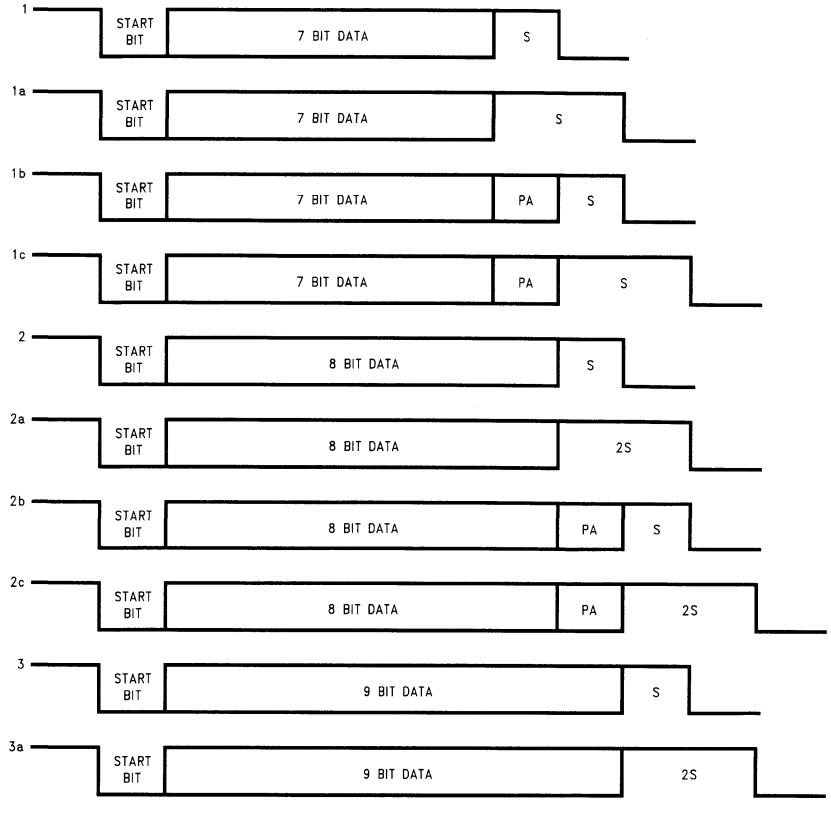
For any of the above framing formats, the last Stop bit can be programmed to be 7/8th of a bit in length. If two Stop bits are selected and the 7/8th bit is set (selected), the second Stop bit will be 7/8th of a bit in length.

The parity is enabled/disabled by PEN bit located in the ENU register. Parity is selected for 7- and 8-bit modes only. If parity is enabled (PEN = 1), the parity selection is then performed by PSEL0 and PSEL1 bits located in the ENU register.

Note that the XBIT9/PSEL0 bit located in the ENU register serves two mutually exclusive functions. This bit programs the ninth bit for transmission when the UART is operating with nine data bits per frame. There is no parity selection in this framing format. For other framing formats XBIT9 is not needed and the bit is PSEL0 used in conjunction with PSEL1 to select parity.

The frame formats for the receiver differ from the transmitter in the number of Stop bits required. The receiver only requires one Stop bit in a frame, regardless of the setting of the Stop bit selection bits in the control register. Note that an implicit assumption is made for full duplex UART operation that the framing formats are the same for the transmitter and receiver.

UART Operation (Continued)



TL/DD/12823–15

FIGURE 13. Framing Formats

UART INTERRUPTS

The UART is capable of generating interrupts. Interrupts are generated on Receive Buffer Full and Transmit Buffer Empty. Both interrupts have individual interrupt vectors. Two bytes of program memory space are reserved for each interrupt vector. The two vectors are located at addresses 0xEC to 0xEF Hex in the program memory space. The interrupts can be individually enabled or disabled using Enable Transmit Interrupt (ETI) and Enable Receive Interrupt (ERI) bits in the ENUI register.

The interrupt from the Transmitter is set pending, and remains pending, as long as both the TBMT and ETI bits are set. To remove this interrupt, software must either clear the ETI bit or write to the TBUF register (thus clearing the TBMT bit).

The interrupt from the receiver is set pending, and remains pending, as long as both the RBFL and ERI bits are set. To remove this interrupt, software must either clear the ERI bit or read from the RBUF register (thus clearing the RBFL bit).

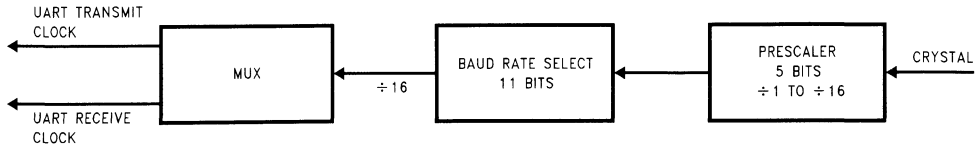
Baud Clock Generation

The clock inputs to the transmitter and receiver sections of the UART can be individually selected to come either from an external source at the CKX pin (port L, pin L1) or from a source selected in the PSR and BAUD registers. Internally, the basic baud clock is created from the oscillator frequency through a two-stage divider chain consisting of a 1–16 (increments of 0.5) prescaler and an 11-bit binary counter. *(Figure 14).* The divide factors are specified through two read/write registers shown in *Figure 15.* Note that the 11-bit Baud Rate Divisor spills over into the Prescaler Select Register (PSR). PSR is cleared upon reset.

As shown in Table IV, a Prescaler Factor of 0 corresponds to NO CLOCK. This condition is the UART power down mode where the UART clock is turned off for power saving purpose. The user must also turn the UART clock off when a different baud rate is chosen.

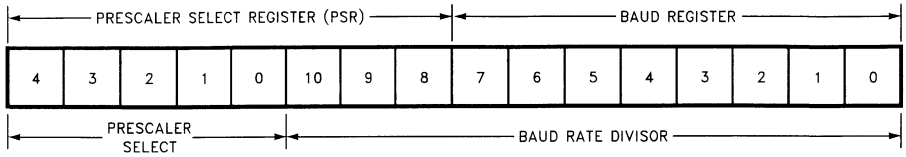
The correspondences between the 5-bit Prescaler Select and Prescaler factors are shown in Table IV. There are many ways to calculate the two divisor factors, but one particularly effective method would be to achieve a 1.8432 MHz frequency coming out of the first stage. The 1.8432 MHz prescaler output is then used to drive the software programmable baud rate counter to create a 16x clock for the following baud rates: 110, 134.5, 150, 300, 600, 1200, 1800, 2400, 3600, 4800, 7200, 9600, 19200 and 38400 (Table III). Other baud rates may be created by using appropriate divisors. The 16x clock is then divided by 16 to provide the rate for the serial shift registers of the transmitter and receiver.

Baud Clock Generation (Continued)



TL/DD/12823–16

FIGURE 14. UART BAUD Clock Generation



TL/DD/12823–17

FIGURE 15. UART BAUD Clock Divisor Registers

Baud Clock Generation (Continued)

TABLE III. Baud Rate Divisors (1.8432 MHz Prescaler Output)

Baud Rate	Baud Rate Divisor − 1 (N-1)
110 (110.03)	1046
134.5 (134.58)	855
150	767
300	383
600	191
1200	95
1800	63
2400	47
3600	31
4800	23
7200	15
9600	11
19200	5
38400	2

Note: The entries in Table IV assume a prescaler output of 1.8432 MHz. In the asynchronous mode the baud rate could be as high as 625k.

TABLE IV. Prescaler Factors

Prescaler Select	Prescaler Factor
00000	NO CLOCK
00001	1
00010	1.5
00011	2
00100	2.5
00101	3
00110	3.5
00111	4
01000	4.5
01001	5
01010	5.5
01011	6
01100	6.5
01101	7
01110	7.5
01111	8
10000	8.5
10001	9
10010	9.5
10011	10
10100	10.5
10101	11
10110	11.5
10111	12
11000	12.5
11001	13
11010	13.5
11011	14
11100	14.5
11101	15
11110	15.5
11111	16

As an example, considering Asynchronous Mode and a CKI clock of 4.608 MHz, the prescaler factor selected is:

$$4.608/1.8432 = 2.5$$

The 2.5 entry is available in Table IV. The 1.8432 MHz prescaler output is then used with proper Baud Rate Divisor (Table IV) to obtain different baud rates. For a baud rate of 19200 e.g., the entry in Table IV is 5.

$N - 1 = 5$ ($N - 1$ is the value from Table III)

$N = 6$ (N is the Baud Rate Divisor)

$$\text{Baud Rate} = 1.8432\ \text{MHz}/(16 \times 6) = 19200$$

The divide by 16 is performed because in the asynchronous mode, the input frequency to the UART is 16 times the baud rate. The equation to calculate baud rates is given below.

The actual Baud Rate may be found from:

$$BR = Fc/(16 \times N \times P)$$

5

Baud Clock Generation (Continued)

Where:

BR is the Baud Rate

Fc is the CKI frequency

N is the Baud Rate Divisor (Table III).

P is the Prescaler Divide Factor selected by the value in the Prescaler Select Register (Table IV)

Note: In the Synchronous Mode, the divisor 16 is replaced by two.

Example:

Asynchronous Mode:

Crystal Frequency = 5 MHz

Desired baud rate = 9600

Using the above equation N × P can be calculated first.

$$N \times P = (5 \times 106)/(16 \times 9600) = 32.552$$

Now 32.552 is divided by each Prescaler Factor (Table IV) to obtain a value closest to an integer. This factor happens to be 6.5 (P = 6.5).

$$N = 32.552/6.5 = 5.008 \ (N = 5)$$

The programmed value (from Table III) should be 4 (N − 1).

Using the above values calculated for N and P:

$$BR = (5 \times 106)/(16 \times 5 \times 6.5) = 9615.384$$

$$\% \text{ error} = (9615.385 - 9600)/9600 = 0.16$$

Effect of HALT/IDLE

The UART logic is reinitialized when either the HALT or IDLE modes are entered. This reinitialization sets the TBMT flag and resets all read only bits in the UART control and status registers. Read/Write bits remain unchanged. The Transmit Buffer (TBUF) is not affected, but the Transmit Shift register (TSFT) bits are set to one. The receiver registers RBUF and RSFT are not affected.

The device will exit from the HALT/IDLE modes when the Start bit of a character is detected at the RDX (L3) pin. This feature is obtained by using the Multi-Input Wakeup scheme provided on the device.

Before entering the HALT or IDLE modes the user program must select the Wakeup source to be on the RDX pin. This selection is done by setting bit 3 of WKEN (Wakeup Enable) register. The Wakeup trigger condition is then selected to be high to low transition. This is done via the WKEDG register (Bit 3 is one.)

If the device is halted and crystal oscillator is used, the Wakeup signal will not start the chip running immediately because of the finite start up time requirement of the crystal oscillator. The idle timer (T0) generates a fixed (256 t_c) delay to ensure that the oscillator has indeed stabilized before allowing the device to execute code. The user has to consider this delay when data transfer is expected immediately after exiting the HALT mode.

Diagnostic

Bits CHARL0 and CHARL1 in the ENU register provide a loopback feature for diagnostic testing of the UART. When these bits are set to one, the following occur: The receiver input pin (RDX) is internally connected to the transmitter output pin (TDX); the output of the Transmitter Shift Register is "looped back" into the Receive Shift Register input. In this mode, data that is transmitted is immediately received. This feature allows the processor to verify the transmit and receive data paths of the UART.

Note that the framing format for this mode is the nine bit format; one Start bit, nine data bits, and 7/8, one or two Stop bits. Parity is not generated or verified in this mode.

Attention Mode

The UART Receiver section supports an alternate mode of operation, referred to as ATTENTION Mode. This mode of operation is selected by the ATTN bit in the ENUR register. The data format for transmission must also be selected as having nine Data bits and either 7/8, one or two Stop bits.

The ATTENTION mode of operation is intended for use in networking the device with other processors. Typically in such environments the messages consists of device addresses, indicating which of several destinations should receive them, and the actual data. This Mode supports a scheme in which addresses are flagged by having the ninth bit of the data field set to a 1. If the ninth bit is reset to a zero the byte is a Data byte.

While in ATTENTION mode, the UART monitors the communication flow, but ignores all characters until an address character is received. Upon receiving an address character, the UART signals that the character is ready by setting the RBFL flag, which in turn interrupts the processor if UART Receiver interrupts are enabled. The ATTN bit is also cleared automatically at this point, so that data characters as well as address characters are recognized. Software examines the contents of the RBUF and responds by deciding either to accept the subsequent data stream (by leaving the ATTN bit reset) or to wait until the next address character is seen (by setting the ATTN bit again).

Operation of the UART Transmitter is not affected by selection of this Mode. The value of the ninth bit to be transmitted is programmed by setting XBIT9 appropriately. The value of the ninth bit received is obtained by reading RBIT9. Since this bit is located in ENUR register where the error flags reside, a bit operation on it will reset the error flags.

Comparators

The device contains two differential comparators, each with a pair of inputs (positive and negative) and an output. Ports I1–I3 and I4–I6 are used for the comparators. The following is the Port I assignment:

- I1 Comparator1 negative input
- I2 Comparator1 positive input
- I3 Comparator1 output
- I4 Comparator2 negative input
- I5 Comparator2 positive input
- I6 Comparator2 output

A Comparator Select Register (CMPSL) is used to enable the comparators, read the outputs of the comparators internally, and enable the outputs of the comparators to the pins. Two control bits (enable and output enable) and one result bit are associated with each comparator. The comparator result bits (CMP1RD and CMP2RD) are read only bits which will read as zero if the associated comparator is not enabled. The Comparator Select Register is cleared with reset, resulting in the comparators being disabled. The comparators should also be disabled before entering either the HALT or IDLE modes in order to save power. The configuration of the CMPSL register is as follows:

Comparators (Continued)

CMPSL REGISTER (ADDRESS X'00B7)

The CMPSL register contains the following bits:

CMP1EN Enable comparator 1

CMP1RD Comparator 1 result (this is a read only bit, which will read as 0 if the comparator is not enabled)

CMP10E Selects pin I3 as comparator 1 output provided that CMPIEN is set to enable the comparator

CMP2EN Enable comparator 2

CMP2RD Comparator 2 result (this is a read only bit, which will read as 0 if the comparator is not enabled)

CMP20E Selects pin I6 as comparator 2 output provided that CMP2EN is set to enable the comparator

Unused	CMP20E	CMP2RD	CMP2EN	CMP10E	CMP1RD	CMP1EN	Unused
Bit 7							Bit 0

Note that the two unused bits of CMPSL may be used as software flags.

Comparator outputs have the same spec as Ports L and G except that the rise and fall times are symmetrical.

Interrupts

The device supports a vectored interrupt scheme. It supports a total of fourteen interrupt sources. The following table lists all the possible interrupt sources, their arbitration ranking and the memory locations reserved for the interrupt vector for each source.

Two bytes of program memory space are reserved for each interrupt source. All interrupt sources except the software interrupt are maskable. Each of the maskable interrupts have an Enable bit and one or more Pending bits. A maskable interrupt is active if its associated enable and pending bits are set. If GIE = 1 and an interrupt is active, then the processor will be interrupted as soon as it is ready to start executing an instruction except if the above conditions happen during the Software Trap service routine. This exception is described in the Software Trap sub-section.

The interruption process is accomplished with the INTR instruction (opcode 00), which is jammed inside the Instruction Register and replaces the opcode about to be executed. The following steps are performed for every interrupt:

1. The GIE (Global Interrupt Enable) bit is reset.
2. The address of the instruction about to be executed is pushed into the stack.
3. The PC (Program Counter) branches to address 00FF.

This procedure takes 7 t_C cycles to execute.

Arbitration Ranking	Source	Description	Vector* Address Hi-Low Byte
(1) Highest	Software	INTR Instruction	0yFE–0yFF
(2)	Reserved		0yFC–0yFD
(3)	External	G0	0yFA–0yFB
(4)	Timer T0	Underflow	0yF8–0yF9
(5)	Timer T1	T1A/Underflow	0yF6–0yF7
(6)	Timer T1	T1B	0yF4–0yF5
(7)	MICROWIRE/PLUS	BUSY Low	0yF2–0yF3
(8)	Reserved		0yF0–0yF1
(9)	UART	Receive	0yEE–0yEF
(10)	UART	Transmit	0yEC–0yED
(11)	Timer T2	T2A/Underflow	0yEA–0yEB
(12)	Timer T2	T2B	0yE8–0yE9
(13)	Timer T3	T3A/Underflow	0yE6–0yE7
(14)	Timer T3	T3B	0yE4–0yE5
(15)	Port L/Wakeup	Port L Edge	0yE2–0yE3
(16) Lowest	Default VIS	Reserved	0yE0–0yE1

*y is a variable which represents the VIS block. VIS and the vector table must be located in the same 256-byte block except if VIS is located at the last address of a block. In this case, the table must be in the next block.

Interrupts (Continued)

At this time, since GIE = 0, other maskable interrupts are disabled. The user is now free to do whatever context switching is required by saving the context of the machine in the stack with PUSH instructions. The user would then program a VIS (Vector Interrupt Select) instruction in order to branch to the interrupt service routine of the highest priority interrupt enabled and pending at the time of the VIS. Note that this is not necessarily the interrupt that caused the branch to address location 00FF Hex prior to the context switching.

Thus, if an interrupt with a higher rank than the one which caused the interruption becomes active before the decision of which interrupt to service is made by the VIS, then the interrupt with the higher rank will override any lower ones and will be acknowledged. The lower priority interrupt(s) are still pending, however, and will cause another interrupt immediately following the completion of the interrupt service routine associated with the higher priority interrupt just serviced. This lower priority interrupt will occur immediately following the RETI (Return from Interrupt) instruction at the end of the interrupt service routine just completed.

Inside the interrupt service routine, the associated pending bit has to be cleared by software. The RETI (Return from Interrupt) instruction at the end of the interrupt service routine will set the GIE (Global Interrupt Enable) bit, allowing the processor to be interrupted again if another interrupt is active and pending.

The VIS instruction looks at all the active interrupts at the time it is executed and performs an indirect jump to the beginning of the service routine of the one with the highest rank.

The addresses of the different interrupt service routines, called vectors, are chosen by the user and stored in ROM in a table starting at 01E0 (assuming that VIS is located between 00FF and 01DF). The vectors are 15-bit wide and therefore occupy 2 ROM locations.

VIS and the vector table must be located in the same 256-byte block (0y00 to 0yFF) except if VIS is located at the last address of a block. In this case, the table must be in the next block. The vector table cannot be inserted in the first 256-byte block (y ≠ 0).

The vector of the maskable interrupt with the lowest rank is located at 0yE0 (Hi-Order byte) and 0yE1 (Lo-Order byte) and so forth in increasing rank number. The vector of the maskable interrupt with the highest rank is located at 0yFA (Hi-Order byte) and 0yFB (Lo-Order byte).

The Software Trap has the highest rank and its vector is located at 0yFE and 0yFF.

If, by accident, a VIS gets executed and no interrupt is active, then the PC (Program Counter) will branch to a vector located at 0yE0–0yE1.

WARNING

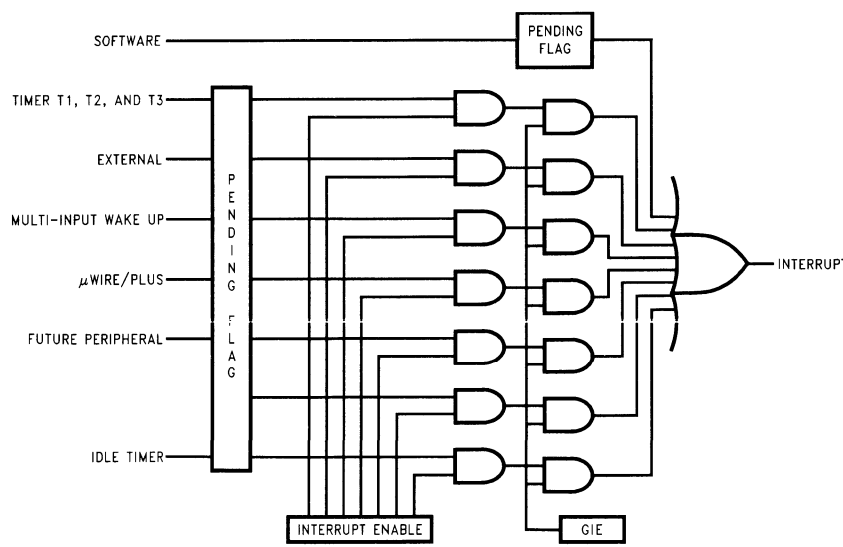
A Default VIS interrupt handler routine must be present. As a minimum, this handler should confirm that the GIE bit is cleared (this indicates that the interrupt sequence has been taken), take care of any required housekeeping, restore context and return. Some sort of Warm Restart procedure should be implemented. These events can occur without any error on the part of the system designer or programmer.

Note: There is always the possibility of an interrupt occurring during an instruction which is attempting to reset the GIE bit or any other interrupt enable bit. If this occurs when a single cycle instruction is being used to reset the interrupt enable bit, the interrupt enable bit will be reset but an interrupt may still occur. This is because interrupt processing is started at the same time as the interrupt bit is being reset. To avoid this scenario, the user should always use a two, three, or four cycle instruction to reset interrupt enable bits.

Figure 16 shows the Interrupt block diagram.

SOFTWARE TRAP

The Software Trap (ST) is a special kind of non-maskable interrupt which occurs when the INTR instruction (used to acknowledge interrupts) is fetched from ROM and placed inside the instruction register. This may happen when the PC is pointing beyond the available ROM address space or when the stack is over-popped.



TL/DD/12823–18

FIGURE 16. Interrupt Block Diagram

Interrupts (Continued)

When an ST occurs, the user can re-initialize the stack pointer and do a recovery procedure (similar to reset, but not necessarily containing all of the same initialization procedures) before restarting.

The occurrence of an ST is latched into the ST pending bit. The GIE bit is not affected and the ST pending bit **(not accessible by the user)** is used to inhibit other interrupts and to direct the program to the ST service routine with the VIS instruction. The RPND instruction is used to clear the software interrupt pending bit. This pending bit is also cleared on reset.

The ST has the highest rank among all interrupts.

Nothing (except another ST) can interrupt an ST being serviced.

WATCHDOG

The device contains a WATCHDOG and clock monitor. The WATCHDOG is designed to detect the user program getting stuck in infinite loops resulting in loss of program control or "runaway" programs. The Clock Monitor is used to detect the absence of a clock or a very slow clock below a specified rate on the CKI pin.

The WATCHDOG consists of two independent logic blocks: WD UPPER and WD LOWER. WD UPPER establishes the upper limit on the service window and WD LOWER defines the lower limit of the service window.

Servicing the WATCHDOG consists of writing a specific value to a WATCHDOG Service Register named WDSVR which is memory mapped in the RAM. This value is composed of three fields, consisting of a 2-bit Window Select, a 5-bit Key Data field, and the 1-bit Clock Monitor Select field. Table V shows the WDSVR register.

TABLE V. WATCHDOG Service Register (WDSVR)

Window Select		Key Data					Clock Monitor
X	X	0	1	1	0	0	Y
7	6	5	4	3	2	1	0

The lower limit of the service window is fixed at 2048 instruction cycles. Bits 7 and 6 of the WDSVR register allow the user to pick an upper limit of the service window.

Table VI shows the four possible combinations of lower and upper limits for the WATCHDOG service window. This flexibility in choosing the WATCHDOG service window prevents any undue burden on the user software.

Bits 5, 4, 3, 2 and 1 of the WDSVR register represent the 5-bit Key Data field. The key data is fixed at 01100. Bit 0 of the WDSVR Register is the Clock Monitor Select bit

TABLE VI. WATCHDOG Service Window Select

WDSVR Bit 7	WDSVR Bit 6	Service Window (Lower-Upper Limits)
0	0	2k–8k t_C Cycles
0	1	2k–16k t_C Cycles
1	0	2k–32k t_C Cycles
1	1	2k–64k t_C Cycles

Clock Monitor

The Clock Monitor aboard the device can be selected or deselected under program control. The Clock Monitor is guaranteed not to reject the clock if the instruction cycle clock ($1/t_C$) is greater or equal to 10 kHz. This equates to a clock input rate on CKI of greater or equal to 100 kHz.

WATCHDOG Operation

The WATCHDOG and Clock Monitor are disabled during reset. The device comes out of reset with the WATCHDOG armed, the WATCHDOG Window Select bits (bits 6, 7 of the WDSVR Register) set, and the Clock Monitor bit (bit 0 of the WDSVR Register) enabled. Thus, a Clock Monitor error will occur after coming out of reset, if the instruction cycle clock frequency has not reached a minimum specified value, including the case where the oscillator fails to start.

The WDSVR register can be written to only once after reset and the key data (bits 5 through 1 of the WDSVR Register) must match to be a valid write. This write to the WDSVR register involves two irrevocable choices: (i) the selection of the WATCHDOG service window (ii) enabling or disabling of the Clock Monitor. Hence, the first write to WDSVR Register involves selecting or deselecting the Clock Monitor, select the WATCHDOG service window and match the WATCHDOG key data. Subsequent writes to the WDSVR register will compare the value being written by the user to the WATCHDOG service window value and the key data (bits 7 through 1) in the WDSVR Register. Table VII shows the sequence of events that can occur.

The user must service the WATCHDOG at least once before the upper limit of the service window expires. The WATCHDOG may not be serviced more than once in every lower limit of the service window. The user may service the WATCHDOG as many times as wished in the time period between the lower and upper limits of the service window. The first write to the WDSVR Register is also counted as a WATCHDOG service.

The WATCHDOG has an output pin associated with it. This is the WDOUT pin, on pin 1 of the port G. WDOUT is active low. The WDOUT pin is in the high impedance state in the inactive state. Upon triggering the WATCHDOG, the logic will pull the WDOUT (G1) pin low for an additional 16 t_C–32 t_C cycles after the signal level on WDOUT pin goes below the lower Schmitt trigger threshold. After this delay, the device will stop forcing the WDOUT output low.

The WATCHDOG service window will restart when the WDOUT pin goes high. It is recommended that the user tie the WDOUT pin back to V_{CC} through a resistor in order to pull WDOUT high.

A WATCHDOG service while the WDOUT signal is active will be ignored. The state of the WDOUT pin is not guaranteed on reset, but if it powers up low then the WATCHDOG will time out and WDOUT will enter high impedance state.

The Clock Monitor forces the G1 pin low upon detecting a clock frequency error. The Clock Monitor error will continue until the clock frequency has reached the minimum specified value, after which the G1 output will enter the high impedance TRI-STATE mode following 16 t_C–32 t_C clock cycles. The Clock Monitor generates a continual Clock Monitor error if the oscillator fails to start, or fails to reach the minimum specified frequency. The specification for the Clock Monitor is as follows:

$1/t_C > 10$ kHz—No clock rejection.

$1/t_C < 10$ Hz—Guaranteed clock rejection.

WATCHDOG Operation (Continued)

WATCHDOG AND CLOCK MONITOR SUMMARY

The following salient points regarding the WATCHDOG and CLOCK MONITOR should be noted:

- Both the WATCHDOG and CLOCK MONITOR detector circuits are inhibited during RESET.
- Following RESET, the WATCHDOG and CLOCK MONITOR are both enabled, with the WATCHDOG having he maximum service window selected.
- The WATCHDOG service window and CLOCK MONITOR enable/disable option can only be changed once, during the initial WATCHDOG service following RESET.
- The initial WATCHDOG service must match the key data value in the WATCHDOG Service register WDSVR in order to avoid a WATCHDOG error.
- Subsequent WATCHDOG services must match all three data fields in WDSVR in order to avoid WATCHDOG errors.
- The correct key data value cannot be read from the WATCHDOG Service register WDSVR. Any attempt to read this key data value of 01100 from WDSVR will read as key data value of all 0's.
- The WATCHDOG detector circuit is inhibited during both the HALT and IDLE modes.
- The CLOCK MONITOR detector circuit is active during both the HALT and IDLE modes. Consequently, the device inadvertently entering the HALT mode will be detected as a CLOCK MONITOR error (provided that the CLOCK MONITOR enable option has been selected by the program).
- With the single-pin R/C oscillator mask option selected and the CLKDLY bit reset, the WATCHDOG service window will resume following HALT mode from where it left off before entering the HALT mode.
- With the crystal oscillator mask option selected, or with the single-pin R/C oscillator mask option selected and the CLKDLY bit set, the WATCHDOG service window will be set to its selected value from WDSVR following HALT. Consequently, the WATCHDOG should not be serviced for at least 2048 instruction cycles following HALT, but must be serviced within the selected window to avoid a WATCHDOG error.
- The IDLE timer T0 is not initialized with RESET.
- The user can sync in to the IDLE counter cycle with an IDLE counter (T0) interrupt or by monitoring the T0PND flag. The T0PND flag is set whenever the thirteenth bit of the IDLE counter toggles (every 4096 instruction cycles). The user is responsible for resetting the T0PND flag.
- A hardware WATCHDOG service occurs just as the device exits the IDLE mode. Consequently, the WATCHDOG should not be serviced for at least 2048 instruction cycles following IDLE, but must be serviced within the selected window to avoid a WATCHDOG error.
- Following RESET, the initial WATCHDOG service (where the service window and the CLOCK MONITOR enable/disable must be selected) may be programmed anywhere within the maximum service window (65,536 instruction cycles) initialized by RESET. Note that this initial WATCHDOG service may be programmed within the initial 2048 instruction cycles without causing a WATCHDOG error.

Detection of Illegal Conditions

The device can detect various illegal conditions resulting from coding errors, transient noise, power supply voltage drops, runaway programs, etc.

Reading of undefined ROM gets zeros. The opcode for software interrupt is zero. If the program fetches instructions from undefined ROM, this will force a software interrupt, thus signaling that an illegal condition has occurred.

The subroutine stack grows down for each call (jump to subroutine), interrupt, or PUSH, and grows up for each return or POP. The stack pointer is initialized to RAM location 06F Hex during reset. Consequently, if there are more returns than calls, the stack pointer will point to addresses 070 and 071 Hex (which are undefined RAM). Undefined RAM from addresses 070 to 07F (Segment 0), and all other segments (i.e., Segments 4 . . . etc.) is read as all 1's, which in turn will cause the program to return to address 7FFF Hex. This is an undefined ROM location and the instruction fetched (all 0's) from this location will generate a software interrupt signaling an illegal condition.

Thus, the chip can detect the following illegal conditions:

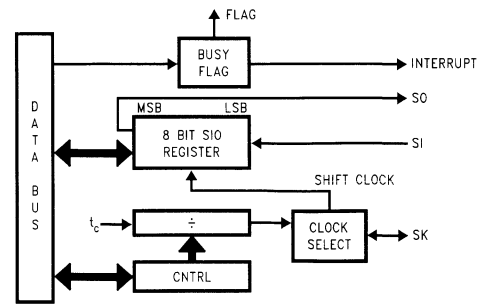
a. Executing from undefined ROM

b. Over "POP"ing the stack by having more returns than calls.

When the software interrupt occurs, the user can re-initialize the stack pointer and do a recovery procedure before restarting (this recovery program is probably similar to that following reset, but might not contain the same program initialization procedures). The recovery program should reset the software interrupt pending bit using the RPND instruction.

MICROWIRE/PLUS

MICROWIRE/PLUS is a serial synchronous communications interface. The MICROWIRE/PLUS capability enables the device to interface with any of National Semiconductor's MICROWIRE peripherals (i.e. A/D converters, display drivers, E^{2}PROMs etc.) and with other microcontrollers which support the MICROWIRE interface. It consists of an 8-bit serial shift register (SIO) with serial data input (SI), serial data output (SO) and serial shift clock (SK). *Figure 17* shows a block diagram of the MICROWIRE/PLUS logic.



TL/DD/12823–19

FIGURE 17. MICROWIRE/PLUS Block Diagram

The shift clock can be selected from either an internal source or an external source. Operating the MICROWIRE/PLUS arrangement with the internal clock source is called the Master mode of operation. Similarly, operating the MICROWIRE/PLUS arrangement with an external shift clock is called the Slave mode of operation.

The CNTRL register is used to configure and control the MICROWIRE/PLUS mode. To use the MICROWIRE/PLUS, the MSEL bit in the CNTRL register is set to one. In the master mode, the SK clock rate is selected by the two bits, SL0 and SL1, in the CNTRL register. Table VIII details the different clock rates that may be selected.

TABLE VII. WATCHDOG Service Actions

Key Data	Window Data	Clock Monitor	Action
Match	Match	Match	Valid Service: Restart Service Window
Don't Care	Mismatch	Don't Care	Error: Generate WATCHDOG Output
Mismatch	Don't Care	Don't Care	Error: Generate WATCHDOG Output
Don't Care	Don't Care	Mismatch	Error: Generate WATCHDOG Output

TABLE VIII. MICROWIRE/PLUS Master Mode Clock Select

SL1	SL0	SK
0	0	$2 \times t_c$
0	1	$4 \times t_c$
1	x	$8 \times t_c$

Where t_c is the instruction cycle clock

5

MICROWIRE/PLUS (Continued)

MICROWIRE/PLUS OPERATION

Setting the BUSY bit in the PSW register causes the MICROWIRE/PLUS to start shifting the data. It gets reset when eight data bits have been shifted. The user may reset the BUSY bit by software to allow less than 8 bits to shift. If enabled, an interrupt is generated when eight data bits have been shifted. The device may enter the MICROWIRE/PLUS mode either as a Master or as a Slave. *Figure 18* shows how two microcontroller devices and several peripherals may be interconnected using the MICROWIRE/PLUS arrangements.

Warning:

The SIO register should only be loaded when the SK clock is low. Loading the SIO register while the SK clock is high will result in undefined data in the SIO register. SK clock is normally low when not shifting.

Setting the BUSY flag when the input SK clock is high in the MICROWIRE/PLUS slave mode may cause the current SK clock for the SIO shift register to be narrow. For safety, the BUSY flag should only be set when the input SK clock is low.

MICROWIRE/PLUS Master Mode Operation

In the MICROWIRE/PLUS Master mode of operation the shift clock (SK) is generated internally by the device. The MICROWIRE Master always initiates all data exchanges. The MSEL bit in the CNTRL register must be set to enable the SO and SK functions onto the G Port. The SO and SK pins must also be selected as outputs by setting appropriate bits in the Port G configuration register. Table IX summarizes the bit settings required for Master mode of operation.

MICROWIRE/PLUS Slave Mode Operation

In the MICROWIRE/PLUS Slave mode of operation the SK clock is generated by an external source. Setting the MSEL bit in the CNTRL register enables the SO and SK functions onto the G Port. The SK pin must be selected as an input and the SO pin is selected as an output pin by setting and resetting the appropriate bits in the Port G configuration register. Table IX summarizes the settings required to enter the Slave mode of operation.

The user must set the BUSY flag immediately upon entering the Slave mode. This will ensure that all data bits sent by the Master will be shifted properly. After eight clock pulses the BUSY flag will be cleared and the sequence may be repeated.

Alternate SK Phase Operation

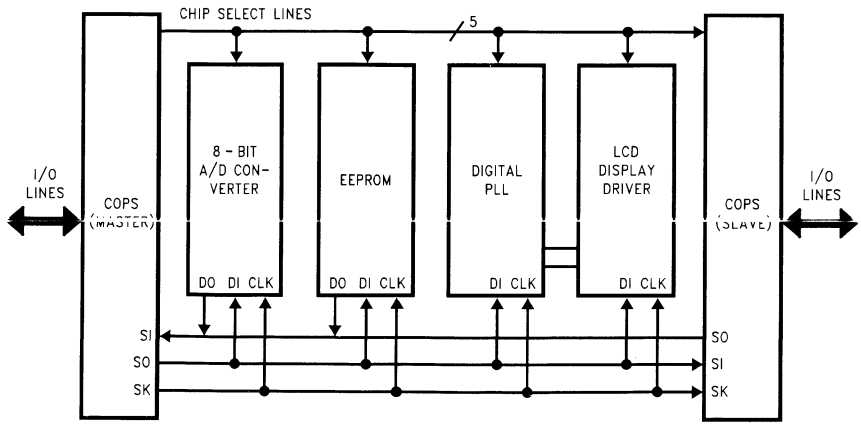
The device allows either the normal SK clock or an alternate phase SK clock to shift data in and out of the SIO register. In both the modes the SK is normally low. In the normal mode data is shifted in on the rising edge of the SK clock and the data is shifted out on the falling edge of the SK clock. The SIO register is shifted on each falling edge of the SK clock. In the alternate SK phase operation, data is shifted in on the falling edge of the SK clock and shifted out on the rising edge of the SK clock.

A control flag, SKSEL, allows either the normal SK clock or the alternate SK clock to be selected. Resetting SKSEL causes the MICROWIRE/PLUS logic to be clocked from the normal SK signal. Setting the SKSEL flag selects the alternate SK clock. The SKSEL is mapped into the G6 configuration bit. The SKSEL flag will power up in the reset condition, selecting the normal SK signal.

TABLE IX

This table assumes that the control flag MSEL is set.

G4 (SO) Config. Bit	G5 (SK) Config. Bit	G4 Fun.	G5 Fun.	Operation
1	1	SO	Int. SK	MICROWIRE/PLUS Master
0	1	TRI-STATE	Int. SK	MICROWIRE/PLUS Master
1	0	SO	Ext. SK	MICROWIRE/PLUS Slave
0	0	TRI-STATE	Ext. SK	MICROWIRE/PLUS Slave



TL/DD/12823–20

FIGURE 18. MICROWIRE/PLUS Application

Memory Map

All RAM, ports and registers (except A and PC) are mapped into data memory address space.

Address S/ADD REG	Contents
0000 to 006F	On-Chip RAM bytes (112 bytes)
0070 to 007F	Unused RAM Address Space (Reads As All Ones)
xx80 to xxAF	Unused RAM Address Space (Reads Undefined Data)
xxB0	Timer T3 Lower Byte
XXB1	Timer T3 Upper Byte
xxB2	Timer T3 Autoload Register T3RA Lower Byte
xxB3	Timer T3 Autoload Register T3RA Upper Byte
xxB4	Timer T3 Autoload Register T3RB Lower Byte
xxB5	Timer T3 Autoload Register T3RB Upper Byte
xxB6	Timer T3 Control Register
xxB7	Comparator Select Register (CMPSL)
xxB8	UART Transmit Buffer (TBUF)
xxB9	UART Receive Buffer (RBUF)
xxBA	UART Control and Status Register (ENU)
xxBB	UART Receive Control and Status Register (ENUR)
xxBC	UART Interrupt and Clock Source Register (ENUI)
xxBD	UART Baud Register (BAUD)
xxBE	UART Prescale Select Register (PSR)
xxBF	Reserved for UART
xxC0	Timer T2 Lower Byte
xxC1	Timer T2 Upper Byte
xxC2	Timer T2 Autoload Register T2RA Lower Byte
xxC3	Timer T2 Autoload Register T2RA Upper Byte
xxC4	Timer T2 Autoload Register T2RB Lower Byte
xxC5	Timer T2 Autoload Register T2RB Upper Byte
xxC6	Timer T2 Control Register
xxC7	WATCHDOG Service Register (Reg:WDSVR)
xxC8	MIWU Edge Select Register (Reg:WKEDG)
xxC9	MIWU Enable Register (Reg:WKEN)
xxCA	MIWU Pending Register (Reg:WKPND)
xxCB	Reserved
xxCC	Reserved
xxCD to xxCF	Reserved

Address S/ADD REG	Contents
xxD0	Port L Data Register
xxD1	Port L Configuration Register
xxD2	Port L Input Pins (Read Only)
xxD3	Reserved for Port L
xxD4	Port G Data Register
xxD5	Port G Configuration Register
xxD6	Port G Input Pins (Read Only)
xxD7	Port I Input Pins (Read Only)
xxD8	Port C Data Register
xxD9	Port C Configuration Register
xxDA	Port C Input Pins (Read Only)
xxDB	Reserved for Port C
xxDC	Port D
xxDD to xxDF	Reserved for Port D
xxE0 to xxE5	Reserved for EE Control Registers
xxE6	Timer T1 Autoload Register T1RB Lower Byte
xxE7	Timer T1 Autoload Register T1RB Upper Byte
xxE8	ICNTRL Register
xxE9	MICROWIRE/PLUS Shift Register
xxEA	Timer T1 Lower Byte
xxEB	Timer T1 Upper Byte
xxEC	Timer T1 Autoload Register T1RA Lower Byte
xxED	Timer T1 Autoload Register T1RA Upper Byte
xxEE	CNTRL Control Register
xxEF	PSW Register
xxF0 to FB	On-Chip RAM Mapped as Registers
xxFC	X Register
xxFD	SP Register
xxFE	B Register
xxFF	S Register
0100 to 017F	On-Chip 128 RAM Bytes
0200 to 027F	On-Chip 128 RAM Bytes
0300 to 037F	On-Chip 128 RAM Bytes

Reading memory locations 0070H–007FH (Segment 0) will return all ones. Reading unused memory locations 0080H–00AFH (Segment 0) will return undefined data. Reading memory locations from other Segments (i.e., Segment 4, Segment 5, ... etc.) will return all ones.

Addressing Modes

There are ten addressing modes, six for operand addressing and four for transfer of control.

OPERAND ADDRESSING MODES

Register Indirect

This is the "normal" addressing mode. The operand is the data memory addressed by the B pointer or X pointer.

Register Indirect (with auto post increment or decrement of pointer)

This addressing mode is used with the LD and X instructions. The operand is the data memory addressed by the B pointer or X pointer. This is a register indirect mode that automatically post increments or decrements the B or X register after executing the instruction.

Direct

The instruction contains an 8-bit address field that directly points to the data memory for the operand.

Immediate

The instruction contains an 8-bit immediate field as the operand.

Short Immediate

This addressing mode is used with the Load B Immediate instruction. The instruction contains a 4-bit immediate field as the operand.

Indirect

This addressing mode is used with the LAID instruction. The contents of the accumulator are used as a partial address (lower 8 bits of PC) for accessing a data operand from the program memory.

TRANSFER OF CONTROL ADDRESSING MODES

Relative

This mode is used for the JP instruction, with the instruction field being added to the program counter to get the new program location. JP has a range from −31 to +32 to allow a 1-byte relative jump (JP + 1 is implemented by a NOP instruction). There are no "pages" when using JP, since all 15 bits of PC are used.

Absolute

This mode is used with the JMP and JSR instructions, with the instruction field of 12 bits replacing the lower 12 bits of the program counter (PC). This allows jumping to any location in the current 4k program memory segment.

Absolute Long

This mode is used with the JMPL and JSRL instructions, with the instruction field of 15 bits replacing the entire 15 bits of the program counter (PC). This allows jumping to any location up to 32k in the program memory space.

Indirect

This mode is used with the JID instruction. The contents of the accumulator are used as a partial address (lower 8 bits of PC) for accessing a location in the program memory. The contents of this program memory location serve as a partial address (lower 8 bits of PC) for the jump to the next instruction.

Note: The VIS is a special case of the Indirect Transfer of Control addressing mode, where the double byte vector associated with the interrupt is transferred from adjacent addresses in the program memory into the program counter (PC) in order to jump to the associated interrupt service routine.

Instruction Set

Register and Symbol Definition

Registers	
A	8-Bit Accumulator Register
B	8-Bit Address Register
X	8-Bit Address Register
S	8-Bit Segment Register
SP	8-Bit Stack Pointer Register
PC	15-Bit Program Counter Register
PU	Upper 7 Bits of PC
PL	Lower 8 Bits of PC
C	1 Bit of PSW Register for Carry
HC	1 Bit of PSW Register for Half Carry
GIE	1 Bit of PSW Register for Global Interrupt Enable
VU	Interrupt Vector Upper Byte
VL	Interrupt Vector Lower Byte

Symbols	
[B]	Memory Indirectly Addressed by B Register
[X]	Memory Indirectly Addressed by X Register
MD	Direct Addressed Memory
Mem	Direct Addressed Memory or [B]
Meml	Direct Addressed Memory or [B] or Immediate Data
Imm	8-Bit Immediate Data
Reg	Register Memory: Addresses F0 to FF (Includes B, X and SP)
Bit	Bit Number (0 to 7)
←	Loaded with
⟷	Exchanged with

Instruction Set (Continued)

INSTRUCTION SET

ADD	A,Meml	ADD	A ← A + Meml
ADC	A,Meml	ADD with Carry	A ← A + Meml + C, C ← Carry HC ← Half Carry
SUBC	A,Meml	Subtract with Carry	A ← A − $\overline{\text{Meml}}$ + C, C ← Carry HC ← Half Carry
AND	A,Meml	Logical AND	A ← A and Meml
ANDSZ	A,Imm	Logical AND Immed., Skip if Zero	Skip next if (A and Imm) = 0
OR	A,Meml	Logical OR	A ← A or Meml
XOR	A,Meml	Logical EXclusive OR	A ← A xor Meml
IFEQ	MD,Imm	IF EQual	Compare MD and Imm, Do next if MD = Imm
IFEQ	A,Meml	IF EQual	Compare A and Meml, Do next if A = Meml
IFNE	A,Meml	IF Not Equal	Compare A and Meml, Do next if A ≠ Meml
IFGT	A,Meml	IF Greater Than	Compare A and Meml, Do next if A > Meml
IFBNE	#	If B Not Equal	Do next if lower 4 bits of B ≠ Imm
DRSZ	Reg	Decrement Reg., Skip if Zero	Reg ← Reg − 1, Skip if Reg = 0
SBIT	#,Mem	Set BIT	1 to bit, Mem (bit = 0 to 7 immediate)
RBIT	#,Mem	Reset BIT	0 to bit, Mem
IFBIT	#,Mem	IF BIT	If bit in A or Mem is true do next instruction
RPND		Reset PeNDing Flag	Reset Software Interrupt Pending Flag
X	A,Mem	EXchange A with Memory	A ⟷ Mem
X	A,[X]	EXchange A with Memory [X]	A ⟷ [X]
LD	A,Meml	LoaD A with Memory	A ← Meml
LD	A,[X]	LoaD A with Memory [X]	A ← [X]
LD	B,Imm	LoaD B with Immed.	B ← Imm
LD	Mem,Imm	LoaD Memory Immed	Mem ← Imm
LD	Reg,Imm	LoaD Register Memory Immed.	Reg ← Imm
X	A, [B ±]	EXchange A with Memory [B]	A ⟷ [B], (B ← B ±1)
X	A, [X ±]	EXchange A with Memory [X]	A ⟷ [X], (X ← X ±1)
LD	A, [B±]	LoaD A with Memory [B]	A ← [B], (B ← B ±1)
LD	A, [X±]	LoaD A with Memory [X]	A ← [X], (X ← X±1)
LD	[B±],Imm	LoaD Memory [B] Immed.	[B] ← Imm, (B ← B±1)
CLR	A	CLeaR A	A ← 0
INC	A	INCrement A	A ← A + 1
DEC	A	DECrement A	A ← A − 1
LAID		Load A InDirect from ROM	A ← ROM (PU,A)
DCOR	A	Decimal CORrect A	A ← BCD correction of A (follows ADC, SUBC)
RRC	A	Rotate A Right thru C	C → A7 → . . . → A0 → C
RLC	A	Rotate A Left thru C	C ← A7 ← . . . ← A0 ← C
SWAP	A	SWAP nibbles of A	A7 . . . A4 ⟷ A3 . . . A0
SC		Set C	C ← 1, HC ← 1
RC		Reset C	C ← 0, HC ← 0
IFC		IF C	IF C is true, do next instruction
IFNC		IF Not C	If C is not true, do next instruction
POP	A	POP the stack into A	SP ← SP + 1, A ← [SP]
PUSH	A	PUSH A onto the stack	[SP] ← A, SP ← SP − 1
VIS		Vector to Interrupt Service Routine	PU ← [VU], PL ← [VL]
JMPL	Addr.	Jump absolute Long	PC ← ii (ii = 15 bits, 0 to 32k)
JMP	Addr.	Jump absolute	PC9 . . . 0 ← i (i = 12 bits)
JP	Disp.	Jump relative short	PC ← PC + r (r is −31 to +32, except 1)
JSRL	Addr.	Jump SubRoutine Long	[SP] ← PL, [SP−1] ← PU,SP−2, PC ← ii
JSR	Addr	Jump SubRoutine	[SP] ← PL, [SP−1] ← PU,SP−2, PC9 . . . 0 ← i
JID		Jump InDirect	PL ← ROM (PU,A)
RET		RETurn from subroutine	SP + 2, PL ← [SP], PU ← [SP−1]
RETSK		RETurn and SKip	SP + 2, PL ← [SP],PU ← [SP−1], skip next instruction
RETI		RETurn from Interrupt	SP + 2, PL ← [SP],PU ← [SP−1],GIE ← 1
INTR		Generate an Interrupt	[SP] ← PL, [SP−1] ← PU, SP−2, PC ← 0FF
NOP		No OPeration	PC ← PC + 1

Instruction Execution Time

Most instructions are single byte (with immediate addressing mode instructions taking two bytes).

Most single byte instructions take one cycle time to execute.

Skipped instructions require x number of cycles to be skipped, where x equals the number of bytes in the skipped instruction opcode.

See the BYTES and CYCLES per INSTRUCTION table for details.

Bytes and Cycles per Instruction

The following table shows the number of bytes and cycles for each instruction in the format of byte/cycle.

Arithmetic and Logic Instructions

	[B]	Direct	Immed.
ADD	1/1	3/4	2/2
ADC	1/1	3/4	2/2
SUBC	1/1	3/4	2/2
AND	1/1	3/4	2/2
OR	1/1	3/4	2/2
XOR	1/1	3/4	2/2
IFEQ	1/1	3/4	2/2
IFNE	1/1	3/4	2/2
IFGT	1/1	3/4	2/2
IFBNE	1/1		
DRSZ		1/3	
SBIT	1/1	3/4	
RBIT	1/1	3/4	
IFBIT	1/1	3/4	

RPND	1/1

Instructions Using A & C

CLRA	1/1
INCA	1/1
DECA	1/1
LAID	1/3
DCOR	1/1
RRCA	1/1
RLCA	1/1
SWAPA	1/1
SC	1/1
RC	1/1
IFC	1/1
IFNC	1/1
PUSHA	1/3
POPA	1/3
ANDSZ	2/2

Transfer of Control Instructions

JMPL	3/4
JMP	2/3
JP	1/3
JSRL	3/5
JSR	2/5
JID	1/3
VIS	1/5
RET	1/5
RETSK	1/5
RETI	1/5
INTR	1/7
NOP	1/1

Memory Transfer Instructions

	Register Indirect		Direct	Immed.	Register Indirect Auto Incr. & Decr.		
	[B]	[X]			[B+, B−]	[X+, X−]	
X A,*	1/1	1/3	2/3		1/2	1/3	
LD A,*	1/1	1/3	2/3	2/2	1/2	1/3	
LD B, Imm				1/1			(IF B < 16)
LD B, Imm				2/2			(IF B > 15)
LD Mem, Imm	2/2		3/3		2/2		
LD Reg, Imm			2/3				
IFEQ MD, Imm			3/3				

* = > Memory location addressed by B or X or directly.

Opcode Table

Upper Nibble																Lower Nibble
F	E	D	C	B	A	9	8	7	6	5	4	3	2	1	0	
JP−15	JP−31	LD 0F0, # i	DRSZ 0F0	RRCA	RC	ADC A, #i	ADC A,[B]	IFBIT 0,[B]	ANDSZ A, #i	LD B, #0F	IFBNE 0	JSR x000−x0FF	JMP x000−x0FF	JP+17	INTR	0
JP−14	JP−30	LD 0F1, # i	DRSZ 0F1	*	SC	SUBC A, #i	SUB A,[B]	IFBIT 1,[B]	*	LD B, #0E	IFBNE 1	JSR x100−x1FF	JMP x100−x1FF	JP+18	JP+2	1
JP−13	JP−29	LD 0F2, # i	DRSZ 0F2	X A,[X+]	X A,[B+]	IFEQ A, #i	IFEQ A,[B]	IFBIT 2,[B]	*	LD B, #0D	IFBNE 2	JSR x200−x2FF	JMP x200−x2FF	JP+19	JP+3	2
JP−12	JP−28	LD 0F3, # i	DRSZ 0F3	X A,[X−]	X A,[B−]	IFGT A, #i	IFGT A,[B]	IFBIT 3,[B]	*	LD B, #0C	IFBNE 3	JSR x300−x3FF	JMP x300−x3FF	JP+20	JP+4	3
JP−11	JP−27	LD 0F4, # i	DRSZ 0F4	VIS	LAID	ADD A, #i	ADD A,[B]	IFBIT 4,[B]	CLRA	LD B, #0B	IFBNE 4	JSR x400−x4FF	JMP x400−x4FF	JP+21	JP+5	4
JP−10	JP−26	LD 0F5, # i	DRSZ 0F5	RPND	JID	AND A, #i	AND A,[B]	IFBIT 5,[B]	SWAPA	LD B, #0A	IFBNE 5	JSR x500−x5FF	JMP x500−x5FF	JP+22	JP+6	5
JP−9	JP−25	LD 0F6, # i	DRSZ 0F6	X A,[X]	X A,[B]	XOR A, #i	XOR A,[B]	IFBIT 6,[B]	DCORA	LD B, #09	IFBNE 6	JSR x600−x6FF	JMP x600−x6FF	JP+23	JP+7	6
JP−8	JP−24	LD 0F7, # i	DRSZ 0F7	*	*	OR A, #i	OR A,[B]	IFBIT 7,[B]	PUSHA	LD B, #08	IFBNE 7	JSR x700−x7FF	JMP x700−x7FF	JP+24	JP+8	7
JP−7	JP−23	LD 0F8, # i	DRSZ 0F8	NOP	RLCA	LD A, #i	IFC	SBIT 0,[B]	RBIT 0,[B]	LD B, #07	IFBNE 8	JSR x800−x8FF	JMP x800−x8FF	JP+25	JP+9	8
JP−6	JP−22	LD 0F9, # i	DRSZ 0F9	IFNE A,[B]	IFEQ Md, #i	IFNE A, #i	IFNC	SBIT 1,[B]	RBIT 1,[B]	LD B, #06	IFBNE 9	JSR x900−x9FF	JMP x900−x9FF	JP+26	JP+10	9
JP−5	JP−21	LD 0FA, # i	DRSZ 0FA	LD A,[X+]	LD A,[B+]	LD [B+], #i	INCA	SBIT 2,[B]	RBIT 2,[B]	LD B, #05	IFBNE 0A	JSR xA00−xAFF	JMP xA00−xAFF	JP+27	JP+11	A
JP−4	JP−20	LD 0FB, # i	DRSZ 0FB	LD A,[X−]	LD A,[B−]	LD [B−], #i	DECA	SBIT 3,[B]	RBIT 3,[B]	LD B, #04	IFBNE 0B	JSR xB00−xBFF	JMP xB00−xBFF	JP+28	JP+12	B
JP−3	JP−19	LD 0FC, # i	DRSZ 0FC	LD Md, #i	JMPL	X A,Md	POPA	SBIT 4,[B]	RBIT 4,[B]	LD B, #03	IFBNE 0C	JSR xC00−xCFF	JMP xC00−xCFF	JP+29	JP+13	C
JP−2	JP−18	LD 0FD, # i	DRSZ 0FD	DIR	JSRL	LD A,Md	RETSK	SBIT 5,[B]	RBIT 5,[B]	LD B, #02	IFBNE 0D	JSR xD00−xDFF	JMP xD00−xDFF	JP+30	JP+14	D
JP−1	JP−17	LD 0FE, # i	DRSZ 0FE	LD A,[X]	LD A,[B]	LD [B], #i	RET	SBIT 6,[B]	RBIT 6,[B]	LD B, #01	IFBNE 0E	JSR xE00−xEFF	JMP xE00−xEFF	JP+31	JP+15	E
JP−0	JP−16	LD 0FF, # i	DRSZ 0FF	*	*	LD B, #i	RETI	SBIT 7,[B]	RBIT 7,[B]	LD B, #00	IFBNE 0F	JSR xF00−xFFF	JMP xF00−xFFF	JP+32	JP+16	F

Where,

i is the immediate data

Md is a directly addressed memory location

* is an unused opcode

Note: The opcode 60 Hex is also the opcode for IFBIT #i,A

Mask Options

The mask programmable options are shown below. The options are programmed at the same time as the ROM pattern submission.

OPTION 1: CLOCK CONFIGURATION

= 1 Crystal Oscillator (CKI/10)
G7 (CKO) is clock generator output to crystal/ resonator with CKI being the clock input

= 2 Single-pin RC controlled oscillator (CKI/10)
G7 is available as a HALT restart and/or general purpose input

OPTION 2: HALT

= 1 Enable HALT mode

= 2 Disable HALT mode

OPTION 3: BONDING

= 1 44-Pin PLCC

= 2 40-Pin DIP

The chip can be driven by a clock input on the CKI input pin which can be between DC and 10 MHz. The CKO output clock is on pin G7 (if clock option = 1 has been selected). The CKI input frequency is divided down by 10 to produce the instruction cycle clock (1/tc).

Development Support

Summary

- iceMASTER: IM-COP8/400—Full feature in-circuit emulation for all COP8 products. A full set of COP8 Basic and Feature Family device and package specific probes are available.
- COP8 Debug Module: Moderate cost in-circuit emulation and development programming unit.
- COP8 Evaluation and Programming Unit: EPU-COP888GG—low cost In-circuit simulation and development programming unit.
- Assembler: COP8-DEV-IBMA. A DOS installable cross development Assembler, Linker, Librarian and Utility Software Development Tool Kit.
- C Compiler: COP8C. A DOS installable cross development Software Tool Kit.
- OTP/EPROM Programmer Support: Covering needs from engineering prototype, pilot production to full production environments.
- In-factory Programming Support: Covering high volume production OTP programming with 2 week turn-on capability.

Development Support (Continued)

IceMASTER (IM) IN-CIRCUIT EMULATION

The iceMASTER IM-COP8/400 is a full feature, PC based, in-circuit emulation tool developed and marketed by MetaLink Corporation to support the whole COP8 family of products. National is a resale vendor for these products.

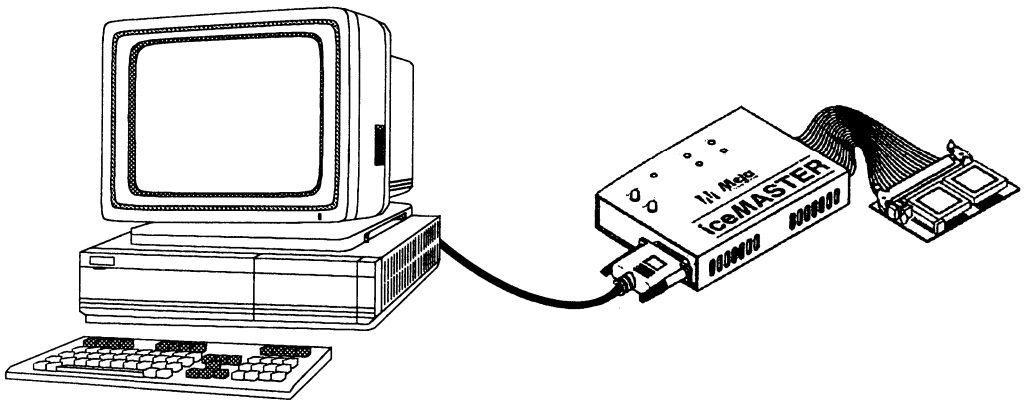
See *Figure 19* for configuration.

The iceMASTER IM-COP8/400 with its device specific COP8 Probe provides a rich feature set for developing, testing and maintaining product:

- Real-time in-circuit emulation; full 2.4V–5.5V operation range, full DC-10 MHz clock. Chip options are programmable or jumper selectable.
- Direct connection to application board by package compatible socket or surface mount assembly.
- Full 32 kbytes of loadable programming space that overlays (replaces) the on-chip ROM or EPROM. On-chip RAM and I/O blocks are used directly or recreated on the probe as necessary.
- Full 4K frame synchronous trace memory. Address, instruction, and 8 unspecified, circuit connectable trace lines. Display can be HLL source (e.g., C source), assembly or mixed.
- A full 64K hardware configurable break, trace on, trace off control, and pass count increment events.
- Tool set integrated interactive symbolic debugger—supports both assembler (COFF) and C Compiler (.COD) linked object formats.
- Real time peformance profiling analysis; selectable bucket definition.
- Watch windows, content updated automatically at each execution break.
- Instruction by instruction memory/register changes displayed on source window when in single step operation.
- Single base unit and debugger software reconfigurable to support the entire COP8 family; only the probe personality needs to change. Debugger software is processor customized, and reconfigured from a master model file.
- Processor specific symbolic display of registers and bit level assignments, configured from master model file.
- Halt/Idle mode notification.
- On-line HELP customized to specific processor using master model file.
- Includes a copy of COP8-DEV-IBMA assembler and linker SDK.

IM Order Information

Base Unit	
IM-COP8/400-1	iceMASTER base unit, 110V power supply
IM-COP8/400-2	iceMASTER base unit, 220V power supply
iceMASTER Probe	
MHW-888GG40DWPC MHW-888GG44PWPC	40 DIP 40 PLCC



TL/DD/12823–21

FIGURE 19. COP iceMASTER Environment

5

Development Support (Continued)

IceMASTER DEBUG MODULE (DM)

The iceMASTER Debug Module is a PC based, combination in-circuit emulation tool and COP8 based OTP/EPROM programming tool developed and marketed by MetaLink Corporation to support the whole COP8 family of products. National is a resale vendor for these products.

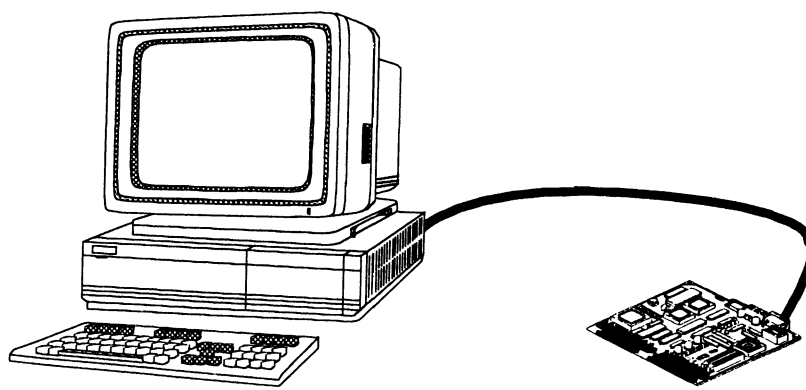
See *Figure 20* for configuration.

The iceMASTER Debug Module is a moderate cost development tool. It has the capability of in-circuit emulation for a specific COP8 microcontroller and in addition serves as a programming tool for COP8 OTP and EPROM product families. Summary of features is as follows:

- Real-time in-circuit emulation; full operating voltage range operation, full DC-10 MHz clock.
- All processor I/O pins can be cabled to an application development board with package compatible cable to socket and surface mount assembly.
- Full 32 kbytes of loadable programming space that overlays (replaces) the on-chip ROM or EPROM. On-chip RAM and I/O blocks are used directly or recreated as necessary.
- 100 frames of synchronous trace memory. The display can be HLL source (C source), assembly or mixed. The most recent history prior to a break is available in the trace memory.
- Configured break points; uses INTR instruction which is modestly intrusive.
- Software—only supported features are selectable.
- Tool set integrated interactive symbolic debugger—supports both
- Assembler (COFF) and C Compiler (.COD) SDK linked object formats.
- Instruction by instruction memory/register changes displayed when in single step operation.
- Debugger software is processor customized, and reconfigured from a master model file.
- Processor specific symbolic display of registers and bit level assignments, configured from master model file.
- Halt/Idle mode notification.
- Programming menu supports full product line of programmable OTP and EPROM COP8 products. Program data is taken directly from the overlay RAM.
- Programming of 44 PLCC and 68 PLCC parts requires external programming adapters.
- Includes wall-mount power supply
- On-board V_{PP} generator from 5V input or connection to external supply supported. Requires V_{PP} level adjustment per the family programming specification (correct level is provided on an on-screen pop-down display). On-line HELP customized to specific processor using master model file.
- On-line HELP customized to specific processor using master model file
- Includes a copy of COP8-DEV-IBMA assembler and linker SDK.

DM Order Information

Debug Module Unit	
COP8-DM/888GG	
Cable Adapters	
DM-COP8/40D DM-COP8/44P	40 DIP 44 PLCC



TL/DD/12823-22

FIGURE 20. COP8-DM Environment

Development Support (Continued)

IceMASTER EVALUATION PROGRAMMING UNIT (EPU)

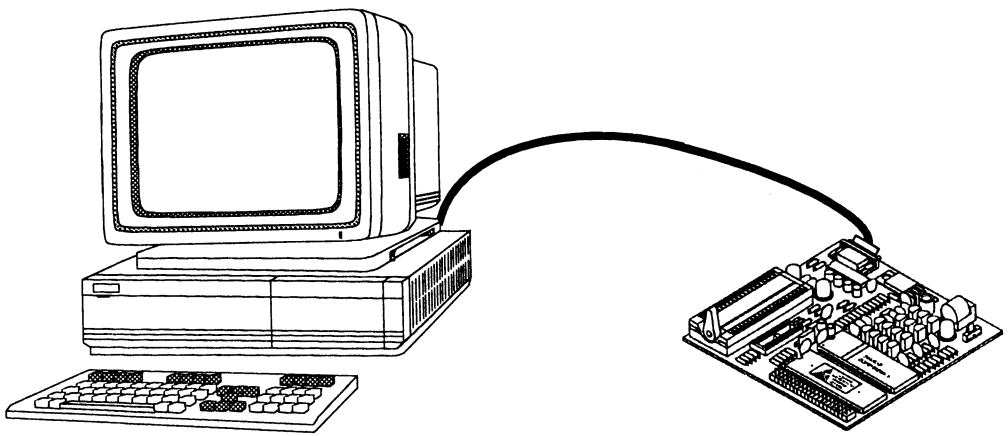
The iceMASTER EPU-COP888GG is a PC based, in-circuit. simulation tool to support the feature family COP8 products. See *Figure 21* for configuration.

The simulation capability is a very low cost means of evaluating the general COP8 architecture. In addition, the EPU has programming capability, with added adapters, for programming the whole COP8 product family of OTP and EPROM products. The product includes the following features:

- Non-real-time in-circuit simulation. Program overlay memory is PC resident; instructions are downloaded over RS-232 as executed. Approximate performance is 20 kHz.
- Includes a 40-pin DIP cable adapter. Other target packages are not supported. All processor I/O pins are cabled to the application development environment.
- Full 32 kbytes of loadable programming space that overlays (replaces) the on-chip ROM or EPROM. On-chip RAM and I/O blocks are used directly or recreated as necessary.
- On-chip timer and watch-dog execution are not well synchronized to the instruction simulation.
- 100 frames of synchronous trace memory. The display can be HLL source (e.g., C source), assembly or mixed. The most recent history prior to a break is available in the trace memory.
- Up to 8 software configured break points; uses INTR instruction which is modestly intrusive.
- Common look-feel debugger software across all MetaLink products—only supported features are selectable.
- Tool set integrated interactive symbolic debugger—supports both assembler (COFF) and C Compiler (.COD) SDK linked object formats.
- Instruction by instruction memory/register changes displayed when in single step operation.
- Processor specific symbolic display of registers and bit level assignments, configured from master model file.
- Halt/Idle mode notification. Restart requires special handling.
- Programming menu supports full product line of programmable OTP and EPROM COP8 products. Only a 40 ZIF socket is available on the EPU unit. Adapters are available for other part package configurations.
- Integral wall mount power supply provides 5V and develops the required V_{PP} to program parts.
- Includes a copy of COP8-DEV-IBMA assembler, linker SDK.

EPU Order Information

Evaluation Programming Unit	
EPU-COP888GG	Evaluation Programming Unit with debugger and programmer control software and 40 ZIF programming socket
General Programming Adapters	
COP8-PGMA-DS44P	28 & 20 DIP and SOIC plus 44 PLCC adapter



TL/DD/12823–23

FIGURE 21. CPU-COP8 Tool Environment

Development Support (Continued)

COP8 ASSEMBLER/LINKER SOFTWARE DEVELOPMENT TOOL KIT

National Semiconductor offers a relocateable COP8 macro cross assembler, linker, librarian and utility software development tool kit. Features are summarized as follows:

- Basic and Feature Family instruction set by "device" type.
- Nested macro capability.
- Extensive set of assembler directives.
- Supported on PC/DOS platform.
- Generates National standard COFF output files.
- Integrated Linker and Librarian.
- Integrated utilities to generate ROM code file outputs.
- DUMPCOFF utility.

This product is integrated as a part of MetaLink tools as a development kit, fully supported by the MetaLink debugger. It may be ordered separately or it is bundled with the MetaLink products at no additional cost.

Order Information

Assembler SDK	
COP8-DEV-IBMA	Assembler SDK on installable 3.5" PC/DOS Floppy Disk Drive format. Periodic upgrades and most recent version is available on National's BBS and Internet.

COP8 C COMPILER

A C Compiler is developed and marketed by Byte Craft Limited. The COP8C compiler is a fully integrated development tool specifically designed to support the compact embedded configuration of the COP8 family of products.

Features are summarized as follows:

- ANSI C with some restrictions and extensions that optimize development for the COP8 embedded application.
- BITS data type extension. Register declaration #pragma with direct bit level definitions.
- C language support for interrupt routines.
- Expert system, rule based code geration and optimization.
- Performs consistency checks against the architectural definitions of the target COP8 device.
- Generates program memory code.
- Supports linking of compiled object or COP8 assembled object formats.
- Global optimization of linked code.
- Symbolic debug load format fully source level supported by the MetaLink debugger.

Approved List

Manufacturer	North America	Europe	Asia
BP Microsystems	(800) 225-2102 (713) 688-4600 Fax: (713) 688-0920	+49-8152-4183 +49-8856-932616	+852-234-16611 +852-2710-8121
Data I/O	(800) 426-1045 (206) 881-6444 Fax: (206) 882-1043	+44-0734-440011	Call North America
HI-LO	(510) 623-8860	Call Asia	+886-2-764-0215 Fax: +886-2-756-6403
ICE Technology	(800) 624-8949 (919) 430-7915	+44-1226-767404 Fax: 0-1226-370-434	
MetaLink	(800) 638-2423 (602) 926-0797 Fax: (602) 693-0681	+49-80 9156 96-0 Fax: +49-80 9123 86	+852-737-1800
Systems General	(408) 263-6667	+41-1-9450300	+886-2-917-3005 Fax: +886-2-911-1283
Needhams	(916) 924-8037 Fax: (916) 924-8065		

Development Support (Continued)

SINGLE-CHIP OTP/EMULATOR SUPPORT

The COP8 family is supported by single-chip OTP emulators. For detailed information, refer to the emulator specific datasheet and the emulator selection table below:

OTP Emulator Ordering Information

Device Number	Clock Option	Package	Emulates
COP87L88GGV-XE	Crystal/ HALT En	44 PLCC	COP888GG
COP87L88GGN-XE	Crystal/ HALT En	40 DIP	COP888GG

INDUSTRY WIDE OTP/EPROM PROGRAMMING SUPPORT

Programming support, in addition to the MetaLink development tools, is provided by a full range of independent approved vendors to meet the needs from the engineering laboratory to full production.

AVAILABLE LITERATURE

For more information, please see the COP8 Basic Family User's Manual, Literature Number 620895, COP8 Feature Family User's Manual Literature Number 620897 and National's Family of 8-bit Microcontrollers COP8 Selection Guide, Literature Number 630009.

DIAL-A-HELPER

Dial-A-Helper is a service provided by the Microcontroller Applications group. The Dial-A-Helper is an Electronic Information System that may be accessed as a Bulletin Board System (BBS) via data modem, as an FTP site on the Internet via standard FTP client application or as an FTP site on the Internet using a standard Internet browser such as Netscape or Mosaic.

The Dial-A-Helper system provides access to an automated information storage and retrieval system. The system capabilities include a MESSAGE SECTION (electronic mail, when accessed as a BBS) for communication to and from the Microcontroller Applications Group and a FILE SECTION which consists of several file areas where valuable application software and utilities could be found.

DIAL-A-HELPER BBS via a Standard Modem

Modem: CANADA/U.S.: (800) NSC-MICRO
(800) 672-6427
EUROPE: (+49) 0-8141-351332
Baud: 14.4k
Set-up: Length: 8-Bit
Parity: None
Stop Bit: 1
Operation: 24 Hrs., 7 Days

DIAL-A-HELPER via FTP

ftp nscmicro.nsc.com
user: anonymous
password: username@yourhost.site.domain

DIAL-A-HELPER via a WorldWide Web Browser

ftp://nscmicro.nsc.com

National Semiconductor on the WorldWide Web

See us on the WorldWide Web at: http://www.national.com

CUSTOMER RESPONSE CENTER

Complete product information and technical support is available from National's customer response centers.

CANADA/US:	Tel:	(800) 272-9959
	email:	support @tevm2.nsc.com
EUROPE:	email:	europe.support@nsc.com
	Deutsch Tel:	+49 (0) 180-530 85 85
	English Tel:	+49 (0) 180-532 78 32
	Français Tel:	+49 (0) 180-532 93 58
	Italiano Tel:	+49 (0) 180-534 16 80
JAPAN:	Tel:	+81-043-299-2309
S.E. ASIA:	Beijing Tel:	(+86) 10-6856-8601
	Shanghai Tel:	(+86) 21-6415-4092
	Hong Kong Tel:	(+852) 2737-1600
	Korea Tel:	(+82) 2-3771-6909
	Malaysia Tel:	(+60-4) 644-9061
	Singapore Tel:	(+65) 255-2226
	Taiwan Tel:	+886-2-521-3288
AUSTRALIA:	Tel:	(+61) 3-9558-9999
INDIA:	Tel:	(+91) 80-559-9467

National Semiconductor

COP688HG/COP888HG
8-Bit Microcontroller with UART and Three Multi-Function Timers

General Description

The COP8™ feature family of microcontrollers uses an 8-bit single chip core architecture fabricated with National Semiconductor's M²CMOS™ process technology. The COP888HG is a member of this expandable 8-bit core processor family of microcontrollers. (Continued)

Key Features

- Full duplex UART
- Three 16-bit timers, each with two 16-bit registers supporting:
  - — Processor Independent PWM mode
  - — External Event counter mode
  - — Input Capture mode
- Quiet design (low radiated emissions)
- 20 kbytes on-board ROM
- 512 bytes on-board RAM

Additional Peripheral Features

- Idle Timer
- Multi-Input Wake-Up (MIWU) with optional interrupts (8)
- Two analog comparators
- WATCHDOG™ and Clock Monitor logic
- MICROWIRE/PLUS™ serial I/O

I/O Features

- Memory mapped I/O
- Software selectable I/O options (TRI-STATE® Output, Push-Pull Output, Weak Pull-Up Input, High Impedance Input)
- Schmitt trigger inputs on ports G and L
- Packages:
  - — 40 DIP with 35 I/O pins
  - — 44 PLCC with 39 I/O pins

CPU/Instruction Set Features

- 1 μs instruction cycle time
- Fourteen multi-source vectored interrupts servicing
  - — External Interrupt with selectable edge
  - — Idle Timer T0
  - — Three Timers (Each with 2 Interrupts)
  - — MICROWIRE/PLUS™
  - — Multi-Input Wake-Up
  - — Software Trap
  - — UART (2)
  - — Default VIS (default interrupt)
- Versatile and easy-to-use instruction set
- 8-bit Stack Pointer (SP) — stack in RAM
- Two 8-bit Register Indirect Data Memory Pointers (B and X)

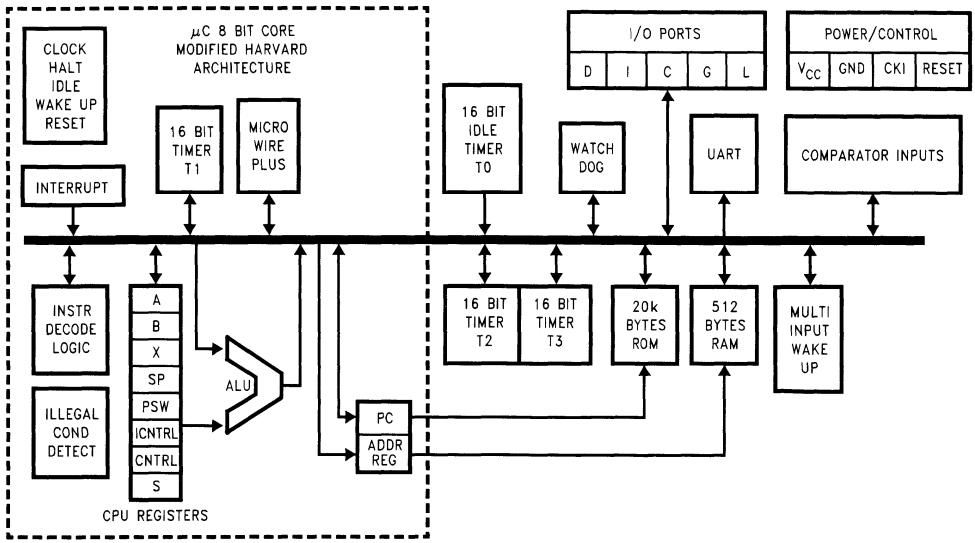
Fully Static CMOS

- Two power saving modes: HALT and IDLE
- Low current drain (typically <1 μA)
- Single supply operation: 2.5V to 5.5V
- Temperature ranges:
  $-40°C$ to $+85°C$, $-55°C$ to $+125°C$

Development Support

- Emulation and OTP devices
- Real time emulation and full program debug offered by MetaLink's Development System

Block Diagram



TL/DD/12858–1

FIGURE 1. COP888HG Block Diagram

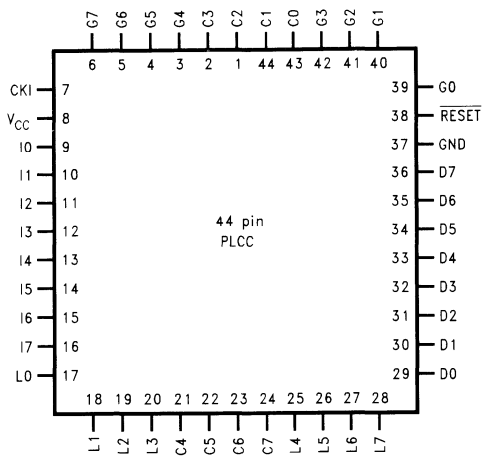
General Description (Continued)

It is a fully static part, fabricated using double-metal silicon gate microCMOS technology. Features include an 8-bit memory mapped architecture, MICROWIRE/PLUS serial I/O, three 16-bit timer/counters supporting three modes (Processor Independent PWM generation, External Event counter, and Input Capture mode capabilities), full duplex UART, two comparators, and two power saving modes (HALT and IDLE), both with a multi-sourced wakeup/interrupt capability. This multi-sourced interrupt capability may also be used independent of the HALT or IDLE modes. Each I/O pin has software selectable configurations. The devices operate over a voltage range of 2.5V to 5.5V. High throughput is achieved with an efficient, regular instruction set operating at a maximum rate of 1 μs per instruction.

Low radiated emissions are achieved by gradual turn-on output drivers and internal I_{CC} smoothing filters on the chip logic and crystal oscillator.

Connection Diagrams

Plastic Chip Carrier

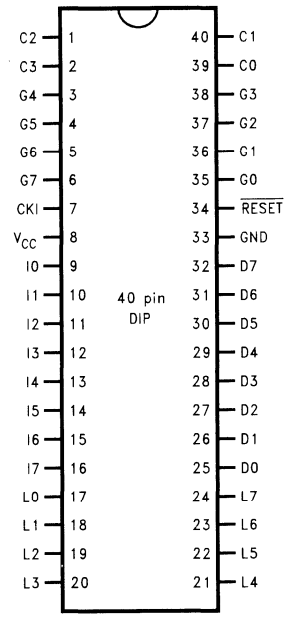


TL/DD/12858-2

Top View

Order Number COP688HG-XXX/N, COP888HG-XXX/V
See NS Package Number V44A

Dual-In-Line Package



TL/DD/12858-3

Top View

Order Number COP688HG-XXX/N, COP888HG-XXX/N
See NS Package Number N40A

FIGURE 2. Connection Diagrams

Connection Diagrams (Continued)

Pinouts for 40- and 44-Pin Packages

Port	Type	Alt. Fun	Alt. Fun	40-Pin DIP	44-Pin PLCC
L0	I/O	MIWU		17	17
L1	I/O	MIWU	CKX	18	18
L2	I/O	MIWU	TDX	19	19
L3	I/O	MIWU	RDX	20	20
L4	I/O	MIWU	T2A	21	25
L5	I/O	MIWU	T2B	22	26
L6	I/O	MIWU	T3A	23	27
L7	I/O	MIWU	T3B	24	28
G0	I/O	INT		35	39
G1	WDOUT			36	40
G2	I/O	T1B		37	41
G3	I/O	T1A		38	42
G4	I/O	SO		3	3
G5	I/O	SK		4	4
G6	I	SI		5	5
G7	I/CKO	HALT Restart		6	6
D0	O			25	29
D1	O			26	30
D2	O			27	31
D3	O			28	32
D4	O			29	33
D5	O			30	34
D6	O			31	35
D7	O			32	36
I0	I			9	9
I1	I	COMP1IN−		10	10
I2	I	COMP1IN+		11	11
I3	I	COMP1OUT		12	12
I4	I	COMP2IN−		13	13
I5	I	COMP2IN+		14	14
I6	I	COMP2OUT		15	15
I7	I			16	16
C0	I/O			39	43
C1	I/O			40	44
C2	I/O			1	1
C3	I/O			2	2
C4	I/O				21
C5	I/O				22
C6	I/O				23
C7	I/O				24
V_{CC}				8	8
GND				33	37
CKI				7	7
$\overline{\text{RESET}}$				34	38

Absolute Maximum Ratings

Supply Voltage (V_{CC})	7V	Total Current out of GND Pin (Sink)	110 mA
Voltage at Any Pin	−0.3V to V_{CC} + 0.3V	Storage Temperature Range	−65°C to +140°C
Total Current into V_{CC} Pin (Source)	100 mA		

Note: *Absolute maximum ratings indicate limits beyond which damage to the device may occur. DC and AC electrical specifications are not ensured when operating the device at absolute maximum ratings.*

DC Electrical Characteristics

COP888HG: $-40°C \le T_A \le +85°C$ unless otherwise specified

Parameter	Conditions	Min	Typ	Max	Units
Operating Voltage		2.5		5.5	V
Power Supply Ripple (Note 1)	Peak-to-Peak			0.1 V_{CC}	V
Supply Current (Note 2)					
CKI = 10 MHz	V_{CC} = 5.5V, t_c = 1 μs			12.5	mA
CKI = 4 MHz	V_{CC} = 5.5V, t_c = 2.5 μs			5.5	mA
CKI = 4 MHz	V_{CC} = 4.0V, t_c = 2.5 μs			2.5	mA
CKI = 1 MHz	V_{CC} = 4.0V, t_c = 10 μs			1.4	mA
HALT Current (Note 3)	V_{CC} = 5.5V, CKI = 0 MHz		<1	12	μA
	V_{CC} = 4.0V, CKI = 0 MHz		<0.5	8	μA
IDLE Current					
CKI = 10 MHz	V_{CC} = 5.5V, t_c = 1 μs			3.5	mA
CKI = 4 MHz	V_{CC} = 5.5V, t_c = 2.5 μs			2.5	mA
CKI = 1 MHz	V_{CC} = 4.0V, t_c = 10 μs			0.7	mA
Input Levels (V_{IH}, V_{IL})					
$\overline{\text{RESET}}$					
Logic High		0.8 V_{CC}			V
Logic Low				0.2 V_{CC}	V
All Other Inputs					
Logic High		0.7 V_{CC}			V
Logic Low				0.2 V_{CC}	V
Hi-Z Input Leakage	V_{CC} = 5.5V, V_{IN} = 0V	−2		+2	μA
Input Pullup Current	V_{CC} = 5.5V, V_{IN} = 0V	−40		−250	μA
G and L Port Input Hysteresis (Note 7)				0.35 V_{CC}	V
Output Current Levels					
D Outputs					
Source	V_{CC} = 4.0V, V_{OH} = 3.3V	−0.4			mA
	V_{CC} = 2.5V, V_{OH} = 1.8V	−0.2			mA
Sink (Note 4)	V_{CC} = 4.0V, V_{OL} = 1V	10			mA
	V_{CC} = 2.5V, V_{OL} = 0.4V	2.0			mA
All Others					
Source (Weak Pull-Up Mode)	V_{CC} = 4.0V, V_{OH} = 2.7V	−10		−100	μA
	V_{CC} = 2.5V, V_{OH} = 1.8V	−2.5		−33	μA
Source (Push-Pull Mode)	V_{CC} = 4.0V, V_{OH} = 3.3V	−0.4			mA
	V_{CC} = 2.5V, V_{OH} = 1.8V	−0.2			mA
Sink (Push-Pull Mode)	V_{CC} = 4.0V, V_{OL} = 0.4V	1.6			mA
	V_{CC} = 2.5V, V_{OL} = 0.4V	0.7			mA
TRI-STATE Leakage	V_{CC} = 5.5V	−2		+2	μA
Allowable Sink/Source Current per Pin (Note 6)					
D Outputs (Sink)				15	mA
All others				3	mA
Maximum Input Current without Latchup (Notes 5, 7)				±100	mA
RAM Retention Voltage, V_r	500 ns Rise and Fall Time (min)	2			V
Input Capacitance (Note 7)				7	pF
Load Capacitance on D2 (Note 7)				1000	pF

AC Electrical Characteristics COP888HG: $-40°C \le T_A \le +85°C$ unless otherwise specified

Parameter	Conditions	Min	Typ	Max	Units
Instruction Cycle Time (t_c)					
Crystal, Resonator	$2.5V \le V_{CC} \le 4.0V$	2.5		DC	μs
	$4.0V \le V_{CC} \le 5.5V$	1		DC	μs
R/C Oscillator	$2.5V \le V_{CC} \le 4.0V$	7.5		DC	μs
	$4.0V \le V_{CC} \le 5.5V$	3		DC	μs
Inputs					
t_{SETUP}	$4.0V \le V_{CC} \le 5.5V$	200			ns
	$2.5V \le V_{CC} \le 4.0V$	500			ns
t_{HOLD}	$4.5V \le V_{CC} \le 5.5V$	60			ns
	$2.5V \le V_{CC} \le 4.0V$	150			ns
Output Propagation Delay	$R_L = 2.2k, C_L = 100$ pF				
t_{PD1}, t_{PD0}					
SO, SK	$4.0V \le V_{CC} \le 5.5V$			0.7	μs
	$2.5V \le V_{CC} \le 4.0V$			1.75	μs
All Others	$4.0V \le V_{CC} \le 5.5V$			1.0	μs
	$2.5V \le V_{CC} \le 4.0V$			2.5	μs
MICROWIRE Setup Time (t_{UWS}) (Note 7)	$V_{CC} \ge 4.0V$	20			ns
MICROWIRE Hold Time (t_{UWH}) (Note 7)	$V_{CC} \ge 4.0V$	56			ns
MICROWIRE Output Propagation Delay (t_{UPD})	$V_{CC} \ge 4.0V$			220	ns
Input Pulse Width					
Interrupt Input High Time		1.0			t_c
Interrupt Input Low Time		1.0			t_c
Timer 1, 2, 3 Input High Time		1.0			t_c
Timer 1, 2, 3 Input Low Time		1.0			t_c
Reset Pulse Width		1.0			μs

t_c = Instruction Cycle Time

Note 1: Maximum rate of voltage change must be less than 0.5 V/ms.

Note 2: Supply and IDLE currents are measured with CKI driven with a square wave Oscillator, CKO driven 180° out of phase with CKI, inputs connected to V_{CC} and outputs driven low but not connected to a load.

Note 3: The HALT mode will stop CKI from oscillating in the RC and the Crystal configurations by bringing CKI high. Test Conditions: All inputs tied to V_{CC}, L and G ports in the TRI-STATE mode and tied to ground, all outputs low and tied to ground. The clock monitor is disabled.

Note 4: The user must guarantee that D2 pin does not source more than 10 mA during RESET. If D2 sources more than 10 mA during reset, the device will go into programming mode.

Note 5: Pins G6 and $\overline{\text{RESET}}$ are designed with a high voltage input network. These pins allow input voltages $> V_{CC}$ and the pins will have sink current to V_{CC} when biased at voltages $> V_{CC}$ (the pins do not have source current when biased at a voltage below V_{CC}). The effective resistance to V_{CC} is 750Ω (typical). These two pins will not latch up. The voltage at the pins must be limited to $< 14V$. **WARNING: Voltages in excess of 14V will cause damage to the pins. This warning excludes ESD transients.**

Note 6: The output propagation delay is referenced to the end of the instruction cycle where the output change occurs.

Note 7: Parameter characterized but not tested.

Absolute Maximum Ratings

If Military/Aerospace specified devices are required, please contact the National Semiconductor Sales Office/Distributors for availability and specifications.

Supply Voltage (V_{CC})	7V
Voltage at Any Pin	$-0.3V$ to $V_{CC} + 0.3V$
Total Current into V_{CC} Pin (Source)	100 mA
Total Current out of GND Pin (Sink)	110 mA
Storage Temperature Range	$-65°C$ to $+140°C$

Note: *Absolute maximum ratings indicate limits beyond which damage to the device may occur. DC and AC electrical specifications are not ensured when operating the device at absolute maximum ratings.*

DC Electrical Characteristics

COP688HG: $-55°C \le T_A \le +125°C$ unless otherwise specified

Parameter	Conditions	Min	Typ	Max	Units
Operating Voltage		4.5		5.5	V
Power Supply Ripple (Note 1)	Peak-to-Peak			0.1 V_{CC}	V
Supply Current (Note 2)					
CKI = 10 MHz	$V_{CC} = 5.5V$, $t_c = 1\ \mu s$			12.5	mA
CKI = 4 MHz	$V_{CC} = 5.5V$, $t_c = 2.5\ \mu s$			5.5	mA
HALT Current (Note 3)	$V_{CC} = 5.5V$, CKI = 0 MHz		<10	30	μA
IDLE Current					
CKI = 10 MHz	$V_{CC} = 5.5V$, $t_c = 1\ \mu s$			3.5	mA
CKI = 4 MHz	$V_{CC} = 5.5V$, $t_c = 2.5\ \mu s$			2.5	mA
Input Levels (V_{IH}, V_{IL})					
$\overline{\text{RESET}}$					
Logic High		0.8 V_{CC}			V
Logic Low				0.2 V_{CC}	V
CKI, All Other Inputs					
Logic High		0.7 V_{CC}			V
Logic Low				0.2 V_{CC}	V
Hi-Z Input Leakage	$V_{CC} = 5.5V$, $V_{IN} = 0V$	−5		+5	μA
Input Pullup Current	$V_{CC} = 5.5V$, $V_{IN} = 0V$	−35		−400	μA
G and L Port Input Hysteresis	(Note 5)			0.35 V_{CC}	V
Output Current Levels					
D Outputs					
Source	$V_{CC} = 4.5V$, $V_{OH} = 3.3V$	0.4			mA
Sink	$V_{CC} = 4.5V$, $V_{OL} = 1V$	9			mA
All Others					
Source (Weak Pull-Up Mode)	$V_{CC} = 4.5V$, $V_{OH} = 2.7V$	−9		−140	μA
Source (Push-Pull Mode)	$V_{CC} = 4.5V$, $V_{OH} = 3.3V$	0.4			mA
Sink (Push-Pull Mode)	$V_{CC} = 4.5V$, $V_{OL} = 0.4V$	1.4			mA
TRI-STATE Leakage	$V_{CC} = 5.5V$	−5		+5	μA
Allowable Sink/Source Current per Pin					
D Outputs (Sink)				12	mA
All others				2.5	mA
Maximum Input Current without Latchup (Notes 4, 5)	Room Temperature			±100	mA
RAM Retention Voltage, V_r	500 ns Rise and Fall Time (min)	2			V
Input Capacitance	(Note 5)			7	pF
Load Capacitance on D2	(Note 5)			1000	pF

AC Electrical Characteristics COP688HG: $-55°C \le T_A \le +125°C$ unless otherwise specified

Parameter	Conditions	Min	Typ	Max	Units
Instruction Cycle Time (t_c)					
Crystal, Resonator	$V_{CC} \ge 4.5V$	1.0		DC	μs
Inputs					
t_{SETUP}	$V_{CC} \ge 4.5V$	200			ns
t_{HOLD}	$V_{CC} \ge 4.5V$	60			ns
Output Propagation Delay	$R_L = 2.2k, C_L = 100$ pF				
t_{PD1}, t_{PD0}					
SO, SK	$V_{CC} \ge 4.5V$			0.7	μs
All Others	$V_{CC} \ge 4.5V$			1	μs
MICROWIRE Setup Time (t_{UWS}) (Note 5)		20			ns
MICROWIRE Hold Time (t_{UWH}) (Note 5)		56			ns
MICROWIRE Output Propagation Delay (t_{UPD})				220	ns
Input Pulse Width (Note 6)					
Interrupt Input High Time		1			t_c
Interrupt Input Low Time		1			t_c
Timer 1, 2, 3 Input High Time		1			t_c
Timer 1, 2, 3 Input Low Time		1			t_c
Reset Pulse Width		1			μs

Note 1: Maximum rate of voltage change must be less than 0.5 V/ms.

Note 2: Supply current is measured after running 2000 cycles with a square wave, CKI input, CKO open, inputs at rails and outputs open.

Note 3: The HALT mode will stop CKI from oscillating in the RC and the Crystal configurations. Measurement of IDD HALT is done with device neither sourcing or sinking current; with L, C, and G0–G5 programmed as low outputs and not driving a load; all outputs programmed low and not driving a load; all inputs tied to V_{CC}; clock monitor and comparators disabled. Parameter refers to HALT mode entered via setting bit 7 of the G Port data register. Part will pull up CKI during HALT in crystal clock mode.

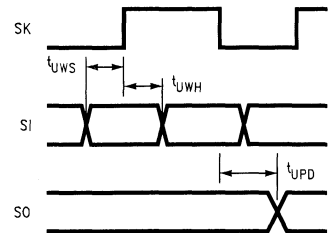
Note 4: Pins G6 and $\overline{\text{RESET}}$ are designed with a high voltage input network. These pins allow input voltages $> V_{CC}$ and the pins will have sink current to V_{CC} when biased at voltages $> V_{CC}$ (the pins do not have source current when biased at a voltage below V_{CC}). The effective resistance to V_{CC} is 750Ω (typical). These two pins will not latch up. The voltage at the pins must be limited to $< 14V$. **WARNING: Voltages in excess of 14V will cause damage to the pins. This warning excludes ESD transients.**

Note 5: Parameter characterized but not tested.

Note 6: t_c = Instruction Cycle Time.

Comparators AC and DC Characteristics $V_{CC} = 5V, -40°C \leq T_A \leq \pm 85°C$.

Parameter	Conditions	Min	Typ	Max	Units
Input Offset Voltage	$0.4V \leq V_{IN} \leq V_{CC} - 1.5V$		±10	±25	mV
Input Common Mode Voltage Range		0.4		$V_{CC} - 1.5$	V
Low Level Output Current	$V_{OL} = 0.4V$	1.6			mA
High Level Output Current	$V_{OH} = 4.6V$	1.6			mA
DC Supply Current per Comparator (When Enabled)				250	μA
Response Time	100 mV Overdrive, 100 pF Load			1	μs



TL/DD/12858–4

FIGURE 3. MICROWIRE/PLUS Timing

Pin Descriptions

V_{CC} and GND are the power supply pins. All V_{CC} and GND pins must be connected.

CKI is the clock input. This can come from an R/C generated oscillator, or a crystal oscillator (in conjunction with CKO). See Oscillator Description section.

$\overline{\text{RESET}}$ is the master reset input. See Reset Description section.

The device contains three bidirectional 8-bit I/O ports (C, G and L), where each individual bit may be independently configured as an input (Schmitt Trigger inputs on ports L and G), output or TRI-STATE under program control. Three data memory address locations are allocated for each of these I/O ports. Each I/O port has two associated 8-bit memory mapped registers, the CONFIGURATION register and the output DATA register. A memory mapped address is also reserved for the input pins of each I/O port. (See the memory map for the various addresses associated with the I/O ports.) *Figure 4* shows the I/O port configurations. The DATA and CONFIGURATION registers allow for each port bit to be individually configured under software control as shown below:

CONFIGURATION Register	DATA Register	Port Set-Up
0	0	Hi-Z Input (TRI-STATE Output)
0	1	Input with Weak Pull-Up
1	0	Push-Pull Zero Output
1	1	Push-Pull One Output

PORT L is an 8-bit I/O port. All L-pins have Schmitt triggers on the inputs.

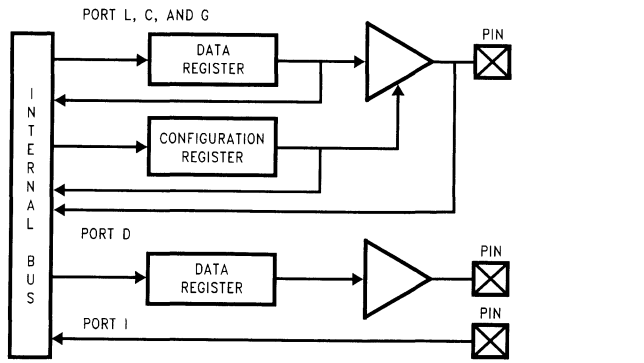
The Port L supports Multi-Input Wake Up on all eight pins. L1 is used for the UART external clock. L2 and L3 are used for the UART transmit and receive. L4 and L5 are used for the timer input functions T2A and T2B. L6 and L7 are used for the timer input functions T3A and T3B.

The Port L has the following alternate features:

- L0 MIWU
- L1 MIWU or CKX
- L2 MIWU or TDX
- L3 MIWU or RDX
- L4 MIWU or T2A
- L5 MIWU or T2B
- L6 MIWU or T3A
- L7 MIWU or T3B

Port G is an 8-bit port with 5 I/O pins (G0, G2–G5), an input pin (G6), and a dedicated output pin (G7). Pins G0 and G2–G6 all have Schmitt Triggers on their inputs. Pin G1 serves as the dedicated WDOUT WATCHDOG output, while pin G7 is either input or output depending on the oscillator mask option selected. With the crystal oscillator option selected, G7 serves as the dedicated output pin for the CKO clock output. With the single-pin R/C oscillator mask option selected, G7 serves as a general purpose input pin but is also used to bring the device out of HALT mode with a low to high transition on G7. There are two registers associated with the G Port, a data register and a configuration register. Therefore, each of the 5 I/O bits (G0, G2–G5) can be individually configured under software control.

Since G6 is an input only pin and G7 is the dedicated CKO clock output pin (crystal clock option) or general purpose input (R/C clock option), the associated bits in the data and configuration registers for G6 and G7 are used for special purpose functions as outlined on the next page. Reading the G6 and G7 data bits will return zeros.



TL/DD/12858–5

FIGURE 4. I/O Port Configurations

Pin Descriptions (Continued)

Note that the chip will be placed in the HALT mode by writing a "1" to bit 7 of the Port G Data Register. Similarly the chip will be placed in the IDLE mode by writing a "1" to bit 6 of the Port G Data Register.

Writing a "1" to bit 6 of the Port G Configuration Register enables the MICROWIRE/PLUS to operate with the alternate phase of the SK clock. The G7 configuration bit, if set high, enables the clock start up delay after HALT when the R/C clock configuration is used.

	Config Reg.	Data Reg.
G7	CLKDLY	HALT
G6	Alternate SK	IDLE

Port G has the following alternate features:

- G0 INTR (External Interrupt Input)
- G2 T1B (Timer T1 Capture Input)
- G3 T1A (Timer T1 I/O)
- G4 SO (MICROWIRE Serial Data Output)
- G5 SK (MICROWIRE Serial Clock)
- G6 SI (MICROWIRE Serial Data Input)

Port G has the following dedicated functions:

- G1 WDOUT WATCHDOG and/or Clock Monitor dedicated output
- G7 CKO Oscillator dedicated output or general purpose input

Port C is an 8-bit I/O port. The 40-pin device does not have a full complement of Port C pins. The unavailable pins are not terminated. A read operation for these unterminated pins will return unpredicatable values.

Port I is an eight-bit Hi-Z input port.

Port I1–I3 are used for Comparator 1. Port I4–I6 are used for Comparator 2.

The Port I has the following alternate features:

- I1 COMP1 − IN (Comparator 1 Negative Input)
- I2 COMP1 + IN (Comparator 1 Positive Input)
- I3 COMP1OUT (Comparator 1 Output)
- I4 COMP2 − IN (Comparator 2 Negative Input)
- I5 COMP2 + IN (Comparator 2 Positive Input)
- I6 COMP2OUT (Comparator 2 Output)

Port D is an 8-bit output port that is preset high when $\overline{\text{RESET}}$ goes low. The user can tie two or more D port outputs (except D2) together in order to get a higher drive.

Functional Description

The architecture of the device is modified Harvard architecture. With the Harvard architecture, the control store program memory (ROM) is separated from the data store memory (RAM). Both ROM and RAM have their own separate addressing space with separate address buses. The architecture, though based on Harvard architecture, permits transfer of data from ROM to RAM.

CPU REGISTERS

The CPU can do an 8-bit addition, subtraction, logical or shift operation in one instruction (t_c) cycle time.

There are six CPU registers:

A is the 8-bit Accumulator Register

PC is the 15-bit Program Counter Register

PU is the upper 7 bits of the program counter (PC)
PL is the lower 8 bits of the program counter (PC)

B is an 8-bit RAM address pointer, which can be optionally post auto incremented or decremented.

X is an 8-bit alternate RAM address pointer, which can be optionally post auto incremented or decremented.

SP is the 8-bit stack pointer, which points to the subroutine/interrupt stack (in RAM). The SP is initialized to RAM address 06F with reset.

S is the 8-bit Data Segment Address Register used to extend the lower half of the address range (00 to 7F) into 256 data segments of 128 bytes each.

All the CPU registers are memory mapped with the exception of the Accumulator (A) and the Program Counter (PC).

PROGRAM MEMORY

The program memory consists of 20 kbytes of ROM. These bytes may hold program instructions or constant data (data tables for the LAID instruction, jump vectors for the JID instruction, and interrupt vectors for the VIS instruction). The program memory is addressed by the 15-bit program counter (PC). All interrupts in the devices vector to program memory location 0FF Hex.

DATA MEMORY

The data memory address space includes the on-chip RAM and data registers, the I/O registers (Configuration, Data and Pin), the control registers, the MICROWIRE/PLUS SIO shift register, and the various registers, and counters associated with the timers (with the exception of the IDLE timer). Data memory is addressed directly by the instruction or indirectly by the B, X, SP pointers and S register.

The data memory consists of 512 bytes of RAM. Sixteen bytes of RAM are mapped as "registers" at addresses 0F0 to 0FF Hex. These registers can be loaded immediately, and also decremented and tested with the DRSZ (decrement register and skip if zero) instruction. The memory pointer registers X, SP, B and S are memory mapped into this space at address locations 0FC to 0FF Hex respectively, with the other registers being available for general usage.

The instruction set permits any bit in memory to be set, reset or tested. All I/O and registers (except A and PC) are memory mapped; therefore, I/O bits and register bits can be directly and individually set, reset and tested. The accumulator (A) bits can also be directly and individually tested.

Note: RAM contents are undefined upon power-up.

Data Memory Segment RAM Extension

Data memory address 0FF is used as a memory mapped location for the Data Segment Address Register (S).

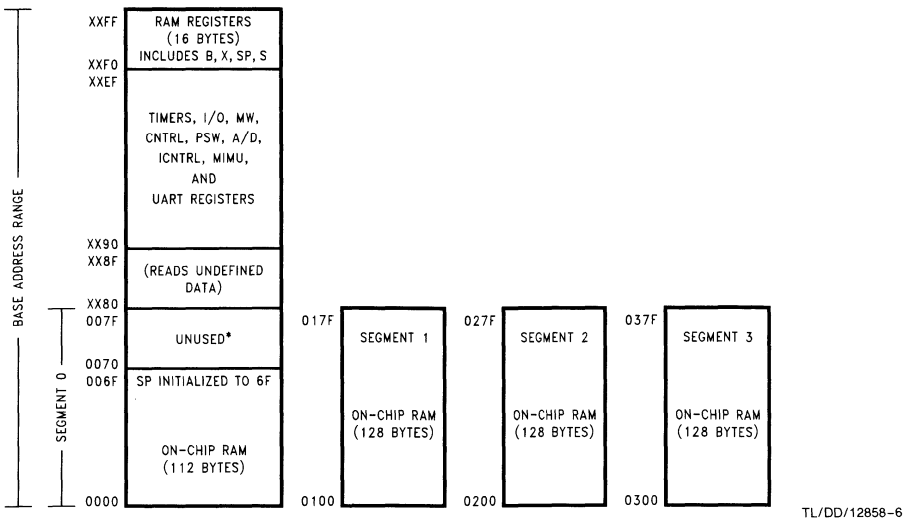
The data store memory is either addressed directly by a single byte address within the instruction, or indirectly relative to the reference of the B, X, or SP pointers (each contains a single-byte address). This single-byte address allows an addressing range of 256 locations from 00 to FF hex. The upper bit of this single-byte address divides the data store memory into two separate sections as outlined previously. With the exception of the RAM register memory from address locations 00F0 to 00FF, all RAM memory is memory mapped with the upper bit of the single-byte address being equal to zero. This allows the upper bit of the single-byte address to determine whether or not the base address range (from 0000 to 00FF) is extended. If this upper bit equals one (representing address range 0080 to 00FF), then address extension does not take place. Alternatively, if this upper bit equals zero, then the data segment extension register S is used to extend the base address range (from 0000 to 007F) from XX00 to XX7F, where XX represents the 8 bits from the S register. Thus the 128-byte data segment extensions are located from addresses 0100 to 017F for data segment 1, 0200 to 027F for data segment 2, etc., up to FF00 to FF7F for data segment 255. The base address range from 0000 to 007F represents data segment 0.

Figure 5 illustrates how the S register data memory extension is used in extending the lower half of the base address range (00 to 7F hex) into 256 data segments of 128 bytes each, with a total addressing range of 32 kbytes from XX00 to XX7F. This organization allows a total of 256 data segments of 128 bytes each with an additional upper base segment of 128 bytes. Furthermore, all addressing modes are available for all data segments. The S register must be changed under program control to move from one data segment (128 bytes) to another. However, the upper base segment (containing the 16 memory registers, I/O registers, control registers, etc.) is always available regardless of the contents of the S register, since the upper base segment (address range 0080 to 00FF) is independent of data segment extension.

The instructions that utilize the stack pointer (SP) always reference the stack as part of the base segment (Segment 0), regardless of the contents of the S register. The S register is not changed by these instructions. Consequently, the stack (used with subroutine linkage and interrupts) is always located in the base segment. The stack pointer will be initialized to point at data memory location 006F as a result of reset.

The 128 bytes of RAM contained in the base segment are split between the lower and upper base segments. The first 112 bytes of RAM are resident from address 0000 to 006F in the lower base segment, while the remaining 16 bytes of RAM represent the 16 data memory registers located at addresses 00F0 to 00FF of the upper base segment. No RAM is located at the upper sixteen addresses (0070 to 007F) of the lower base segment.

Additional RAM beyond these initial 128 bytes, however, will always be memory mapped in groups of 128 bytes (or less) at the data segment address extensions (XX00 to XX7F) of the lower base segment. The additional 128 bytes of RAM are memory mapped at address locations 0100 to 017F hex.



*Reads as all ones.

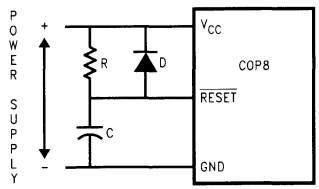
FIGURE 5. RAM Organization

Reset

The $\overline{\text{RESET}}$ input when pulled low initializes the microcontroller. Initialization will occur whenever the $\overline{\text{RESET}}$ input is pulled low. Upon initialization, the data and configuration registers for ports L, G and C are cleared, resulting in these Ports being initialized to the TRI-STATE mode. Pin G1 of the G Port is an exception (as noted below) since pin G1 is dedicated as the WATCHDOG and/or Clock Monitor error output pin. Port D is set high. The PC, PSW, ICNTRL, CNTRL, T2CNTRL and T3CNTRL control registers are cleared. The Comparator Select Register is cleared. The S register is initialized to zero. The Multi-Input Wakeup registers WKEN and WKEDG are cleared. Wakeup register WKPND is unknown. The stack pointer, SP, is initialized to 6F hex.

The device comes out of reset with both the WATCHDOG logic and the Clock Monitor detector armed, with the WATCHDOG service window bits set and the Clock Monitor bit set. The WATCHDOG and Clock Monitor circuits are inhibited during reset. The WATCHDOG service window bits being initialized high default to the maximum WATCHDOG service window of 64k t_C clock cycles. The Clock Monitor bit being initialized high will cause a Clock Monitor error following reset if the clock has not reached the minimum specified frequency at the termination of reset. A Clock Monitor error will cause an active low error output on pin G1. This error output will continue until 16 t_C–32 t_C clock cycles following the clock frequency reaching the minimum specified value, at which time the G1 output will enter the TRI-STATE mode.

The external RC network shown in *Figure 6* should be used to ensure that the $\overline{\text{RESET}}$ pin is held low until the power supply to the chip stabilizes.



TL/DD/12858–7

RC > 5 × Power Supply Rise Time

FIGURE 6. Recommended Reset Circuit

Oscillator Circuits

The chip can be driven by a clock input on the CKI input pin which can be between DC and 10 MHz. The CKO output clock is on pin G7 (crystal configuration). The CKI input frequency is divided down by 10 to produce the instruction cycle clock ($1/t_C$).

Figure 7 shows the Crystal and R/C oscillator diagrams.

CRYSTAL OSCILLATOR

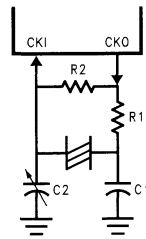
CKI and CKO can be connected to make a closed loop crystal (or resonator) controlled oscillator.

Table I shows the component values required for various standard crystal values.

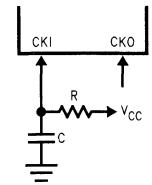
R/C OSCILLATOR

By selecting CKI as a single pin oscillator input, a single pin R/C oscillator circuit can be connected to it. CKO is available as a general purpose input, and/or HALT restart input.

Table II shows the variation in the oscillator frequencies as functions of the component (R and C) values.



TL/DD/12858–8



TL/DD/12858–9

FIGURE 7. Crystal and R/C Oscillator Diagrams

TABLE I. Crystal Oscillator Configuration, T_A = 25°C

R1 (kΩ)	R2 (MΩ)	C1 (pF)	C2 (pF)	CKI Freq (MHz)	Conditions
0	1	30	30–36	10	V_{CC} = 5V
0	1	30	30–36	4	V_{CC} = 5V
0	1	200	100–150	0.455	V_{CC} = 5V

TABLE II. RC Oscillator Configuration, T_A = 25°C

R (kΩ)	C (pF)	CKI Freq (MHz)	Instr. Cycle (μs)	Conditions
3.3	82	2.2 to 2.7	3.7 to 4.6	V_{CC} = 5V
5.6	100	1.1 to 1.3	7.4 to 9.0	V_{CC} = 5V
6.8	100	0.9 to 1.1	8.8 to 10.8	V_{CC} = 5V

Note: 3k ≤ R ≤ 200k

50 pF ≤ C ≤ 200 pF

Control Registers

CNTRL Register (Address X'00EE)

The Timer1 (T1) and MICROWIRE/PLUS control register contains the following bits:

- SL1 & SL0 Select the MICROWIRE/PLUS clock divide by (00 = 2, 01 = 4, 1x = 8)
- IEDG External interrupt edge polarity select (0 = Rising edge, 1 = Falling edge)
- MSEL Selects G5 and G4 as MICROWIRE/PLUS signals SK and SO respectively
- T1C0 Timer T1 Start/Stop control in timer modes 1 and 2
  Timer T1 Underflow Interrupt Pending Flag in timer mode 3
- T1C1 Timer T1 mode control bit
- T1C2 Timer T1 mode control bit
- T1C3 Timer T1 mode control bit

T1C3	T1C2	T1C1	T1C0	MSEL	IEDG	SL1	SL0

Bit 7 Bit 0

PSW Register (Address X'00EF)

The PSW register contains the following select bits:

- GIE Global interrupt enable (enables interrupts)
- EXEN Enable external interrupt
- BUSY MICROWIRE/PLUS busy shifting flag
- EXPND External interrupt pending
- T1ENA Timer T1 Interrupt Enable for Timer Underflow or T1A Input capture edge
- T1PNDA Timer T1 Interrupt Pending Flag (Autoreload RA in mode 1, T1 Underflow in Mode 2, T1A capture edge in mode 3)
- C Carry Flag
- HC Half Carry Flag

HC	C	T1PNDA	T1ENA	EXPND	BUSY	EXEN	GIE

Bit 7 Bit 0

The Half-Carry bit is also affected by all the instructions that affect the Carry flag. The SC (Set Carry) and RC (Reset Carry) instructions will respectively set or clear both the carry flags. In addition to the SC and RC instructions, ADC, SUBC, RRC and RLC instructions affect the carry and Half Carry flags.

ICNTRL Register (Address X'00E8)

The ICNTRL register contains the following bits:

- T1ENB Timer T1 Interrupt Enable for T1B Input capture edge
- T1PNDB Timer T1 Interrupt Pending Flag for T1B capture edge
- μWEN Enable MICROWIRE/PLUS interrupt
- μWPND MICROWIRE/PLUS interrupt pending
- T0EN Timer T0 Interrupt Enable (Bit 12 toggle)
- T0PND Timer T0 Interrupt pending
- LPEN L Port Interrupt Enable (Multi-Input Wakeup/Interrupt)
  Bit 7 could be used as a flag

Unused	LPEN	T0PND	T0EN	μWPND	μWEN	T1PNDB	T1ENB

Bit 7 Bit 0

T2CNTRL Register (Address X'00C6)

The T2CNTRL register contains the following bits:

- T2ENB Timer T2 Interrupt Enable for T2B Input capture edge
- T2PNDB Timer T2 Interrupt Pending Flag for T2B capture edge
- T2ENA Timer T2 Interrupt Enable for Timer Underflow or T2A Input capture edge
- T2PNDA Timer T2 Interrupt Pending Flag (Autoreload RA in mode 1, T2 Underflow in mode 2, T2A capture edge in mode 3)
- T2C0 Timer T2 Start/Stop control in timer modes 1 and 2 Timer T2 Underflow Interrupt Pending Flag in timer mode 3
- T2C1 Timer T2 mode control bit
- T2C2 Timer T2 mode control bit
- T2C3 Timer T2 mode control bit

T2C3	T2C2	T2C1	T2C0	T2PNDA	T2ENA	T2PNDB	T2ENB

Bit 7 Bit 0

T3CNTRL Register (Address X'00B6)

The T3CNTRL register contains the following bits:

- T3ENB Timer T3 Interrupt Enable for T3B Input capture edge
- T3PNDB Timer T3 Interrupt Pending Flag for T3B capture edge
- T3ENA Timer T3 Interrupt Enable for Timer Underflow or T3A Input capture edge
- T3PNDA Timer T3 Interrupt Pending Flag (Autoload RA in mode 1, T3 Underflow in mode 2, T3A capture edge in mode 3)
- T3C0 Timer T3 Start/Stop control in timer modes 1 and 2
  Timer T3 Underflow Interrupt Pending Flag in timer mode 3
- T3C1 Timer T3 mode control bit
- T3C2 Timer T3 mode control bit
- T3C3 Timer T3 mode control bit

T3C3	T3C2	T3C1	T3C0	T3PNDA	T3ENA	T3PNDB	T3ENB

Bit 7 Bit 0

Timers

The device contains a very versatile set of timers (T0, T1, T2, T3). All timers and associated autoreload/capture registers power up containing random data.

TIMER T0 (IDLE TIMER)

The device supports applications that require maintaining real time and low power with the IDLE mode. This IDLE mode support is furnished by the IDLE timer T0, which is a 16-bit timer. The Timer T0 runs continuously at the fixed rate of the instruction cycle clock, t_C. The user cannot read or write to the IDLE Timer T0, which is a count down timer.

The Timer T0 supports the following functions:

- Exit out of the Idle Mode (See Idle Mode description)
- WATCHDOG logic (See WATCHDOG description)
- Start up delay out of the HALT mode

The IDLE Timer T0 can generate an interrupt when the thirteenth bit toggles. This toggle is latched into the T0PND pending flag, and will occur every 4 ms at the maximum clock frequency (t_C = 1 μs). A control flag T0EN allows the interrupt from the thirteenth bit of Timer T0 to be enabled or disabled. Setting T0EN will enable the interrupt, while resetting it will disable the interrupt.

TIMER T1, TIMER T2 AND TIMER T3

The device has a set of three powerful timer/counter blocks, T1, T2 and T3. The associated features and functioning of a timer block are described by referring to the timer block Tx. Since the three timer blocks, T1, T2 and T3 are identical, all comments are equally applicable to any of the three timer blocks.

Each timer block consists of a 16-bit timer, Tx, and two supporting 16-bit autoreload/capture registers, RxA and RxB. Each timer block has two pins associated with it, TxA and TxB. The pin TxA supports I/O required by the timer block, while the pin TxB is an input to the timer block. The powerful and flexible timer block allows the device to easily perform all timer functions with minimal software overhead. The timer block has three operating modes: Processor Independent PWM mode, External Event Counter mode, and Input Capture mode.

The control bits TxC3, TxC2, and TxC1 allow selection of the different modes of operation.

Mode 1. Processor Independent PWM Mode

As the name suggests, this mode allows the device to generate a PWM signal with very minimal user intervention. The user only has to define the parameters of the PWM signal (ON time and OFF time). Once begun, the timer block will continuously generate the PWM signal completely independent of the microcontroller. The user software services the timer block only when the PWM parameters require updating.

In this mode the timer Tx counts down at a fixed rate of t_C. Upon every underflow the timer is alternately reloaded with the contents of supporting registers, RxA and RxB. The very first underflow of the timer causes the timer to reload from the register RxA. Subsequent underflows cause the timer to be reloaded from the registers alternately beginning with the register RxB.

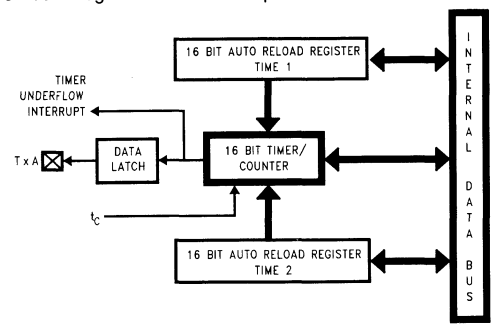
The Tx Timer control bits, TxC3, TxC2 and TxC1 set up the timer for PWM mode operation.

Figure 8 shows a block diagram of the timer in PWM mode.

The underflows can be programmed to toggle the TxA output pin. The underflows can also be programmed to generate interrupts.

Underflows from the timer are alternately latched into two pending flags, TxPNDA and TxPNDB. The user must reset these pending flags under software control. Two control enable flags, TxENA and TxENB, allow the interrupts from the timer underflow to be enabled or disabled. Setting the timer enable flag TxENA will cause an interrupt when a timer underflow causes the RxA register to be reloaded into the timer. Setting the timer enable flag TxENB will cause an interrupt when a timer underflow causes the RxB register to be reloaded into the timer. Resetting the timer enable flags will disable the associated interrupts.

Either or both of the timer underflow interrupts may be enabled. This gives the user the flexibility of interrupting once per PWM period on either the rising or falling edge of the PWM output. Alternatively, the user may choose to interrupt on both edges of the PWM output.



TL/DD/12858–10

FIGURE 8. Timer in PWM Mode

Mode 2. External Event Counter Mode

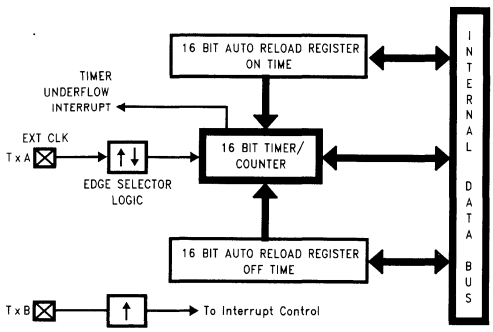
This mode is quite similar to the processor independent PWM mode described above. The main difference is that the timer, Tx, is clocked by the input signal from the TxA pin. The Tx timer control bits, TxC3, TxC2 and TxC1 allow the timer to be clocked either on a positive or negative edge from the TxA pin. Underflows from the timer are latched into the TxPNDA pending flag. Setting the TxENA control flag will cause an interrupt when the timer underflows.

In this mode the input pin TxB can be used as an independent positive edge sensitive interrupt input if the TxENB control flag is set. The occurrence of a positive edge on the TxB input pin is latched into the TxPNDB flag.

Timers (Continued)

Figure 9 shows a block diagram of the timer in External Event Counter mode.

Note: The PWM output is not available in this mode since the TxA pin is being used as the counter input clock.



TL/DD/12858–11

FIGURE 9. Timer in External Event Counter Mode

Mode 3. Input Capture Mode

The device can precisely measure external frequencies or time external events by placing the timer block, Tx, in the input capture mode.

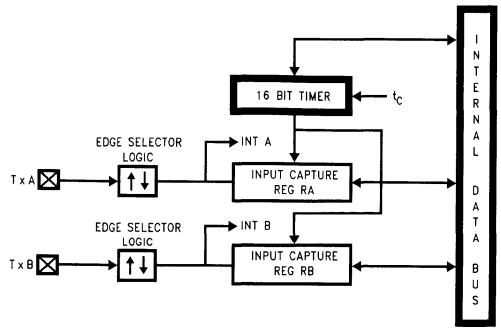
In this mode, the timer Tx is constantly running at the fixed t_C rate. The two registers, RxA and RxB, act as capture registers. Each register acts in conjunction with a pin. The register RxA acts in conjunction with the TxA pin and the register RxB acts in conjunction with the TxB pin.

The timer value gets copied over into the register when a trigger event occurs on its corresponding pin. Control bits, TxC3, TxC2 and TxC1, allow the trigger events to be specified either as a positive or a negative edge. The trigger condition for each input pin can be specified independently.

The trigger conditions can also be programmed to generate interrupts. The occurrence of the specified trigger condition on the TxA and TxB pins will be respectively latched into the pending flags, TxPNDA and TxPNDB. The control flag TxENA allows the interrupt on TxA to be either enabled or disabled. Setting the TxENA flag enables interrupts to be generated when the selected trigger condition occurs on the TxA pin. Similarly, the flag TxENB controls the interrupts from the TxB pin.

Underflows from the timer can also be programmed to generate interrupts. Underflows are latched into the timer TxC0 pending flag (the TxC0 control bit serves as the timer underflow interrupt pending flag in the Input Capture mode). Consequently, the TxC0 control bit should be reset when entering the Input Capture mode. The timer underflow interrupt is enabled with the TxENA control flag. When a TxA interrupt occurs in the Input Capture mode, the user must check both the TxPNDA and TxC0 pending flags in order to determine whether a TxA input capture or a timer underflow (or both) caused the interrupt.

Figure 10 shows a block diagram of the timer in Input Capture mode.



TL/DD/12858–12

FIGURE 10. Timer in Input Capture Mode

TIMER CONTROL FLAGS

The timers T1, T2 and T3 have indentical control structures. The control bits and their functions are summarized below.

TxC0	Timer Start/Stop control in Modes 1 and 2 (Processor Independent PWM and External Event Counter), where 1 = Start, 0 = Stop Timer Underflow Interrupt Pending Flag in Mode 3 (Input Capture)
TxPNDA	Timer Interrupt Pending Flag
TxPNDB	Timer Interrupt Pending Flag
TxENA	Timer Interrupt Enable Flag
TxENB	Timer Interrupt Enable Flag 1 = Timer Interrupt Enabled 0 = Timer Interrupt Disabled
TxC3	Timer mode control
TxC2	Timer mode control
TxC1	Timer mode control

Timers (Continued)

The timer mode control bits (TxC3, TxC2 and TxC1) are detailed below:

TxC3	TxC2	TxC1	Timer Mode	Interrupt A Source	Interrupt B Source	Timer Counts On
0	0	0	MODE 2 (External Event Counter)	Timer Underflow	Pos. TxB Edge	TxA Pos. Edge
0	0	1	MODE 2 (External Event Counter)	Timer Underflow	Pos. TxB Edge	TxA Neg. Edge
1	0	1	MODE 1 (PWM) TxA Toggle	Autoreload RA	Autoreload RB	t_C
1	0	0	MODE 1 (PWM) No TxA Toggle	Autoreload RA	Autoreload RB	t_C
0	1	0	MODE 3 (Capture) Captures: TxA Pos. Edge TxB Pos. Edge	Pos. TxA Edge or Timer Underflow	Pos. TxB Edge	t_C
1	1	0	MODE 3 (Capture) Captures: TxA Pos. Edge TxB Neg. Edge	Pos. TxA Edge or Timer Underflow	Neg. TxB Edge	t_C
0	1	1	MODE 3 (Capture) Captures: TxA Neg. Edge TxB Pos. Edge	Neg. TxA Edge or Timer Underflow	Pos. TxB Edge	t_C
1	1	1	MODE 3 (Capture) Captures: TxA Neg. Edge TxB Neg. Edge	Neg. TxA Edge or Timer Underflow	Neg. TxB Edge	t_C

Power Save Modes

The device offers the user two power save modes of operation: HALT and IDLE. In the HALT mode, all microcontroller activities are stopped. In the IDLE mode, the on-board oscillator circuitry the WATCHDOG logic, the Clock Monitor and timer T0 are active but all other microcontroller activities are stopped. In either mode, all on-board RAM, registers, I/O states, and timers (with the exception of T0) are unaltered.

HALT MODE

The device can be placed in the HALT mode by writing a "1" to the HALT flag (G7 data bit). All microcontroller activities, including the clock and timers, are stopped. The WATCHDOG logic is disabled during the HALT mode. However, the clock monitor circuitry if enabled remains active and will cause the WATCHDOG output pin (WDOUT) to go low. If the HALT mode is used and the user does not want to activate the WDOUT pin, the Clock Monitor should be disabled after the device comes out of reset (resetting the Clock Monitor control bit with the first write to the WDSVR register). In the HALT mode, the power requirements of the device are minimal and the applied voltage (V_{CC}) may be decreased to V_r (V_r = 2.0V) without altering the state of the machine.

The device supports three different ways of exiting the HALT mode. The first method of exiting the HALT mode is with the Multi-Input Wakeup feature on the L port. The second method is with a low to high transition on the CKO (G7) pin. This method precludes the use of the crystal clock configuration (since CKO becomes a dedicated output), and so may be used with an RC clock configuration. The third method of exiting the HALT mode is by pulling the $\overline{\text{RESET}}$ pin low.

Since a crystal or ceramic resonator may be selected as the oscillator, the Wakeup signal is not allowed to start the chip running immediately since crystal oscillators and ceramic resonators have a delayed start up time to reach full amplitude and frequency stability. The IDLE timer is used to generate a fixed delay to ensure that the oscillator has indeed stabilized before allowing instruction execution. In this case, upon detecting a valid Wakeup signal, only the oscillator circuitry is enabled. The IDLE timer is loaded with a value of 256 and is clocked with the t_C instruction cycle clock. The t_C clock is derived by dividing the oscillator clock down by a factor of 10. The Schmitt trigger following the CKI inverter on the chip ensures that the IDLE timer is clocked only when the oscillator has a sufficiently large amplitude to meet the Schmitt trigger specifications. This Schmitt trigger is not part of the oscillator closed loop. The startup timeout from the IDLE timer enables the clock signals to be routed to the rest of the chip.

If an RC clock option is being used, the fixed delay is introduced optionally. A control bit, CLKDLY, mapped as configuration bit G7, controls whether the delay is to be introduced or not. The delay is included if CLKDLY is set, and excluded if CLKDLY is reset. The CLKDLY bit is cleared on reset.

Power Save Modes (Continued)

The device has two mask options associated with the HALT mode. The first mask option enables the HALT mode feature, while the second mask option disables the HALT mode. With the HALT mode enable mask option, the device will enter and exit the HALT mode as described above. With the HALT disable mask option, the device cannot be placed in the HALT mode (writing a "1" to the HALT flag will have no effect, the HALT flag will remain "0").

IDLE MODE

The device is placed in the IDLE mode by writing a "1" to the IDLE flag (G6 data bit). In this mode, all activities, except the associated on-board oscillator circuitry and the IDLE Timer T0, are stopped. The power supply requirements of the micro-controller in this mode of operation are typically around 30% of normal power requirement of the microcontroller.

As with the HALT mode, the device can be returned to normal operation with a reset, or with a Multi-Input Wakeup from the L Port. Alternately, the microcontroller resumes normal operation from the IDLE mode when the thirteenth bit (representing 4.096 ms at internal clock frequency of 1 MHz, t_c = 1 μs) of the IDLE Timer toggles.

This toggle condition of the thirteenth bit of the IDLE Timer T0 is latched into the T0PND pending flag.

The user has the option of being interrupted with a transition on the thirteenth bit of the IDLE Timer T0. The interrupt can be enabled or disabled via the T0EN control bit. Setting the T0EN flag enables the interrupt and vice versa.

The user can enter the IDLE mode with the Timer T0 interrupt enabled. In this case, when the T0PND bit gets set, the device will first execute the Timer T0 interrupt service routine and then return to the instruction following the "Enter Idle Mode" instruction.

Alternatively, the user can enter the IDLE mode with the IDLE Timer T0 interrupt disabled. In this case, the device will resume normal operation with the instruction immediately following the "Enter IDLE Mode" instruction.

Note: It is necessary to program two NOP instructions following both the set HALT mode and set IDLE mode instructions. These NOP instructions are necessary to allow clock resynchronization following the HALT or IDLE modes.

Multi-Input Wakeup

The Multi-Input Wakeup feature is ued to return (wakeup) the device from either the HALT or IDLE modes. Alternately Multi-Input Wakeup/Interrupt feature may also be used to generate up to 8 edge selectable external interrupts.

Figure 11 shows the Multi-Input Wakeup logic.

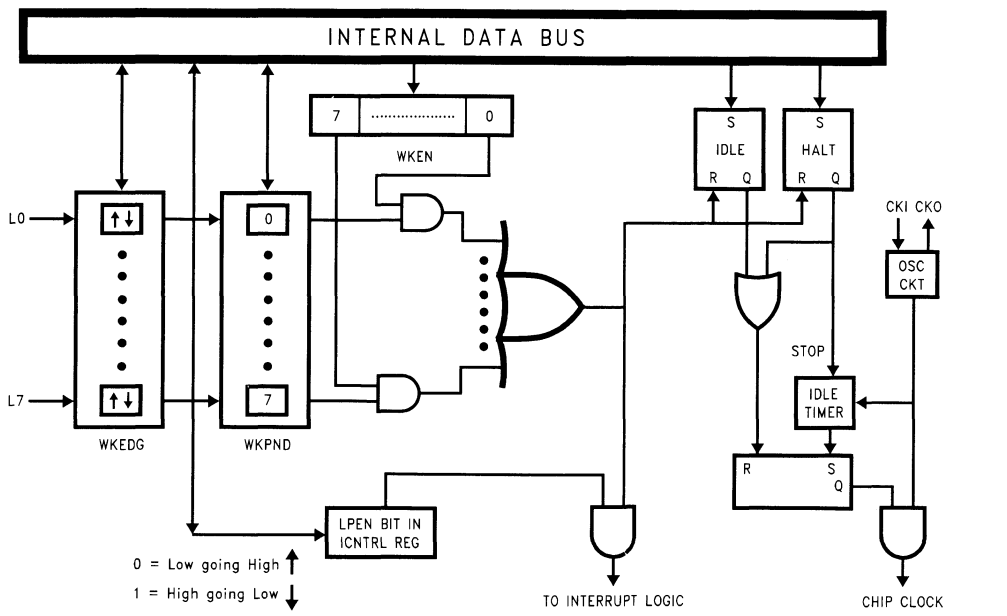


FIGURE 11. Multi-Input Wake Up Logic

Multi-Input Wakeup (Continued)

The Multi-Input Wakeup feature utilizes the L Port. The user selects which particular L port bit (or combination of L Port bits) will cause the device to exit the HALT or IDLE modes. The selection is done through the Reg: WKEN. The Reg: WKEN is an 8-bit read/write register, which contains a control bit for every L port bit. Setting a particular WKEN bit enables a Wakeup from the associated L port pin.

The user can select whether the trigger condition on the selected L Port pin is going to be either a positive edge (low to high transition) or a negative edge (high to low transition). This selection is made via the Reg: WKEDG, which is an 8-bit control register with a bit assigned to each L Port pin. Setting the control bit will select the trigger condition to be a negative edge on that particular L Port pin. Resetting the bit selects the trigger condition to be a positive edge. Changing an edge select entails several steps in order to avoid a pseudo Wakeup condition as a result of the edge change. First, the associated WKEN bit should be reset, followed by the edge select change in WKEDG. Next, the associated WKPND bit should be cleared, followed by the associated WKEN bit being re-enabled.

An example may serve to clarify this procedure. Suppose we wish to change the edge select from positive (low going high) to negative (high going low) for L Port bit 5, where bit 5 has previously been enabled for an input interrupt. The program would be as follows:

```
RBIT  5, WKEN      ; Disable Port bit 5 for
                   ; Wakeup
SBIT  5, WKEDG     ; Select neg going edge
                   ; sensitivity
RBIT  5, WKPND     ; Clear pending bit
SBIT  5, WKEN      ; Re-enable the bit for
                   ; Wakeup
```

If the L port bits have been used as outputs and then changed to inputs with Multi-Input Wakeup/Interrupt, a safety procedure should also be followed to avoid inherited pseudo wakeup conditions. After the selected L port bits have been changed from output to input but before the associated WKEN bits are enabled, the associated edge select bits in WKEDG should be set or reset for the desired edge selects, followed by the associated WKPND bits being cleared.

This same procedure should be used following reset, since the L port inputs are left floating as a result of reset.

The occurrence of the selected trigger condition for Multi-Input Wakeup is latched into a pending register called WKPND. The respective bits of the WKPND register will be set on the occurrence of the selected trigger edge on the corresponding Port L pin. The user has the responsibility of clearing these pending flags. Since WKPND is a pending register for the occurrence of selected wakeup conditions, the device will not enter the HALT mode if any Wakeup bit is both enabled and pending. Consequently, the user has the responsibility of clearing the pending flags before attempting to enter the HALT mode.

WKEN, WKPND and WKEDG are all read/write registers, and are cleared at reset.

PORT L INTERRUPTS

Port L provides the user with an additional eight fully selectable, edge sensitive interrupts which are all vectored into the same service subroutine.

The interrupt from Port L shares logic with the wake up circuitry. The register WKEN allows interrupts from Port L to be individually enabled or disabled. The register WKEDG specifies the trigger condition to be either a positive or a negative edge. Finally, the register WKPND latches in the pending trigger conditions.

The GIE (Global Interrupt Enable) bit enables the interrupt function.

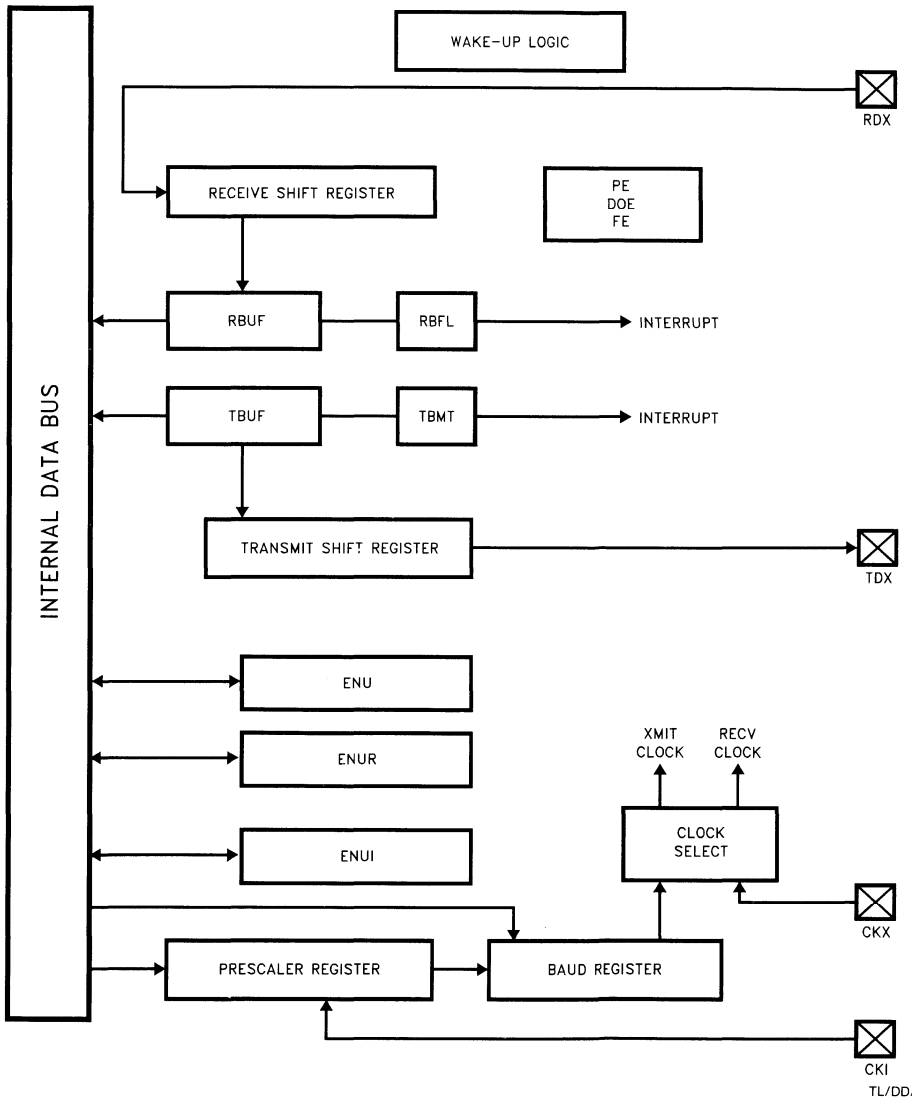
A control flag, LPEN, functions as a global interrupt enable for Port L interrupts. Setting the LPEN flag will enable interrupts and vice versa. A separate global pending flag is not needed since the register WKPND is adequate.

Since Port L is also used for waking the device out of the HALT or IDLE modes, the user can elect to exit the HALT or IDLE modes either with or without the interrupt enabled. If he elects to disable the interrupt, then the device will restart execution from the instruction immediately following the instruction that placed the microcontroller in the HALT or IDLE modes. In the other case, the device will first execute the interrupt service routine and then revert to normal operation. (See HALT MODE for clock option wakeup information.)

UART

The device contains a full-duplex software programmable UART. The UART *(Figure 12)* consists of a transmit shift register, a receive shift register and seven addressable registers, as follows: a transmit buffer register (TBUF), a receiver buffer register (RBUF), a UART control and status register (ENU), a UART receive control and status register (ENUR), a UART interrupt and clock source register (ENUI), a prescaler select register (PSR) and baud (BAUD) register. The ENU register contains flags for transmit and receive functions; this register also determines the length of the data frame (7, 8 or 9 bits), the value of the ninth bit in transmission, and parity selection bits. The ENUR register flags framing, data overrun and parity errors while the UART is receiving.

Other functions of the ENUR register include saving the ninth bit received in the data frame, enabling or disabling the UART's attention mode of operation and providing additional receiver/transmitter status information via RCVG and XMTG bits. The determination of an internal or external clock source is done by the ENUI register, as well as selecting the number of stop bits and enabling or disabling transmit and receive interrupts. A control flag in this register can also select the UART mode of operation: asynchronous or synchronous.



TL/DD/12858–14

FIGURE 12. UART Block Diagram

UART (Continued)

UART CONTROL AND STATUS REGISTERS

The operation of the UART is programmed through three registers: ENU, ENUR and ENUI. The function of the individual bits in these registers is as follows:

ENU-UART Control and Status Register (Address at 0BA)

PEN	PSEL1	XBIT9/ PSEL0	CHL1	CHL0	ERR	RBFL	TBMT
0RW	0RW	0RW	0RW	0RW	0R	0R	1R

Bit 7 Bit 0

ENUR-UART Receive Control and Status Register (Address at 0BB)

DOE	FE	PE	SPARE	RBIT9	ATTN	XMTG	RCVG
0RD	0RD	0RD	0RW*	0R	0RW	0R	0R

Bit 7 Bit 0

ENUI-UART Interrupt and Clock Source Register (Address at 0BC)

STP2	STP78	ETDX	SSEL	XRCLK	XTCLK	ERI	ETI
0RW	0RW	0RW	0RW	0RW	0RW	0RW	0RW

Bit 7 Bit 0

*Bit is not used.

0 Bit is cleared on reset.

1 Bit is set to one on reset.

R Bit is read-only; it cannot be written by software.

RW Bit is read/write.

D Bit is cleared on read; when read by software as a one, it is cleared automatically. Writing to the bit does not affect its state.

DESCRIPTION OF UART REGISTER BITS

ENU—UART CONTROL AND STATUS REGISTER

TBMT: This bit is set when the UART transfers a byte of data from the TBUF register into the TSFT register for transmission. It is automatically reset when software writes into the TBUF register.

RBFL: This bit is set when the UART has received a complete character and has copied it into the RBUF register. It is automatically reset when software reads the character from RBUF.

ERR: This bit is a global UART error flag which gets set if any or a combination of the errors (DOE, FE, PE) occur.

CHL1, CHL0: These bits select the character frame format. Parity is not included and is generated/verified by hardware.

CHL1 = 0, CHL0 = 0 The frame contains eight data bits.
CHL1 = 0, CHL0 = 1 The frame contains seven data bits.
CHL1 = 1, CHL0 = 0 The frame contains nine data bits.
CHL1 = 1, CHL0 = 1 Loopback Mode selected. Transmitter output internally looped back to receiver input. Nine bit framing format is used.

XBIT9/PSEL0: Programs the ninth bit for transmission when the UART is operating with nine data bits per frame. For seven or eight data bits per frame, this bit in conjunction with PSEL1 selects parity.

PSEL1, PSEL0: Parity select bits.

PSEL1 = 0, PSEL0 = 0 Odd Parity (if Parity enabled)
PSEL1 = 0, PSEL0 = 1 Odd Parity (if Parity enabled)
PSEL1 = 1, PSEL0 = 0 Mark(1) (if Parity enabled)
PSEL1 = 1, PSEL0 = 1 Space(0) (if Parity enabled)

PEN: This bit enables/disables Parity (7- and 8-bit modes only).

PEN = 0 Parity disabled.
PEN = 1 Parity enabled.

ENUR—UART RECEIVE CONTROL AND STATUS REGISTER

RCVG: This bit is set high whenever a framing error occurs and goes low when RDX goes high.

XMTG: This bit is set to indicate that the UART is transmitting. It gets reset at the end of the last frame (end of last Stop bit).

ATTN: ATTENTION Mode is enabled while this bit is set. This bit is cleared automatically on receiving a character with data bit nine set.

RBIT9: Contains the ninth data bit received when the UART is operating with nine data bits per frame.

SPARE: Reserved for future use.

PE: Flags a Parity Error.

PE = 0 Indicates no Parity Error has been detected since the last time the ENUR register was read.
PE = 1 Indicates the occurrence of a Parity Error.

FE: Flags a Framing Error.

FE = 0 Indicates no Framing Error has been detected since the last time the ENUR register was read.
FE = 1 Indicates the occurrence of a Framing Error.

DOE: Flags a Data Overrun Error.

DOE = 0 Indicates no Data Overrun Error has been detected since the last time the ENUR register was read.
DOE = 1 Indicates the occurrence of a Data Overrun Error.

ENUI—UART INTERRUPT AND CLOCK SOURCE REGISTER

ETI: This bit enables/disables interrupt from the transmitter section.

ETI = 0 Interrupt from the transmitter is disabled.
ETI = 1 Interrupt from the transmitter is enabled.

ERI: This bit enables/disables interrupt from the receiver section.

ERI = 0 Interrupt from the receiver is disabled.
ERI = 1 Interrupt from the receiver is enabled.

XTCLK: This bit selects the clock source for the transmitter section.

XTCLK = 0 The clock source is selected through the PSR and BAUD registers.
XTCLK = 1 Signal on CKX (L1) pin is used as the clock.

XRCLK: This bit selects the clock source for the receiver section.

XRCLK = 0 The clock source is selected through the PSR and BAUD registers.
XRCLK = 1 Signal on CKX (L1) pin is used as the clock.

SSEL: UART mode select.

SSEL = 0 Asynchronous Mode.
SSEL = 1 Synchronous Mode.

UART (Continued)

ETDX: TDX (UART Transmit Pin) is the alternate function assigned to Port L pin L2; it is selected by setting ETDX bit. To simulate line break generation, software should reset ETDX bit and output logic zero to TDX pin through Port L data and configuration registers.

STP78: This bit is set to program the last Stop bit to be 7/8th of a bit in length.

STP2: This bit programs the number of Stop bits to be transmitted.

STP2 = 0 One Stop bit transmitted.
STP2 = 1 Two Stop bits transmitted.

Associated I/O Pins

Data is transmitted on the TDX pin and received on the RDX pin. TDX is the alternate function assigned to Port L pin L2; it is selected by setting ETDX (in the ENUI register) to one. RDX is an inherent function of Port L pin L3, requiring no setup.

The baud rate clock for the UART can be generated on-chip, or can be taken from an external source. Port L pin L1 (CKX) is the external clock I/O pin. The CKX pin can be either an input or an output, as determined by Port L Configuration and Data registers (Bit 1). As an input, it accepts a clock signal which may be selected to drive the transmitter and/or receiver. As an output, it presents the internal Baud Rate Generator output.

UART Operation

The UART has two modes of operation: asynchronous mode and synchronous mode.

ASYNCHRONOUS MODE

This mode is selected by resetting the SSEL (in the ENUI register) bit to zero. The input frequency to the UART is 16 times the baud rate.

The TSFT and TBUF registers double-buffer data for transmission. While TSFT is shifting out the current character on the TDX pin, the TBUF register may be loaded by software with the next byte to be transmitted. When TSFT finishes transmitting the current character the contents of TBUF are transferred to the TSFT register and the Transmit Buffer Empty Flag (TBMT in the ENU register) is set. The TBMT flag is automatically reset by the UART when software loads a new character into the TBUF register. There is also the XMTG bit which is set to indicate that the UART is transmitting. This bit gets reset at the end of the last frame (end of last Stop bit). TBUF is a read/write register.

The RSFT and RBUF registers double-buffer data being received. The UART receiver continually monitors the signal on the RDX pin for a low level to detect the beginning of a Start bit. Upon sensing this low level, it waits for half a bit time and samples again. If the RDX pin is still low, the receiver considers this to be a valid Start bit, and the remaining bits in the character frame are each sampled a single time, at the mid-bit position. Serial data input on the RDX pin is shifted into the RSFT register. Upon receiving the complete character, the contents of the RSFT register are copied into the RBUF register and the Received Buffer Full Flag (RBFL) is set. RBFL is automatically reset when software reads the character from the RBUF register. RBUF is a read only register. There is also the RCVG bit which is set high when a framing error occurs and goes low once RDX goes high. TBMT, XMTG, RBFL and RCVG are read only bits.

SYNCHRONOUS MODE

In this mode data is transferred synchronously with the clock. Data is transmitted on the rising edge and received on the falling edge of the synchronous clock.

This mode is selected by setting SSEL bit in the ENUI register. The input frequency to the UART is the same as the baud rate.

When an external clock input is selected at the CKX pin, data transmit and receive are performed synchronously with this clock through TDX/RDX pins.

If data transmit and receive are selected with the CKX pin as clock output, the device generates the synchronous clock output at the CKX pin. The internal baud rate generator is used to produce the synchronous clock. Data transmit and receive are performed synchronously with this clock.

FRAMING FORMATS

The UART supports several serial framing formats *(Figure 13).* The format is selected using control bits in the ENU, ENUR and ENUI registers.

The first format (1, 1a, 1b, 1c) for data transmission (CHL0 = 1, CHL1 = 0) consists of Start bit, seven Data bits (excluding parity) and 7/8, one or two Stop bits. In applications using parity, the parity bit is generated and verified by hardware.

The second format (CHL0 = 0, CHL1 = 0) consists of one Start bit, eight Data bits (excluding parity) and 7/8, one or two Stop bits. Parity bit is generated and verified by hardware.

The third format for transmission (CHL0 = 0, CHL1 = 1) consists of one Start bit, nine Data bits and 7/8, one or two Stop bits. This format also supports the UART "ATTENTION" feature. When operating in this format, all eight bits of TBUF and RBUF are used for data. The ninth data bit is transmitted and received using two bits in the ENU and ENUR registers, called XBIT9 and RBIT9. RBIT9 is a read only bit. Parity is not generated or verified in this mode.

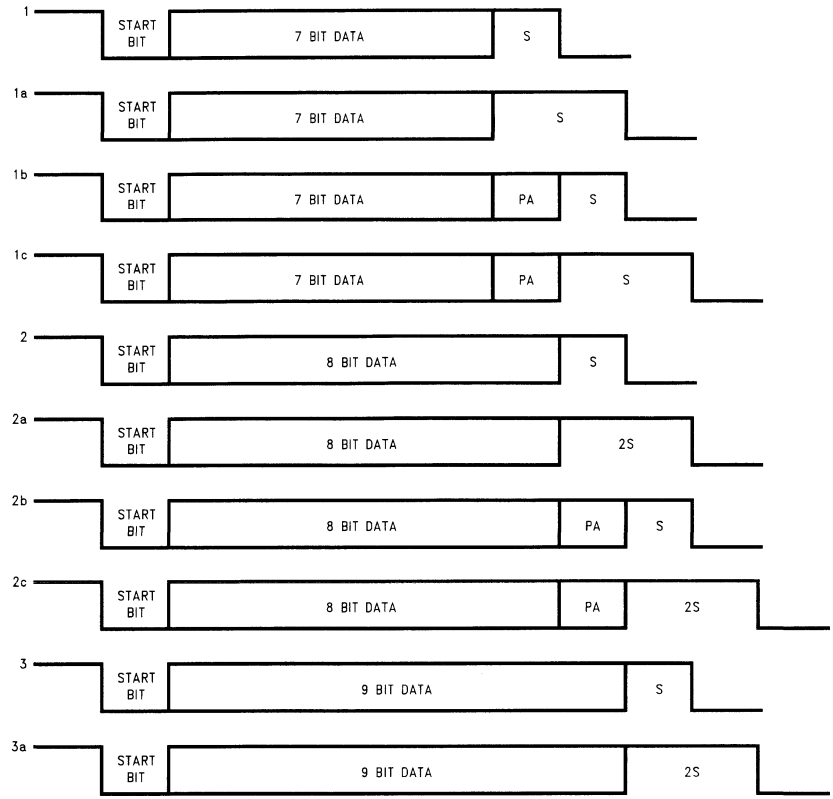
For any of the above framing formats, the last Stop bit can be programmed to be 7/8th of a bit in length. If two Stop bits are selected and the 7/8th bit is set (selected), the second Stop bit will be 7/8th of a bit in length.

The parity is enabled/disabled by PEN bit located in the ENU register. Parity is selected for 7- and 8-bit modes only. If parity is enabled (PEN = 1), the parity selection is then performed by PSEL0 and PSEL1 bits located in the ENU register.

Note that the XBIT9/PSEL0 bit located in the ENU register serves two mutually exclusive functions. This bit programs the ninth bit for transmission when the UART is operating with nine data bits per frame. There is no parity selection in this framing format. For other framing formats XBIT9 is not needed and the bit is PSEL0 used in conjunction with PSEL1 to select parity.

The frame formats for the receiver differ from the transmitter in the number of Stop bits required. The receiver only requires one Stop bit in a frame, regardless of the setting of the Stop bit selection bits in the control register. Note that an implicit assumption is made for full duplex UART operation that the framing formats are the same for the transmitter and receiver.

UART Operation (Continued)



TL/DD/12858–15

FIGURE 13. Framing Formats

UART INTERRUPTS

The UART is capable of generating interrupts. Interrupts are generated on Receive Buffer Full and Transmit Buffer Empty. Both interrupts have individual interrupt vectors. Two bytes of program memory space are reserved for each interrupt vector. The two vectors are located at addresses 0xEC to 0xEF Hex in the program memory space. The interrupts can be individually enabled or disabled using Enable Transmit Interrupt (ETI) and Enable Receive Interrupt (ERI) bits in the ENUI register.

The interrupt from the Transmitter is set pending, and remains pending, as long as both the TBMT and ETI bits are set. To remove this interrupt, software must either clear the ETI bit or write to the TBUF register (thus clearing the TBMT bit).

The interrupt from the receiver is set pending, and remains pending, as long as both the RBFL and ERI bits are set. To remove this interrupt, software must either clear the ERI bit or read from the RBUF register (thus clearing the RBFL bit).

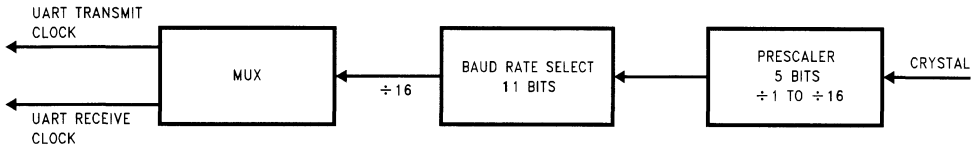
Baud Clock Generation

The clock inputs to the transmitter and receiver sections of the UART can be individually selected to come either from an external source at the CKX pin (port L, pin L1) or from a source selected in the PSR and BAUD registers. Internally, the basic baud clock is created from the oscillator frequency through a two-stage divider chain consisting of a 1–16 (increments of 0.5) prescaler and an 11-bit binary counter. *(Figure 14).* The divide factors are specified through two read/write registers shown in *Figure 15.* Note that the 11-bit Baud Rate Divisor spills over into the Prescaler Select Register (PSR). PSR is cleared upon reset.

As shown in Table IV, a Prescaler Factor of 0 corresponds to NO CLOCK. This condition is the UART power down mode where the UART clock is turned off for power saving purpose. The user must also turn the UART clock off when a different baud rate is chosen.

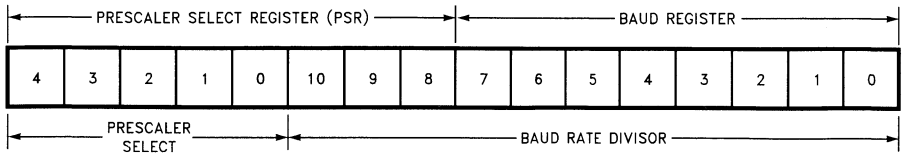
The correspondences between the 5-bit Prescaler Select and Prescaler factors are shown in Table IV. There are many ways to calculate the two divisor factors, but one particularly effective method would be to achieve a 1.8432 MHz frequency coming out of the first stage. The 1.8432 MHz prescaler output is then used to drive the software programmable baud rate counter to create a 16x clock for the following baud rates: 110, 134.5, 150, 300, 600, 1200, 1800, 2400, 3600, 4800, 7200, 9600, 19200 and 38400 (Table III). Other baud rates may be created by using appropriate divisors. The 16x clock is then divided by 16 to provide the rate for the serial shift registers of the transmitter and receiver.

Baud Clock Generation (Continued)



TL/DD/12858–16

FIGURE 14. UART BAUD Clock Generation



TL/DD/12858–17

FIGURE 15. UART BAUD Clock Divisor Registers

Baud Clock Generation (Continued)

TABLE III. Baud Rate Divisors (1.8432 MHz Prescaler Output)

Baud Rate	Baud Rate Divisor − 1 (N-1)
110 (110.03)	1046
134.5 (134.58)	855
150	767
300	383
600	191
1200	95
1800	63
2400	47
3600	31
4800	23
7200	15
9600	11
19200	5
38400	2

Note: The entries in Table IV assume a prescaler output of 1.8432 MHz. In the asynchronous mode the baud rate could be as high as 625k.

TABLE IV. Prescaler Factors

Prescaler Select	Prescaler Factor
00000	NO CLOCK
00001	1
00010	1.5
00011	2
00100	2.5
00101	3
00110	3.5
00111	4
01000	4.5
01001	5
01010	5.5
01011	6
01100	6.5
01101	7
01110	7.5
01111	8
10000	8.5
10001	9
10010	9.5
10011	10
10100	10.5
10101	11
10110	11.5
10111	12
11000	12.5
11001	13
11010	13.5
11011	14
11100	14.5
11101	15
11110	15.5
11111	16

As an example, considering Asynchronous Mode and a CKI clock of 4.608 MHz, the prescaler factor selected is:

$$4.608/1.8432 = 2.5$$

The 2.5 entry is available in Table IV. The 1.8432 MHz prescaler output is then used with proper Baud Rate Divisor (Table IV) to obtain different baud rates. For a baud rate of 19200 e.g., the entry in Table IV is 5.

$$N - 1 = 5 \text{ (N} - 1 \text{ is the value from Table III)}$$

$$N = 6 \text{ (N is the Baud Rate Divisor)}$$

$$\text{Baud Rate} = 1.8432 \text{ MHz}/(16 \times 6) = 19200$$

The divide by 16 is performed because in the asynchronous mode, the input frequency to the UART is 16 times the baud rate. The equation to calculate baud rates is given below.

The actual Baud Rate may be found from:

$$BR = Fc/(16 \times N \times P)$$

Baud Clock Generation (Continued)

Where:

BR is the Baud Rate

Fc is the CKI frequency

N is the Baud Rate Divisor (Table III).

P is the Prescaler Divide Factor selected by the value in the Prescaler Select Register (Table IV)

Note: In the Synchronous Mode, the divisor 16 is replaced by two.

Example:

Asynchronous Mode:

Crystal Frequency = 5 MHz

Desired baud rate = 9600

Using the above equation N × P can be calculated first.

$$N \times P = (5 \times 106)/(16 \times 9600) = 32.552$$

Now 32.552 is divided by each Prescaler Factor (Table IV) to obtain a value closest to an integer. This factor happens to be 6.5 (P = 6.5).

$$N = 32.552/6.5 = 5.008 \ (N = 5)$$

The programmed value (from Table III) should be 4 (N − 1).

Using the above values calculated for N and P:

$$BR = (5 \times 106)/(16 \times 5 \times 6.5) = 9615.384$$

$$\% \text{ error} = (9615.385 - 9600)/9600 = 0.16$$

Effect of HALT/IDLE

The UART logic is reinitialized when either the HALT or IDLE modes are entered. This reinitialization sets the TBMT flag and resets all read only bits in the UART control and status registers. Read/Write bits remain unchanged. The Transmit Buffer (TBUF) is not affected, but the Transmit Shift register (TSFT) bits are set to one. The receiver registers RBUF and RSFT are not affected.

The device will exit from the HALT/IDLE modes when the Start bit of a character is detected at the RDX (L3) pin. This feature is obtained by using the Multi-Input Wakeup scheme provided on the device.

Before entering the HALT or IDLE modes the user program must select the Wakeup source to be on the RDX pin. This selection is done by setting bit 3 of WKEN (Wakeup Enable) register. The Wakeup trigger condition is then selected to be high to low transition. This is done via the WKEDG register (Bit 3 is one.)

If the device is halted and crystal oscillator is used, the Wakeup signal will not start the chip running immediately because of the finite start up time requirement of the crystal oscillator. The idle timer (T0) generates a fixed (256 t_C) delay to ensure that the oscillator has indeed stabilized before allowing the device to execute code. The user has to consider this delay when data transfer is expected immediately after exiting the HALT mode.

Diagnostic

Bits CHARL0 and CHARL1 in the ENU register provide a loopback feature for diagnostic testing of the UART. When these bits are set to one, the following occur: The receiver input pin (RDX) is internally connected to the transmitter output pin (TDX); the output of the Transmitter Shift Register is "looped back" into the Receive Shift Register input. In this mode, data that is transmitted is immediately received. This feature allows the processor to verify the transmit and receive data paths of the UART.

Note that the framing format for this mode is the nine bit format; one Start bit, nine data bits, and 7/8, one or two Stop bits. Parity is not generated or verified in this mode.

Attention Mode

The UART Receiver section supports an alternate mode of operation, referred to as ATTENTION Mode. This mode of operation is selected by the ATTN bit in the ENUR register. The data format for transmission must also be selected as having nine Data bits and either 7/8, one or two Stop bits.

The ATTENTION mode of operation is intended for use in networking the device with other processors. Typically in such environments the messages consists of device addresses, indicating which of several destinations should receive them, and the actual data. This Mode supports a scheme in which addresses are flagged by having the ninth bit of the data field set to a 1. If the ninth bit is reset to a zero the byte is a Data byte.

While in ATTENTION mode, the UART monitors the communication flow, but ignores all characters until an address character is received. Upon receiving an address character, the UART signals that the character is ready by setting the RBFL flag, which in turn interrupts the processor if UART Receiver interrupts are enabled. The ATTN bit is also cleared automatically at this point, so that data characters as well as address characters are recognized. Software examines the contents of the RBUF and responds by deciding either to accept the subsequent data stream (by leaving the ATTN bit reset) or to wait until the next address character is seen (by setting the ATTN bit again).

Operation of the UART Transmitter is not affected by selection of this Mode. The value of the ninth bit to be transmitted is programmed by setting XBIT9 appropriately. The value of the ninth bit received is obtained by reading RBIT9. Since this bit is located in ENUR register where the error flags reside, a bit operation on it will reset the error flags.

Comparators

The device contains two differential comparators, each with a pair of inputs (positive and negative) and an output. Ports I1-I3 and I4-I6 are used for the comparators. The following is the Port I assignment:

- I1 Comparator1 negative input
- I2 Comparator1 positive input
- I3 Comparator1 output
- I4 Comparator2 negative input
- I5 Comparator2 positive input
- I6 Comparator2 output

A Comparator Select Register (CMPSL) is used to enable the comparators, read the outputs of the comparators internally, and enable the outputs of the comparators to the pins. Two control bits (enable and output enable) and one result bit are associated with each comparator. The comparator result bits (CMP1RD and CMP2RD) are read only bits which will read as zero if the associated comparator is not enabled. The Comparator Select Register is cleared with reset, resulting in the comparators being disabled. The comparators should also be disabled before entering either the HALT or IDLE modes in order to save power. The configuration of the CMPSL register is as follows:

Comparators (Continued)

CMPSL REGISTER (ADDRESS X'00B7)

The CMPSL register contains the following bits:

CMP1EN Enable comparator 1

CMP1RD Comparator 1 result (this is a read only bit, which will read as 0 if the comparator is not enabled)

CMP10E Selects pin I3 as comparator 1 output provided that CMPIEN is set to enable the comparator

CMP2EN Enable comparator 2

CMP2RD Comparator 2 result (this is a read only bit, which will read as 0 if the comparator is not enabled)

CMP20E Selects pin I6 as comparator 2 output provided that CMP2EN is set to enable the comparator

Unused	CMP20E	CMP2RD	CMP2EN	CMP10E	CMP1RD	CMP1EN	Unused
Bit 7							Bit 0

Note that the two unused bits of CMPSL may be used as software flags.

Comparator outputs have the same spec as Ports L and G except that the rise and fall times are symmetrical.

Interrupts

The device supports a vectored interrupt scheme. It supports a total of fourteen interrupt sources. The following table lists all the possible interrupt sources, their arbitration ranking and the memory locations reserved for the interrupt vector for each source.

Two bytes of program memory space are reserved for each interrupt source. All interrupt sources except the software interrupt are maskable. Each of the maskable interrupts have an Enable bit and one or more Pending bits. A maskable interrupt is active if its associated enable and pending bits are set. If GIE = 1 and an interrupt is active, then the processor will be interrupted as soon as it is ready to start executing an instruction except if the above conditions happen during the Software Trap service routine. This exception is described in the Software Trap sub-section.

The interruption process is accomplished with the INTR instruction (opcode 00), which is jammed inside the Instruction Register and replaces the opcode about to be executed. The following steps are performed for every interrupt:

1. The GIE (Global Interrupt Enable) bit is reset.
2. The address of the instruction about to be executed is pushed into the stack.
3. The PC (Program Counter) branches to address 00FF.

This procedure takes 7 t_C cycles to execute.

Arbitration Ranking	Source	Description	Vector* Address Hi-Low Byte
(1) Highest	Software	INTR Instruction	0yFE–0yFF
(2)	Reserved		0yFC–0yFD
(3)	External	G0	0yFA–0yFB
(4)	Timer T0	Underflow	0yF8–0yF9
(5)	Timer T1	T1A/Underflow	0yF6–0yF7
(6)	Timer T1	T1B	0yF4–0yF5
(7)	MICROWIRE/PLUS	BUSY Low	0yF2–0yF3
(8)	Reserved		0yF0–0yF1
(9)	UART	Receive	0yEE–0yEF
(10)	UART	Transmit	0yEC–0yED
(11)	Timer T2	T2A/Underflow	0yEA–0yEB
(12)	Timer T2	T2B	0yE8–0yE9
(13)	Timer T3	T3A/Underflow	0yE6–0yE7
(14)	Timer T3	T3B	0yE4–0yE5
(15)	Port L/Wakeup	Port L Edge	0yE2–0yE3
(16) Lowest	Default VIS	Reserved	0yE0–0yE1

*y is a variable which represents the VIS block. VIS and the vector table must be located in the same 256-byte block except if VIS is located at the last address of a block. In this case, the table must be in the next block.

Interrupts (Continued)

At this time, since GIE = 0, other maskable interrupts are disabled. The user is now free to do whatever context switching is required by saving the context of the machine in the stack with PUSH instructions. The user would then program a VIS (Vector Interrupt Select) instruction in order to branch to the interrupt service routine of the highest priority interrupt enabled and pending at the time of the VIS. Note that this is not necessarily the interrupt that caused the branch to address location 00FF Hex prior to the context switching.

Thus, if an interrupt with a higher rank than the one which caused the interruption becomes active before the decision of which interrupt to service is made by the VIS, then the interrupt with the higher rank will override any lower ones and will be acknowledged. The lower priority interrupt(s) are still pending, however, and will cause another interrupt immediately following the completion of the interrupt service routine associated with the higher priority interrupt just serviced. This lower priority interrupt will occur immediately following the RETI (Return from Interrupt) instruction at the end of the interrupt service routine just completed.

Inside the interrupt service routine, the associated pending bit has to be cleared by software. The RETI (Return from Interrupt) instruction at the end of the interrupt service routine will set the GIE (Global Interrupt Enable) bit, allowing the processor to be interrupted again if another interrupt is active and pending.

The VIS instruction looks at all the active interrupts at the time it is executed and performs an indirect jump to the beginning of the service routine of the one with the highest rank.

The addresses of the different interrupt service routines, called vectors, are chosen by the user and stored in ROM in a table starting at 01E0 (assuming that VIS is located between 00FF and 01DF). The vectors are 15-bit wide and therefore occupy 2 ROM locations.

VIS and the vector table must be located in the same 256-byte block (0y00 to 0yFF) except if VIS is located at the last address of a block. In this case, the table must be in the next block. The vector table cannot be inserted in the first 256-byte block (y ≠ 0).

The vector of the maskable interrupt with the lowest rank is located at 0yE0 (Hi-Order byte) and 0yE1 (Lo-Order byte) and so forth in increasing rank number. The vector of the maskable interrupt with the highest rank is located at 0yFA (Hi-Order byte) and 0yFB (Lo-Order byte).

The Software Trap has the highest rank and its vector is located at 0yFE and 0yFF.

If, by accident, a VIS gets executed and no interrupt is active, then the PC (Program Counter) will branch to a vector located at 0yE0–0yE1.

WARNING

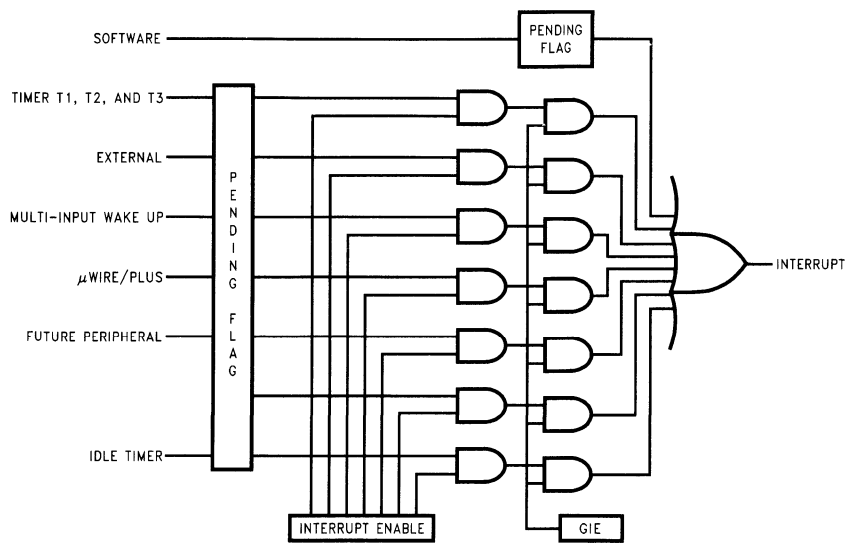
A Default VIS interrupt handler routine must be present. As a minimum, this handler should confirm that the GIE bit is cleared (this indicates that the interrupt sequence has been taken), take care of any required housekeeping, restore context and return. Some sort of Warm Restart procedure should be implemented. These events can occur without any error on the part of the system designer or programmer.

Note: There is always the possibility of an interrupt occurring during an instruction which is attempting to reset the GIE bit or any other interrupt enable bit. If this occurs when a single cycle instruction is being used to reset the interrupt enable bit, the interrupt enable bit will be reset but an interrupt may still occur. This is because interrupt processing is started at the same time as the interrupt bit is being reset. To avoid this scenario, the user should always use a two, three, or four cycle instruction to reset interrupt enable bits.

Figure 16 shows the Interrupt block diagram.

SOFTWARE TRAP

The Software Trap (ST) is a special kind of non-maskable interrupt which occurs when the INTR instruction (used to acknowledge interrupts) is fetched from ROM and placed inside the instruction register. This may happen when the PC is pointing beyond the available ROM address space or when the stack is over-popped.



TL/DD/12858–18

FIGURE 16. Interrupt Block Diagram

5

Interrupts (Continued)

When an ST occurs, the user can re-initialize the stack pointer and do a recovery procedure (similar to reset, but not necessarily containing all of the same initialization procedures) before restarting.

The occurrence of an ST is latched into the ST pending bit. The GIE bit is not affected and the ST pending bit **(not accessible by the user)** is used to inhibit other interrupts and to direct the program to the ST service routine with the VIS instruction. The RPND instruction is used to clear the software interrupt pending bit. This pending bit is also cleared on reset.

The ST has the highest rank among all interrupts.

Nothing (except another ST) can interrupt an ST being serviced.

WATCHDOG

The device contains a WATCHDOG and clock monitor. The WATCHDOG is designed to detect the user program getting stuck in infinite loops resulting in loss of program control or "runaway" programs. The Clock Monitor is used to detect the absence of a clock or a very slow clock below a specified rate on the CKI pin.

The WATCHDOG consists of two independent logic blocks: WD UPPER and WD LOWER. WD UPPER establishes the upper limit on the service window and WD LOWER defines the lower limit of the service window.

Servicing the WATCHDOG consists of writing a specific value to a WATCHDOG Service Register named WDSVR which is memory mapped in the RAM. This value is composed of three fields, consisting of a 2-bit Window Select, a 5-bit Key Data field, and the 1-bit Clock Monitor Select field. Table V shows the WDSVR register.

TABLE V. WATCHDOG Service Register (WDSVR)

Window Select		Key Data					Clock Monitor
X	X	0	1	1	0	0	Y
7	6	5	4	3	2	1	0

The lower limit of the service window is fixed at 2048 instruction cycles. Bits 7 and 6 of the WDSVR register allow the user to pick an upper limit of the service window.

Table VI shows the four possible combinations of lower and upper limits for the WATCHDOG service window. This flexibility in choosing the WATCHDOG service window prevents any undue burden on the user software.

Bits 5, 4, 3, 2 and 1 of the WDSVR register represent the 5-bit Key Data field. The key data is fixed at 01100. Bit 0 of the WDSVR Register is the Clock Monitor Select bit.

TABLE VI. WATCHDOG Service Window Select

WDSVR Bit 7	WDSVR Bit 6	Service Window (Lower-Upper Limits)
0	0	2k–8k t_C Cycles
0	1	2k–16k t_C Cycles
1	0	2k–32k t_C Cycles
1	1	2k–64k t_C Cycles

Clock Monitor

The Clock Monitor aboard the device can be selected or deselected under program control. The Clock Monitor is guaranteed not to reject the clock if the instruction cycle clock ($1/t_C$) is greater or equal to 10 kHz. This equates to a clock input rate on CKI of greater or equal to 100 kHz.

WATCHDOG Operation

The WATCHDOG and Clock Monitor are disabled during reset. The device comes out of reset with the WATCHDOG armed, the WATCHDOG Window Select bits (bits 6, 7 of the WDSVR Register) set, and the Clock Monitor bit (bit 0 of the WDSVR Register) enabled. Thus, a Clock Monitor error will occur after coming out of reset, if the instruction cycle clock frequency has not reached a minimum specified value, including the case where the oscillator fails to start.

The WDSVR register can be written to only once after reset and the key data (bits 5 through 1 of the WDSVR Register) must match to be a valid write. This write to the WDSVR register involves two irrevocable choices: (i) the selection of the WATCHDOG service window (ii) enabling or disabling of the Clock Monitor. Hence, the first write to WDSVR Register involves selecting or deselecting the Clock Monitor, select the WATCHDOG service window and match the WATCHDOG key data. Subsequent writes to the WDSVR register will compare the value being written by the user to the WATCHDOG service window value and the key data (bits 7 through 1) in the WDSVR Register. Table VII shows the sequence of events that can occur.

The user must service the WATCHDOG at least once before the upper limit of the service window expires. The WATCHDOG may not be serviced more than once in every lower limit of the service window. The user may service the WATCHDOG as many times as wished in the time period between the lower and upper limits of the service window. The first write to the WDSVR Register is also counted as a WATCHDOG service.

The WATCHDOG has an output pin associated with it. This is the WDOUT pin, on pin 1 of the port G. WDOUT is active low. The WDOUT pin is in the high impedance state in the inactive state. Upon triggering the WATCHDOG, the logic will pull the WDOUT (G1) pin low for an additional 16 t_C–32 t_C cycles after the signal level on WDOUT pin goes below the lower Schmitt trigger threshold. After this delay, the device will stop forcing the WDOUT output low.

The WATCHDOG service window will restart when the WDOUT pin goes high. It is recommended that the user tie the WDOUT pin back to V_{CC} through a resistor in order to pull WDOUT high.

A WATCHDOG service while the WDOUT signal is active will be ignored. The state of the WDOUT pin is not guaranteed on reset, but if it powers up low then the WATCHDOG will time out and WDOUT will enter high impedance state.

The Clock Monitor forces the G1 pin low upon detecting a clock frequency error. The Clock Monitor error will continue until the clock frequency has reached the minimum specified value, after which the G1 output will enter the high impedance TRI-STATE mode following 16 t_C–32 t_C clock cycles. The Clock Monitor generates a continual Clock Monitor error if the oscillator fails to start, or fails to reach the minimum specified frequency. The specification for the Clock Monitor is as follows:

$1/t_C > 10$ kHz—No clock rejection.

$1/t_C < 10$ Hz—Guaranteed clock rejection.

WATCHDOG Operation (Continued)

WATCHDOG AND CLOCK MONITOR SUMMARY

The following salient points regarding the WATCHDOG and CLOCK MONITOR should be noted:

- Both the WATCHDOG and CLOCK MONITOR detector circuits are inhibited during RESET.
- Following RESET, the WATCHDOG and CLOCK MONITOR are both enabled, with the WATCHDOG having he maximum service window selected.
- The WATCHDOG service window and CLOCK MONITOR enable/disable option can only be changed once, during the initial WATCHDOG service following RESET.
- The initial WATCHDOG service must match the key data value in the WATCHDOG Service register WDSVR in order to avoid a WATCHDOG error.
- Subsequent WATCHDOG services must match all three data fields in WDSVR in order to avoid WATCHDOG errors.
- The correct key data value cannot be read from the WATCHDOG Service register WDSVR. Any attempt to read this key data value of 01100 from WDSVR will read as key data value of all 0's.
- The WATCHDOG detector circuit is inhibited during both the HALT and IDLE modes.
- The CLOCK MONITOR detector circuit is active during both the HALT and IDLE modes. Consequently, the device inadvertently entering the HALT mode will be detected as a CLOCK MONITOR error (provided that the CLOCK MONITOR enable option has been selected by the program).
- With the single-pin R/C oscillator mask option selected and the CLKDLY bit reset, the WATCHDOG service window will resume following HALT mode from where it left off before entering the HALT mode.
- With the crystal oscillator mask option selected, or with the single-pin R/C oscillator mask option selected and the CLKDLY bit set, the WATCHDOG service window will be set to its selected value from WDSVR following HALT. Consequently, the WATCHDOG should not be serviced for at least 2048 instruction cycles following HALT, but must be serviced within the selected window to avoid a WATCHDOG error.
- The IDLE timer T0 is not initialized with RESET.
- The user can sync in to the IDLE counter cycle with an IDLE counter (T0) interrupt or by monitoring the T0PND flag. The T0PND flag is set whenever the thirteenth bit of the IDLE counter toggles (every 4096 instruction cycles). The user is responsible for resetting the T0PND flag.
- A hardware WATCHDOG service occurs just as the device exits the IDLE mode. Consequently, the WATCHDOG should not be serviced for at least 2048 instruction cycles following IDLE, but must be serviced within the selected window to avoid a WATCHDOG error.
- Following RESET, the initial WATCHDOG service (where the service window and the CLOCK MONITOR enable/disable must be selected) may be programmed anywhere within the maximum service window (65,536 instruction cycles) initialized by RESET. Note that this initial WATCHDOG service may be programmed within the initial 2048 instruction cycles without causing a WATCHDOG error.

Detection of Illegal Conditions

The device can detect various illegal conditions resulting from coding errors, transient noise, power supply voltage drops, runaway programs, etc.

Reading of undefined ROM gets zeros. The opcode for software interrupt is zero. If the program fetches instructions from undefined ROM, this will force a software interrupt, thus signaling that an illegal condition has occurred.

The subroutine stack grows down for each call (jump to subroutine), interrupt, or PUSH, and grows up for each return or POP. The stack pointer is initialized to RAM location 06F Hex during reset. Consequently, if there are more returns than calls, the stack pointer will point to addresses 070 and 071 Hex (which are undefined RAM). Undefined RAM from addresses 070 to 07F (Segment 0), and all other segments (i.e., Segments 4 . . . etc.) is read as all 1's, which in turn will cause the program to return to address 7FFF Hex. This is an undefined ROM location and the instruction fetched (all 0's) from this location will generate a software interrupt signaling an illegal condition.

Thus, the chip can detect the following illegal conditions:

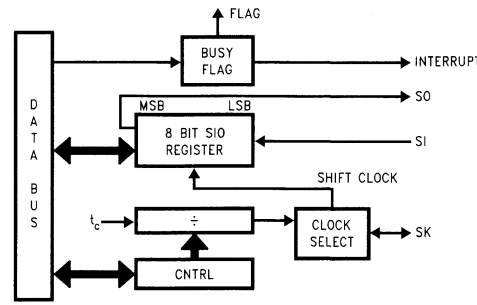
a. Executing from undefined ROM

b. Over "POP"ing the stack by having more returns than calls.

When the software interrupt occurs, the user can re-initialize the stack pointer and do a recovery procedure before restarting (this recovery program is probably similar to that following reset, but might not contain the same program initialization procedures). The recovery program should reset the software interrupt pending bit using the RPND instruction.

MICROWIRE/PLUS

MICROWIRE/PLUS is a serial synchronous communications interface. The MICROWIRE/PLUS capability enables the device to interface with any of National Semiconductor's MICROWIRE peripherals (i.e. A/D converters, display drivers, E^2PROMs etc.) and with other microcontrollers which support the MICROWIRE interface. It consists of an 8-bit serial shift register (SIO) with serial data input (SI), serial data output (SO) and serial shift clock (SK). *Figure 17* shows a block diagram of the MICROWIRE/PLUS logic.



TL/DD/12858–19

FIGURE 17. MICROWIRE/PLUS Block Diagram

The shift clock can be selected from either an internal source or an external source. Operating the MICROWIRE/PLUS arrangement with the internal clock source is called the Master mode of operation. Similarly, operating the MICROWIRE/PLUS arrangement with an external shift clock is called the Slave mode of operation.

The CNTRL register is used to configure and control the MICROWIRE/PLUS mode. To use the MICROWIRE/PLUS, the MSEL bit in the CNTRL register is set to one. In the master mode, the SK clock rate is selected by the two bits, SL0 and SL1, in the CNTRL register. Table VIII details the different clock rates that may be selected.

TABLE VII. WATCHDOG Service Actions

Key Data	Window Data	Clock Monitor	Action
Match	Match	Match	Valid Service: Restart Service Window
Don't Care	Mismatch	Don't Care	Error: Generate WATCHDOG Output
Mismatch	Don't Care	Don't Care	Error: Generate WATCHDOG Output
Don't Care	Don't Care	Mismatch	Error: Generate WATCHDOG Output

TABLE VIII. MICROWIRE/PLUS Master Mode Clock Select

SL1	SL0	SK
0	0	$2 \times t_c$
0	1	$4 \times t_c$
1	x	$8 \times t_c$

Where t_c is the instruction cycle clock

MICROWIRE/PLUS (Continued)

MICROWIRE/PLUS OPERATION

Setting the BUSY bit in the PSW register causes the MICROWIRE/PLUS to start shifting the data. It gets reset when eight data bits have been shifted. The user may reset the BUSY bit by software to allow less than 8 bits to shift. If enabled, an interrupt is generated when eight data bits have been shifted. The device may enter the MICROWIRE/PLUS mode either as a Master or as a Slave. *Figure 18* shows how two microcontroller devices and several peripherals may be interconnected using the MICROWIRE/PLUS arrangements.

Warning:

The SIO register should only be loaded when the SK clock is low. Loading the SIO register while the SK clock is high will result in undefined data in the SIO register. SK clock is normally low when not shifting.

Setting the BUSY flag when the input SK clock is high in the MICROWIRE/PLUS slave mode may cause the current SK clock for the SIO shift register to be narrow. For safety, the BUSY flag should only be set when the input SK clock is low.

MICROWIRE/PLUS Master Mode Operation

In the MICROWIRE/PLUS Master mode of operation the shift clock (SK) is generated internally by the device. The MICROWIRE Master always initiates all data exchanges. The MSEL bit in the CNTRL register must be set to enable the SO and SK functions onto the G Port. The SO and SK pins must also be selected as outputs by setting appropriate bits in the Port G configuration register. Table IX summarizes the bit settings required for Master mode of operation.

MICROWIRE/PLUS Slave Mode Operation

In the MICROWIRE/PLUS Slave mode of operation the SK clock is generated by an external source. Setting the MSEL bit in the CNTRL register enables the SO and SK functions onto the G Port. The SK pin must be selected as an input and the SO pin is selected as an output pin by setting and resetting the appropriate bits in the Port G configuration register. Table IX summarizes the settings required to enter the Slave mode of operation.

The user must set the BUSY flag immediately upon entering the Slave mode. This will ensure that all data bits sent by the Master will be shifted properly. After eight clock pulses the BUSY flag will be cleared and the sequence may be repeated.

Alternate SK Phase Operation

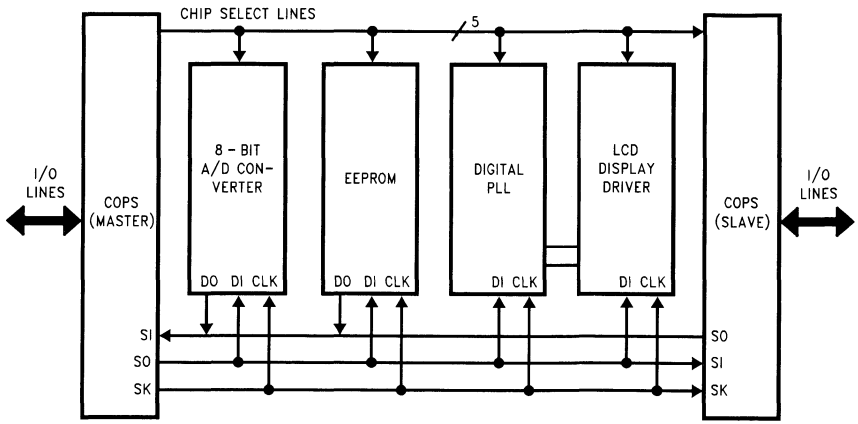
The device allows either the normal SK clock or an alternate phase SK clock to shift data in and out of the SIO register. In both the modes the SK is normally low. In the normal mode data is shifted in on the rising edge of the SK clock and the data is shifted out on the falling edge of the SK clock. The SIO register is shifted on each falling edge of the SK clock. In the alternate SK phase operation, data is shifted in on the falling edge of the SK clock and shifted out on the rising edge of the SK clock.

A control flag, SKSEL, allows either the normal SK clock or the alternate SK clock to be selected. Resetting SKSEL causes the MICROWIRE/PLUS logic to be clocked from the normal SK signal. Setting the SKSEL flag selects the alternate SK clock. The SKSEL is mapped into the G6 configuration bit. The SKSEL flag will power up in the reset condition, selecting the normal SK signal.

TABLE IX

This table assumes that the control flag MSEL is set.

G4 (SO) Config. Bit	G5 (SK) Config. Bit	G4 Fun.	G5 Fun.	Operation
1	1	SO	Int. SK	MICROWIRE/PLUS Master
0	1	TRI-STATE	Int. SK	MICROWIRE/PLUS Master
1	0	SO	Ext. SK	MICROWIRE/PLUS Slave
0	0	TRI-STATE	Ext. SK	MICROWIRE/PLUS Slave



TL/DD/12858–20

FIGURE 18. MICROWIRE/PLUS Application

Memory Map

All RAM, ports and registers (except A and PC) are mapped into data memory address space.

Address S/ADD REG	Contents
0000 to 006F	On-Chip RAM bytes (112 bytes)
0070 to 007F	Unused RAM Address Space (Reads As All Ones)
xx80 to xxAF	Unused RAM Address Space (Reads Undefined Data)
xxB0	Timer T3 Lower Byte
XXB1	Timer T3 Upper Byte
xxB2	Timer T3 Autoload Register T3RA Lower Byte
xxB3	Timer T3 Autoload Register T3RA Upper Byte
xxB4	Timer T3 Autoload Register T3RB Lower Byte
xxB5	Timer T3 Autoload Register T3RB Upper Byte
xxB6	Timer T3 Control Register
xxB7	Comparator Select Register (CMPSL)
xxB8	UART Transmit Buffer (TBUF)
xxB9	UART Receive Buffer (RBUF)
xxBA	UART Control and Status Register (ENU)
xxBB	UART Receive Control and Status Register (ENUR)
xxBC	UART Interrupt and Clock Source Register (ENUI)
xxBD	UART Baud Register (BAUD)
xxBE	UART Prescale Select Register (PSR)
xxBF	Reserved for UART
xxC0	Timer T2 Lower Byte
xxC1	Timer T2 Upper Byte
xxC2	Timer T2 Autoload Register T2RA Lower Byte
xxC3	Timer T2 Autoload Register T2RA Upper Byte
xxC4	Timer T2 Autoload Register T2RB Lower Byte
xxC5	Timer T2 Autoload Register T2RB Upper Byte
xxC6	Timer T2 Control Register
xxC7	WATCHDOG Service Register (Reg:WDSVR)
xxC8	MIWU Edge Select Register (Reg:WKEDG)
xxC9	MIWU Enable Register (Reg:WKEN)
xxCA	MIWU Pending Register (Reg:WKPND)
xxCB	Reserved
xxCC	Reserved
xxCD to xxCF	Reserved

Address S/ADD REG	Contents
xxD0	Port L Data Register
xxD1	Port L Configuration Register
xxD2	Port L Input Pins (Read Only)
xxD3	Reserved for Port L
xxD4	Port G Data Register
xxD5	Port G Configuration Register
xxD6	Port G Input Pins (Read Only)
xxD7	Port I Input Pins (Read Only)
xxD8	Port C Data Register
xxD9	Port C Configuration Register
xxDA	Port C Input Pins (Read Only)
xxDB	Reserved for Port C
xxDC	Port D
xxDD to xxDF	Reserved for Port D
xxE0 to xxE5	Reserved for EE Control Registers
xxE6	Timer T1 Autoload Register T1RB Lower Byte
xxE7	Timer T1 Autoload Register T1RB Upper Byte
xxE8	ICNTRL Register
xxE9	MICROWIRE/PLUS Shift Register
xxEA	Timer T1 Lower Byte
xxEB	Timer T1 Upper Byte
xxEC	Timer T1 Autoload Register T1RA Lower Byte
xxED	Timer T1 Autoload Register T1RA Upper Byte
xxEE	CNTRL Control Register
xxEF	PSW Register
xxF0 to FB	On-Chip RAM Mapped as Registers
xxFC	X Register
xxFD	SP Register
xxFE	B Register
xxFF	S Register
0100 to 017F	On-Chip 128 RAM Bytes
0200 to 027F	On-Chip 128 RAM Bytes
0300 to 037F	On-Chip 128 RAM Bytes

Reading memory locations 0070H–007FH (Segment 0) will return all ones. Reading unused memory locations 0080H–00AFH (Segment 0) will return undefined data. Reading memory locations from other Segments (i.e., Segment 4, Segment 5, ... etc.) will return all ones.

Addressing Modes

There are ten addressing modes, six for operand addressing and four for transfer of control.

OPERAND ADDRESSING MODES

Register Indirect

This is the "normal" addressing mode. The operand is the data memory addressed by the B pointer or X pointer.

Register Indirect (with auto post increment or decrement of pointer)

This addressing mode is used with the LD and X instructions. The operand is the data memory addressed by the B pointer or X pointer. This is a register indirect mode that automatically post increments or decrements the B or X register after executing the instruction.

Direct

The instruction contains an 8-bit address field that directly points to the data memory for the operand.

Immediate

The instruction contains an 8-bit immediate field as the operand.

Short Immediate

This addressing mode is used with the Load B Immediate instruction. The instruction contains a 4-bit immediate field as the operand.

Indirect

This addressing mode is used with the LAID instruction. The contents of the accumulator are used as a partial address (lower 8 bits of PC) for accessing a data operand from the program memory.

TRANSFER OF CONTROL ADDRESSING MODES

Relative

This mode is used for the JP instruction, with the instruction field being added to the program counter to get the new program location. JP has a range from −31 to +32 to allow a 1-byte relative jump (JP + 1 is implemented by a NOP instruction). There are no "pages" when using JP, since all 15 bits of PC are used.

Absolute

This mode is used with the JMP and JSR instructions, with the instruction field of 12 bits replacing the lower 12 bits of the program counter (PC). This allows jumping to any location in the current 4k program memory segment.

Absolute Long

This mode is used with the JMPL and JSRL instructions, with the instruction field of 15 bits replacing the entire 15 bits of the program counter (PC). This allows jumping to any location up to 32k in the program memory space.

Indirect

This mode is used with the JID instruction. The contents of the accumulator are used as a partial address (lower 8 bits of PC) for accessing a location in the program memory. The contents of this program memory location serve as a partial address (lower 8 bits of PC) for the jump to the next instruction.

Note: The VIS is a special case of the Indirect Transfer of Control addressing mode, where the double byte vector associated with the interrupt is transferred from adjacent addresses in the program memory into the program counter (PC) in order to jump to the associated interrupt service routine.

Instruction Set

Register and Symbol Definition

Registers	
A	8-Bit Accumulator Register
B	8-Bit Address Register
X	8-Bit Address Register
S	8-Bit Segment Register
SP	8-Bit Stack Pointer Register
PC	15-Bit Program Counter Register
PU	Upper 7 Bits of PC
PL	Lower 8 Bits of PC
C	1 Bit of PSW Register for Carry
HC	1 Bit of PSW Register for Half Carry
GIE	1 Bit of PSW Register for Global Interrupt Enable
VU	Interrupt Vector Upper Byte
VL	Interrupt Vector Lower Byte

Symbols	
[B]	Memory Indirectly Addressed by B Register
[X]	Memory Indirectly Addressed by X Register
MD	Direct Addressed Memory
Mem	Direct Addressed Memory or [B]
Meml	Direct Addressed Memory or [B] or Immediate Data
Imm	8-Bit Immediate Data
Reg	Register Memory: Addresses F0 to FF (Includes B, X and SP)
Bit	Bit Number (0 to 7)
←	Loaded with
⟷	Exchanged with

5

Instruction Set (Continued)

INSTRUCTION SET

ADD	A,Meml	ADD	A ← A + Meml
ADC	A,Meml	ADD with Carry	A ← A + Meml + C, C ← Carry HC ← Half Carry
SUBC	A,Meml	Subtract with Carry	A ← A − $\overline{\text{Meml}}$ + C, C ← Carry HC ← Half Carry
AND	A,Meml	Logical AND	A ← A and Meml
ANDSZ	A,Imm	Logical AND Immed., Skip if Zero	Skip next if (A and Imm) = 0
OR	A,Meml	Logical OR	A ← A or Meml
XOR	A,Meml	Logical EXclusive OR	A ← A xor Meml
IFEQ	MD,Imm	IF EQual	Compare MD and Imm, Do next if MD = Imm
IFEQ	A,Meml	IF EQual	Compare A and Meml, Do next if A = Meml
IFNE	A,Meml	IF Not Equal	Compare A and Meml, Do next if A ≠ Meml
IFGT	A,Meml	IF Greater Than	Compare A and Meml, Do next if A > Meml
IFBNE	#	If B Not Equal	Do next if lower 4 bits of B ≠ Imm
DRSZ	Reg	Decrement Reg., Skip if Zero	Reg ← Reg − 1, Skip if Reg = 0
SBIT	#,Mem	Set BIT	1 to bit, Mem (bit = 0 to 7 immediate)
RBIT	#,Mem	Reset BIT	0 to bit, Mem
IFBIT	#,Mem	IF BIT	If bit in A or Mem is true do next instruction
RPND		Reset PeNDing Flag	Reset Software Interrupt Pending Flag
X	A,Mem	EXchange A with Memory	A ⟷ Mem
X	A,[X]	EXchange A with Memory [X]	A ⟷ [X]
LD	A,Meml	LoaD A with Memory	A ← Meml
LD	A,[X]	LoaD A with Memory [X]	A ← [X]
LD	B,Imm	LoaD B with Immed.	B ← Imm
LD	Mem,Imm	LoaD Memory Immed	Mem ← Imm
LD	Reg,Imm	LoaD Register Memory Immed.	Reg ← Imm
X	A, [B ±]	EXchange A with Memory [B]	A ⟷ [B], (B ← B ±1)
X	A, [X ±]	EXchange A with Memory [X]	A ⟷ [X], (X ← X ±1)
LD	A, [B±]	LoaD A with Memory [B]	A ← [B], (B ← B ±1)
LD	A, [X±]	LoaD A with Memory [X]	A ← [X], (X ← X±1)
LD	[B±],Imm	LoaD Memory [B] Immed.	[B] ← Imm, (B ← B±1)
CLR	A	CLeaR A	A ← 0
INC	A	INCrement A	A ← A + 1
DEC	A	DECrement A	A ← A − 1
LAID		Load A InDirect from ROM	A ← ROM (PU,A)
DCOR	A	Decimal CORrect A	A ← BCD correction of A (follows ADC, SUBC)
RRC	A	Rotate A Right thru C	C → A7 → . . . → A0 → C
RLC	A	Rotate A Left thru C	C ← A7 ← . . . ← A0 ← C
SWAP	A	SWAP nibbles of A	A7 . . . A4 ⟷ A3 . . . A0
SC		Set C	C ← 1, HC ← 1
RC		Reset C	C ← 0, HC ← 0
IFC		IF C	IF C is true, do next instruction
IFNC		IF Not C	If C is not true, do next instruction
POP	A	POP the stack into A	SP ← SP + 1, A ← [SP]
PUSH	A	PUSH A onto the stack	[SP] ← A, SP ← SP − 1
VIS		Vector to Interrupt Service Routine	PU ← [VU], PL ← [VL]
JMPL	Addr.	Jump absolute Long	PC ← ii (ii = 15 bits, 0 to 32k)
JMP	Addr.	Jump absolute	PC9 . . . 0 ← i (i = 12 bits)
JP	Disp.	Jump relative short	PC ← PC + r (r is −31 to +32, except 1)
JSRL	Addr.	Jump SubRoutine Long	[SP] ← PL, [SP−1] ← PU,SP−2, PC ← ii
JSR	Addr	Jump SubRoutine	[SP] ← PL, [SP−1] ← PU,SP−2, PC9 . . . 0 ← i
JID		Jump InDirect	PL ← ROM (PU,A)
RET		RETurn from subroutine	SP + 2, PL ← [SP], PU ← [SP−1]
RETSK		RETurn and SKip	SP + 2, PL ← [SP],PU ← [SP−1], skip next instruction
RETI		RETurn from Interrupt	SP + 2, PL ← [SP],PU ← [SP−1],GIE ← 1
INTR		Generate an Interrupt	[SP] ← PL, [SP−1] ← PU, SP−2, PC ← 0FF
NOP		No OPeration	PC ← PC + 1

Instruction Execution Time

Most instructions are single byte (with immediate addressing mode instructions taking two bytes).

Most single byte instructions take one cycle time to execute.

Skipped instructions require x number of cycles to be skipped, where x equals the number of bytes in the skipped instruction opcode.

See the BYTES and CYCLES per INSTRUCTION table for details.

Bytes and Cycles per Instruction

The following table shows the number of bytes and cycles for each instruction in the format of byte/cycle.

Arithmetic and Logic Instructions

	[B]	Direct	Immed.
ADD	1/1	3/4	2/2
ADC	1/1	3/4	2/2
SUBC	1/1	3/4	2/2
AND	1/1	3/4	2/2
OR	1/1	3/4	2/2
XOR	1/1	3/4	2/2
IFEQ	1/1	3/4	2/2
IFNE	1/1	3/4	2/2
IFGT	1/1	3/4	2/2
IFBNE	1/1		
DRSZ		1/3	
SBIT	1/1	3/4	
RBIT	1/1	3/4	
IFBIT	1/1	3/4	

RPND	1/1

Instructions Using A & C

CLRA	1/1
INCA	1/1
DECA	1/1
LAID	1/3
DCOR	1/1
RRCA	1/1
RLCA	1/1
SWAPA	1/1
SC	1/1
RC	1/1
IFC	1/1
IFNC	1/1
PUSHA	1/3
POPA	1/3
ANDSZ	2/2

Transfer of Control Instructions

JMPL	3/4
JMP	2/3
JP	1/3
JSRL	3/5
JSR	2/5
JID	1/3
VIS	1/5
RET	1/5
RETSK	1/5
RETI	1/5
INTR	1/7
NOP	1/1

Memory Transfer Instructions

	Register Indirect		Direct	Immed.	Register Indirect Auto Incr. & Decr.		
	[B]	[X]			[B+, B−]	[X+, X−]	
X A,*	1/1	1/3	2/3		1/2	1/3	
LD A,*	1/1	1/3	2/3	2/2	1/2	1/3	
LD B, Imm				1/1			(IF B < 16)
LD B, Imm				2/2			(IF B > 15)
LD Mem, Imm	2/2		3/3		2/2		
LD Reg, Imm			2/3				
IFEQ MD, Imm			3/3				

* = > Memory location addressed by B or X or directly.

Opcode Table

Upper Nibble																Lower Nibble
F	E	D	C	B	A	9	8	7	6	5	4	3	2	1	0	
JP−15	JP−31	LD 0F0, # i	DRSZ 0F0	RRCA	RC	ADC A,#i	ADC A,[B]	IFBIT 0,[B]	ANDSZ A, #i	LD B,#0F	IFBNE 0	JSR x000−x0FF	JMP x000−x0FF	JP+17	INTR	0
JP−14	JP−30	LD 0F1, # i	DRSZ 0F1	*	SC	SUBC A, #i	SUB A,[B]	IFBIT 1,[B]	*	LD B,#0E	IFBNE 1	JSR x100−x1FF	JMP x100−x1FF	JP+18	JP+2	1
JP−13	JP−29	LD 0F2, # i	DRSZ 0F2	X A,[X+]	X A,[B+]	IFEQ A,#i	IFEQ A,[B]	IFBIT 2,[B]	*	LD B,#0D	IFBNE 2	JSR x200−x2FF	JMP x200−x2FF	JP+19	JP+3	2
JP−12	JP−28	LD 0F3, # i	DRSZ 0F3	X A,[X−]	X A,[B−]	IFGT A,#i	IFGT A,[B]	IFBIT 3,[B]	*	LD B,#0C	IFBNE 3	JSR x300−x3FF	JMP x300−x3FF	JP+20	JP+4	3
JP−11	JP−27	LD 0F4, # i	DRSZ 0F4	VIS	LAID	ADD A,#i	ADD A,[B]	IFBIT 4,[B]	CLRA	LD B,#0B	IFBNE 4	JSR x400−x4FF	JMP x400−x4FF	JP+21	JP+5	4
JP−10	JP−26	LD 0F5, # i	DRSZ 0F5	RPND	JID	AND A,#i	AND A,[B]	IFBIT 5,[B]	SWAPA	LD B,#0A	IFBNE 5	JSR x500−x5FF	JMP x500−x5FF	JP+22	JP+6	5
JP−9	JP−25	LD 0F6, # i	DRSZ 0F6	X A,[X]	X A,[B]	XOR A,#i	XOR A,[B]	IFBIT 6,[B]	DCORA	LD B,#09	IFBNE 6	JSR x600−x6FF	JMP x600−x6FF	JP+23	JP+7	6
JP−8	JP−24	LD 0F7, # i	DRSZ 0F7	*	*	OR A,#i	OR A,[B]	IFBIT 7,[B]	PUSHA	LD B,#08	IFBNE 7	JSR x700−x7FF	JMP x700−x7FF	JP+24	JP+8	7
JP−7	JP−23	LD 0F8, # i	DRSZ 0F8	NOP	RLCA	LD A,#i	IFC	SBIT 0,[B]	RBIT 0,[B]	LD B,#07	IFBNE 8	JSR x800−x8FF	JMP x800−x8FF	JP+25	JP+9	8
JP−6	JP−22	LD 0F9, # i	DRSZ 0F9	IFNE A,[B]	IFEQ Md,#i	IFNE A,#i	IFNC	SBIT 1,[B]	RBIT 1,[B]	LD B,#06	IFBNE 9	JSR x900−x9FF	JMP x900−x9FF	JP+26	JP+10	9
JP−5	JP−21	LD 0FA, # i	DRSZ 0FA	LD A,[X+]	LD A,[B+]	LD [B+],#i	INCA	SBIT 2,[B]	RBIT 2,[B]	LD B,#05	IFBNE 0A	JSR xA00−xAFF	JMP xA00−xAFF	JP+27	JP+11	A
JP−4	JP−20	LD 0FB, # i	DRSZ 0FB	LD A,[X−]	LD A,[B−]	LD [B−],#i	DECA	SBIT 3,[B]	RBIT 3,[B]	LD B,#04	IFBNE 0B	JSR xB00−xBFF	JMP xB00−xBFF	JP+28	JP+12	B
JP−3	JP−19	LD 0FC, # i	DRSZ 0FC	LD Md,#i	JMPL	X A,Md	POPA	SBIT 4,[B]	RBIT 4,[B]	LD B,#03	IFBNE 0C	JSR xC00−xCFF	JMP xC00−xCFF	JP+29	JP+13	C
JP−2	JP−18	LD 0FD, # i	DRSZ 0FD	DIR	JSRL	LD A,Md	RETSK	SBIT 5,[B]	RBIT 5,[B]	LD B,#02	IFBNE 0D	JSR xD00−xDFF	JMP xD00−xDFF	JP+30	JP+14	D
JP−1	JP−17	LD 0FE, # i	DRSZ 0FE	LD A,[X]	LD A,[B]	LD [B],#i	RET	SBIT 6,[B]	RBIT 6,[B]	LD B,#01	IFBNE 0E	JSR xE00−xEFF	JMP xE00−xEFF	JP+31	JP+15	E
JP−0	JP−16	LD 0FF, # i	DRSZ 0FF	*	*	LD B,#i	RETI	SBIT 7,[B]	RBIT 7,[B]	LD B,#00	IFBNE 0F	JSR xF00−xFFF	JMP xF00−xFFF	JP+32	JP+16	F

Where,

i is the immediate data

Md is a directly addressed me nory location

* is an unused opcode

Note: The opcode 60 Hex is also the opcode for IFBIT #i,A

Mask Options

The mask programmable options are shown below. The options are programmed at the same time as the ROM pattern submission.

OPTION 1: CLOCK CONFIGURATION

= 1 Crystal Oscillator (CKI/10)
G7 (CKO) is clock generator output to crystal/ resonator with CKI being the clock input

= 2 Single-pin RC controlled oscillator (CKI/10)
G7 is available as a HALT restart and/or general purpose input

OPTION 2: HALT

= 1 Enable HALT mode

= 2 Disable HALT mode

OPTION 3: BONDING

= 1 44-Pin PLCC

= 2 40-Pin DIP

The chip can be driven by a clock input on the CKI input pin which can be between DC and 10 MHz. The CKO output clock is on pin G7 (if clock option = 1 has been selected). The CKI input frequency is divided down by 10 to produce the instruction cycle clock (1/tc).

Development Support

Summary

- iceMASTER: IM-COP8/400—Full feature in-circuit emulation for all COP8 products. A full set of COP8 Basic and Feature Family device and package specific probes are available.
- COP8 Debug Module: Moderate cost in-circuit emulation and development programming unit.
- COP8 Evaluation and Programming Unit: EPU-COP888HG—low cost In-circuit simulation and development programming unit.
- Assembler: COP8-DEV-IBMA. A DOS installable cross development Assembler, Linker, Librarian and Utility Software Development Tool Kit.
- C Compiler: COP8C. A DOS installable cross development Software Tool Kit.
- OTP/EPROM Programmer Support: Covering needs from engineering prototype, pilot production to full production environments.

Development Support (Continued)

IceMASTER (IM) IN-CIRCUIT EMULATION

The iceMASTER IM-COP8/400 is a full feature, PC based, in-circuit emulation tool developed and marketed by MetaLink Corporation to support the whole COP8 family of products. National is a resale vendor for these products.

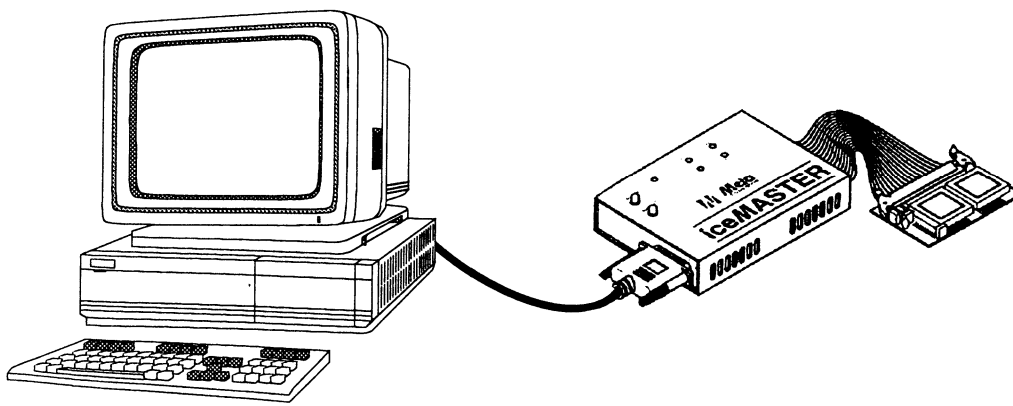
See *Figure 19* for configuration.

The iceMASTER IM-COP8/400 with its device specific COP8 Probe provides a rich feature set for developing, testing and maintaining product:

- Real-time in-circuit emulation; full 2.4V–5.5V operation range, full DC-10 MHz clock. Chip options are programmable or jumper selectable.
- Direct connection to application board by package compatible socket or surface mount assembly.
- Full 32 kbytes of loadable programming space that overlays (replaces) the on-chip ROM or EPROM. On-chip RAM and I/O blocks are used directly or recreated on the probe as necessary.
- Full 4K frame synchronous trace memory. Address, instruction, and 8 unspecified, circuit connectable trace lines. Display can be HLL source (e.g., C source), assembly or mixed.
- A full 64K hardware configurable break, trace on, trace off control, and pass count increment events.
- Tool set integrated interactive symbolic debugger—supports both assembler (COFF) and C Compiler (.COD) linked object formats.
- Real time peformance profiling analysis; selectable bucket definition.
- Watch windows, content updated automatically at each execution break.
- Instruction by instruction memory/register changes displayed on source window when in single step operation.
- Single base unit and debugger software reconfigurable to support the entire COP8 family; only the probe personality needs to change. Debugger software is processor customized, and reconfigured from a master model file.
- Processor specific symbolic display of registers and bit level assignments, configured from master model file.
- Halt/Idle mode notification.
- On-line HELP customized to specific processor using master model file.
- Includes a copy of COP8-DEV-IBMA assembler and linker SDK.

IM Order Information

Base Unit	
IM-COP8/400-1	iceMASTER base unit, 110V power supply
IM-COP8/400-2	iceMASTER base unit, 220V power supply
iceMASTER Probe	
MHW-888GG40DWPC	40 DIP
MHW-888GG44PWPC	40 PLCC



TL/DD/12858–21

FIGURE 19. COP iceMASTER Environment

Development Support (Continued)

IceMASTER DEBUG MODULE (DM)

The iceMASTER Debug Module is a PC based, combination in-circuit emulation tool and COP8 based OTP/EPROM programming tool developed and marketed by MetaLink Corporation to support the whole COP8 family of products. National is a resale vendor for these products.

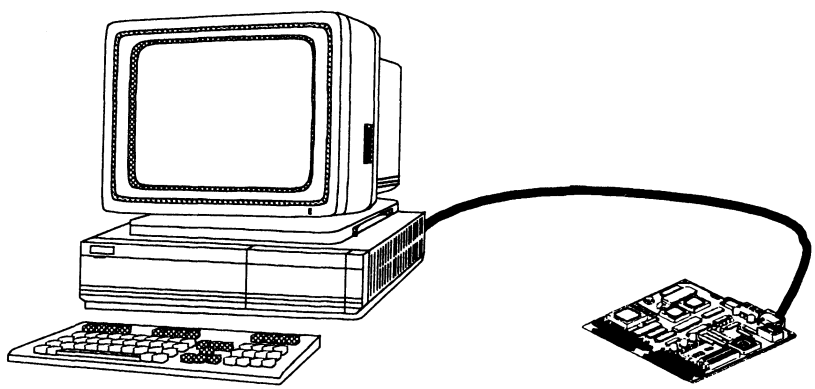
See *Figure 20* for configuration.

The iceMASTER Debug Module is a moderate cost development tool. It has the capability of in-circuit emulation for a specific COP8 microcontroller and in addition serves as a programming tool for COP8 OTP and EPROM product families. Summary of features is as follows:

- Real-time in-circuit emulation; full operating voltage range operation, full DC-10 MHz clock.
- All processor I/O pins can be cabled to an application development board with package compatible cable to socket and surface mount assembly.
- Full 32 kbytes of loadable programming space that overlays (replaces) the on-chip ROM or EPROM. On-chip RAM and I/O blocks are used directly or recreated as necessary.
- 100 frames of synchronous trace memory. The display can be HLL source (C source), assembly or mixed. The most recent history prior to a break is available in the trace memory.
- Configured break points; uses INTR instruction which is modestly intrusive.
- Software—only supported features are selectable.
- Tool set integrated interactive symbolic debugger—supports both
- Assembler (COFF) and C Compiler (.COD) SDK linked object formats.
- Instruction by instruction memory/register changes displayed when in single step operation.
- Debugger software is processor customized, and reconfigured from a master model file.
- Processor specific symbolic display of registers and bit level assignments, configured from master model file.
- Halt/Idle mode notification.
- Programming menu supports full product line of programmable OTP and EPROM COP8 products. Program data is taken directly from the overlay RAM.
- Programming of 44 PLCC and 68 PLCC parts requires external programming adapters.
- Includes wall-mount power supply
- On-board V_{PP} generator from 5V input or connection to external supply supported. Requires V_{PP} level adjustment per the family programming specification (correct level is provided on an on-screen pop-down display). On-line HELP customized to specific processor using master model file.
- On-line HELP customized to specific processor using master model file
- Includes a copy of COP8-DEV-IBMA assembler and linker SDK.

DM Order Information

Debug Module Unit	
COP8-DM/888GG	
Cable Adapters	
DM-COP8/40D	40 DIP
DM-COP8/44P	44 PLCC



TL/DD/12858–22

FIGURE 20. COP8-DM Environment

Development Support (Continued)

IceMASTER EVALUATION PROGRAMMING UNIT (EPU)

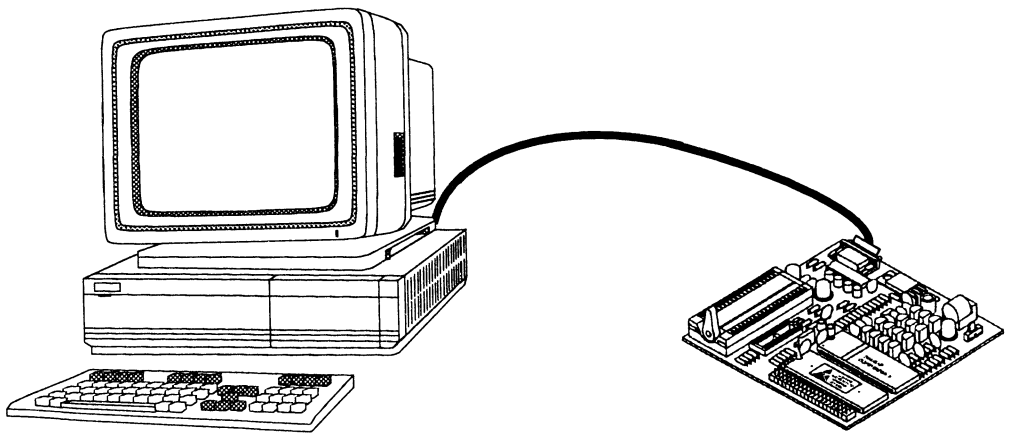
The iceMASTER EPU-COP888HG is a PC based, in-circuit. simulation tool to support the feature family COP8 products. See *Figure 21* for configuration.

The simulation capability is a very low cost means of evaluating the general COP8 architecture. In addition, the EPU has programming capability, with added adapters, for programming the whole COP8 product family of OTP and EPROM products. The product includes the following features:

- Non-real-time in-circuit simulation. Program overlay memory is PC resident; instructions are downloaded over RS-232 as executed. Approximate performance is 20 kHz.
- Includes a 40-pin DIP cable adapter. Other target packages are not supported. All processor I/O pins are cabled to the application development environment.
- Full 32 kbytes of loadable programming space that overlays (replaces) the on-chip ROM or EPROM. On-chip RAM and I/O blocks are used directly or recreated as necessary.
- On-chip timer and watch-dog execution are not well synchronized to the instruction simulation.
- 100 frames of synchronous trace memory. The display can be HLL source (e.g., C source), assembly or mixed. The most recent history prior to a break is available in the trace memory.
- Up to 8 software configured break points; uses INTR instruction which is modestly intrusive.
- Common look-feel debugger software across all MetaLink products—only supported features are selectable.
- Tool set integrated interactive symbolic debugger—supports both assembler (COFF) and C Compiler (.COD) SDK linked object formats.
- Instruction by instruction memory/register changes displayed when in single step operation.
- Processor specific symbolic display of registers and bit level assignments, configured from master model file.
- Halt/Idle mode notification. Restart requires special handling.
- Programming menu supports full product line of programmable OTP and EPROM COP8 products. Only a 40 ZIF socket is available on the EPU unit. Adapters are available for other part package configurations.
- Integral wall mount power supply provides 5V and develops the required V_{PP} to program parts.
- Includes a copy of COP8-DEV-IBMA assembler, linker SDK.

EPU Order Information

Evaluation Programming Unit	
EPU-COP888GG	Evaluation Programming Unit with debugger and programmer control software and 40 ZIF programming socket
General Programming Adapters	
COP8-PGMA-DS	28 & 20 DIP and SOIC adapter
COP8-PGMA-DS44P	28 & 20 DIP and SOIC plus 44 PLCC adapter



TL/DD/12858–23

FIGURE 21. CPU-COP8 Tool Environment

Development Support (Continued)

COP8 ASSEMBLER/LINKER SOFTWARE DEVELOPMENT TOOL KIT

National Semiconductor offers a relocateable COP8 macro cross assembler, linker, librarian and utility software development tool kit. Features are summarized as follows:

- Basic and Feature Family instruction set by "device" type.
- Nested macro capability.
- Extensive set of assembler directives.
- Supported on PC/DOS platform.
- Generates National standard COFF output files.
- Integrated Linker and Librarian.
- Integrated utilities to generate ROM code file outputs.
- DUMPCOFF utility.

This product is integrated as a part of MetaLink tools as a development kit, fully supported by the MetaLink debugger. It may be ordered separately or it is bundled with the MetaLink products at no additional cost.

Order Information

Assembler SDK	
COP8-DEV-IBMA	Assembler SDK on installable 3.5" PC/DOS Floppy Disk Drive format. Periodic upgrades and most recent version is available on National's BBS and Internet.

COP8 C COMPILER

A C Compiler is developed and marketed by Byte Craft Limited. The COP8C compiler is a fully integrated development tool specifically designed to support the compact embedded configuration of the COP8 family of products.

Features are summarized as follows:

- ANSI C with some restrictions and extensions that optimize development for the COP8 embedded application.
- BITS data type extension. Register declaration #pragma with direct bit level definitions.
- C language support for interrupt routines.
- Expert system, rule based code geration and optimization.
- Performs consistency checks against the architectural definitions of the target COP8 device.
- Generates program memory code.
- Supports linking of compiled object or COP8 assembled object formats.
- Global optimization of linked code.
- Symbolic debug load format fully source level supported by the MetaLink debugger.

Approved List

Manufacturer	North America	Europe	Asia
BP Microsystems	(800) 225-2102 (713) 688-4600 Fax: (713) 688-0920	+49-8152-4183 +49-8856-932616	+852-234-16611 +852-2710-8121
Data I/O	(800) 426-1045 (206) 881-6444 Fax: (206) 882-1043	+44-0734-440011	Call North America
HI-LO	(510) 623-8860	Call Asia	+886-2-764-0215 Fax: +886-2-756-6403
ICE Technology	(800) 624-8949 (919) 430-7915	+44-1226-767404 Fax: 0-1226-370-434	
MetaLink	(800) 638-2423 (602) 926-0797 Fax: (602) 693-0681	+49-80 9156 96-0 Fax: +49-80 9123 86	+852-737-1800
Systems General	(408) 263-6667	+41-1-9450300	+886-2-9173005 Fax: +886-2-911-1283
Needhams	(916) 924-8037 Fax: (916) 924-8065		

Development Support (Continued)

SINGLE-CHIP OTP/EMULATOR SUPPORT

The COP8 family is supported by single-chip OTP emulators. For detailed information, refer to the emulator specific datasheet and the emulator selection table below:

OTP Emulator Ordering Information

Device Number	Clock Option	Package	Emulates
COP87L88GGV-XE	Crystal/ HALT En	44 PLCC	COP888HG
COP87L88GGN-XE	Crystal/ HALT En	40 DIP	COP888HG

INDUSTRY WIDE OTP/EPROM PROGRAMMING SUPPORT

Programming support, in addition to the MetaLink development tools, is provided by a full range of independent approved vendors to meet the needs from the engineering laboratory to full production.

AVAILABLE LITERATURE

For more information, please see the COP8 Basic Family User's Manual, Literature Number 620895, COP8 Feature Family User's Manual Literature Number 620897 and National's Family of 8-bit Microcontrollers COP8 Selection Guide, Literature Number 630009.

DIAL-A-HELPER

Dial-A-Helper is a service provided by the Microcontroller Applications group. The Dial-A-Helper is an Electronic Information System that may be accessed as a Bulletin Board System (BBS) via data modem, as an FTP site on the Internet via standard FTP client application or as an FTP site on the Internet using a standard Internet browser such as Netscape or Mosaic.

The Dial-A-Helper system provides access to an automated information storage and retrieval system. The system capabilities include a MESSAGE SECTION (electronic mail, when accessed as a BBS) for communication to and from the Microcontroller Applications Group and a FILE SECTION which consists of several file areas where valuable application software and utilities could be found.

DIAL-A-HELPER BBS via a Standard Modem

Modem: CANADA/U.S.: (800) NSC-MICRO
(800) 672-6427
EUROPE: (+49) 0-8141-351332
Baud: 14.4k
Set-up: Length: 8-Bit
Parity: None
Stop Bit: 1
Operation: 24 Hrs., 7 Days

DIAL-A-HELPER via FTP

ftp nscmicro.nsc.com
user: anonymous
password: username@yourhost.site.domain

DIAL-A-HELPER via a WorldWide Web Browser

ftp://nscmicro.nsc.com

National Semiconductor on the WorldWide Web

See us on the WorldWide Web at: http://www.national.com

CUSTOMER RESPONSE CENTER

Complete product information and technical support is available from National's customer response centers.

CANADA/U.S.:	Tel:	(800) 272-9959
	email:	support @tevm2.nsc.com
EUROPE:	email:	europe.support@nsc.com
	Deutsch Tel:	+49 (0) 180-530 85 85
	English Tel:	+49 (0) 180-532 78 32
	Français Tel:	+49 (0) 180-532 93 58
	Italiano Tel:	+49 (0) 180-534 16 80
JAPAN:	Tel:	+81-043-299-2309
S.E. ASIA:	Beijing Tel:	(+86) 10-6856-8601
	Shanghai Tel:	(+86) 21-6415-4092
	Hong Kong Tel:	(+852) 2737-1600
	Korea Tel:	(+82) 2-3771-6909
	Malaysia Tel:	(+60-4) 644-9061
	Singapore Tel:	(+65) 255-2226
	Taiwan Tel:	+886-2-521-3288
AUSTRALIA:	Tel:	(+61) 3-9558-9999
INDIA:	Tel:	(+91) 80-559-9467

National Semiconductor

PRELIMINARY

COP688KG/COP888KG 8-Bit Microcontroller with UART and Three Multi-Function Timers

General Description

The COP8™ feature family of microcontrollers uses an 8-bit single-chip core architecture fabricated with National Semiconductor's M²CMOS™ process technology. The COP888KG is a member of this expandable 8-bit core processor family of microcontrollers. (Continued)

Key Features

- Full duplex UART
- Three 16-bit timers, each with two 16-bit registers supporting:
  - — Processor Independent PWM mode
  - — External Event counter mode
  - — Input Capture mode
- Quiet design (low radiated emissions)
- 24 kbytes on-board ROM
- 1088 bytes on-board RAM

Additional Peripheral Features

- Idle Timer
- Multi-Input Wake-Up (MIWU) with optional interrupts (8)
- Two analog comparators
- WATCHDOG™ and Clock Monitor logic
- MICROWIRE/PLUS™ serial I/O

I/O Features

- Memory mapped I/O
- Software selectable I/O options (TRI-STATE® Output, Push-Pull Output, Weak Pull-Up Input, High Impedance Input)
- Schmitt trigger inputs on ports G and L
- Packages:
  - — 40 DIP with 35 I/O pins
  - — 44 PLCC with 39 I/O pins

CPU/Instruction Set Features

- 1 μs instruction cycle time
- Fourteen multi-source vectored interrupts servicing
  - — External Interrupt with selectable edge
  - — Idle Timer T0
  - — Three Timers (each with 2 interrupts)
  - — MICROWIRE/PLUS
  - — Multi-Input Wake-Up
  - — Software Trap
  - — UART (2)
  - — Default VIS (default interrupt)
- Versatile and easy to use instruction set
- 8-bit Stack Pointer SP—(stack in RAM)
- Two 8-bit Register Indirect Data Memory Pointers (B and X)

Fully Static CMOS

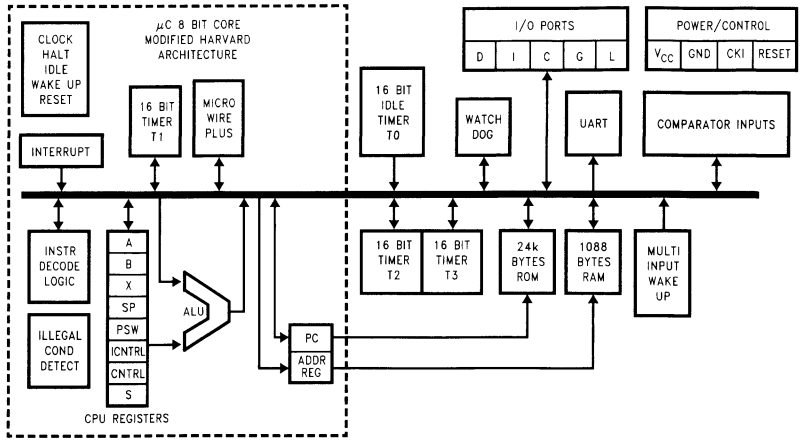
- Two power saving modes: HALT and IDLE
- Low current drain (typically <1 μA)
- Single supply operation: 2.5V–5.5V
- Temperature ranges:
  $-40°C$ to $+85°C$, $-55°C$ to $+125°C$

Development Support

- Emulation and OTP devices
- Real time emulation and full program debug offered by MetaLink's Development System

5

Block Diagram



TL/DD12829–1

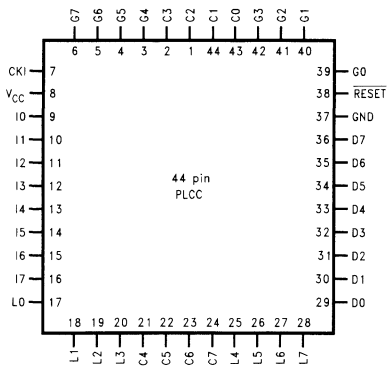
FIGURE 1. COP888KG Block Diagram

General Description (Continued)

It is a fully static part, fabricated using double-metal silicon gate microCMOS technology. Features include an 8-bit memory mapped architecture, MICROWIRE/PLUS serial I/O, three 16-bit timer/counters supporting three modes (Processor Independent PWM generation, External Event counter, and Input Capture mode capabilities), full duplex UART, two comparators, and two power saving modes (HALT and IDLE), both with a multi-sourced wakeup/interrupt capability. This multi-sourced interrupt capability may also be used independent of the HALT or IDLE modes. Each I/O pin has software selectable configurations. The devices operate over a voltage range of 2.5V to 5.5V. High throughput is achieved with an efficient, regular instruction set operating at a maximum rate of 1 μs per instruction. Low radiated emissions are achieved by gradual turn-on output drivers and internal I_{CC} smoothing filters on the chip logic and crystal oscillator.

Connection Diagrams

Plastic Chip Carrier

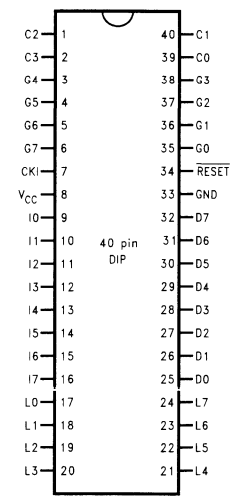


TL/DD12829–2

Top View

Order Number COP688KG-XXX/V,
COP888KG-XXX/V
See NS Package Number V44A

Dual-In-Line Package



TL/DD12829–3

Top View

Order Number COP688KG-XXX/N,
COP888KG-XXX/N
See NS Package Number N40A

FIGURE 2. Connection Diagrams

Connection Diagrams (Continued)

Pinouts for 40- and 44-Pin Packages

Port	Type	Alt. Fun	Alt. Fun	40-Pin DIP	44-Pin PLCC
L0	I/O	MIWU		17	17
L1	I/O	MIWU	CKX	18	18
L2	I/O	MIWU	TDX	19	19
L3	I/O	MIWU	RDX	20	20
L4	I/O	MIWU	T2A	21	25
L5	I/O	MIWU	T2B	22	26
L6	I/O	MIWU	T3A	23	27
L7	I/O	MIWU	T3B	24	28
G0	I/O	INT		35	39
G1	WDOUT			36	40
G2	I/O	T1B		37	41
G3	I/O	T1A		38	42
G4	I/O	SO		3	3
G5	I/O	SK		4	4
G6	I	SI		5	5
G7	I/CKO	HALT Restart		6	6
D0	O			25	29
D1	O			26	30
D2	O			27	31
D3	O			28	32
D4	O			29	33
D5	O			30	34
D6	O			31	35
D7	O			32	36
I0	I			9	9
I1	I	COMP1IN−		10	10
I2	I	COMP1IN+		11	11
I3	I	COMP1OUT		12	12
I4	I	COMP2IN−		13	13
I5	I	COMP2IN+		14	14
I6	I	COMP2OUT		15	15
I7	I			16	16
C0	I/O			39	43
C1	I/O			40	44
C2	I/O			1	1
C3	I/O			2	2
C4	I/O				21
C5	I/O				22
C6	I/O				23
C7	I/O				24
V_{CC}				8	8
GND				33	37
CKI				7	7
$\overline{\text{RESET}}$				34	38

Absolute Maximum Ratings

If Military/Aerospace specified devices are required, please contact the National Semiconductor Sales Office/Distributors for availability and specifications.

Supply Voltage (V_{CC})	7V
Voltage at Any Pin	−0.3V to V_{CC} + 0.3V
Total Current into V_{CC} Pin (Source)	100 mA
Total Current out of GND Pin (Sink)	110 mA
Storage Temperature Range	−65°C to +140°C

Note: *Absolute maximum ratings indicate limits beyond which damage to the device may occur. DC and AC electrical specifications are not ensured when operating the device at absolute maximum ratings.*

DC Electrical Characteristics COP888KG: $-40°C \le T_A \le +85°C$ unless otherwise specified

Parameter	Conditions	Min	Typ	Max	Units
Operating Voltage		2.5		5.5	V
Power Supply Ripple (Note 1)	Peak-to-Peak			0.1 V_{CC}	V
Supply Current (Note 2)					
CKI = 10 MHz	V_{CC} = 5.5V, t_c = 1 μs			12.5	mA
CKI = 4 MHz	V_{CC} = 5.5V, t_c = 2.5 μs			5.5	mA
CKI = 4 MHz	V_{CC} = 4V, t_c = 2.5 μs			2.5	mA
CKI = 1 MHz	V_{CC} = 4V, t_c = 10 μs			1.4	mA
HALT Current (Note 3)	V_{CC} = 5.5V, CKI = 0 MHz		<1	10	μA
	V_{CC} = 4V, CKI = 0 MHz		<0.5	6	μA
IDLE Current					
CKI = 10 MHz	V_{CC} = 5.5V, t_c = 1 μs			3.5	mA
CKI = 4 MHz	V_{CC} = 5.5V, t_c = 2.5 μs			2.5	mA
CKI = 1 MHz	V_{CC} = 4V, t_c = 10 μs			0.7	mA
Input Levels (V_{IH}, V_{IL})					
$\overline{\text{RESET}}$					
Logic High		0.8 V_{CC}			V
Logic Low				0.2 V_{CC}	V
CKI, All Other Inputs					
Logic High		0.7 V_{CC}			V
Logic Low				0.2 V_{CC}	V
Hi-Z Input Leakage	V_{CC} = 5.5V, V_{IN} = 0V	−2		+2	μA
Input Pullup Current	V_{CC} = 5.5V, V_{IN} = 0V	−40		−250	μA
G and L Port Input Hysteresis	(Note 5)			0.35 V_{CC}	V
Output Current Levels					
D Outputs					
Source	V_{CC} = 4V, V_{OH} = 3.3V	−0.4			mA
	V_{CC} = 2.5V, V_{OH} = 1.8V	−0.2			mA
Sink	V_{CC} = 4V, V_{OL} = 1V	10			mA
	V_{CC} = 2.5V, V_{OL} = 0.4V	2.0			mA
All Others					
Source (Weak Pull-Up Mode)	V_{CC} = 4V, V_{OH} = 2.7V	−10		−110	μA
	V_{CC} = 2.5V, V_{OH} = 1.8V	−2.5		−33	μA
Source (Push-Pull Mode)	V_{CC} = 4V, V_{OH} = 3.3V	−0.4			mA
	V_{CC} = 2.5V, V_{OH} = 1.8V	−0.2			mA
Sink (Push-Pull Mode)	V_{CC} = 4V, V_{OL} = 0.4V	1.6			mA
	V_{CC} = 2.5V, V_{OL} = 0.4V	0.7			mA
TRI-STATE Leakage	V_{CC} = 5.5V	−2		+2	μA
Allowable Sink/Source Current per Pin					
D Outputs (Sink)				15	mA
All others				3	mA
Maximum Input Current without Latchup (Notes 4, 5)	Room Temperature			±100	mA
RAM Retention Voltage, V_r	500 ns Rise and Fall Time (min)	2			V
Input Capacitance	(Note 5)			7	pF
Load Capacitance on D2	(Note 5)			1000	pF

AC Electrical Characteristics COP888KG: $-40°C \le T_A \le +85°C$ unless otherwise specified

Parameter	Conditions	Min	Typ	Max	Units
Instruction Cycle Time (t_c)					
Crystal, Resonator	$4V \le V_{CC} \le 5.5V$	2.5		DC	μs
	$2.5V \le V_{CC} < 4V$	1.0		DC	μs
R/C Oscillator	$4V \le V_{CC} \le 5.5V$	7.5		DC	μs
	$2.5V \le V_{CC} < 4V$	3.0		DC	μs
CKI Clock Duty Cycle (Note 5)	f = Max	40		60	%
Rise Time (Note 5)	f = 10 MHz Ext Clock			5	μs
Fall Time (Note 5)	f = 10 MHz Ext Clock			5	μs
Inputs					
t_{SETUP}	$4V \le V_{CC} \le 5.5V$	200			ns
	$2.5V \le V_{CC} < 4V$	500			ns
t_{HOLD}	$4V \le V_{CC} \le 5.5V$	60			ns
	$2.5V \le V_{CC} < 4V$	150			ns
Output Propagation Delay	$R_L = 2.2k, C_L = 100$ pF				
t_{PD1}, t_{PD0}					
SO, SK	$4V \le V_{CC} \le 5.5V$			0.7	μs
	$2.5V \le V_{CC} < 4V$			1.75	μs
All Others	$4V \le V_{CC} \le 5.5V$			1	μs
	$2.5V \le V_{CC} < 4V$			2.5	μs
MICROWIRE Setup Time (t_{UWS}) (Note 5)	$V_{CC} \ge 4V$	20			ns
MICROWIRE Hold Time (t_{UWH}) (Note 5)	$V_{CC} \ge 4V$	56			ns
MICROWIRE Output Propagation Delay (t_{UPD})	$V_{CC} \ge 4V$			220	ns
Input Pulse Width (Note 6)					
Interrupt Input High Time		1			t_c
Interrupt Input Low Time		1			t_c
Timer 1, 2, 3 Input High Time		1			t_c
Timer 1, 2, 3 Input Low Time		1			t_c
Reset Pulse Width		1			μs

Note 1: Maximum rate of voltage change must be less than 0.5 V/ms.

Note 2: Supply current is measured after running 2000 cycles with a square wave CKI input, CKO open, inputs at rails and outputs open.

Note 3: The HALT mode will stop CKI from oscillating in the RC and the Crystal configurations. Measurement of IDD HALT is done with device neither sourcing or sinking current; with L, C, and G0–G5 programmed as low outputs and not driving a load; all outputs programmed low and not driving a load; all inputs tied to V_{CC}; clock monitor and comparators disabled. Parameter refers to HALT mode entered via setting bit 7 of the G Port data register. Part will pull up CKI during HALT in crystal clock mode.

Note 4: Pins G6 and $\overline{\text{RESET}}$ are designed with a high voltage input network. These pins allow input voltages greater than V_{CC} and the pins will have sink current to V_{CC} when biased at voltages greater than V_{CC} (the pins do not have source current when biased at a voltage below V_{CC}). The effective resistance to V_{CC} is 750Ω (typical). These two pins will not latch up. The voltage at the pins must be limited to less than 14V. **WARNING: Voltages in excess of 14V will cause damage to the pins. This warning excludes ESD transients.**

Note 5: Parameter characterized but not tested.

Note 6: t_c Instruction Cycle Time

Absolute Maximum Ratings

If Military/Aerospace specified devices are required, please contact the National Semiconductor Sales Office/Distributors for availability and specifications.

Supply Voltage (V_{CC})	7V
Voltage at Any Pin	−0.3V to V_{CC} + 0.3V
Total Current into V_{CC} Pin (Source)	100 mA
Total Current out of GND Pin (Sink)	110 mA
Storage Temperature Range	−65°C to +140°C

Note: *Absolute maximum ratings indicate limits beyond which damage to the device may occur. DC and AC electrical specifications are not ensured when operating the device at absolute maximum ratings.*

DC Electrical Characteristics COP688KG: $-55°C \le T_A \le +125°C$ unless otherwise specified

Parameter	Conditions	Min	Typ	Max	Units
Operating Voltage		4.5		5.5	V
Power Supply Ripple (Note 1)	Peak-to-Peak			0.1 V_{CC}	V
Supply Current (Note 2)					
CKI = 10 MHz	V_{CC} = 5.5V, t_c = 1 μs			12.5	mA
CKI = 4 MHz	V_{CC} = 5.5V, t_c = 2.5 μs			5.5	mA
HALT Current (Note 3)	V_{CC} = 5.5V, CKI = 0 MHz		<10	30	μA
IDLE Current					
CKI = 10 MHz	V_{CC} = 5.5V, t_c = 1 μs			3.5	mA
CKI = 4 MHz	V_{CC} = 5.5V, t_c = 2.5 μs			2.5	mA
Input Levels (V_{IH}, V_{IL})					
$\overline{RESET}$					
Logic High		0.8 V_{CC}			V
Logic Low				0.2 V_{CC}	V
CKI, All Other Inputs					
Logic High		0.7 V_{CC}			V
Logic Low				0.2 V_{CC}	V
Hi-Z Input Leakage	V_{CC} = 5.5V, V_{IN} = 0V	−5		+5	μA
Input Pullup Current	V_{CC} = 5.5V, V_{IN} = 0V	−35		−400	μA
G and L Port Input Hysteresis	(Note 5)			0.35 V_{CC}	V
Output Current Levels					
D Outputs					
Source	V_{CC} = 4.5V, V_{OH} = 3.3V	−0.4			mA
Sink	V_{CC} = 4.5V, V_{OL} = 1V	9			mA
All Others					
Source (Weak Pull-Up Mode)	V_{CC} = 4.5V, V_{OH} = 2.7V	−9		−140	μA
Source (Push-Pull Mode)	V_{CC} = 4.5V, V_{OH} = 3.3V	−0.4			mA
Sink (Push-Pull Mode)	V_{CC} = 4.5V, V_{OL} = 0.4V	1.4			mA
TRI-STATE Leakage	V_{CC} = 5.5V	−5		+5	μA
Allowable Sink/Source Current per Pin					
D Outputs (Sink)				12	mA
All others				2.5	mA
Maximum Input Current without Latchup (Notes 4, 5)	Room Temp			±100	mA
RAM Retention Voltage, V_r	500 ns Rise and Fall Time (min)	2			V
Input Capacitance	(Note 5)			7	pF
Load Capacitance on D2	(Note 5)			1000	pF

AC Electrical Characteristics COP688KG: $-55°C \leq T_A \leq +125°C$ unless otherwise specified

Parameter	Conditions	Min	Typ	Max	Units
Instruction Cycle Time (t_c)					
Crystal, Resonator	$V_{CC} \geq 4.5V$	1.0		DC	μs
Inputs					
t_{SETUP}	$V_{CC} \geq 4.5V$	200			ns
t_{HOLD}	$V_{CC} \geq 4.5V$	60			ns
Output Propagation Delay	$R_L = 2.2k, C_L = 100$ pF				
t_{PD1}, t_{PD0}					
SO, SK	$V_{CC} \geq 4.5V$			0.7	μs
All Others	$V_{CC} \geq 4.5V$			1	μs
MICROWIRE™ Setup Time (t_{UWS}) (Note 5)		20			ns
MICROWIRE Hold Time (t_{UWH}) (Note 5)		56			ns
MICROWIRE Output Propagation Delay (t_{UPD})				220	ns
Input Pulse Width (Note 6)					
Interrupt Input High Time		1			t_c
Interrupt Input Low Time		1			t_c
Timer 1, 2, 3 Input High Time		1			t_c
Timer 1, 2, 3 Input Low Time		1			t_c
Reset Pulse Width		1			μs

Note 1: Maximum rate of voltage change must be less than 0.5 V/ms.

Note 2: Supply current is measured after running 2000 cycles with a square wave CKI input, CKO open, inputs at rails and outputs open.

Note 3: The HALT mode will stop CKI from oscillating in the RC and the Crystal configurations. Measurement of IDD HALT is done with device neither sourcing or sinking current; with L, C, and G0–G5 programmed as low outputs and not driving a load; all outputs programmed low and not driving a load; all inputs tied to V_{CC}; clock monitor and comparators disabled. Parameter refers to HALT mode entered via setting bit 7 of the G Port data register. Part will pull up CKI during HALT in crystal clock mode.

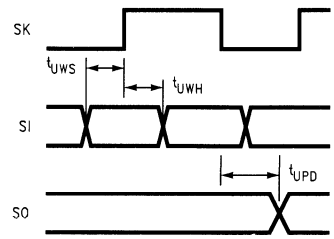
Note 4: Pins G6 and $\overline{\text{RESET}}$ are designed with a high voltage input network. These pins allow input voltages greater than V_{CC} and the pins will have sink current to V_{CC} when biased at voltages greater than V_{CC} (the pins do not have source current when biased at a voltage below V_{CC}). The effective resistance to V_{CC} is 750Ω (typical). These two pins will not latch up. The voltage at the pins must be limited to less than 14V. **WARNING: Voltages in excess of 14V will cause damage to the pins. This warning excludes ESD transients.**

Note 5: Parameter characterized but not tested.

Note 6: t_c = Instruction Cycle Time

Comparators AC and DC Characteristics $V_{CC} = 5V, -40°C < T_A \leq +85°C$.

Parameter	Conditions	Min	Typ	Max	Units
Input Offset Voltage	$0.4V \leq V_{IN} \leq V_{CC} - 1.5V$		±10	±25	mV
Input Common Mode Voltage Range		0.4		$V_{CC} - 1.5$	V
Voltage Gain			300k		V/V
Low Level Output Current	$V_{OL} = 0.4V$	1.6			mA
High Level Output Current	$V_{OH} = 4.6V$	1.6			mA
DC Supply Current per Comparator (When Enabled)				250	μA
Response Time	100 mV Overdrive, 100 pF Load			1	μs



TL/DD12829-4

FIGURE 3. MICROWIRE/PLUS Timing

Pin Descriptions

V_{CC} and GND are the power supply pins. All V_{CC} and GND pins must be connected.

CKI is the clock input. This can come from an R/C generated oscillator, or a crystal oscillator (in conjunction with CKO). See Oscillator Description section.

$\overline{\text{RESET}}$ is the master reset input. See Reset Description section.

The device contains three bidirectional 8-bit I/O ports (C, G and L), where each individual bit may be independently configured as an input (Schmitt Trigger inputs on ports L and G), output or TRI-STATE under program control. Three data memory address locations are allocated for each of these I/O ports. Each I/O port has two associated 8-bit memory mapped registers, the CONFIGURATION register and the output DATA register. A memory mapped address is also reserved for the input pins of each I/O port. (See the memory map for the various addresses associated with the I/O ports.) *Figure 4* shows the I/O port configurations. The DATA and CONFIGURATION registers allow for each port bit to be individually configured under software control as shown below:

CONFIGURATION Register	DATA Register	Port Set-Up
0	0	Hi-Z Input (TRI-STATE Output)
0	1	Input with Weak Pull-Up
1	0	Push-Pull Zero Output
1	1	Push-Pull One Output

PORT L is an 8-bit I/O port. All L-pins have Schmitt triggers on the inputs.

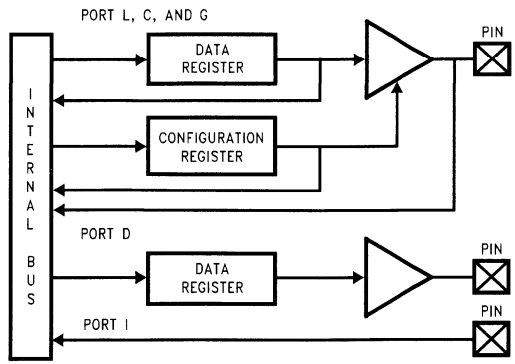
The Port L supports Multi-Input Wake Up on all eight pins. L1 is used for the UART external clock. L2 and L3 are used for the UART transmit and receive. L4 and L5 are used for the timer input functions T2A and T2B. L6 and L7 are used for the timer input functions T3A and T3B.

The Port L has the following alternate features:

- L0 MIWU
- L1 MIWU or CKX
- L2 MIWU or TDX
- L3 MIWU or RDX
- L4 MIWU or T2A
- L5 MIWU or T2B
- L6 MIWU or T3A
- L7 MIWU or T3B

Port G is an 8-bit port with 5 I/O pins (G0, G2–G5), an input pin (G6), and a dedicated output pin (G7). Pins G0 and G2–G6 all have Schmitt Triggers on their inputs. Pin G1 serves as the dedicated WDOUT WATCHDOG output, while pin G7 is either input or output depending on the oscillator mask option selected. With the crystal oscillator option selected, G7 serves as the dedicated output pin for the CKO clock output. With the single-pin R/C oscillator mask option selected, G7 serves as a general purpose input pin but is also used to bring the device out of HALT mode with a low to high transition on G7. There are two registers associated with the G Port, a data register and a configuration register. Therefore, each of the 5 I/O bits (G0, G2–G5) can be individually configured under software control.

Since G6 is an input only pin and G7 is the dedicated CKO clock output pin (crystal clock option) or general purpose input (R/C clock option), the associated bits in the data and configuration registers for G6 and G7 are used for special purpose functions as outlined on the next page. Reading the G6 and G7 data bits will return zeros.



TL/DD12829–5

FIGURE 4. I/O Port Configurations

Pin Descriptions (Continued)

Note that the chip will be placed in the HALT mode by writing a "1" to bit 7 of the Port G Data Register. Similarly the chip will be placed in the IDLE mode by writing a "1" to bit 6 of the Port G Data Register.

Writing a "1" to bit 6 of the Port G Configuration Register enables the MICROWIRE/PLUS to operate with the alternate phase of the SK clock. The G7 configuration bit, if set high, enables the clock start up delay after HALT when the R/C clock configuration is used.

	Config Reg.	Data Reg.
G7	CLKDLY	HALT
G6	Alternate SK	IDLE

Port G has the following alternate features:

- G0 INTR (External Interrupt Input)
- G2 T1B (Timer T1 Capture Input)
- G3 T1A (Timer T1 I/O)
- G4 SO (MICROWIRE Serial Data Output)
- G5 SK (MICROWIRE Serial Clock)
- G6 SI (MICROWIRE Serial Data Input)

Port G has the following dedicated functions:

- G1 WDOUT WATCHDOG and/or Clock Monitor dedicated output
- G7 CKO Oscillator dedicated output or general purpose input

Port C is an 8-bit I/O port. The 40-pin device does not have a full complement of Port C pins. The unavailable pins are not terminated. A read operation for these unterminated pins will return unpredicatable values.

Port I is an eight-bit Hi-Z input port.

Port I1–I3 are used for Comparator 1. Port I4–I6 are used for Comparator 2.

The Port I has the following alternate features:

- I1 COMP1 − IN (Comparator 1 Negative Input)
- I2 COMP1 + IN (Comparator 1 Positive Input)
- I3 COMP1OUT (Comparator 1 Output)
- I4 COMP2 − IN (Comparator 2 Negative Input)
- I5 COMP2 + IN (Comparator 2 Positive Input)
- I6 COMP2OUT (Comparator 2 Output)

Port D is an 8-bit output port that is preset high when $\overline{\text{RESET}}$ goes low. The user can tie two or more D port outputs (except D2) together in order to get a higher drive.

Functional Description

The architecture of the device is modified Harvard architecture. With the Harvard architecture, the control store program memory (ROM) is separated from the data store memory (RAM). Both ROM and RAM have their own separate addressing space with separate address buses. The architecture, though based on Harvard architecture, permits transfer of data from ROM to RAM.

CPU REGISTERS

The CPU can do an 8-bit addition, subtraction, logical or shift operation in one instruction (t_c) cycle time.

There are six CPU registers:

A is the 8-bit Accumulator Register

PC is the 15-bit Program Counter Register

PU is the upper 7 bits of the program counter (PC)
PL is the lower 8 bits of the program counter (PC)

B is an 8-bit RAM address pointer, which can be optionally post auto incremented or decremented.

X is an 8-bit alternate RAM address pointer, which can be optionally post auto incremented or decremented.

SP is the 8-bit stack pointer, which points to the subroutine/interrupt stack (in RAM). The SP is initialized to RAM address 06F with reset.

S is the 8-bit Data Segment Address Register used to extend the lower half of the address range (00 to 7F) into 256 data segments of 128 bytes each.

All the CPU registers are memory mapped with the exception of the Accumulator (A) and the Program Counter (PC).

PROGRAM MEMORY

The program memory consists of 24 kbytes of ROM. These bytes may hold program instructions or constant data (data tables for the LAID instruction, jump vectors for the JID instruction, and interrupt vectors for the VIS instruction). The program memory is addressed by the 15-bit program counter (PC). All interrupts in the devices vector to program memory location 0FF Hex.

DATA MEMORY

The data memory address space includes the on-chip RAM and data registers, the I/O registers (Configuration, Data and Pin), the control registers, the MICROWIRE/PLUS SIO shift register, and the various registers, and counters associated with the timers (with the exception of the IDLE timer). Data memory is addressed directly by the instruction or indirectly by the B, X, SP pointers and S register.

The data memory consists of 1088 bytes of RAM. Sixteen bytes of RAM are mapped as "registers" at addresses 0F0 to 0FF Hex. These registers can be loaded immediately, and also decremented and tested with the DRSZ (decrement register and skip if zero) instruction. The memory pointer registers X, SP, B and S are memory mapped into this space at address locations 0FC to 0FF Hex respectively, with the other registers being available for general usage.

The instruction set permits any bit in memory to be set, reset or tested. All I/O and registers (except A and PC) are memory mapped; therefore, I/O bits and register bits can be directly and individually set, reset and tested. The accumulator (A) bits can also be directly and individually tested.

Note: RAM contents are undefined upon power-up.

Data Memory Segment RAM Extension

Data memory address 0FF is used as a memory mapped location for the Data Segment Address Register (S).

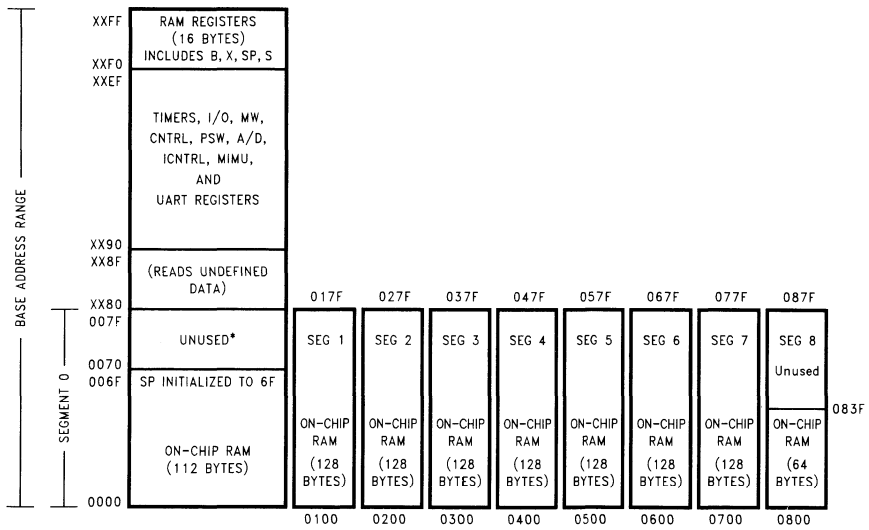
The data store memory is either addressed directly by a single byte address within the instruction, or indirectly relative to the reference of the B, X, or SP pointers (each contains a single-byte address). This single-byte address allows an addressing range of 256 locations from 00 to FF hex. The upper bit of this single-byte address divides the data store memory into two separate sections as outlined previously. With the exception of the RAM register memory from address locations 00F0 to 00FF, all RAM memory is memory mapped with the upper bit of the single-byte address being equal to zero. This allows the upper bit of the single-byte address to determine whether or not the base address range (from 0000 to 00FF) is extended. If this upper bit equals one (representing address range 0080 to 00FF), then address extension does not take place. Alternatively, if this upper bit equals zero, then the data segment extension register S is used to extend the base address range (from 0000 to 007F) from XX00 to XX7F, where XX represents the 8 bits from the S register. Thus the 128-byte data segment extensions are located from addresses 0100 to 017F for data segment 1, 0200 to 027F for data segment 2, etc., up to FF00 to FF7F for data segment 255. The base address range from 0000 to 007F represents data segment 0.

Figure 5 illustrates how the S register data memory extension is used in extending the lower half of the base address range (00 to 7F hex) into 256 data segments of 128 bytes each, with a total addressing range of 32 kbytes from XX00 to XX7F. This organization allows a total of 256 data segments of 128 bytes each with an additional upper base segment of 128 bytes. Furthermore, all addressing modes are available for all data segments. The S register must be changed under program control to move from one data segment (128 bytes) to another. However, the upper base segment (containing the 16 memory registers, I/O registers, control registers, etc.) is always available regardless of the contents of the S register, since the upper base segment (address range 0080 to 00FF) is independent of data segment extension.

The instructions that utilize the stack pointer (SP) always reference the stack as part of the base segment (Segment 0), regardless of the contents of the S register. The S register is not changed by these instructions. Consequently, the stack (used with subroutine linkage and interrupts) is always located in the base segment. The stack pointer will be initialized to point at data memory location 006F as a result of reset.

The 128 bytes of RAM contained in the base segment are split between the lower and upper base segments. The first 112 bytes of RAM are resident from address 0000 to 006F in the lower base segment, while the remaining 16 bytes of RAM represent the 16 data memory registers located at addresses 00F0 to 00FF of the upper base segment. No RAM is located at the upper sixteen addresses (0070 to 007F) of the lower base segment.

Additional RAM beyond these initial 128 bytes, however, will always be memory mapped in groups of 128 bytes (or less) at the data segment address extensions (XX00 to XX7F) of the lower base segment. The additional 960 bytes of RAM are memory mapped at segment 1 through segment 8 (see *Figure 5*).



TL/DD12829-6

*Reads as all ones.

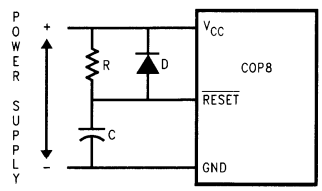
FIGURE 5. RAM Organization

Reset

The $\overline{\text{RESET}}$ input when pulled low initializes the microcontroller. Initialization will occur whenever the $\overline{\text{RESET}}$ input is pulled low. Upon initialization, the data and configuration registers for ports L, G and C are cleared, resulting in these Ports being initialized to the TRI-STATE mode. Pin G1 of the G Port is an exception (as noted below) since pin G1 is dedicated as the WATCHDOG and/or Clock Monitor error output pin. Port D is set high. The PC, PSW, ICNTRL, CNTRL, T2CNTRL and T3CNTRL control registers are cleared. The UART registers PSR, ENU (except that TBMT bit is set), ENUR and ENUI are cleared. The Comparator Select Register is cleared. The S register is initialized to zero. The Multi-Input Wakeup registers WKEN and WKEDG are cleared. Wakeup register WKPND is unknown. The stack pointer, SP, is initialized to 6F hex.

The device comes out of reset with both the WATCHDOG logic and the Clock Monitor detector armed, with the WATCHDOG service window bits set and the Clock Monitor bit set. The WATCHDOG and Clock Monitor circuits are inhibited during reset. The WATCHDOG service window bits being initialized high default to the maximum WATCHDOG service window of 64k t_C clock cycles. The Clock Monitor bit being initialized high will cause a Clock Monitor error following reset if the clock has not reached the minimum specified frequency at the termination of reset. A Clock Monitor error will cause an active low error output on pin G1. This error output will continue until 16 t_C–32 t_C clock cycles following the clock frequency reaching the minimum specified value, at which time the G1 output will enter the TRI-STATE mode.

The external RC network shown in *Figure 6* should be used to ensure that the $\overline{\text{RESET}}$ pin is held low until the power supply to the chip stabilizes.



TL/DD12829–7

RC > 5 × Power Supply Rise Time

FIGURE 6. Recommended Reset Circuit

Oscillator Circuits

The chip can be driven by a clock input on the CKI input pin which can be between DC and 10 MHz. The CKO output clock is on pin G7 (crystal configuration). The CKI input frequency is divided down by 10 to produce the instruction cycle clock ($1/t_C$).

Figure 7 shows the Crystal and R/C oscillator diagrams.

CRYSTAL OSCILLATOR

CKI and CKO can be connected to make a closed loop crystal (or resonator) controlled oscillator.

Table A shows the component values required for various standard crystal values.

R/C OSCILLATOR

By selecting CKI as a single pin oscillator input, a single pin R/C oscillator circuit can be connected to it. CKO is available as a general purpose input, and/or HALT restart input.

Table II shows the variation in the oscillator frequencies as functions of the component (R and C) values.

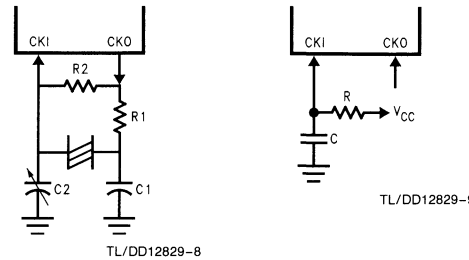


FIGURE 7. Crystal and R/C Oscillator Diagrams

TABLE I. Crystal Oscillator Configuration, T_A = 25°C

R1 (kΩ)	R2 (MΩ)	C1 (pF)	C2 (pF)	CKI Freq (MHz)	Conditions
0	1	30	30–36	10	V_{CC} = 5V
0	1	30	30–36	4	V_{CC} = 5V
0	1	200	100–150	0.455	V_{CC} = 5V

TABLE II. RC Oscillator Configuration, T_A = 25°C

R (kΩ)	C (pF)	CKI Freq (MHz)	Instr. Cycle (μs)	Conditions
3.3	82	2.2 to 2.7	3.7 to 4.6	V_{CC} = 5V
5.6	100	1.1 to 1.3	7.4 to 9.0	V_{CC} = 5V
6.8	100	0.9 to 1.1	8.8 to 10.8	V_{CC} = 5V

Note: 3k ≤ R ≤ 200k

50 pF ≤ C ≤ 200 pF

Control Registers

CNTRL Register (Address X'00EE)

The Timer1 (T1) and MICROWIRE/PLUS control register contains the following bits:

- SL1 & SL0 Select the MICROWIRE/PLUS clock divide by (00 = 2, 01 = 4, 1x = 8)
- IEDG External interrupt edge polarity select (0 = Rising edge, 1 = Falling edge)
- MSEL Selects G5 and G4 as MICROWIRE/PLUS signals SK and SO respectively
- T1C0 Timer T1 Start/Stop control in timer modes 1 and 2
  Timer T1 Underflow Interrupt Pending Flag in timer mode 3
- T1C1 Timer T1 mode control bit
- T1C2 Timer T1 mode control bit
- T1C3 Timer T1 mode control bit

T1C3	T1C2	T1C1	T1C0	MSEL	IEDG	SL1	SL0

Bit 7 Bit 0

PSW Register (Address X'00EF)

The PSW register contains the following select bits:

- GIE Global interrupt enable (enables interrupts)
- EXEN Enable external interrupt
- BUSY MICROWIRE/PLUS busy shifting flag
- EXPND External interrupt pending
- T1ENA Timer T1 Interrupt Enable for Timer Underflow or T1A Input capture edge
- T1PNDA Timer T1 Interrupt Pending Flag (Autoreload RA in mode 1, T1 Underflow in Mode 2, T1A capture edge in mode 3)
- C Carry Flag
- HC Half Carry Flag

HC	C	T1PNDA	T1ENA	EXPND	BUSY	EXEN	GIE

Bit 7 Bit 0

The Half-Carry bit is also affected by all the instructions that affect the Carry flag. The SC (Set Carry) and RC (Reset Carry) instructions will respectively set or clear both the carry flags. In addition to the SC and RC instructions, ADC, SUBC, RRC and RLC instructions affect the carry and Half Carry flags.

ICNTRL Register (Address X'00E8)

The ICNTRL register contains the following bits:

- T1ENB Timer T1 Interrupt Enable for T1B Input capture edge
- T1PNDB Timer T1 Interrupt Pending Flag for T1B capture edge
- μWEN Enable MICROWIRE/PLUS interrupt
- μWPND MICROWIRE/PLUS interrupt pending
- T0EN Timer T0 Interrupt Enable (Bit 12 toggle)
- T0PND Timer T0 Interrupt pending
- LPEN L Port Interrupt Enable (Multi-Input Wakeup/Interrupt)
  Bit 7 could be used as a flag

Unused	LPEN	T0PND	T0EN	μWPND	μWEN	T1PNDB	T1ENB

Bit 7 Bit 0

T2CNTRL Register (Address X'00C6)

The T2CNTRL register contains the following bits:

- T2ENB Timer T2 Interrupt Enable for T2B Input capture edge
- T2PNDB Timer T2 Interrupt Pending Flag for T2B capture edge
- T2ENA Timer T2 Interrupt Enable for Timer Underflow or T2A Input capture edge
- T2PNDA Timer T2 Interrupt Pending Flag (Autoreload RA in mode 1, T2 Underflow in mode 2, T2A capture edge in mode 3)
- T2C0 Timer T2 Start/Stop control in timer modes 1 and 2 Timer T2 Underflow Interrupt Pending Flag in timer mode 3
- T2C1 Timer T2 mode control bit
- T2C2 Timer T2 mode control bit
- T2C3 Timer T2 mode control bit

T2C3	T2C2	T2C1	T2C0	T2PNDA	T2ENA	T2PNDB	T2ENB

Bit 7 Bit 0

T3CNTRL Register (Address X'00B6)

The T3CNTRL register contains the following bits:

- T3ENB Timer T3 Interrupt Enable for T3B Input capture edge
- T3PNDB Timer T3 Interrupt Pending Flag for T3B capture edge
- T3ENA Timer T3 Interrupt Enable for Timer Underflow or T3A Input capture edge
- T3PNDA Timer T3 Interrupt Pending Flag (Autoreload RA in mode 1, T3 Underflow in mode 2, T3A capture edge in mode 3)
- T3C0 Timer T3 Start/Stop control in timer modes 1 and 2
  Timer T3 Underflow Interrupt Pending Flag in timer mode 3
- T3C1 Timer T3 mode control bit
- T3C2 Timer T3 mode control bit
- T3C3 Timer T3 mode control bit

T3C3	T3C2	T3C1	T3C0	T3PNDA	T3ENA	T3PNDB	T3ENB

Bit 7 Bit 0

Timers

The device contains a very versatile set of timers (T0, T1, T2, T3). All timers and associated autoreload/capture registers power up containing random data.

TIMER T0 (IDLE TIMER)

The device supports applications that require maintaining real time and low power with the IDLE mode. This IDLE mode support is furnished by the IDLE timer T0, which is a 16-bit timer. The Timer T0 runs continuously at the fixed rate of the instruction cycle clock, t_C. The user cannot read or write to the IDLE Timer T0, which is a count down timer.

The Timer T0 supports the following functions:

- Exit out of the Idle Mode (See Idle Mode description)
- WATCHDOG logic (See WATCHDOG description)
- Start up delay out of the HALT mode

The IDLE Timer T0 can generate an interrupt when the thirteenth bit toggles. This toggle is latched into the T0PND pending flag, and will occur every 4 ms at the maximum clock frequency (t_C = 1 μs). A control flag T0EN allows the interrupt from the thirteenth bit of Timer T0 to be enabled or disabled. Setting T0EN will enable the interrupt, while resetting it will disable the interrupt.

TIMER T1, TIMER T2 AND TIMER T3

The device has a set of three powerful timer/counter blocks, T1, T2 and T3. The associated features and functioning of a timer block are described by referring to the timer block Tx. Since the three timer blocks, T1, T2 and T3 are identical, all comments are equally applicable to any of the three timer blocks.

Each timer block consists of a 16-bit timer, Tx, and two supporting 16-bit autoreload/capture registers, RxA and RxB. Each timer block has two pins associated with it, TxA and TxB. The pin TxA supports I/O required by the timer block, while the pin TxB is an input to the timer block. The powerful and flexible timer block allows the device to easily perform all timer functions with minimal software overhead. The timer block has three operating modes: Processor Independent PWM mode, External Event Counter mode, and Input Capture mode.

The control bits TxC3, TxC2, and TxC1 allow selection of the different modes of operation.

Mode 1. Processor Independent PWM Mode

As the name suggests, this mode allows the device to generate a PWM signal with very minimal user intervention. The user only has to define the parameters of the PWM signal (ON time and OFF time). Once begun, the timer block will continuously generate the PWM signal completely independent of the microcontroller. The user software services the timer block only when the PWM parameters require updating.

In this mode the timer Tx counts down at a fixed rate of t_C. Upon every underflow the timer is alternately reloaded with the contents of supporting registers, RxA and RxB. The very first underflow of the timer causes the timer to reload from the register RxA. Subsequent underflows cause the timer to be reloaded from the registers alternately beginning with the register RxB.

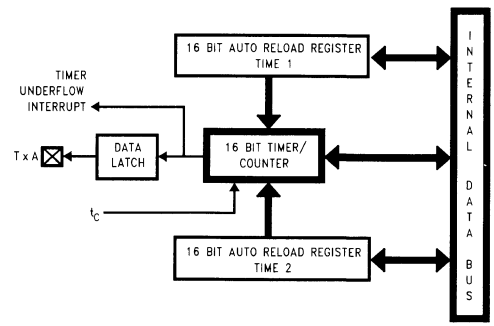
The Tx Timer control bits, TxC3, TxC2 and TxC1 set up the timer for PWM mode operation.

Figure 8 shows a block diagram of the timer in PWM mode.

The underflows can be programmed to toggle the TxA output pin. The underflows can also be programmed to generate interrupts.

Underflows from the timer are alternately latched into two pending flags, TxPNDA and TxPNDB. The user must reset these pending flags under software control. Two control enable flags, TxENA and TxENB, allow the interrupts from the timer underflow to be enabled or disabled. Setting the timer enable flag TxENA will cause an interrupt when a timer underflow causes the RxA register to be reloaded into the timer. Setting the timer enable flag TxENB will cause an interrupt when a timer underflow causes the RxB register to be reloaded into the timer. Resetting the timer enable flags will disable the associated interrupts.

Either or both of the timer underflow interrupts may be enabled. This gives the user the flexibility of interrupting once per PWM period on either the rising or falling edge of the PWM output. Alternatively, the user may choose to interrupt on both edges of the PWM output.



TL/DD/12829–10

FIGURE 8. Timer in PWM Mode

Mode 2. External Event Counter Mode

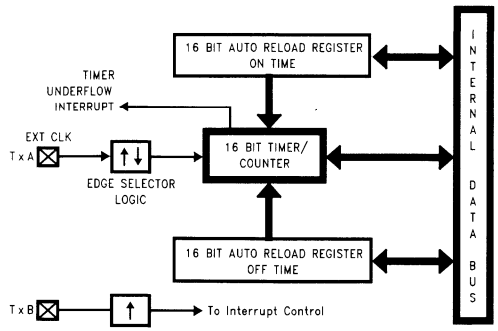
This mode is quite similar to the processor independent PWM mode described above. The main difference is that the timer, Tx, is clocked by the input signal from the TxA pin. The Tx timer control bits, TxC3, TxC2 and TxC1 allow the timer to be clocked either on a positive or negative edge from the TxA pin. Underflows from the timer are latched into the TxPNDA pending flag. Setting the TxENA control flag will cause an interrupt when the timer underflows.

In this mode the input pin TxB can be used as an independent positive edge sensitive interrupt input if the TxENB control flag is set. The occurrence of a positive edge on the TxB input pin is latched into the TxPNDB flag.

Timers (Continued)

Figure 9 shows a block diagram of the timer in External Event Counter mode.

Note: The PWM output is not available in this mode since the TxA pin is being used as the counter input clock.



TL/DD12829–11

FIGURE 9. Timer in External Event Counter Mode

Mode 3. Input Capture Mode

The device can precisely measure external frequencies or time external events by placing the timer block, Tx, in the input capture mode.

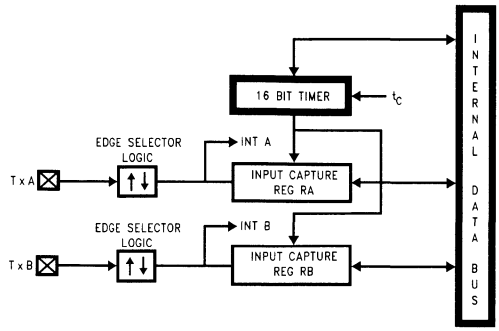
In this mode, the timer Tx is constantly running at the fixed t_C rate. The two registers, RxA and RxB, act as capture registers. Each register acts in conjunction with a pin. The register RxA acts in conjunction with the TxA pin and the register RxB acts in conjunction with the TxB pin.

The timer value gets copied over into the register when a trigger event occurs on its corresponding pin. Control bits, TxC3, TxC2 and TxC1, allow the trigger events to be specified either as a positive or a negative edge. The trigger condition for each input pin can be specified independently.

The trigger conditions can also be programmed to generate interrupts. The occurrence of the specified trigger condition on the TxA and TxB pins will be respectively latched into the pending flags, TxPNDA and TxPNDB. The control flag TxENA allows the interrupt on TxA to be either enabled or disabled. Setting the TxENA flag enables interrupts to be generated when the selected trigger condition occurs on the TxA pin. Similarly, the flag TxENB controls the interrupts from the TxB pin.

Underflows from the timer can also be programmed to generate interrupts. Underflows are latched into the timer TxC0 pending flag (the TxC0 control bit serves as the timer underflow interrupt pending flag in the Input Capture mode). Consequently, the TxC0 control bit should be reset when entering the Input Capture mode. The timer underflow interrupt is enabled with the TxENA control flag. When a TxA interrupt occurs in the Input Capture mode, the user must check both the TxPNDA and TxC0 pending flags in order to determine whether a TxA input capture or a timer underflow (or both) caused the interrupt.

Figure 10 shows a block diagram of the timer in Input Capture mode.



TL/DD12829–12

FIGURE 10. Timer in Input Capture Mode

TIMER CONTROL FLAGS

The timers T1, T2 and T3 have indentical control structures. The control bits and their functions are summarized below.

TxC0	Timer Start/Stop control in Modes 1 and 2 (Processor Independent PWM and External Event Counter), where 1 = Start, 0 = Stop Timer Underflow Interrupt Pending Flag in Mode 3 (Input Capture)
TxPNDA	Timer Interrupt Pending Flag
TxPNDB	Timer Interrupt Pending Flag
TxENA	Timer Interrupt Enable Flag
TxENB	Timer Interrupt Enable Flag 1 = Timer Interrupt Enabled 0 = Timer Interrupt Disabled
TxC3	Timer mode control
TxC2	Timer mode control
TxC1	Timer mode control

Timers (Continued)

The timer mode control bits (TxC3, TxC2 and TxC1) are detailed below:

TxC3	TxC2	TxC1	Timer Mode	Interrupt A Source	Interrupt B Source	Timer Counts On
0	0	0	MODE 2 (External Event Counter)	Timer Underflow	Pos. TxB Edge	TxA Pos. Edge
0	0	1	MODE 2 (External Event Counter)	Timer Underflow	Pos. TxB Edge	TxA Neg. Edge
1	0	1	MODE 1 (PWM) TxA Toggle	Autoreload RA	Autoreload RB	t_C
1	0	0	MODE 1 (PWM) No TxA Toggle	Autoreload RA	Autoreload RB	t_C
0	1	0	MODE 3 (Capture) Captures: TxA Pos. Edge TxB Pos. Edge	Pos. TxA Edge or Timer Underflow	Pos. TxB Edge	t_C
1	1	0	MODE 3 (Capture) Captures: TxA Pos. Edge TxB Neg. Edge	Pos. TxA Edge or Timer Underflow	Neg. TxB Edge	t_C
0	1	1	MODE 3 (Capture) Captures: TxA Neg. Edge TxB Pos. Edge	Neg. TxA Edge or Timer Underflow	Pos. TxB Edge	t_C
1	1	1	MODE 3 (Capture) Captures: TxA Neg. Edge TxB Neg. Edge	Neg. TxA Edge or Timer Underflow	Neg. TxB Edge	t_C

Power Save Modes

The device offers the user two power save modes of operation: HALT and IDLE. In the HALT mode, all microcontroller activities are stopped. In the IDLE mode, the on-board oscillator circuitry and timer T0 are active but all other microcontroller activities are stopped. In either mode, all on-board RAM, registers, I/O states, and timers (with the exception of T0) are unaltered.

HALT MODE

The device can be placed in the HALT mode by writing a "1" to the HALT flag (G7 data bit). All microcontroller activities, including the clock and timers, are stopped. The WATCHDOG logic on the device is disabled during the HALT mode. However, the clock monitor circuitry if enabled remains active and will cause the WATCHDOG output pin (WDOUT) to go low. If the HALT mode is used and the user does not want to activate the WDOUT pin, the Clock Monitor should be disabled after the device comes out of reset (resetting the Clock Monitor control bit with the first write to the WDSVR register). In the HALT mode, the power requirements of the device are minimal and the applied voltage (V_{CC}) may be decreased to V_r (V_r = 2.0V) without altering the state of the machine.

The device supports three different ways of exiting the HALT mode. The first method of exiting the HALT mode is with the Multi-Input Wakeup feature on the L port. The second method is with a low to high transition on the CKO (G7) pin. This method precludes the use of the crystal clock configuration (since CKO becomes a dedicated output), and so may only be used with an RC clock configuration. The third method of exiting the HALT mode is by pulling the $\overline{\text{RESET}}$ pin low.

Since a crystal or ceramic resonator may be selected as the oscillator, the Wakeup signal is not allowed to start the chip running immediately since crystal oscillators and ceramic resonators have a delayed start up time to reach full amplitude and frequency stability. The IDLE timer is used to generate a fixed delay to ensure that the oscillator has indeed stabilized before allowing instruction execution. In this case, upon detecting a valid Wakeup signal, only the oscillator circuitry is enabled. The IDLE timer is loaded with a value of 256 and is clocked with the t_C instruction cycle clock. The t_C clock is derived by dividing the oscillator clock down by a factor of 10. The Schmitt trigger following the CKI inverter on the chip ensures that the IDLE timer is clocked only when the oscillator has a sufficiently large amplitude to meet the Schmitt trigger specifications. This Schmitt trigger is not part of the oscillator closed loop. The startup timeout from the IDLE timer enables the clock signals to be routed to the rest of the chip.

If an RC clock option is being used, the fixed delay is introduced optionally. A control bit, CLKDLY, mapped as configuration bit G7, controls whether the delay is to be introduced or not. The delay is included if CLKDLY is set, and excluded if CLKDLY is reset. The CLKDLY bit is cleared on reset.

Power Save Modes (Continued)

The device has two mask options associated with the HALT mode. The first mask option enables the HALT mode feature, while the second mask option disables the HALT mode. With the HALT mode enable mask option, the device will enter and exit the HALT mode as described above. With the HALT disable mask option, the device cannot be placed in the HALT mode (writing a "1" to the HALT flag will have no effect, the HALT flag will remain "0").

IDLE MODE

The device is placed in the IDLE mode by writing a "1" to the IDLE flag (G6 data bit). In this mode, all activities, except the associated on-board oscillator circuitry and the IDLE Timer T0, are stopped. The power supply requirements of the micro-controller in this mode of operation are typically around 30% of normal power requirement of the microcontroller.

As with the HALT mode, the device can be returned to normal operation with a reset, or with a Multi-Input Wakeup from the L Port. Alternately, the microcontroller resumes normal operation from the IDLE mode when the thirteenth bit (representing 4.096 ms at internal clock frequency of 10 MHz, t_C = 1 μs) of the IDLE Timer toggles.

This toggle condition of the thirteenth bit of the IDLE Timer T0 is latched into the T0PND pending flag.

The user has the option of being interrupted with a transition on the thirteenth bit of the IDLE Timer T0. The interrupt can be enabled or disabled via the T0EN control bit. Setting the T0EN flag enables the interrupt and vice versa.

The user can enter the IDLE mode with the Timer T0 interrupt enabled. In this case, when the T0PND bit gets set, the device will first execute the Timer T0 interrupt service routine and then return to the instruction following the "Enter Idle Mode" instruction.

Alternatively, the user can enter the IDLE mode with the IDLE Timer T0 interrupt disabled. In this case, the device will resume normal operation with the instruction immediately following the "Enter IDLE Mode" instruction.

Note: It is necessary to program two NOP instructions following both the set HALT mode and set IDLE mode instructions. These NOP instructions are necessary to allow clock resynchronization following the HALT or IDLE modes.

Multi-Input Wakeup

The Multi-Input Wakeup feature is ued to return (wakeup) the device from either the HALT or IDLE modes. Alternately Multi-Input Wakeup/Interrupt feature may also be used to generate up to 8 edge selectable external interrupts.

Figure 11 shows the Multi-Input Wakeup logic.

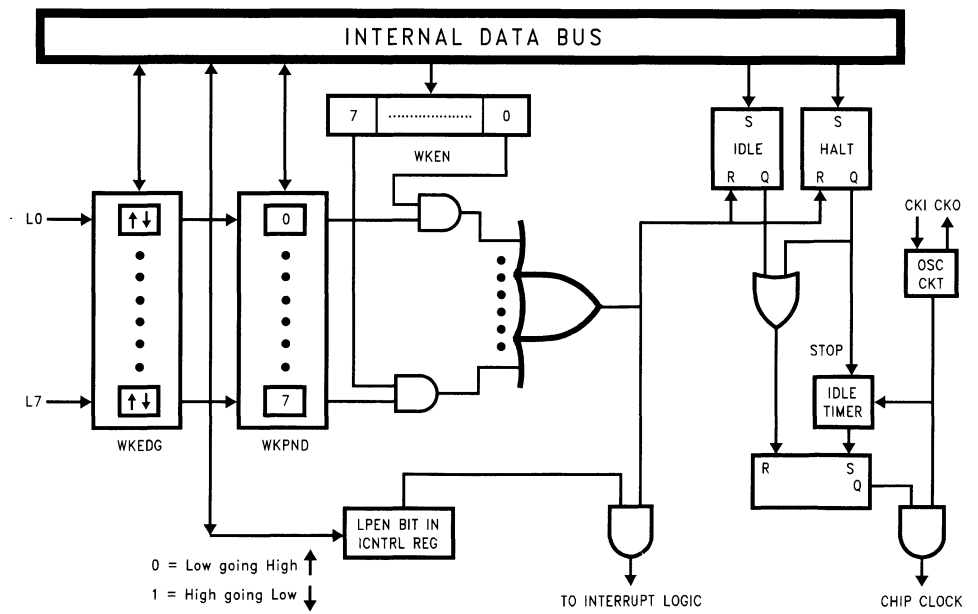


FIGURE 11. Multi-Input Wake Up Logic

Multi-Input Wakeup (Continued)

The Multi-Input Wakeup feature utilizes the L Port. The user selects which particular L port bit (or combination of L Port bits) will cause the device to exit the HALT or IDLE modes. The selection is done through the Register WKEN. The Register WKEN is an 8-bit read/write register, which contains a control bit for every L port bit. Setting a particular WKEN bit enables a Wakeup from the associated L port pin.

The user can select whether the trigger condition on the selected L Port pin is going to be either a positive edge (low to high transition) or a negative edge (high to low transition). This selection is made via the Register WKEDG, which is an 8-bit control register with a bit assigned to each L Port pin. Setting the control bit will select the trigger condition to be a negative edge on that particular L Port pin. Resetting the bit selects the trigger condition to be a positive edge. Changing an edge select entails several steps in order to avoid a Wakeup condition as a result of the edge change. First, the associated WKEN bit should be reset, followed by the edge select change in WKEDG. Next, the associated WKPND bit should be cleared, followed by the associated WKEN bit being re-enabled.

An example may serve to clarify this procedure. Suppose we wish to change the edge select from positive (low going high) to negative (high going low) for L Port bit 5, where bit 5 has previously been enabled for an input interrupt. The program would be as follows:

```
RBIT  5, WKEN      ; Disable Port bit 5 for
                   ; Wakeup
SBIT  5, WKEDG     ; Select neg going edge
                   ; sensitivity
RBIT  5, WKPND     ; Clear pending bit
SBIT  5, WKEN      ; Re-enable the bit for
                   ; Wakeup
```

If the L port bits have been used as outputs and then changed to inputs with Multi-Input Wakeup/Interrupt, a safety procedure should also be followed to avoid wakeup conditions. After the selected L port bits have been changed from output to input but before the associated WKEN bits are enabled, the associated edge select bits in WKEDG should be set or reset for the desired edge selects, followed by the associated WKPND bits being cleared.

This same procedure should be used following reset, since the L port inputs are left floating as a result of reset.

The occurrence of the selected trigger condition for Multi-Input Wakeup is latched into a pending register called WKPND. The respective bits of the WKPND register will be set on the occurrence of the selected trigger edge on the corresponding Port L pin. The user has the responsibility of clearing these pending flags. Since WKPND is a pending register for the occurrence of selected wakeup conditions, the device will not enter the HALT mode if any Wakeup bit is both enabled and pending. Consequently, the user must clear the pending flags before attempting to enter the HALT mode.

WKEN, WKPND and WKEDG are all read/write registers, and are cleared at reset.

PORT L INTERRUPTS

Port L provides the user with an additional eight fully selectable, edge sensitive interrupts which are all vectored into the same service subroutine.

The interrupt from Port L shares logic with the wake up circuitry. The register WKEN allows interrupts from Port L to be individually enabled or disabled. The register WKEDG specifies the trigger condition to be either a positive or a negative edge. Finally, the register WKPND latches in the pending trigger conditions.

The GIE (Global Interrupt Enable) bit enables the interrupt function.

A control flag, LPEN, functions as a global interrupt enable for Port L interrupts. Setting the LPEN flag will enable interrupts and vice versa. A separate global pending flag is not needed since the register WKPND is adequate.

Since Port L is also used for waking the device out of the HALT or IDLE modes, the user can elect to exit the HALT or IDLE modes either with or without the interrupt enabled. If he elects to disable the interrupt, then the device will restart execution from the instruction immediately following the instruction that placed the microcontroller in the HALT or IDLE modes. In the other case, the device will first execute the interrupt service routine and then revert to normal operation. (See HALT MODE for clock option wakeup information.)

UART

The device contains a full-duplex software programmable UART. The UART *(Figure 12)* consists of a transmit shift register, a receive shift register and seven addressable registers, as follows: a transmit buffer register (TBUF), a receiver buffer register (RBUF), a UART control and status register (ENU), a UART receive control and status register (ENUR), a UART interrupt and clock source register (ENUI), a prescaler select register (PSR) and baud (BAUD) register. The ENU register contains flags for transmit and receive functions; this register also determines the length of the data frame (7, 8 or 9 bits), the value of the ninth bit in transmission, and parity selection bits. The ENUR register flags framing, data overrun and parity errors while the UART is receiving.

Other functions of the ENUR register include saving the ninth bit received in the data frame, enabling or disabling the UART's attention mode of operation and providing additional receiver/transmitter status information via RCVG and XMTG bits. The determination of an internal or external clock source is done by the ENUI register, as well as selecting the number of stop bits and enabling or disabling transmit and receive interrupts. A control flag in this register can also select the UART mode of operation: asynchronous or synchronous.

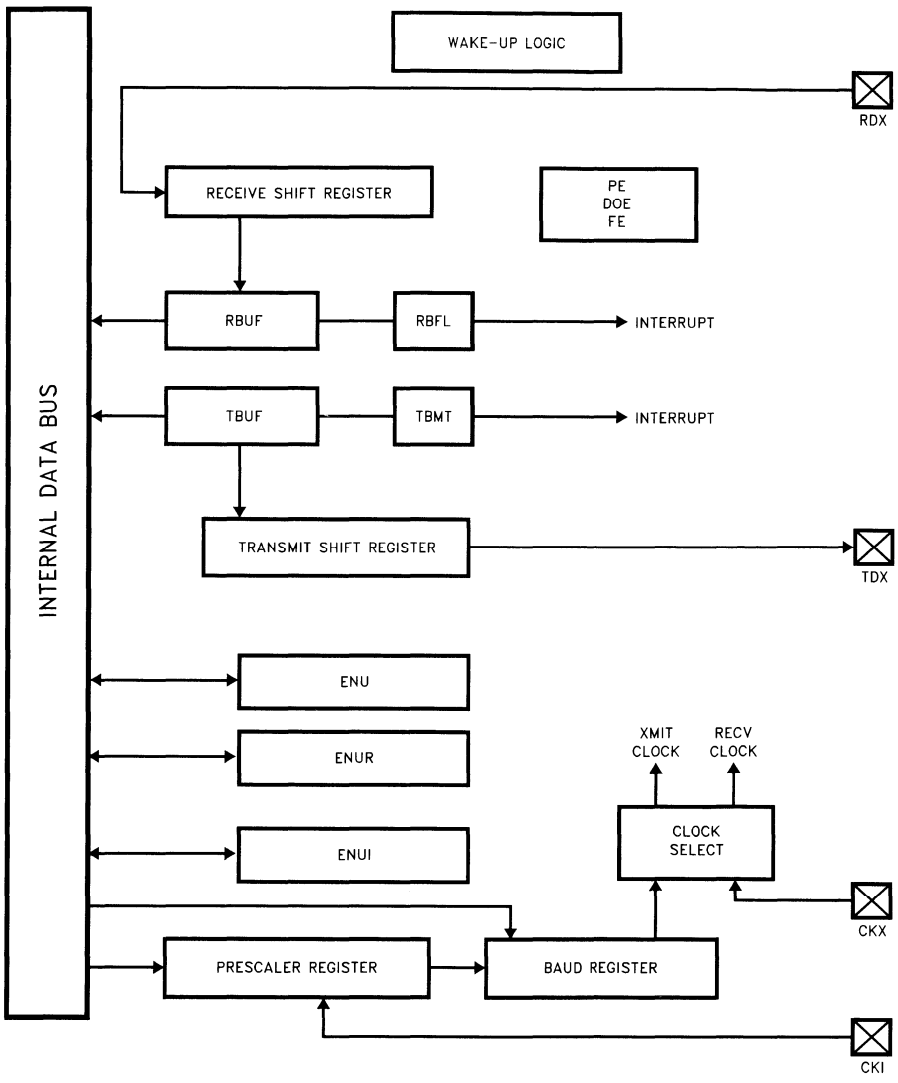


FIGURE 12. UART Block Diagram

UART (Continued)

UART CONTROL AND STATUS REGISTERS

The operation of the UART is programmed through three registers: ENU, ENUR and ENUI. The function of the individual bits in these registers is as follows:

ENU-UART Control and Status Register (Address at 0BA)

PEN	PSEL1	XBIT9/ PSEL0	CHL1	CHL0	ERR	RBFL	TBMT
0RW	0RW	0RW	0RW	0RW	0R	0R	1R

Bit 7 Bit 0

ENUR-UART Receive Control and Status Register (Address at 0BB)

DOE	FE	PE	SPARE	RBIT9	ATTN	XMTG	RCVG
0RD	0RD	0RD	0RW*	0R	0RW	0R	0R

Bit7 Bit0

ENUI-UART Interrupt and Clock Source Register (Address at 0BC)

STP2	STP78	ETDX	SSEL	XRCLK	XTCLK	ERI	ETI
0RW	0RW	0RW	0RW	0RW	0RW	0RW	0RW

Bit7 Bit0

*Bit is not used.

0 Bit is cleared on reset.

1 Bit is set to one on reset.

R Bit is read-only; it cannot be written by software.

RW Bit is read/write.

D Bit is cleared on read; when read by software as a one, it is cleared automatically. Writing to the bit does not affect its state.

DESCRIPTION OF UART REGISTER BITS

ENU—UART CONTROL AND STATUS REGISTER

TBMT: This bit is set when the UART transfers a byte of data from the TBUF register into the TSFT register for transmission. It is automatically reset when software writes into the TBUF register.

RBFL: This bit is set when the UART has received a complete character and has copied it into the RBUF register. It is automatically reset when software reads the character from RBUF.

ERR: This bit is a global UART error flag which gets set if any or a combination of the errors (DOE, FE, PE) occur.

CHL1, CHL0: These bits select the character frame format. Parity is not included and is generated/verified by hardware.

CHL1 = 0, CHL0 = 0 The frame contains eight data bits.
CHL1 = 0, CHL0 = 1 The frame contains seven data bits.
CHL1 = 1, CHL0 = 0 The frame contains nine data bits.
CHL1 = 1, CHL0 = 1 Loopback Mode selected. Transmitter output internally looped back to receiver input. Nine bit framing format is used.

XBIT9/PSEL0: Programs the ninth bit for transmission when the UART is operating with nine data bits per frame. For seven or eight data bits per frame, this bit in conjunction with PSEL1 selects parity.

PSEL1, PSEL0: Parity select bits.

PSEL1 = 0, PSEL0 = 0 Odd Parity (if Parity enabled)
PSEL1 = 0, PSEL0 = 1 Odd Parity (if Parity enabled)
PSEL1 = 1, PSEL0 = 0 Mark(1) (if Parity enabled)
PSEL1 = 1, PSEL0 = 1 Space(0) (if Parity enabled)

PEN: This bit enables/disables Parity (7- and 8-bit modes only).

PEN = 0 Parity disabled.
PEN = 1 Parity enabled.

ENUR—UART RECEIVE CONTROL AND STATUS REGISTER

RCVG: This bit is set high whenever a framing error occurs and goes low when RDX goes high.

XMTG: This bit is set to indicate that the UART is transmitting. It gets reset at the end of the last frame (end of last Stop bit).

ATTN: ATTENTION Mode is enabled while this bit is set. This bit is cleared automatically on receiving a character with data bit nine set.

RBIT9: Contains the ninth data bit received when the UART is operating with nine data bits per frame.

SPARE: Reserved for future use.

PE: Flags a Parity Error.

PE = 0 Indicates no Parity Error has been detected since the last time the ENUR register was read.
PE = 1 Indicates the occurrence of a Parity Error.

FE: Flags a Framing Error.

FE = 0 Indicates no Framing Error has been detected since the last time the ENUR register was read.
FE = 1 Indicates the occurrence of a Framing Error.

DOE: Flags a Data Overrun Error.

DOE = 0 Indicates no Data Overrun Error has been detected since the last time the ENUR register was read.
DOE = 1 Indicates the occurrence of a Data Overrun Error.

ENUI—UART INTERRUPT AND CLOCK SOURCE REGISTER

ETI: This bit enables/disables interrupt from the transmitter section.

ETI = 0 Interrupt from the transmitter is disabled.
ETI = 1 Interrupt from the transmitter is enabled.

ERI: This bit enables/disables interrupt from the receiver section.

ERI = 0 Interrupt from the receiver is disabled.
ERI = 1 Interrupt from the receiver is enabled.

XTCLK: This bit selects the clock source for the transmitter section.

XTCLK = 0 The clock source is selected through the PSR and BAUD registers.
XTCLK = 1 Signal on CKX (L1) pin is used as the clock.

XRCLK: This bit selects the clock source for the receiver section.

XRCLK = 0 The clock source is selected through the PSR and BAUD registers.
XRCLK = 1 Signal on CKX (L1) pin is used as the clock.

SSEL: UART mode select.

SSEL = 0 Asynchronous Mode.
SSEL = 1 Synchronous Mode.

UART (Continued)

ETDX: TDX (UART Transmit Pin) is the alternate function assigned to Port L pin L2; it is selected by setting ETDX bit. To simulate line break generation, software should reset ETDX bit and output logic zero to TDX pin through Port L data and configuration registers.

STP78: This bit is set to program the last Stop bit to be 7/8th of a bit in length.

STP2: This bit programs the number of Stop bits to be transmitted.

STP2 = 0 One Stop bit transmitted.
STP2 = 1 Two Stop bits transmitted.

Associated I/O Pins

Data is transmitted on the TDX pin and received on the RDX pin. TDX is the alternate function assigned to Port L pin L2; it is selected by setting ETDX (in the ENUI register) to one. RDX is an inherent function of Port L pin L3, requiring no setup.

The baud rate clock for the UART can be generated on-chip, or can be taken from an external source. Port L pin L1 (CKX) is the external clock I/O pin. The CKX pin can be either an input or an output, as determined by Port L Configuration and Data registers (Bit 1). As an input, it accepts a clock signal which may be selected to drive the transmitter and/or receiver. As an output, it presents the internal Baud Rate Generator output.

UART Operation

The UART has two modes of operation: asynchronous mode and synchronous mode.

ASYNCHRONOUS MODE

This mode is selected by resetting the SSEL (in the ENUI register) bit to zero. The input frequency to the UART is 16 times the baud rate.

The TSFT and TBUF registers double-buffer data for transmission. While TSFT is shifting out the current character on the TDX pin, the TBUF register may be loaded by software with the next byte to be transmitted. When TSFT finishes transmitting the current character the contents of TBUF are transferred to the TSFT register and the Transmit Buffer Empty Flag (TBMT in the ENU register) is set. The TBMT flag is automatically reset by the UART when software loads a new character into the TBUF register. There is also the XMTG bit which is set to indicate that the UART is transmitting. This bit gets reset at the end of the last frame (end of last Stop bit). TBUF is a read/write register.

The RSFT and RBUF registers double-buffer data being received. The UART receiver continually monitors the signal on the RDX pin for a low level to detect the beginning of a Start bit. Upon sensing this low level, it waits for half a bit time and samples again. If the RDX pin is still low, the receiver considers this to be a valid Start bit, and the remaining bits in the character frame are each sampled a single time, at the mid-bit position. Serial data input on the RDX pin is shifted into the RSFT register. Upon receiving the complete character, the contents of the RSFT register are copied into the RBUF register and the Received Buffer Full Flag (RBFL) is set. RBFL is automatically reset when software reads the character from the RBUF register. RBUF is a read only register. There is also the RCVG bit which is set high when a framing error occurs and goes low once RDX goes high. TBMT, XMTG, RBFL and RCVG are read only bits.

SYNCHRONOUS MODE

In this mode data is transferred synchronously with the clock. Data is transmitted on the rising edge and received on the falling edge of the synchronous clock.

This mode is selected by setting SSEL bit in the ENUI register. The input frequency to the UART is the same as the baud rate.

When an external clock input is selected at the CKX pin, data transmit and receive are performed synchronously with this clock through TDX/RDX pins.

If data transmit and receive are selected with the CKX pin as clock output, the device generates the synchronous clock output at the CKX pin. The internal baud rate generator is used to produce the synchronous clock. Data transmit and receive are performed synchronously with this clock.

FRAMING FORMATS

The UART supports several serial framing formats *(Figure 13).* The format is selected using control bits in the ENU, ENUR and ENUI registers.

The first format (1, 1a, 1b, 1c) for data transmission (CHL0 = 1, CHL1 = 0) consists of Start bit, seven Data bits (excluding parity) and 7/8, one or two Stop bits. In applications using parity, the parity bit is generated and verified by hardware.

The second format (CHL0 = 0, CHL1 = 0) consists of one Start bit, eight Data bits (excluding parity) and 7/8, one or two Stop bits. Parity bit is generated and verified by hardware.

The third format for transmission (CHL0 = 0, CHL1 = 1) consists of one Start bit, nine Data bits and 7/8, one or two Stop bits. This format also supports the UART "ATTENTION" feature. When operating in this format, all eight bits of TBUF and RBUF are used for data. The ninth data bit is transmitted and received using two bits in the ENU and ENUR registers, called XBIT9 and RBIT9. RBIT9 is a read only bit. Parity is not generated or verified in this mode.

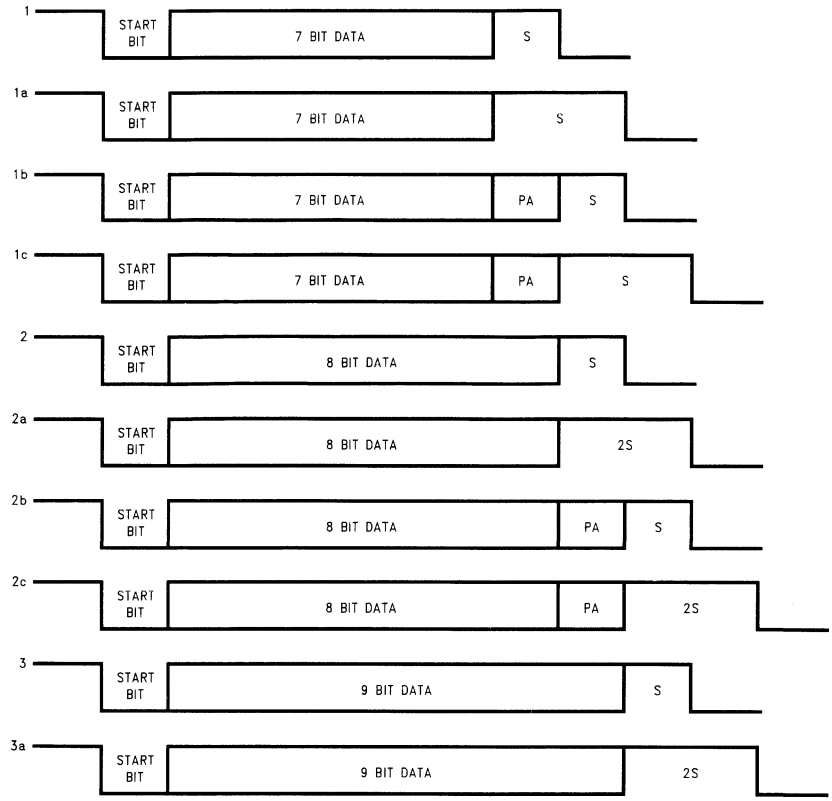
For any of the above framing formats, the last Stop bit can be programmed to be 7/8th of a bit in length. If two Stop bits are selected and the 7/8th bit is set (selected), the second Stop bit will be 7/8th of a bit in length.

The parity is enabled/disabled by PEN bit located in the ENU register. Parity is selected for 7- and 8-bit modes only. If parity is enabled (PEN = 1), the parity selection is then performed by PSEL0 and PSEL1 bits located in the ENU register.

Note that the XBIT9/PSEL0 bit located in the ENU register serves two mutually exclusive functions. This bit programs the ninth bit for transmission when the UART is operating with nine data bits per frame. There is no parity selection in this framing format. For other framing formats XBIT9 is not needed and the bit is PSEL0 used in conjunction with PSEL1 to select parity.

The frame formats for the receiver differ from the transmitter in the number of Stop bits required. The receiver only requires one Stop bit in a frame, regardless of the setting of the Stop bit selection bits in the control register. Note that an implicit assumption is made for full duplex UART operation that the framing formats are the same for the transmitter and receiver.

UART Operation (Continued)



TL/DD12829–15

FIGURE 13. Framing Formats

UART INTERRUPTS

The UART is capable of generating interrupts. Interrupts are generated on Receive Buffer Full and Transmit Buffer Empty. Both interrupts have individual interrupt vectors. Two bytes of program memory space are reserved for each interrupt vector. The two vectors are located at addresses 0xEC to 0xEF Hex in the program memory space. The interrupts can be individually enabled or disabled using Enable Transmit Interrupt (ETI) and Enable Receive Interrupt (ERI) bits in the ENUI register.

The interrupt from the Transmitter is set pending, and remains pending, as long as both the TBMT and ETI bits are set. To remove this interrupt, software must either clear the ETI bit or write to the TBUF register (thus clearing the TBMT bit).

The interrupt from the receiver is set pending, and remains pending, as long as both the RBFL and ERI bits are set. To remove this interrupt, software must either clear the ERI bit or read from the RBUF register (thus clearing the RBFL bit).

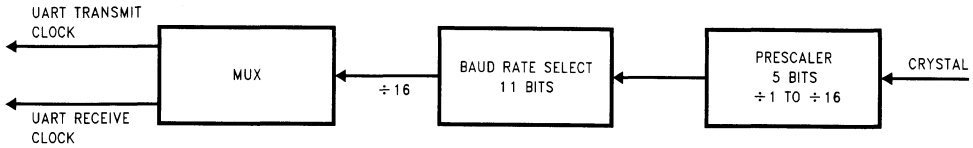
Baud Clock Generation

The clock inputs to the transmitter and receiver sections of the UART can be individually selected to come either from an external source at the CKX pin (port L, pin L1) or from a source selected in the PSR and BAUD registers. Internally, the basic baud clock is created from the oscillator frequency through a two-stage divider chain consisting of a 1–16 (increments of 0.5) prescaler and an 11-bit binary counter. *(Figure 14).* The divide factors are specified through two read/write registers shown in *Figure 15*. Note that the 11-bit Baud Rate Divisor spills over into the Prescaler Select Register (PSR). PSR is cleared upon reset.

As shown in Table V, a Prescaler Factor of 0 corresponds to NO CLOCK. This condition is the UART power down mode where the UART clock is turned off for power saving purpose. The user must also turn the UART clock off when a different baud rate is chosen.

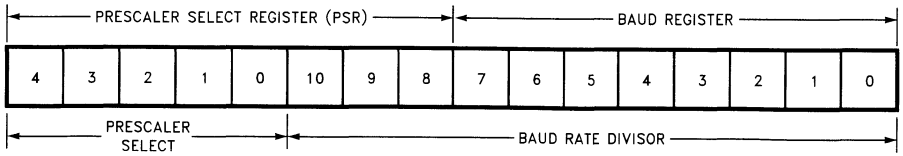
The correspondences between the 5-bit Prescaler Select and Prescaler factors are shown in Table V. There are many ways to calculate the two divisor factors, but one particularly effective method would be to achieve a 1.8432 MHz frequency coming out of the first stage. The 1.8432 MHz prescaler output is then used to drive the software programmable baud rate counter to create a 16x clock for the following baud rates: 110, 134.5, 150, 300, 600, 1200, 1800, 2400, 3600, 4800, 7200, 9600, 19200 and 38400 (Table I). Other baud rates may be created by using appropriate divisors. The 16x clock is then divided by 16 to provide the rate for the serial shift registers of the transmitter and receiver.

Baud Clock Generation (Continued)



TL/DD12829–16

FIGURE 14. UART BAUD Clock Generation



TL/DD12829–17

FIGURE 15. UART BAUD Clock Divisor Registers

Baud Clock Generation (Continued)

TABLE III. Baud Rate Divisors (1.8432 MHz Prescaler Output)

Baud Rate	Baud Rate Divisor − 1 (N-1)
110 (110.03)	1046
134.5 (134.58)	855
150	767
300	383
600	191
1200	95
1800	63
2400	47
3600	31
4800	23
7200	15
9600	11
19200	5
38400	2

Note: The entries in Table IV assume a prescaler output of 1.8432 MHz. In the asynchronous mode the baud rate could be as high as 625k.

TABLE IV. Prescaler Factors

Prescaler Select	Prescaler Factor
00000	NO CLOCK
00001	1
00010	1.5
00011	2
00100	2.5
00101	3
00110	3.5
00111	4
01000	4.5
01001	5
01010	5.5
01011	6
01100	6.5
01101	7
01110	7.5
01111	8
10000	8.5
10001	9
10010	9.5
10011	10
10100	10.5
10101	11
10110	11.5
10111	12
11000	12.5
11001	13
11010	13.5
11011	14
11100	14.5
11101	15
11110	15.5
11111	16

As an example, considering Asynchronous Mode and a CKI clock of 4.608 MHz, the prescaler factor selected is:

$$4.608/1.8432 = 2.5$$

The 2.5 entry is available in Table IV. The 1.8432 MHz prescaler output is then used with proper Baud Rate Divisor (Table V) to obtain different baud rates. For a baud rate of 19200 e.g., the entry in Table IV is 5.

$N - 1 = 5$ ($N - 1$ is the value from Table III)

$N = 6$ (N is the Baud Rate Divisor)

$$\text{Baud Rate} = 1.8432\ \text{MHz}/(16 \times 6) = 19200$$

The divide by 16 is performed because in the asynchronous mode, the input frequency to the UART is 16 times the baud rate. The equation to calculate baud rates is given below.

The actual Baud Rate may be found from:

$$BR = Fc/(16 \times N \times P)$$

Baud Clock Generation (Continued)

Where:

BR is the Baud Rate

Fc is the CKI frequency

N is the Baud Rate Divisor (Table IV).

P is the Prescaler Divide Factor selected by the value in the Prescaler Select Register (Table III)

Note: In the Synchronous Mode, the divisor 16 is replaced by two.

Example:

Asynchronous Mode:

Crystal Frequency = 5 MHz

Desired baud rate = 9600

Using the above equation N × P can be calculated first.

$$N \times P = (5 \times 106)/(16 \times 9600) = 32.552$$

Now 32.552 is divided by each Prescaler Factor (Table III) to obtain a value closest to an integer. This factor happens to be 6.5 (P = 6.5).

$$N = 32.552/6.5 = 5.008\ (N = 5)$$

The programmed value (from Table III) should be 4 (N − 1).

Using the above values calculated for N and P:

$$BR = (5 \times 106)/(16 \times 5 \times 6.5) = 9615.384$$

$$\%\ \text{error} = (9615.385 - 9600)/9600 = 0.16$$

Effect of HALT/IDLE

The UART logic is reinitialized when either the HALT or IDLE modes are entered. This reinitialization sets the TBMT flag and resets all read only bits in the UART control and status registers. Read/Write bits remain unchanged. The Transmit Buffer (TBUF) is not affected, but the Transmit Shift register (TSFT) bits are set to one. The receiver registers RBUF and RSFT are not affected.

The device will exit from the HALT/IDLE modes when the Start bit of a character is detected at the RDX (L3) pin. This feature is obtained by using the Multi-Input Wakeup scheme provided on the device.

Before entering the HALT or IDLE modes the user program must select the Wakeup source to be on the RDX pin. This selection is done by setting bit 3 of WKEN (Wakeup Enable) register. The Wakeup trigger condition is then selected to be high to low transition. This is done via the WKEDG register (Bit 3 is one.)

If the device is halted and crystal oscillator is used, the Wakeup signal will not start the chip running immediately because of the finite start up time requirement of the crystal oscillator. The idle timer (T0) generates a fixed (256 t_c) delay to ensure that the oscillator has indeed stabilized before allowing the device to execute code. The user has to consider this delay when data transfer is expected immediately after exiting the HALT mode.

Diagnostic

Bits CHARL0 and CHARL1 in the ENU register provide a loopback feature for diagnostic testing of the UART. When these bits are set to one, the following occur: The receiver input pin (RDX) is internally connected to the transmitter output pin (TDX); the output of the Transmitter Shift Register is "looped back" into the Receive Shift Register input. In this mode, data that is transmitted is immediately received. This feature allows the processor to verify the transmit and receive data paths of the UART.

Note that the framing format for this mode is the nine bit format; one Start bit, nine data bits, and 7/8, one or two Stop bits. Parity is not generated or verified in this mode.

Attention Mode

The UART Receiver section supports an alternate mode of operation, referred to as ATTENTION Mode. This mode of operation is selected by the ATTN bit in the ENUR register. The data format for transmission must also be selected as having nine Data bits and either 7/8, one or two Stop bits.

The ATTENTION mode of operation is intended for use in networking the device with other processors. Typically in such environments the messages consists of device addresses, indicating which of several destinations should receive them, and the actual data. This Mode supports a scheme in which addresses are flagged by having the ninth bit of the data field set to a 1. If the ninth bit is reset to a zero the byte is a Data byte.

While in ATTENTION mode, the UART monitors the communication flow, but ignores all characters until an address character is received. Upon receiving an address character, the UART signals that the character is ready by setting the RBFL flag, which in turn interrupts the processor if UART Receiver interrupts are enabled. The ATTN bit is also cleared automatically at this point, so that data characters as well as address characters are recognized. Software examines the contents of the RBUF and responds by deciding either to accept the subsequent data stream (by leaving the ATTN bit reset) or to wait until the next address character is seen (by setting the ATTN bit again).

Operation of the UART Transmitter is not affected by selection of this Mode. The value of the ninth bit to be transmitted is programmed by setting XBIT9 appropriately. The value of the ninth bit received is obtained by reading RBIT9. Since this bit is located in ENUR register where the error flags reside, a bit operation on it will reset the error flags.

Comparators

The device contains two differential comparators, each with a pair of inputs (positive and negative) and an output. Ports I1–I3 and I4–I6 are used for the comparators. The following is the Port I assignment:

- I1 Comparator1 negative input
- I2 Comparator1 positive input
- I3 Comparator1 output
- I4 Comparator2 negative input
- I5 Comparator2 positive input
- I6 Comparator2 output

A Comparator Select Register (CMPSL) is used to enable the comparators, read the outputs of the comparators internally, and enable the outputs of the comparators to the pins. Two control bits (enable and output enable) and one result bit are associated with each comparator. The comparator result bits (CMP1RD and CMP2RD) are read only bits which will read as zero if the associated comparator is not enabled. The Comparator Select Register is cleared with reset, resulting in the comparators being disabled. The comparators should also be disabled before entering either the HALT or IDLE modes in order to save power. The configuration of the CMPSL register is as follows:

Comparators (Continued)

CMPSL REGISTER (ADDRESS X'00B7)

The CMPSL register contains the following bits:

CMP1EN Enable comparator 1

CMP1RD Comparator 1 result (this is a read only bit, which will read as 0 if the comparator is not enabled)

CMP10E Selects pin I3 as comparator 1 output provided that CMPIEN is set to enable the comparator

CMP2EN Enable comparator 2

CMP2RD Comparator 2 result (this is a read only bit, which will read as 0 if the comparator is not enabled)

CMP20E Selects pin I6 as comparator 2 output provided that CMP2EN is set to enable the comparator

Unused	CMP20E	CMP2RD	CMP2EN	CMP10E	CMP1RD	CMP1EN	Unused
Bit 7							Bit 0

Note that the two unused bits of CMPSL may be used as software flags.

Comparator outputs have the same spec as Ports L and G except that the rise and fall times are symmetrical.

Interrupts

The device supports a vectored interrupt scheme. It supports a total of fourteen interrupt sources. The following table lists all the possible interrupt sources, their arbitration ranking and the memory locations reserved for the interrupt vector for each source.

Two bytes of program memory space are reserved for each interrupt source. All interrupt sources except the software interrupt are maskable. Each of the maskable interrupts have an Enable bit and one or more Pending bit. A maskable interrupt is active if its associated enable and pending bits are set. If GIE = 1 and an interrupt is active, then the processor will be interrupted as soon as it is ready to start executing an instruction except if the above conditions happen during the Software Trap service routine. This exception is described in the Software Trap sub-section.

The interruption process is accomplished with the INTR instruction (opcode 00), which is jammed inside the Instruction Register and replaces the opcode about to be executed. The following steps are performed for every interrupt:

1. The GIE (Global Interrupt Enable) bit is reset.
2. The address of the instruction about to be executed is pushed into the stack.
3. The PC (Program Counter) branches to address 00FF.

This procedure takes 7 t_C cycles to execute.

Arbitration Ranking	Source	Description	Vector* Address Hi-Low Byte
(1) Highest	Software	INTR Instruction	0yFE–0yFF
(2)	Reserved		0yFC–0yFD
(3)	External	G0	0yFA–0yFB
(4)	Timer T0	Underflow	0yF8–0yF9
(5)	Timer T1	T1A/Underflow	0yF6–0yF7
(6)	Timer T1	T1B	0yF4–0yF5
(7)	MICROWIRE/PLUS	BUSY Low	0yF2–0yF3
(8)	Reserved		0yF0–0yF1
(9)	UART	Receive	0yEE–0yEF
(10)	UART	Transmit	0yEC–0yED
(11)	Timer T2	T2A/Underflow	0yEA–0yEB
(12)	Timer T2	T2B	0yE8–0yE9
(13)	Timer T3	T3A/Underflow	0yE6–0yE7
(14)	Timer T3	T3B	0yE4–0yE5
(15)	Port L/Wakeup	Port L Edge	0yE2–0yE3
(16) Lowest	Default VIS	Reserved	0yE0–0yE1

*y is a variable which represents the VIS block. VIS and the vector table must be located in the same 256-byte block except if VIS is located at the last address of a block. In this case, the table must be in the next block.

Interrupts (Continued)

At this time, since GIE = 0, other maskable interrupts are disabled. The user is now free to do whatever context switching is required by saving the context of the machine in the stack with PUSH instructions. The user would then program a VIS (Vector Interrupt Select) instruction in order to branch to the interrupt service routine of the highest priority interrupt enabled and pending at the time of the VIS. Note that this is not necessarily the interrupt that caused the branch to address location 00FF Hex prior to the context switching.

Thus, if an interrupt with a higher rank than the one which caused the interruption becomes active before the decision of which interrupt to service is made by the VIS, then the interrupt with the higher rank will override any lower ones and will be acknowledged. The lower priority interrupt(s) are still pending, however, and will cause another interrupt immediately following the completion of the interrupt service routine associated with the higher priority interrupt just serviced. This lower priority interrupt will occur immediately following the RETI (Return from Interrupt) instruction at the end of the interrupt service routine just completed.

Inside the interrupt service routine, the associated pending bit has to be cleared by software. The RETI (Return from Interrupt) instruction at the end of the interrupt service routine will set the GIE (Global Interrupt Enable) bit, allowing the processor to be interrupted again if another interrupt is active and pending.

The VIS instruction looks at all the active interrupts at the time it is executed and performs an indirect jump to the beginning of the service routine of the one with the highest rank.

The addresses of the different interrupt service routines, called vectors, are chosen by the user and stored in ROM in a table starting at 01E0 (assuming that VIS is located between 00FF and 01DF). The vectors are 15-bit wide and therefore occupy 2 ROM locations.

VIS and the vector table must be located in the same 256-byte block (0y00 to 0yFF) except if VIS is located at the last address of a block. In this case, the table must be in the next block. The vector table cannot be inserted in the first 256-byte block (y ≠ 0).

The vector of the maskable interrupt with the lowest rank is located at 0yE0 (Hi-Order byte) and 0yE1 (Lo-Order byte) and so forth in increasing rank number. The vector of the maskable interrupt with the highest rank is located at 0yFA (Hi-Order byte) and 0yFB (Lo-Order byte).

The Software Trap has the highest rank and its vector is located at 0yFE and 0yFF.

If, by accident, a VIS gets executed and no interrupt is active, then the PC (Program Counter) will branch to a vector located at 0yE0–0yE1.

WARNING

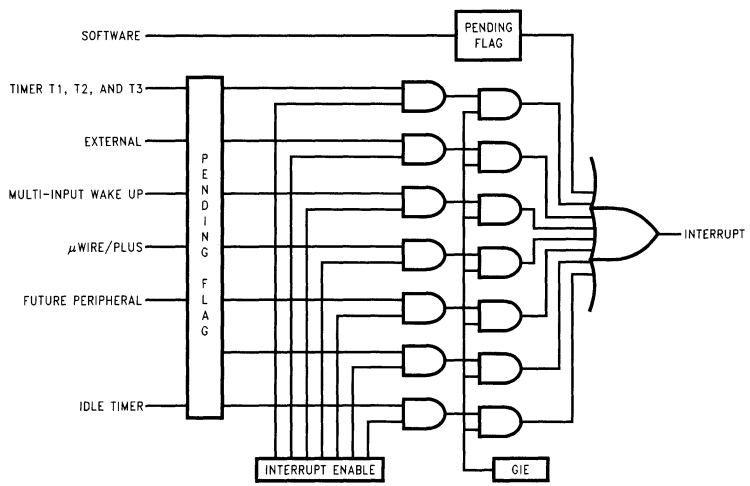
A Default VIS interrupt handler routine must be present. As a minimum, this handler should confirm that the GIE bit is cleared (this indicates that the interrupt sequence has been taken), take care of any required housekeeping, restore context and return. Some sort of Warm Restart procedure should be implemented. These events can occur without any error on the part of the system designer or programmer.

Note: There is always the possibility of an interrupt occurring during an instruction which is attempting to reset the GIE bit or any other interrupt enable bit. If this occurs when a single cycle instruction is being used to reset the interrupt enable bit, the interrupt enable bit will be reset but an interrupt may still occur. This is because interrupt processing is started at the same time as the interrupt bit is being reset. To avoid this scenario, the user should always use a two, three, or four cycle instruction to reset interrupt enable bits.

Figure 16 shows the Interrupt block diagram.

SOFTWARE TRAP

The Software Trap (ST) is a special kind of non-maskable interrupt which occurs when the INTR instruction (used to acknowledge interrupts) is fetched from ROM and placed inside the instruction register. This may happen when the PC is pointing beyond the available ROM address space or when the stack is over-popped.



TL/DD12829–18

FIGURE 16. Interrupt Block Diagram

Interrupts (Continued)

When an ST occurs, the user can re-initialize the stack pointer and do a recovery procedure (similar to reset, but not necessarily containing all of the same initialization procedures) before restarting.

The occurrence of an ST is latched into the ST pending bit. The GIE bit is not affected and the ST pending bit **(not accessible by the user)** is used to inhibit other interrupts and to direct the program to the ST service routine with the VIS instruction. The RPND instruction is used to clear the software interrupt pending bit. This pending bit is also cleared on reset.

The ST has the highest rank among all interrupts.

Nothing (except another ST) can interrupt an ST being serviced.

WATCHDOG

The device contains a WATCHDOG and clock monitor. The WATCHDOG is designed to detect the user program getting stuck in infinite loops resulting in loss of program control or "runaway" programs. The Clock Monitor is used to detect the absence of a clock or a very slow clock below a specified rate on the CKI pin.

The WATCHDOG consists of two independent logic blocks: WD UPPER and WD LOWER. WD UPPER establishes the upper limit on the service window and WD LOWER defines the lower limit of the service window.

Servicing the WATCHDOG consists of writing a specific value to a WATCHDOG Service Register named WDSVR which is memory mapped in the RAM. This value is composed of three fields, consisting of a 2-bit Window Select, a 5-bit Key Data field, and the 1-bit Clock Monitor Select field. Table II shows the WDSVR register.

TABLE V. WATCHDOG Service Register (WDSVR)

Window Select		Key Data					Clock Monitor
X	X	0	1	1	0	0	Y
7	6	5	4	3	2	1	0

The lower limit of the service window is fixed at 2048 instruction cycles. Bits 7 and 6 of the WDSVR register allow the user to pick an upper limit of the service window.

Table III shows the four possible combinations of lower and upper limits for the WATCHDOG service window. This flexibility in choosing the WATCHDOG service window prevents any undue burden on the user software.

Bits 5, 4, 3, 2 and 1 of the WDSVR register represent the 5-bit Key Data field. The key data is fixed at 01100. Bit 0 of the WDSVR Register is the Clock Monitor Select bit.

TABLE VI. WATCHDOG Service Window Select

WDSVR Bit 7	WDSVR Bit 6	Service Window (Lower-Upper Limits)
0	0	2k–8k t_C Cycles
0	1	2k–16k t_C Cycles
1	0	2k–32k t_C Cycles
1	1	2k–64k t_C Cycles

Clock Monitor

The Clock Monitor aboard the device can be selected or deselected under program control. The Clock Monitor is guaranteed not to reject the clock if the instruction cycle clock ($1/t_C$) is greater or equal to 10 kHz. This equates to a clock input rate on CKI of greater or equal to 100 kHz.

WATCHDOG Operation

The WATCHDOG and Clock Monitor are disabled during reset. The device comes out of reset with the WATCHDOG armed, the WATCHDOG Window Select bits (bits 6, 7 of the WDSVR Register) set, and the Clock Monitor bit (bit 0 of the WDSVR Register) enabled. Thus, a Clock Monitor error will occur after coming out of reset, if the instruction cycle clock frequency has not reached a minimum specified value, including the case where the oscillator fails to start.

The WDSVR register can be written to only once after reset and the key data (bits 5 through 1 of the WDSVR Register) must match to be a valid write. This write to the WDSVR register involves two irrevocable choices: (i) the selection of the WATCHDOG service window (ii) enabling or disabling of the Clock Monitor. Hence, the first write to WDSVR Register involves selecting or deselecting the Clock Monitor, select the WATCHDOG service window and match the WATCHDOG key data. Subsequent writes to the WDSVR register will compare the value being written by the user to the WATCHDOG service window value and the key data (bits 7 through 1) in the WDSVR Register. Table IV shows the sequence of events that can occur.

The user must service the WATCHDOG at least once before the upper limit of the service window expires. The WATCHDOG may not be serviced more than once in every lower limit of the service window. The user may service the WATCHDOG as many times as wished in the time period between the lower and upper limits of the service window. The first write to the WDSVR Register is also counted as a WATCHDOG service.

The WATCHDOG has an output pin associated with it. This is the WDOUT pin, on pin 1 of the port G. WDOUT is active low. The WDOUT pin is in the high impedance state in the inactive state. Upon triggering the WATCHDOG, the logic will pull the WDOUT (G1) pin low for an additional 16 t_C–32 t_C cycles after the signal level on WDOUT pin goes below the lower Schmitt trigger threshold. After this delay, the device will stop forcing the WDOUT output low.

The WATCHDOG service window will restart when the WDOUT pin goes high. It is recommended that the user tie the WDOUT pin back to V_{CC} through a resistor in order to pull WDOUT high.

A WATCHDOG service while the WDOUT signal is active will be ignored. The state of the WDOUT pin is not guaranteed on reset, but if it powers up low then the WATCHDOG will time out and WDOUT will enter high impedance state.

The Clock Monitor forces the G1 pin low upon detecting a clock frequency error. The Clock Monitor error will continue until the clock frequency has reached the minimum specified value, after which the G1 output will enter the high impedance TRI-STATE mode following 16 t_C–32 t_C clock cycles. The Clock Monitor generates a continual Clock Monitor error if the oscillator fails to start, or fails to reach the minimum specified frequency. The specification for the Clock Monitor is as follows:

$1/t_C > 10$ kHz—No clock rejection.

$1/t_C < 10$ Hz—Guaranteed clock rejection.

WATCHDOG Operation (Continued)

WATCHDOG AND CLOCK MONITOR SUMMARY

The following salient points regarding the WATCHDOG and CLOCK MONITOR should be noted:

- Both the WATCHDOG and CLOCK MONITOR detector circuits are inhibited during RESET.
- Following RESET, the WATCHDOG and CLOCK MONITOR are both enabled, with the WATCHDOG having he maximum service window selected.
- The WATCHDOG service window and CLOCK MONITOR enable/disable option can only be changed once, during the initial WATCHDOG service following RESET.
- The initial WATCHDOG service must match the key data value in the WATCHDOG Service register WDSVR in order to avoid a WATCHDOG error.
- Subsequent WATCHDOG services must match all three data fields in WDSVR in order to avoid WATCHDOG errors.
- The correct key data value cannot be read from the WATCHDOG Service register WDSVR. Any attempt to read this key data value of 01100 from WDSVR will read as key data value of all 0's.
- The WATCHDOG detector circuit is inhibited during both the HALT and IDLE modes.
- The CLOCK MONITOR detector circuit is active during both the HALT and IDLE modes. Consequently, the device inadvertently entering the HALT mode will be detected as a CLOCK MONITOR error (provided that the CLOCK MONITOR enable option has been selected by the program).
- With the single-pin R/C oscillator mask option selected and the CLKDLY bit reset, the WATCHDOG service window will resume following HALT mode from where it left off before entering the HALT mode.
- With the crystal oscillator mask option selected, or with the single-pin R/C oscillator mask option selected and the CLKDLY bit set, the WATCHDOG service window will be set to its selected value from WDSVR following HALT. Consequently, the WATCHDOG should not be serviced for at least 2048 instruction cycles following HALT, but must be serviced within the selected window to avoid a WATCHDOG error.
- The IDLE timer T0 is not initialized with RESET.
- The user can sync in to the IDLE counter cycle with an IDLE counter (T0) interrupt or by monitoring the T0PND flag. The T0PND flag is set whenever the thirteenth bit of the IDLE counter toggles (every 4096 instruction cycles). The user is responsible for resetting the T0PND flag.
- A hardware WATCHDOG service occurs just as the device exits the IDLE mode. Consequently, the WATCHDOG should not be serviced for at least 2048 instruction cycles following IDLE, but must be serviced within the selected window to avoid a WATCHDOG error.
- Following RESET, the initial WATCHDOG service (where the service window and the CLOCK MONITOR enable/disable must be selected) may be programmed anywhere within the maximum service window (65,536 instruction cycles) initialized by RESET. Note that this initial WATCHDOG service may be programmed within the initial 2048 instruction cycles without causing a WATCHDOG error.

Detection of Illegal Conditions

The device can detect various illegal conditions resulting from coding errors, transient noise, power supply voltage drops, runaway programs, etc.

Reading of undefined ROM gets zeros. The opcode for software interrupt is zero. If the program fetches instructions from undefined ROM, this will force a software interrupt, thus signaling that an illegal condition has occurred.

The subroutine stack grows down for each call (jump to subroutine), interrupt, or PUSH, and grows up for each return or POP. The stack pointer is initialized to RAM location 06F Hex during reset. Consequently, if there are more returns than calls, the stack pointer will point to addresses 070 and 071 Hex (which are undefined RAM). Undefined RAM from addresses 070 to 07F (Segment 0), and all other segments (i.e., Segments 4 . . . etc.) is read as all 1's, which in turn will cause the program to return to address 7FFF Hex. This is an undefined ROM location and the instruction fetched (all 0's) from this location will generate a software interrupt signaling an illegal condition.

Thus, the chip can detect the following illegal conditions:

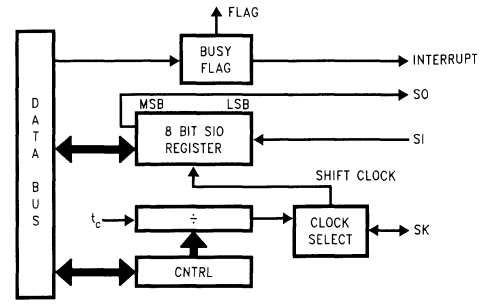
a. Executing from undefined ROM

b. Over "POP"ing the stack by having more returns than calls.

When the software interrupt occurs, the user can re-initialize the stack pointer and do a recovery procedure before restarting (this recovery program is probably similar to that following reset, but might not contain the same program initialization procedures). The recovery program should reset the software interrupt pending bit using the RPND instruction.

MICROWIRE/PLUS

MICROWIRE/PLUS is a serial synchronous communications interface. The MICROWIRE/PLUS capability enables the device to interface with any of National Semiconductor's MICROWIRE peripherals (i.e. A/D converters, display drivers, E^2PROMs etc.) and with other microcontrollers which support the MICROWIRE interface. It consists of an 8-bit serial shift register (SIO) with serial data input (SI), serial data output (SO) and serial shift clock (SK). *Figure 17* shows a block diagram of the MICROWIRE/PLUS logic.



TL/DD12829–19

FIGURE 17. MICROWIRE/PLUS Block Diagram

The shift clock can be selected from either an internal source or an external source. Operating the MICROWIRE/PLUS arrangement with the internal clock source is called the Master mode of operation. Similarly, operating the MICROWIRE/PLUS arrangement with an external shift clock is called the Slave mode of operation.

The CNTRL register is used to configure and control the MICROWIRE/PLUS mode. To use the MICROWIRE/PLUS, the MSEL bit in the CNTRL register is set to one. In the master mode, the SK clock rate is selected by the two bits, SL0 and SL1, in the CNTRL register. Table VIII details the different clock rates that may be selected.

TABLE VII. WATCHDOG Service Actions

Key Data	Window Data	Clock Monitor	Action
Match	Match	Match	Valid Service: Restart Service Window
Don't Care	Mismatch	Don't Care	Error: Generate WATCHDOG Output
Mismatch	Don't Care	Don't Care	Error: Generate WATCHDOG Output
Don't Care	Don't Care	Mismatch	Error: Generate WATCHDOG Output

TABLE VIII. MICROWIRE/PLUS Master Mode Clock Select

SL1	SL0	SK
0	0	$2 \times t_c$
0	1	$4 \times t_c$
1	x	$8 \times t_c$

Where t_c is the instruction cycle clock

MICROWIRE/PLUS (Continued)

MICROWIRE/PLUS OPERATION

Setting the BUSY bit in the PSW register causes the MICROWIRE/PLUS to start shifting the data. It gets reset when eight data bits have been shifted. The user may reset the BUSY bit by software to allow less than 8 bits to shift. If enabled, an interrupt is generated when eight data bits have been shifted. The device may enter the MICROWIRE/PLUS mode either as a Master or as a Slave. *Figure 18* shows how two microcontroller devices and several peripherals may be interconnected using the MICROWIRE/PLUS arrangements.

Warning:

The SIO register should only be loaded when the SK clock is low. Loading the SIO register while the SK clock is high will result in undefined data in the SIO register. SK clock is normally low when not shifting.

Setting the BUSY flag when the input SK clock is high in the MICROWIRE/PLUS slave mode may cause the current SK clock for the SIO shift register to be narrow. For safety, the BUSY flag should only be set when the input SK clock is low.

MICROWIRE/PLUS Master Mode Operation

In the MICROWIRE/PLUS Master mode of operation the shift clock (SK) is generated internally by the device. The MICROWIRE Master always initiates all data exchanges. The MSEL bit in the CNTRL register must be set to enable the SO and SK functions onto the G Port. The SO and SK pins must also be selected as outputs by setting appropriate bits in the Port G configuration register. Table IX summarizes the bit settings required for Master mode of operation.

MICROWIRE/PLUS Slave Mode Operation

In the MICROWIRE/PLUS Slave mode of operation the SK clock is generated by an external source. Setting the MSEL bit in the CNTRL register enables the SO and SK functions onto the G Port. The SK pin must be selected as an input and the SO pin is selected as an output pin by setting and resetting the appropriate bits in the Port G configuration register. Table VI summarizes the settings required to enter the Slave mode of operation.

The user must set the BUSY flag immediately upon entering the Slave mode. This will ensure that all data bits sent by the Master will be shifted properly. After eight clock pulses the BUSY flag will be cleared and the sequence may be repeated.

Alternate SK Phase Operation

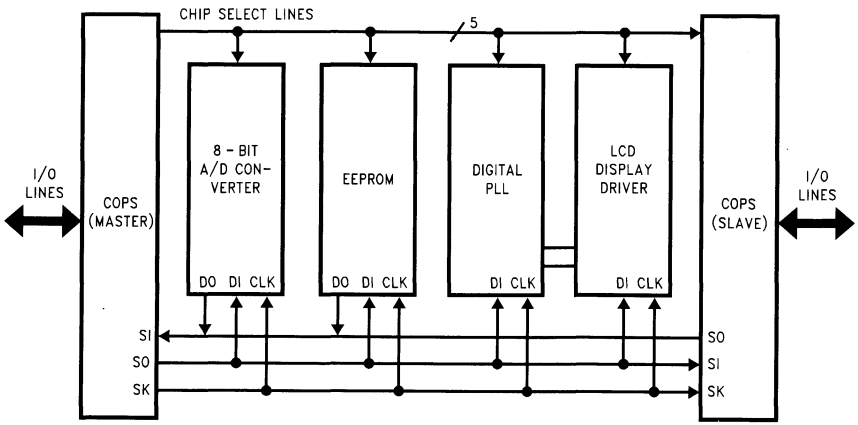
The device allows either the normal SK clock or an alternate phase SK clock to shift data in and out of the SIO register. In both the modes the SK is normally low. In the normal mode data is shifted in on the rising edge of the SK clock and the data is shifted out on the falling edge of the SK clock. The SIO register is shifted on each falling edge of the SK clock. In the alternate SK phase operation, data is shifted in on the falling edge of the SK clock and shifted out on the rising edge of the SK clock.

A control flag, SKSEL, allows either the normal SK clock or the alternate SK clock to be selected. Resetting SKSEL causes the MICROWIRE/PLUS logic to be clocked from the normal SK signal. Setting the SKSEL flag selects the alternate SK clock. The SKSEL is mapped into the G6 configuration bit. The SKSEL flag will power up in the reset condition, selecting the normal SK signal.

TABLE IX

This table assumes that the control flag MSEL is set.

G4 (SO) Config. Bit	G5 (SK) Config. Bit	G4 Fun.	G5 Fun.	Operation
1	1	SO	Int. SK	MICROWIRE/PLUS Master
0	1	TRI-STATE	Int. SK	MICROWIRE/PLUS Master
1	0	SO	Ext. SK	MICROWIRE/PLUS Slave
0	0	TRI-STATE	Ext. SK	MICROWIRE/PLUS Slave



TL/DD12829-20

FIGURE 18. MICROWIRE/PLUS Application

Memory Map

All RAM, ports and registers (except A and PC) are mapped into data memory address space.

Address S/ADD REG	Contents
0000 to 006F	On-Chip RAM bytes (112 bytes)
0070 to 007F	Unused RAM Address Space (Reads As All Ones)
xx80 to xxAF	Unused RAM Address Space (Reads Undefined Data)
xxB0	Timer T3 Lower Byte
XXB1	Timer T3 Upper Byte
xxB2	Timer T3 Autoload Register T3RA Lower Byte
xxB3	Timer T3 Autoload Register T3RA Upper Byte
xxB4	Timer T3 Autoload Register T3RB Lower Byte
xxB5	Timer T3 Autoload Register T3RB Upper Byte
xxB6	Timer T3 Control Register
xxB7	Comparator Select Register (CMPSL)
xxB8	UART Transmit Buffer (TBUF)
xxB9	UART Receive Buffer (RBUF)
xxBA	UART Control and Status Register (ENU)
xxBB	UART Receive Control and Status Register (ENUR)
xxBC	UART Interrupt and Clock Source Register (ENUI)
xxBD	UART Baud Register (BAUD)
xxBE	UART Prescale Select Register (PSR)
xxBF	Reserved for UART
xxC0	Timer T2 Lower Byte
xxC1	Timer T2 Upper Byte
xxC2	Timer T2 Autoload Register T2RA Lower Byte
xxC3	Timer T2 Autoload Register T2RA Upper Byte
xxC4	Timer T2 Autoload Register T2RB Lower Byte
xxC5	Timer T2 Autoload Register T2RB Upper Byte
xxC6	Timer T2 Control Register
xxC7	WATCHDOG Service Register (Reg:WDSVR)
xxC8	MIWU Edge Select Register (Reg:WKEDG)
xxC9	MIWU Enable Register (Reg:WKEN)
xxCA	MIWU Pending Register (Reg:WKPND)
xxCB	Reserved
xxCC	Reserved
xxCD to xxCF	Reserved

Address S/ADD REG	Contents
xxD0	Port L Data Register
xxD1	Port L Configuration Register
xxD2	Port L Input Pins (Read Only)
xxD3	Reserved for Port L
xxD4	Port G Data Register
xxD5	Port G Configuration Register
xxD6	Port G Input Pins (Read Only)
xxD7	Port I Input Pins (Read Only)
xxD8	Port C Data Register
xxD9	Port C Configuration Register
xxDA	Port C Input Pins (Read Only)
xxDB	Reserved for Port C
xxDC	Port D
xxDD to xxDF	Reserved for Port D
xxE0 to xxE5	Reserved for EE Control Registers
xxE6	Timer T1 Autoload Register T1RB Lower Byte
xxE7	Timer T1 Autoload Register T1RB Upper Byte
xxE8	ICNTRL Register
xxE9	MICROWIRE/PLUS Shift Register
xxEA	Timer T1 Lower Byte
xxEB	Timer T1 Upper Byte
xxEC	Timer T1 Autoload Register T1RA Lower Byte
xxED	Timer T1 Autoload Register T1RA Upper Byte
xxEE	CNTRL Control Register
xxEF	PSW Register
xxF0 to FB	On-Chip RAM Mapped as Registers
xxFC	X Register
xxFD	SP Register
xxFE	B Register
xxFF	S Register
0100–017F	On-Chip 128 RAM Bytes
0200–027F	On-Chip 128 RAM Bytes
0300–037F	On-Chip 128 RAM Bytes
0400–047F	On-Chip 128 RAM Bytes
0500–057F	On-Chip 128 RAM Bytes
0600–067F	On-Chip 128 RAM Bytes
0700–077F	On-Chip 128 RAM Bytes
0800–083F	On-Chip 64 RAM Bytes

Reading memory locations 0070H–007FH (Segment 0) will return all ones. Reading unused memory locations 0080H–00AFH (Segment 0) will return undefined data. Reading memory locations from other Segments (i.e., Segment 8 (084H–087FH), Segment 9, ... etc.) will return all ones.

Addressing Modes

There are ten addressing modes, six for operand addressing and four for transfer of control.

OPERAND ADDRESSING MODES

Register Indirect

This is the "normal" addressing mode. The operand is the data memory addressed by the B pointer or X pointer.

Register Indirect (with auto post increment or decrement of pointer)

This addressing mode is used with the LD and X instructions. The operand is the data memory addressed by the B pointer or X pointer. This is a register indirect mode that automatically post increments or decrements the B or X register after executing the instruction.

Direct

The instruction contains an 8-bit address field that directly points to the data memory for the operand.

Immediate

The instruction contains an 8-bit immediate field as the operand.

Short Immediate

This addressing mode is used with the Load B Immediate instruction. The instruction contains a 4-bit immediate field as the operand.

Indirect

This addressing mode is used with the LAID instruction. The contents of the accumulator are used as a partial address (lower 8 bits of PC) for accessing a data operand from the program memory.

TRANSFER OF CONTROL ADDRESSING MODES

Relative

This mode is used for the JP instruction, with the instruction field being added to the program counter to get the new program location. JP has a range from −31 to +32 to allow a 1-byte relative jump (JP + 1 is implemented by a NOP instruction). There are no "pages" when using JP, since all 15 bits of PC are used.

Absolute

This mode is used with the JMP and JSR instructions, with the instruction field of 12 bits replacing the lower 12 bits of the program counter (PC). This allows jumping to any location in the current 4k program memory segment.

Absolute Long

This mode is used with the JMPL and JSRL instructions, with the instruction field of 15 bits replacing the entire 15 bits of the program counter (PC). This allows jumping to any location up to 32k in the program memory space.

Indirect

This mode is used with the JID instruction. The contents of the accumulator are used as a partial address (lower 8 bits of PC) for accessing a location in the program memory. The contents of this program memory location serve as a partial address (lower 8 bits of PC) for the jump to the next instruction.

Note: The VIS is a special case of the Indirect Transfer of Control addressing mode, where the double byte vector associated with the interrupt is transferred from adjacent addresses in the program memory into the program counter (PC) in order to jump to the associated interrupt service routine.

Instruction Set

Register and Symbol Definition

Registers	
A	8-Bit Accumulator Register
B	8-Bit Address Register
X	8-Bit Address Register
S	8-Bit Segment Register
SP	8-Bit Stack Pointer Register
PC	15-Bit Program Counter Register
PU	Upper 7 Bits of PC
PL	Lower 8 Bits of PC
C	1 Bit of PSW Register for Carry
HC	1 Bit of PSW Register for Half Carry
GIE	1 Bit of PSW Register for Global Interrupt Enable
VU	Interrupt Vector Upper Byte
VL	Interrupt Vector Lower Byte

Symbols	
[B]	Memory Indirectly Addressed by B Register
[X]	Memory Indirectly Addressed by X Register
MD	Direct Addressed Memory
Mem	Direct Addressed Memory or [B]
Meml	Direct Addressed Memory or [B] or Immediate Data
Imm	8-Bit Immediate Data
Reg	Register Memory: Addresses F0 to FF (Includes B, X and SP)
Bit	Bit Number (0 to 7)
←	Loaded with
↔	Exchanged with

5

Instruction Set (Continued)

INSTRUCTION SET

ADD	A,Meml	ADD	A ← A + Meml
ADC	A,Meml	ADD with Carry	A ← A + Meml + C, C ← Carry HC ← Half Carry
SUBC	A,Meml	Subtract with Carry	A ← A − $\overline{\text{Meml}}$ + C, C ← Carry HC ← Half Carry
AND	A,Meml	Logical AND	A ← A and Meml
ANDSZ	A,Imm	Logical AND Immed., Skip if Zero	Skip next if (A and Imm) = 0
OR	A,Meml	Logical OR	A ← A or Meml
XOR	A,Meml	Logical EXclusive OR	A ← A xor Meml
IFEQ	MD,Imm	IF EQual	Compare MD and Imm, Do next if MD = Imm
IFEQ	A,Meml	IF EQual	Compare A and Meml, Do next if A = Meml
IFNE	A,Meml	IF Not Equal	Compare A and Meml, Do next if A ≠ Meml
IFGT	A,Meml	IF Greater Than	Compare A and Meml, Do next if A > Meml
IFBNE	#	If B Not Equal	Do next if lower 4 bits of B ≠ Imm
DRSZ	Reg	Decrement Reg., Skip if Zero	Reg ← Reg − 1, Skip if Reg = 0
SBIT	#,Mem	Set BIT	1 to bit, Mem (bit = 0 to 7 immediate)
RBIT	#,Mem	Reset BIT	0 to bit, Mem
IFBIT	#,Mem	IF BIT	If bit in A or Mem is true do next instruction
RPND		Reset PeNDing Flag	Reset Software Interrupt Pending Flag
X	A,Mem	EXchange A with Memory	A ⟷ Mem
X	A,[X]	EXchange A with Memory [X]	A ⟷ [X]
LD	A,Meml	LoaD A with Memory	A ← Meml
LD	A,[X]	LoaD A with Memory [X]	A ← [X]
LD	B,Imm	LoaD B with Immed.	B ← Imm
LD	Mem,Imm	LoaD Memory Immed	Mem ← Imm
LD	Reg,Imm	LoaD Register Memory Immed.	Reg ← Imm
X	A, [B ±]	EXchange A with Memory [B]	A ⟷ [B], (B ← B ±1)
X	A, [X ±]	EXchange A with Memory [X]	A ⟷ [X], (X ← X ±1)
LD	A, [B±]	LoaD A with Memory [B]	A ← [B], (B ← B ±1)
LD	A, [X±]	LoaD A with Memory [X]	A ← [X], (X ← X±1)
LD	[B±],Imm	LoaD Memory [B] Immed.	[B] ← Imm, (B ← B±1)
CLR	A	CLeaR A	A ← 0
INC	A	INCrement A	A ← A + 1
DEC	A	DECrement A	A ← A − 1
LAID		Load A InDirect from ROM	A ← ROM (PU,A)
DCOR	A	Decimal CORrect A	A ← BCD correction of A (follows ADC, SUBC)
RRC	A	Rotate A Right thru C	C → A7 → . . . → A0 → C
RLC	A	Rotate A Left thru C	C ← A7 ← . . . ← A0 ← C
SWAP	A	SWAP nibbles of A	A7 . . . A4 ⟷ A3 . . . A0
SC		Set C	C ← 1, HC ← 1
RC		Reset C	C ← 0, HC ← 0
IFC		IF C	IF C is true, do next instruction
IFNC		IF Not C	If C is not true, do next instruction
POP	A	POP the stack into A	SP ← SP + 1, A ← [SP]
PUSH	A	PUSH A onto the stack	[SP] ← A, SP ← SP − 1
VIS		Vector to Interrupt Service Routine	PU ← [VU], PL ← [VL]
JMPL	Addr.	Jump absolute Long	PC ← ii (ii = 15 bits, 0 to 32k)
JMP	Addr.	Jump absolute	PC9 . . . 0 ← i (i = 12 bits)
JP	Disp.	Jump relative short	PC ← PC + r (r is −31 to +32, except 1)
JSRL	Addr.	Jump SubRoutine Long	[SP] ← PL, [SP−1] ← PU,SP−2, PC ← ii
JSR	Addr	Jump SubRoutine	[SP] ← PL, [SP−1] ← PU,SP−2, PC9 . . . 0 ← i
JID		Jump InDirect	PL ← ROM (PU,A)
RET		RETurn from subroutine	SP + 2, PL ← [SP], PU ← [SP−1]
RETSK		RETurn and SKip	SP + 2, PL ← [SP],PU ← [SP−1], skip next instruction
RETI		RETurn from Interrupt	SP + 2, PL ← [SP],PU ← [SP−1],GIE ← 1
INTR		Generate an Interrupt	[SP] ← PL, [SP−1] ← PU, SP−2, PC ← 0FF
NOP		No OPeration	PC ← PC + 1

Instruction Execution Time

Most instructions are single byte (with immediate addressing mode instructions taking two bytes).

Most single byte instructions take one cycle time to execute.

Skipped instructions require x number of cycles to be skipped, where x equals the number of bytes in the skipped instruction opcode.

See the BYTES and CYCLES per INSTRUCTION table for details.

Bytes and Cycles per Instruction

The following table shows the number of bytes and cycles for each instruction in the format of byte/cycle.

Arithmetic and Logic Instructions

	[B]	Direct	Immed.
ADD	1/1	3/4	2/2
ADC	1/1	3/4	2/2
SUBC	1/1	3/4	2/2
AND	1/1	3/4	2/2
OR	1/1	3/4	2/2
XOR	1/1	3/4	2/2
IFEQ	1/1	3/4	2/2
IFNE	1/1	3/4	2/2
IFGT	1/1	3/4	2/2
IFBNE	1/1		
DRSZ		1/3	
SBIT	1/1	3/4	
RBIT	1/1	3/4	
IFBIT	1/1	3/4	

RPND	1/1

Instructions Using A & C

CLRA	1/1
INCA	1/1
DECA	1/1
LAID	1/3
DCOR	1/1
RRCA	1/1
RLCA	1/1
SWAPA	1/1
SC	1/1
RC	1/1
IFC	1/1
IFNC	1/1
PUSHA	1/3
POPA	1/3
ANDSZ	2/2

Transfer of Control Instructions

JMPL	3/4
JMP	2/3
JP	1/3
JSRL	3/5
JSR	2/5
JID	1/3
VIS	1/5
RET	1/5
RETSK	1/5
RETI	1/5
INTR	1/7
NOP	1/1

Memory Transfer Instructions

	Register Indirect		Direct	Immed.	Register Indirect Auto Incr. & Decr.		
	[B]	[X]			[B+, B−]	[X+, X−]	
X A,*	1/1	1/3	2/3		1/2	1/3	
LD A,*	1/1	1/3	2/3	2/2	1/2	1/3	
LD B, Imm				1/1			(IF B < 16)
LD B, Imm				2/2			(IF B > 15)
LD Mem, Imm	2/2		3/3		2/2		
LD Reg, Imm			2/3				
IFEQ MD, Imm			3/3				

* = > Memory location addressed by B or X or directly.

5

Opcode Table

	Upper Nibble															Lower Nibble
F	E	D	C	B	A	9	8	7	6	5	4	3	2	1	0	
JP−15	JP−31	LD 0F0, # i	DRSZ 0F0	RRCA	RC	ADC A, #i	ADC A,[B]	IFBIT 0,[B]	ANDSZ A, #i	LD B, #0F	IFBNE 0	JSR x000−x0FF	JMP x000−x0FF	JP+17	INTR	0
JP−14	JP−30	LD 0F1, # i	DRSZ 0F1	*	SC	SUBC A, #i	SUB A,[B]	IFBIT 1,[B]	*	LD B, #0E	IFBNE 1	JSR x100−x1FF	JMP x100−x1FF	JP+18	JP+2	1
JP−13	JP−29	LD 0F2, # i	DRSZ 0F2	X A,[X+]	X A,[B+]	IFEQ A, #i	IFEQ A,[B]	IFBIT 2,[B]	*	LD B, #0D	IFBNE 2	JSR x200−x2FF	JMP x200−x2FF	JP+19	JP+3	2
JP−12	JP−28	LD 0F3, # i	DRSZ 0F3	X A,[X−]	X A,[B−]	IFGT A, #i	IFGT A,[B]	IFBIT 3,[B]	*	LD B, #0C	IFBNE 3	JSR x300−x3FF	JMP x300−x3FF	JP+20	JP+4	3
JP−11	JP−27	LD 0F4, # i	DRSZ 0F4	VIS	LAID	ADD A, #i	ADD A,[B]	IFBIT 4,[B]	CLRA	LD B, #0B	IFBNE 4	JSR x400−x4FF	JMP x400−x4FF	JP+21	JP+5	4
JP−10	JP−26	LD 0F5, # i	DRSZ 0F5	RPND	JID	AND A, #i	AND A,[B]	IFBIT 5,[B]	SWAPA	LD B, #0A	IFBNE 5	JSR x500−x5FF	JMP x500−x5FF	JP+22	JP+6	5
JP−9	JP−25	LD 0F6, # i	DRSZ 0F6	X A,[X]	X A,[B]	XOR A, #i	XOR A,[B]	IFBIT 6,[B]	DCORA	LD B, #09	IFBNE 6	JSR x600−x6FF	JMP x600−x6FF	JP+23	JP+7	6
JP−8	JP−24	LD 0F7, # i	DRSZ 0F7	*	*	OR A, #i	OR A,[B]	IFBIT 7,[B]	PUSHA	LD B, #08	IFBNE 7	JSR x700−x7FF	JMP x700−x7FF	JP+24	JP+8	7
JP−7	JP−23	LD 0F8, # i	DRSZ 0F8	NOP	RLCA	LD A, #i	IFC	SBIT 0,[B]	RBIT 0,[B]	LD B, #07	IFBNE 8	JSR x800−x8FF	JMP x800−x8FF	JP+25	JP+9	8
JP−6	JP−22	LD 0F9, # i	DRSZ 0F9	IFNE A,[B]	IFEQ Md, #i	IFNE A, #i	IFNC	SBIT 1,[B]	RBIT 1,[B]	LD B, #06	IFBNE 9	JSR x900−x9FF	JMP x900−x9FF	JP+26	JP+10	9
JP−5	JP−21	LD 0FA, # i	DRSZ 0FA	LD A,[X+]	LD A,[B+]	LD [B+], #i	INCA	SBIT 2,[B]	RBIT 2,[B]	LD B, #05	IFBNE 0A	JSR xA00−xAFF	JMP xA00−xAFF	JP+27	JP+11	A
JP−4	JP−20	LD 0FB, # i	DRSZ 0FB	LD A,[X−]	LD A,[B−]	LD [B−], #i	DECA	SBIT 3,[B]	RBIT 3,[B]	LD B, #04	IFBNE 0B	JSR xB00−xBFF	JMP xB00−xBFF	JP+28	JP+12	B
JP−3	JP−19	LD 0FC, # i	DRSZ 0FC	LD Md, #i	JMPL	X A,Md	POPA	SBIT 4,[B]	RBIT 4,[B]	LD B, #03	IFBNE 0C	JSR xC00−xCFF	JMP xC00−xCFF	JP+29	JP+13	C
JP−2	JP−18	LD 0FD, # i	DRSZ 0FD	DIR	JSRL	LD A,Md	RETSK	SBIT 5,[B]	RBIT 5,[B]	LD B, #02	IFBNE 0D	JSR xD00−xDFF	JMP xD00−xDFF	JP+30	JP+14	D
JP−1	JP−17	LD 0FE, # i	DRSZ 0FE	LD A,[X]	LD A,[B]	LD [B], #i	RET	SBIT 6,[B]	RBIT 6,[B]	LD B, #01	IFBNE 0E	JSR xE00−xEFF	JMP xE00−xEFF	JP+31	JP+15	E
JP−0	JP−16	LD 0FF, # i	DRSZ 0FF	*	*	LD B, #i	RETI	SBIT 7,[B]	RBIT 7,[B]	LD B, #00	IFBNE 0F	JSR xF00−xFFF	JMP xF00−xFFF	JP+32	JP+16	F

Where,

i is the immediate data

Md is a directly addressed memory location

* is an unused opcode

Note: The opcode 60 Hex is also the opcode for IFBIT #i,A

Mask Options

The mask programmable options are shown below. The options are programmed at the same time as the ROM pattern submission.

OPTION 1: CLOCK CONFIGURATION

= 1 Crystal Oscillator (CKI/10)
G7 (CKO) is clock generator output to crystal/ resonator with CKI being the clock input

= 2 Single-pin RC controlled oscillator (CKI/10)
G7 is available as a HALT restart and/or general purpose input

OPTION 2: HALT

= 1 Enable HALT mode

= 2 Disable HALT mode

The chip can be driven by a clock input on the CKI input pin which can be between DC and 10 MHz. The CKO output clock is on pin G7 (if clock option = 1 has been selected). The CKI input frequency is divided down by 10 to produce the instruction cycle clock (1/tc).

Development Support

Summary

- iceMASTER: IM-COP8/400—Full feature in-circuit emulation for all COP8 products. A full set of COP8 Basic and Feature Family device and package specific probes are available.
- COP8 Debug Module: Moderate cost in-circuit emulation and development programming unit.
- COP8 Evaluation and Programming Unit: EPU-COP888GG—low cost In-circuit simulation and development programming unit.
- Assembler: COP8-DEV-IBMA. A DOS installable cross development Assembler, Linker, Librarian and Utility Software Development Tool Kit.
- C Compiler: COP8C. A DOS installable cross development Software Tool Kit.
- OTP/EPROM Programmer Support: Covering needs from engineering prototype, pilot production to full production environments.

Development Support (Continued)

IceMASTER (IM) IN-CIRCUIT EMULATION

The iceMASTER IM-COP8/400 is a full feature, PC based, in-circuit emulation tool developed and marketed by MetaLink Corporation to support the whole COP8 family of products. National is a resale vendor for these products.

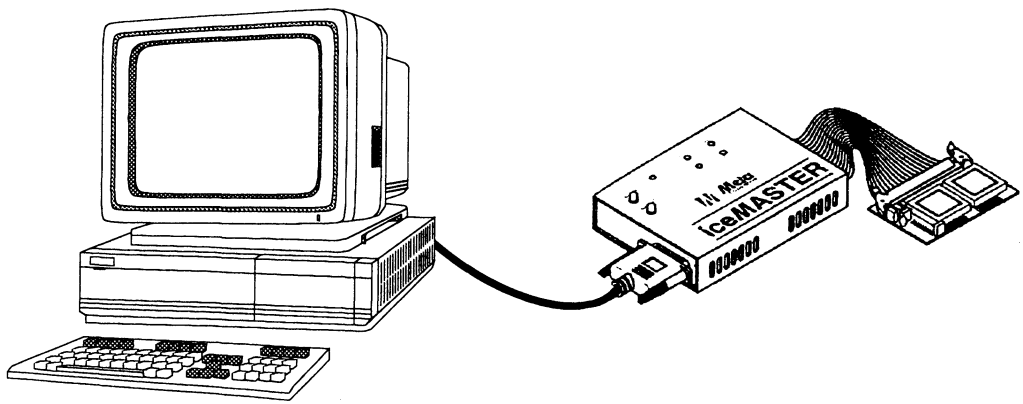
See *Figure 19* for configuration.

The iceMASTER IM-COP8/400 with its device specific COP8 Probe provides a rich feature set for developing, testing and maintaining product:

- Real-time in-circuit emulation; full 2.4V–5.5V operation range, full DC-10 MHz clock. Chip options are programmable or jumper selectable.
- Direct connection to application board by package compatible socket or surface mount assembly.
- Full 32 kbytes of loadable programming space that overlays (replaces) the on-chip ROM or EPROM. On-chip RAM and I/O blocks are used directly or recreated on the probe as necessary.
- Full 4K frame synchronous trace memory. Address, instruction, and 8 unspecified, circuit connectable trace lines. Display can be HLL source (e.g., C source), assembly or mixed.
- A full 64K hardware configurable break, trace on, trace off control, and pass count increment events.
- Tool set integrated interactive symbolic debugger—supports both assembler (COFF) and C Compiler (.COD) linked object formats.
- Real time peformance profiling analysis; selectable bucket definition.
- Watch windows, content updated automatically at each execution break.
- Instruction by instruction memory/register changes displayed on source window when in single step operation.
- Single base unit and debugger software reconfigurable to support the entire COP8 family; only the probe personality needs to change. Debugger software is processor customized, and reconfigured from a master model file.
- Processor specific symbolic display of registers and bit level assignments, configured from master model file.
- Halt/Idle mode notification.
- On-line HELP customized to specific processor using master model file.
- Includes a copy of COP8-DEV-IBMA assembler and linker SDK.

IM Order Information

Base Unit	
IM-COP8/400-1	iceMASTER base unit, 110V power supply
IM-COP8/400-2	iceMASTER base unit, 220V power supply
iceMASTER Probe	
MHW-888KG40DWPC	40 DIP
MHW-888KG44PWPC	44 PLCC



TL/DD/12829–21

FIGURE 19. COP8 iceMASTER Environment

Development Support (Continued)

IceMASTER DEBUG MODULE (DM)

The iceMASTER Debug Module is a PC based, combination in-circuit emulation tool and COP8 based OTP/EPROM programming tool developed and marketed by MetaLink Corporation to support the whole COP8 family of products. National is a resale vendor for these products.

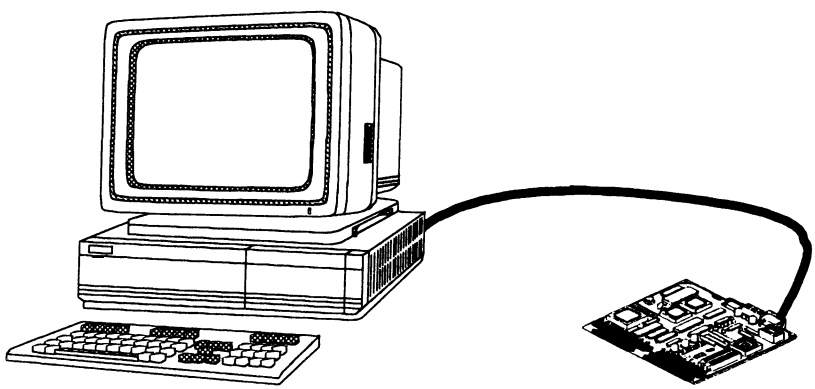
See *Figure 20* for configuration.

The iceMASTER Debug Module is a moderate cost development tool. It has the capability of in-circuit emulation for a specific COP8 microcontroller and in addition serves as a programming tool for COP8 OTP and EPROM product families. Summary of features is as follows:

- Real-time in-circuit emulation; full operating voltage range operation, full DC-10 MHz clock.
- All processor I/O pins can be cabled to an application development board with package compatible cable to socket and surface mount assembly.
- Full 32 kbytes of loadable programming space that overlays (replaces) the on-chip ROM or EPROM. On-chip RAM and I/O blocks are used directly or recreated as necessary.
- 100 frames of synchronous trace memory. The display can be HLL source (C source), assembly or mixed. The most recent history prior to a break is available in the trace memory.
- Configured break points; uses INTR instruction which is modestly intrusive.
- Software—only supported features are selectable.
- Tool set integrated interactive symbolic debugger—supports both Assembler (COFF) and C Compiler (.COD) SDK linked object formats.
- Instruction by instruction memory/register changes displayed when in single step operation.
- Debugger software is processor customized, and reconfigured from a master model file.
- Processor specific symbolic display of registers and bit level assignments, configured from master model file.
- Halt/Idle mode notification.
- Programming menu supports full product line of programmable OTP and EPROM COP8 products. Program data is taken directly from the overlay RAM.
- Programming of 44 PLCC and 68 PLCC parts requires external programming adapters.
- Includes wall mount power supply
- On-board V_{PP} generator from 5V input or connection to external supply supported. Requires V_{PP} level adjustment per the family programming specification (correct level is provided on an on-screen pop-down display).
- On-line HELP customized to specific processor using master model file.
- Includes a copy of COP8-DEV-IBMA assembler and linker SDK.

DM Order Information

Debug Module Unit	
COP8-DM/888KG	
Cable Adapters	
DM-COP8/40D	40 DIP
DM-COP8/44P	44 PLCC



TL/DD/12829–22

FIGURE 20. COP8-DM Environment

Development Support (Continued)

COP8 ASSEMBLER/LINKER SOFTWARE DEVELOPMENT TOOL KIT

National Semiconductor offers a relocateable COP8 macro cross assembler, linker, librarian and utility software development tool kit. Features are summarized as follows:

- Basic and Feature Family instruction set by "device" type.
- Nested macro capability.
- Extensive set of assembler directives.
- Supported on PC/DOS platform.
- Generates National standard COFF output files.
- Integrated Linker & Librarian.
- Integrated utilities to generate ROM code file outputs.
- DUMPCOFF utility.

This product is integrated as a part of MetaLink tools as a development kit, fully supported by the MetaLink debugger. It may be ordered separately or it is bundled with the MetaLink products at no additional cost.

Order-Information

Assembler SDK:	
COP8-DEV-IBMA	Assembler SDK on installable 3.5" PC/DOS Floppy Disk Drive format. Periodic upgrades and most recent version is available on National's BBS and Internet.

COP8 C COMPILER

A C Compiler is developed and marketed by Byte Craft Limited. The COP8C compiler is a fully integrated development tool specifically designed to support the compact embedded configuration of the COP8 family of products.

Features are summarized as follows:

- ANSI C wrth some restrictions and extensions that optimize development for the COP8 embedded application.
- BITS data type extension. Register declaration #pragma with direct bit level definitions.
- C language support for interrupt routines.
- Expert system, rule based code generation and optimization.
- Performs consistency checks against the architectural definitions of the target COP8 device.
- Generates program memory code.
- Supports linking of compiled object or COP8 assembled object formats.
- Global optimization of linked code.
- Symbolic debug load format fully source level supported by the MetaLink debugger.

SINGLE CHIP OTP/EMULATOR SUPPORT

The COP8 family is supported by single chip OTP emulators. For detailed information refer to the emulator specific datasheet and the emulator selection table below:

OTP Emulator Ordering Information

Device Number	Clock Option	Package	Emulates
COP87L88KGV-XE	Crystal/ HALT En	44 PLCC	COP888KG
COP87L88KGN-XE	Crystal/ HALT En	40 DIP	COP888KG

INDUSTRY WIDE OTP/EPROM PROGRAMMING SUPPORT

Programming support, in addition to the MetaLink development tools, is provided by a full range of independent approved vendors to meet the needs from the engineering laboratory to full production.

Approved List

Manufacturer	North America	Europe	Asia
BP Microsystems	(800) 225-2102 (713) 688-4600 Fax: (713) 688-0920	+49-8152-4183 +49-8856-932616	+852-234-16611 +852-2710-8121
Data I/O	(800) 426-1045 (206) 881-6444 Fax: (206) 882-1043	+44-0734-440011	Call North America
HI-LO	(510) 623-8860	Call Asia	+886-2-764-0215 Fax: +886-2-750-0403
ICE Technology	(800) 624-8949 (919) 430-7915	+44-1226-767404 Fax: 0-1226-370-434	
MetaLink	(800) 638-2423 (602) 926-0797 Fax: (602) 693-0681	+49-80 9156 96-0 Fax: 49-80 9123 86	+852-737-1800
Systems General	(408) 263-6667	+41-1-9450300	+886-2-917-3005 Fax: +886-2-911-1283
Needhams	(916) 924-8037 Fax: (916) 924-8065		

Development Support (Continued)

AVAILABLE LITERATURE

For more information, please see the COP8 Basic Family User's Manual, Literature Number 620895, COP8 Feature Family User's Manual Literature Number 620897 and National's Family of 8-bit Microcontrollers COP8 Selection Guide, Literature Number 630009.

DIAL-A-HELPER

Dial-A-Helper is a service provided by the Microcontroller Applications group. The Dial-A-Helper is an Electronic Information System that may be accessed as a Bulletin Board System (BBS) via data modem, as an FTP site on the Internet via standard FTP client application or as an FTP site on the Internet using a standard Internet browser such as Netscape or Mosaic.

The Dial-A-Helper system provides access to an automated information storage and retrieval system. The system capabilities include a MESSAGE SECTION (electronic mail, when accessed as a BBS) for communication to and from the Microcontroller Applications Group and a FILE SECTION which consists of several file areas where valuable application software and utilities could be found.

DIAL-A-HELPER BBS via a Standard Modem

Modem: CANADA/U.S.: (800) NSC-MICRO
(800) 672-6427
EUROPE: (+49) 0-8141-351332
Baud: 14.4k
Set-up: Length: 8-Bit
Parity: None
Stop Bit: 1
Operation: 24 Hrs., 7 Days

DIAL-A-HELPER via FTP

ftp nscmicro.nsc.com
user: anonymous
password: username@yourhost.site.domain

DIAL-A-HELPER via a WorldWide Web Browser

ftp://nscmicro.nsc.com

National Semiconductor on the WorldWide Web

See us on the WorldWide Web at: http://www.national.com

CUSTOMER RESPONSE CENTER

Complete product information and technical support is available from National's customer response centers.

CANADA/US:	Tel:	(800) 272-9959
	email:	support @tevm2.nsc.com
EUROPE:	email:	europe.support@nsc.com
	Deutsch Tel:	+49 (0) 180-530 85 85
	English Tel:	+49 (0) 180-532 78 32
	Français Tel:	+49 (0) 180-532 93 58
	Italiano Tel:	+49 (0) 180-534 16 80
JAPAN:	Tel:	+81-043-299-2309
S.E. ASIA:	Beijing Tel:	(+86) 10-6856-8601
	Shanghai Tel:	(+86) 21-6415-4092
	Hong Kong Tel:	(+852) 2737-1600
	Korea Tel:	(+82) 2-3771-6909
	Malaysia Tel:	(+60-4) 644-9061
	Singapore Tel:	(+65) 255-2226
	Taiwan Tel:	+886-2-521-3288
AUSTRALIA:	Tel:	(+61) 3-9558-9999
INDIA:	Tel:	(+91) 80-559-9467

National Semiconductor

PRELIMINARY

COP888GW
8-Bit Microcontroller with Pulse Train Generators and Capture Modules

General Description

The COP888 family of microcontrollers uses an 8-bit single chip core architecture fabricated with National Semiconductor's M²CMOS™ process technology. The COP888GW is a member of this expandable 8-bit core processor family of microcontrollers. It is a fully static part, fabricated using double-metal silicon gate microCMOS technology.

Features include an 8-bit memory mapped architecture, MICROWIRE/PLUS serial I/O, two 16-bit timer/counters supporting three modes (Processor Independent PWM generation, External Event counter and Input Capture mode capabilities), four independent 16-bit pulse train generators with 16-bit prescalers, two independent 16-bit input capture modules with 8-bit prescalers, multiply and divide functions, full duplex UART, and two power savings modes (HALT and IDLE), both with a multi-sourced wake up/interrupt capability. This multi-sourced interrupt capability may also be used independent of the HALT or IDLE modes.

Each I/O pin has software selectable configurations. The devices operate over a voltage range of 2.5V–6V. High throughput is achieved with an efficient, regular instruction set operating at a maximum of 1 μs per instruction rate. The device has low EMI emissions. Low radiated emissions are achieved by gradual turn-on output drivers and internal I_{CC} filters on the chip logic and crystal oscillator. The device is available in 68-pin PLCC package.

Key Features

- Two 16-bit input capture modules with 8-bit prescalers
- Four Pulse Train Generators with 16-bit prescalers
- Full duplex UART
- Two 16-bit timers, each with two 16-bit registers supporting:
  - — Processor independent PWM mode
  - — External event counter mode
  - — Input capture mode
- Quiet design (low radiated emissions)
- 16 kbytes on-board ROM
- 512 bytes on-board RAM

Additional Peripheral Features

- Idle Timer
- Multi-Input Wake-Up (MIWU) with optional interrupts (8)
- MICROWIRE/PLUS™ serial I/O

I/O Features

- Memory mapped I/O
- Software selectable I/O options (TRI-STATE® Output, Push-Pull Output, Weak Pull-Up Input, High Impedance Input)
- Schmitt trigger inputs on port G
- Package: 68-pin PLCC

CPU/Instruction Set Features

- 1 μs instruction cycle time
- Fourteen multi-source vectored interrupts servicing:
  - — External Interrupt with selectable edge
  - — Idle Timer T0
  - — Two Timers (each with 2 interrupts)
  - — MICROWIRE/PLUS
  - — Multi-Input Wake-Up
  - — Software Trap
  - — UART (2)
  - — Capture Timers
  - — Counters (one vector for all four counters)
  - — Default VIS (default interrupt)
- Versatile and easy-to-use instruction set
- 8-bit Stack Pointer SP—(stack in RAM)
- Two 8-bit register indirect data memory pointers (B and X)

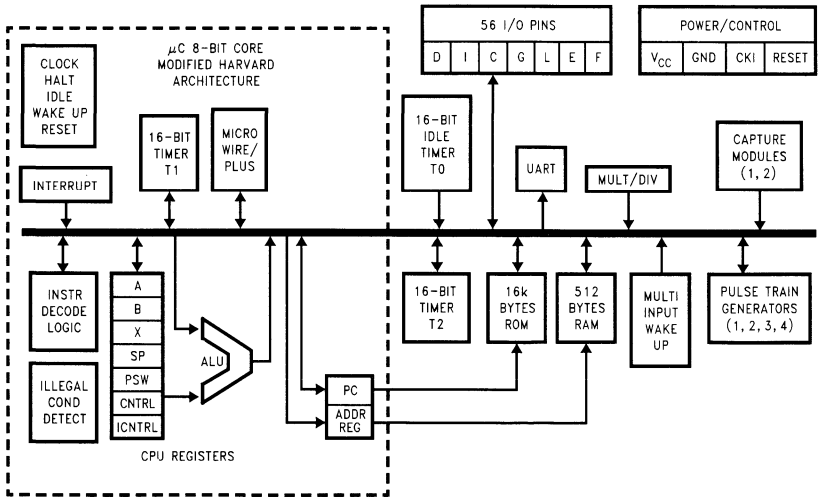
Fully Static CMOS

- Two power saving modes: HALT and IDLE
- Low current drain (typically <1 μA)
- Single supply operation: 2.5V–5.5V
- Temperature range: $-40°C$ to $+85°C$

Development Support

- Emulation and OTP device
- Real time emulation and full program debug offered by MetaLink's Development System

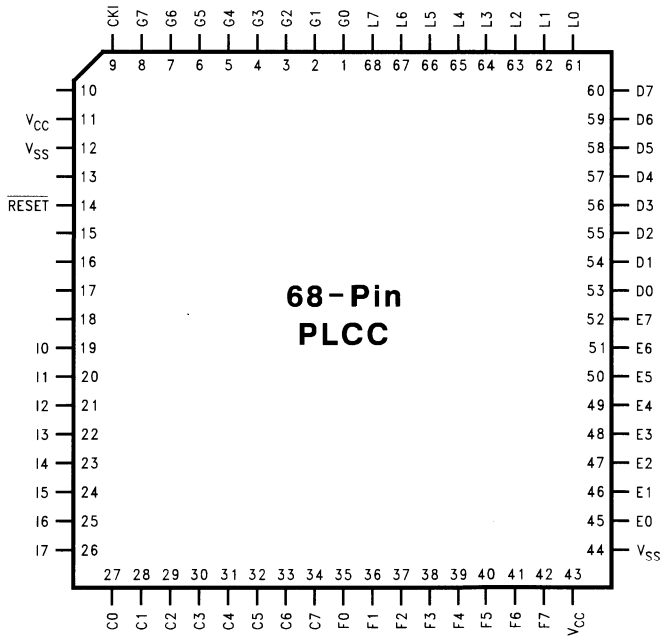
Block Diagram



TL/DD/12065–1

FIGURE 1. COP888GW Block Diagram

Connection Diagram



TL/DD/12065–2

Top View

Order Number COP888GW-XXX/V
See NS Package Number V68A

Absolute Maximum Ratings (Note)

Supply Voltage (V_{CC})	7V
Voltage at Any Pin	−0.3V to V_{CC} +0.3V
Total Current into V_{CC} Pin (Source)	100 mA
Total Current out of GND Pin (Sink)	110 mA
Storage Temperature Range	−65°C to +150°C

Note: *Absolute maximum ratings indicate limits beyond which damage to the device may occur. DC and AC electrical specifications are not ensured when operating the device at absolute maximum ratings.*

DC Electrical Characteristics COP888GW: −40°C ≤ T_A ≤ +85°C unless otherwise specified

Parameter	Conditions	Min	Typ	Max	Units
Operating Voltage		2.5		6.0	V
Power Supply Ripple (Note 1)	Peak-to-Peak			0.1 V_{CC}	V
Supply Current (Note 2)					
CKI = 10 MHz	V_{CC} = 6V, t_c = 1 μs			10	mA
CKI = 4 MHz	V_{CC} = 2.5V, t_c = 2.5 μs			1.7	mA
HALT Current (Note 3)	V_{CC} = 6V, CKI = 0 MHz		<1	10	μA
IDLE Current					
CKI = 10 MHz	V_{CC} = 6V			1.7	mA
CKI = 4 MHz	V_{CC} = 2.5V			0.4	mA
Input Levels (V_{IH}, V_{IL})					
$\overline{\text{RESET}}$, CKI					
Logic High		0.8 V_{CC}			V
Logic Low				0.2 V_{CC}	V
All Other Inputs					
Logic High		0.7 V_{CC}			V
Logic Low				0.2 V_{CC}	V
Hi-Z Input Leakage	V_{CC} = 6V	−2		+2	μA
Input Pullup Current	V_{CC} = 6V, V_{IN} = 0V	−40		−250	μA
G Port Input Hysteresis	(Note 6)		0.05 V_{CC}	0.35 V_{CC}	V
Output Current Levels					
D Outputs					
Source	V_{CC} = 4V, V_{OH} = 3.3V	−0.4			mA
	V_{CC} = 2.5V, V_{OH} = 1.8V	−0.2			mA
Sink	V_{CC} = 4V, V_{OL} = 1V	10			mA
	V_{CC} = 2.5V, V_{OL} = 0.4V	2.0			mA
All Others					
Source (Weak Pull-Up Mode)	V_{CC} = 4V, V_{OH} = 2.7V	−10		−100	μA
	V_{CC} = 2.5V, V_{OH} = 1.8V	−2.5		−33	μA
Source (Push-Pull Mode)	V_{CC} = 4V, V_{OH} = 3.3V	−0.4			mA
	V_{CC} = 2.5V, V_{OH} = 1.8V	−0.2			mA
Sink (Push-Pull Mode)	V_{CC} = 4V, V_{OL} = 0.4V	1.6			mA
	V_{CC} = 2.5V, V_{OL} = 0.4V	0.7			mA
TRI-STATE Leakage	V_{CC} = 6.0V	−2		+2	μA
Allowable Sink/Source Current per Pin					
D Outputs (Sink)				15	mA
All others				3	mA
Maximum Input Current without Latchup (Note 4, 6)	Room Temp			±200	mA
RAM Retention Voltage, V_R (Note 5)	500 ns Rise and Fall Time (min)	2			V
Input Capacitance	(Note 6)			7	pF
Load Capacitance on D2	(Note 6)			1000	pF

AC Electrical Characteristics COP888GW: $-40°C \le T_A \le +85°C$ unless otherwise specified

Parameter	Conditions	Min	Typ	Max	Units
Instruction Cycle Time (t_c)					
Crystal, Resonator	$2.5V \le V_{CC} < 4V$	2.5		DC	μs
Ceramic	$V_{CC} \ge 4V$	1.0		DC	μs
CKI Clock Duty Cycle (Note 5)	f = Max	40		60	%
Rise Time (Note 5)	f = 10 MHz Ext Clock			5	μs
Fall Time (Note 5)	f = 10 MHz Ext Clock			5	μs
Inputs					
t_{SETUP}	$V_{CC} \ge 4V$	200			ns
	$2.5V \le V_{CC} < 4V$	500			ns
t_{HOLD}	$V_{CC} \ge 4V$	60			ns
	$2.5V \le V_{CC} < 4V$	150			ns
Output Propagation Delay (Note 8)	R_L = 2.2k, C_L = 100 pF				
t_{PD1}, t_{PD0}					
SO, SK	$V_{CC} \ge 4V$			0.7	μs
	$2.5V \le V_{CC} < 4V$			1.8	μs
All Others	$V_{CC} \ge 4V$			1	μs
	$2.5V \le V_{CC} < 4V$			2.5	μs
MICROWIRE™ Setup Time (t_{UWS}) (Note 6)	$V_{CC} \ge 4V$	20			ns
MICROWIRE Hold Time (t_{UWH}) (Note 6)	$V_{CC} \ge 4V$	56			ns
MICROWIRE Output Propagation Delay (t_{UPD})	$V_{CC} \ge 4V$			220	ns
Input Pulse Width (Note 7)					
Interrupt Input High Time		1			t_c
Interrupt Input Low Time		1			t_c
Timer 1, 2 Input High Time		1			t_c
Timer 1, 2 Input Low Time		1			t_c
Capture Timer High Time		1			CKI
Capture Timer Low Time		1			CKI
Reset Pause Width		1			t_c

Note 1: Maximum rate of voltage change to be defined.

Note 2: Supply current is measured after running 2000 cydes with a square wave CKI input, CKO open, inputs at rails and outputs open.

Note 3: The HALT mode will stop CKI from oscillatng. Test conditions: All inputs tied to V_{CC}, L, C, E, F, and G port I/O's configured as outputs and programmed low and not driving a load; D outputs programmed low and not driving a load. Parameter refers to HALT mode entered via setting bit 7 of the G Port data register. Part will pull up CKI during HALT in crystal clock mode.

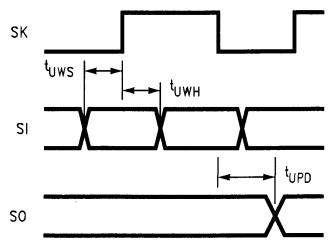
Note 4: Pins G6 and $\overline{\text{RESET}}$ are designed with a high voltage input network. These pins allow input voltages greater than V_{CC} and the pins will have sink current to V_{CC} when biased at voltages greater than V_{CC} (the pins do not have source current when biased at a voltage below V_{CC}.) The effective resistance to V_{CC} is 750Ω (typical). These two pins will not latch up. The voltage at the pins must be limited to less than 14 volts. WARNING: Voltages in excess of 14 volts will cause damage to the pins. This warning excludes ESD transients.

Note 5: Condition and parameter valid only for part in HALT mode.

Note 6: Parameter characterized but not tested.

Note 7: t_c = Instruction Cycle Time

Note 8: The output propagation delay is referenced to the end of the instruction cycle where the output change occurs.

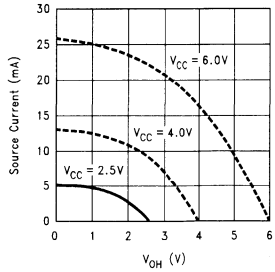


TL/DD/12065–3

FIGURE 2. MICROWIRE/PLUS Timing

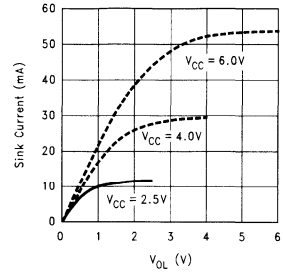
Typical Performance Characteristics $-40°C \leq T_A \leq +85°C$

Port D Source Current



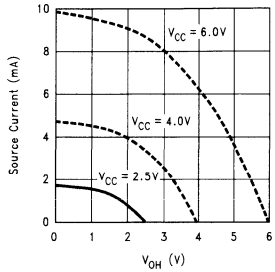
TL/DD/12065–23

Port D Sink Current



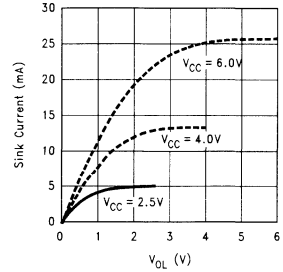
TL/DD/12065–24

Ports C/G/L/E/F Source Current



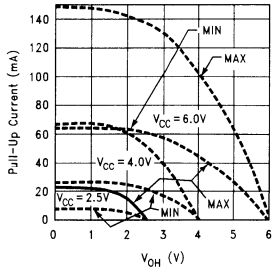
TL/DD/12065–25

Ports C/G/L/E/F Sink Current



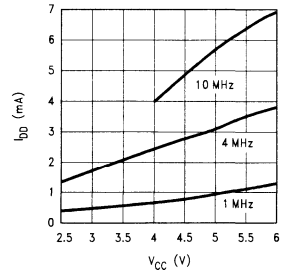
TL/DD/12065–26

Ports C/G/L/E/F Weak Pull-Up Source Current



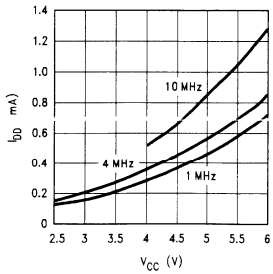
TL/DD/12065–27

Dynamic — I_{DD} vs V_{CC}



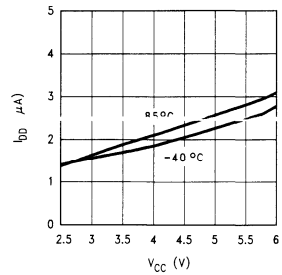
TL/DD/12065–28

Idle — I_{DD} vs V_{CC}



TL/DD/12065–29

HALT — I_{DD} vs V_{CC}



TL/DD/12065–30

Pin Descriptions

V_{CC} and GND are the power supply pins. All V_{CC} and GND pins must be connected.

CKI is the clock input. This comes from an R/C generated oscillator, or a crystal oscillator (in conjunction with CKO). See Oscillator Description section.

$\overline{\text{RESET}}$ is the master reset input. See Reset description section.

The device contains five bidirectional 8-bit I/O ports (C, E, F, G and L), where each individual bit may be independently configured as an input (Schmitt trigger inputs on ports L and G), output or TRI-STATE under program control. Three data memory address locations are allocated for each of these I/O ports. Each I/O port has two associated 8-bit memory mapped registers, the CONFIGURATION register and the output DATA register. A memory mapped address is also reserved for the input pins of each I/O port. (See the memory map for the various addresses associated with the I/O ports.) *Figure 3* shows the I/O port configurations. The DATA and CONFIGURATION registers allow for each port bit to be individually configured under software control as shown below:

Configuration Register	Data Register	Port Set-Up
0	0	Hi-Z Input (TRI-STATE Output)
0	1	Input with Weak Pull-Up
1	0	Push-Pull Zero Output
1	1	Push-Pull One Output

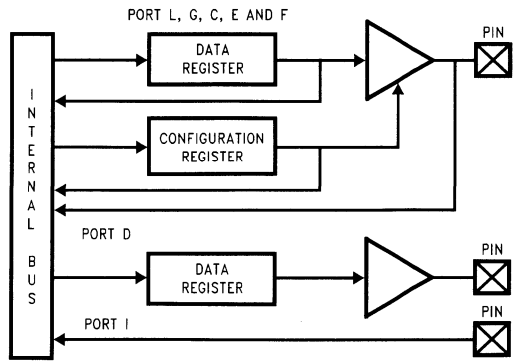
PORT L is an 8-bit I/O port. All L-pins have Schmitt triggers on the inputs.

The Port L supports Multi-Input Wake Up on all eight pins. L1 is used for the UART external clock. L2 and L3 are used for the UART transmit and receive. L4 and L5 are used for the timer input functions T2A and T2B. L6 and L7 are used for the capture timer input functions CAP1 and CAP2.

The Port L has the following alternate features:

- L0 MIWU
- L1 MIWU or CKX
- L2 MIWU or TDX
- L3 MIWU or RDX
- L4 MIWU or T2A
- L5 MIWU or T2B
- L6 MIWU or CAP1
- L7 MIWU or CAP2

Port G is an 8-bit port with 6 I/O pins (G0–G5), an input pin (G6), and a dedicated output pin (G7). Pins G0–G6 all have Schmitt Triggers on their inputs. Pin G7 serves as the dedicated output pin for the CKO clock output. There are two registers associated with the G Port, a data register and a configuration register. Therefore, each of the 6 I/O bits (G0–G5) can be individually configured under software control.



TL/DD/12065–4

FIGURE 3. I/O Port Configurations

5

Pin Descriptions (Continued)

Since G6 is an input only pin and G7 is dedicated CKO clock output pin, the associated bits in the data and configuration registers for G6 and G7 are used for special purpose functions as outlined below. Reading the G6 and G7 data bits will return zeros.

Note that the chip will be placed in the HALT mode by writing a "1" to bit 7 of the Port G Data Register. Similarly the chip will be placed in the IDLE mode by writing a "1" to bit 6 of the Port G Data Register.

Writing a "1" to bit 6 of the Port G Configuration Register enables the MICROWIRE/PLUS to operate with the alternate phase of the SK clock.

	Config Reg.	Data Reg.
G7	Not Used	HALT
G6	Alternate SK	IDLE

Port G has the following alternate features:

- G0 INTR (External Interrupt Input)
- G2 T1B (Timer T1 Capture Input)
- G3 T1A (Timer T1 I/O)
- G4 SO (MICROWIRE Serial Data Output)
- G5 SK (MICROWIRE Serial Clock)
- G6 SI (MICROWIRE Serial Data Input)

Port G has the following dedicated functions:

- G7 CKO Oscillator dedicated output

Ports C and F are 8-bit I/O ports.

Port E is an 8-bit I/O port. It has the following alternate features:

- E0 CT1 (Output for counter1, Pulse Train Generator)
- E1 CT2 (Output for counter2, Pulse Train Generator)
- E2 CT3 (Output for counter3, Pulse Train Generator)
- E3 CT4 (Output for counter4, Pulse Train Generator)

Port I is an eight-bit Hi-Z input port.

Port D is an 8-bit output port that is preset high when $\overline{\text{RESET}}$ goes low. The user can tie two or more D port outputs (except D2) together in order to get a higher drive.

Functional Description

The architecture of the device is modified Harvard architecture. With the Harvard architecture, the control store program memory (ROM) is separated from the data store memory (RAM). Both ROM and RAM have their own separate addressing space with separate address buses. The architecture, though based on Harvard architecture, permits transfer of data from ROM to RAM.

CPU REGISTERS

The CPU can do an 8-bit addition, subtraction, logical or shift operation in one instruction (t_c) cycle time.

There are six CPU registers:

A is the 8-bit Aocumulator Register

PC is the 15-bit Program Counter Register

- PU is the upper 7 bits of the program counter (PC)
- PL is the lower 8 bits of the program counter (PC)

B is an 8-bit RAM address pointer, which can be optionally post auto incremented or decremented.

X is an 8-bit alternate RAM address pointer, which can be optionally post auto incremented or decremented.

SP is the 8-bit stack pointer, which points to the subroutine/interrupt stack (in RAM). The SP is initialized to RAM address 06F with reset.

S is the 8-bit Data Segment Address Register used to extend the lower half of the address range (00 to 7F) into 256 data segments of 128 bytes each.

All the CPU registers are memory mapped with the exception of the Accumulator (A) and the Program Counter (PC).

PROGRAM MEMORY

The program memory consists of 16384 bytes of ROM. These bytes may hold program instructions or constant data (data tables for the LAID instruction, jump vectors for the JID instruction, and interrupt vectors for the VIS instruction). The program memory is addressed by the 15-bit program counter (PC). All interrupts in the devices Vector to program memory location OFF Hex.

DATA MEMORY

The data memory address space includes the on-chip RAM and data registers, the I/O registers (Configuration, Data and Pin), the control registers, the MICROWIRE/PLUS SIO shift register, and the various registers, and counters associated with the timers (with the exception of the IDLE timer). Data memory is addressed directly by the instruction or indirectly by the B, X, SP pointers and S register.

The data memory consists of 512 bytes of RAM. Sixteen bytes of RAM are mapped as "registers" at addresses 0F0 to 0FF Hex. These registers can be loaded immediately, and also decremented and tested with the DRSZ (decrement register and skip if zero) instruction. The memory pointer registers X, SP, B and S are memory mapped into this space at address locations 0FC to 0FF Hex respectively, with the other registers being available for general usage.

Note: RAM contents are undefined upon power-up.

Data Memory Segment RAM Extension

Data memory address 0FF is used as a memory mapped location for the Data Segment Address Register (S).

The data store memory is either addressed directly by a single-byte address within the instruction, or indirectly relative to the reference of the B, X, or SP pointers (each contains a single-byte address). This single-byte address allows an addressing range of 256 locations from 00 to FF hex. The upper bit of this single-byte address divides the data store memory into two separate sections as outlined previously. With the exception of the RAM register memory from address locations 00F0 to 00FF, all RAM memory is memory mapped with the upper bit of the single-byte address being equal to zero. This allows the upper bit of the single-byte address to determine whether or not the base address range (from 0000 to 00FF) is extended. If this upper bit equals one (representing address range 0080 to 00FF), then address extension does not take place. Alternatively, if this upper bit equals zero, then the data segment extension

Data Memory Segment RAM Extension (Continued)

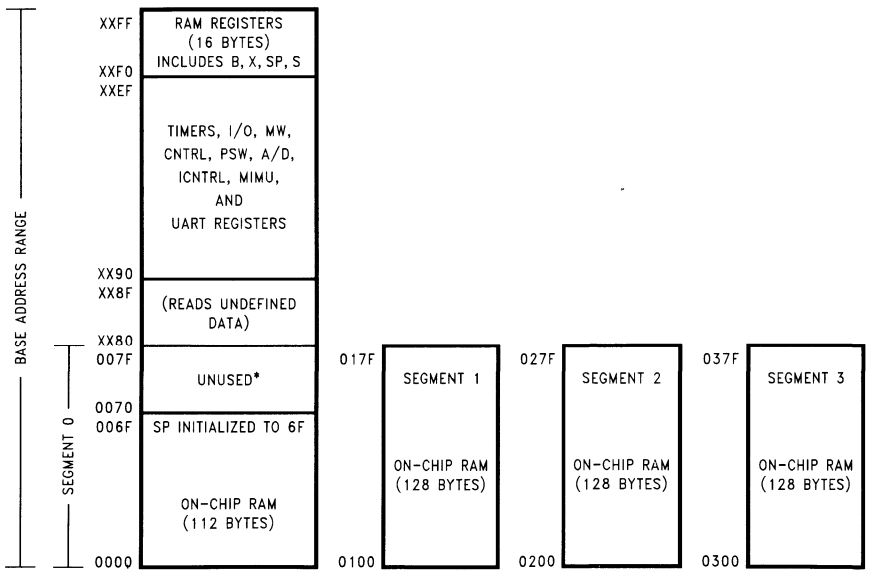
register S is used to extend the base address range (from 0000 to 007F) from XX00 to XX7F, where XX represents the 8 bits from the S register. Thus the 128-byte data segment extensions are located from addresses 0100 to 017F for data segment 1, 0200 to 027F for data segment 2, etc., up to FF00 to FF7F for data segment 255. The base address range from 0000 to 007F represents data segment 0.

Figure 4 illustrates how the S register data memory extension is used in extending the lower half of the base address range (00 to 7F hex) into 256 data segments of 128 bytes each, with a total addressing range of 32 kbytes from XX00 to XX7F. This organization allows a total of 256 data segments of 128-bytes each with an additional upper base segment of 128 bytes. Furthermore, all addressing modes are available for all data segments. The S register must be changed under program control to move from one data segment (128 bytes) to another. However, the upper base segment (containing the 16 memory registers, I/O registers, control registers, etc.) is always available regardless of the contents of the S register, since the upper base segment (address range 0080 to 00FF) is independent of data segment extension.

The instructions that utilize the stack pointer (SP) always reference the stack as part of the base segment (Segment 0), regardless of the contents of the S register. The S register is not changed by these instructions. Consequently, the stack (used with subroutine linkage and interrupts) is always located in the base segment. The stack pointer will be initialized to point at data memory location 006F as a result of reset.

The 128 bytes of RAM contained in the base segment are split between the lower and upper base segments. The first 112 bytes of RAM are resident from address 0000 to 006F in the lower base segment, while the remaining 16 bytes of RAM represent the 16 data memory registers located at addresses 00F0 to 00FF of the upper base segment. No RAM is located at the upper sixteen addresses (0070 to 007F) of the lower base segment.

Additional RAM beyond these initial 128 bytes, however, will always be memory mapped in groups of 128 bytes (or less) at the data segment address extensions (XX00 to XX7F) of the lower base segment. The additional 384 bytes of RAM in this device are memory mapped at address locations 0100 to 017F° 0200 to 027F, and 0300 to 037F hex.



TL/DD/12065–5

*Reads as all ones.

FIGURE 4. RAM Organization

Reset

This device enters a reset state immediately upon detecting a logic low on the $\overline{\text{RESET}}$ pin. The $\overline{\text{RESET}}$ pin must be held low for a minimum of one instruction cycle to guarantee a valid reset. During power-up initialization, the user must insure that the $\overline{\text{RESET}}$ pin is held low until this device is within the specified V_{CC} voltage. An R/C circuit on the $\overline{\text{RESET}}$ pin with a delay 5 times (5x) greater than the power supply rise time is recommended.

When the $\overline{\text{RESET}}$ input goes low, the I/O ports are initialized immediately, with any observed delay being only propagation delay. When the $\overline{\text{RESET}}$ pin goes high, this device comes out of the reset state synchronously. This device will be running within two instruction cycles of the $\overline{\text{RESET}}$ pin going high.

$\overline{\text{RESET}}$ may also be used to exit this device from the HALT mode.

Some registers are reset to a known state, whereas other registers and RAM are "unchanged" by reset. When the controller goes into reset state while it is performing a write operation to one of these registers or RAM that are "unchanged" by reset, the register or RAM value will become unknown (i.e. not unchanged). This is because the write operation is terminated prematurely by reset and the results become uncertain. These registers and RAM locations are unchanged by reset only if they are not written to when the controller resets.

The following initializations occur with $\overline{\text{RESET}}$:

Port L: TRI-STATE

Port C: TRI-STATE

Port G: TRI-STATE

Port E: TRI-STATE

Port F: TRI-STATE

Port D: HIGH

PC: CLEARED

PSW, CNTRL and ICNTRL registers: CLEARED

SIOR:

UNAFFECTED after RESET with power already applied

RANDOM after RESET at power-on

T1CNTRL: CLEARED

T2CNTRL: CLEARED

TxRA, TxRB: RANDOM

CCMR1, CCMR2: CLEARED

CM1PSC, CM1CRL, CM1CRH, CM2PSC, CM2CRL, and CM2CRH:

UNAFFECTED after RESET with power already applied

RANDOM after RESET at power-on

CCR1 and CCR2: CLEARED

CxPRH, CxPRL, CxCTH, and CxCTL:

UNAFFECTED after RESET with RC clock option (power already applied)

RANDOM after RESET at power-on

PSR, ENUR and ENUI: CLEARED

ENU: CLEARED except Bit 1 (TBMT) = 1

Accumulator, Timer 1 and Timer 2:

RANDOM after RESET with crystal clock option (power already applied)

RANDOM after RESET at power-on

MDCR: CLEARED

MDR1, MDR2, MDR3, MDR4, MDR5: RANDOM

WKEN, WKEDG: CLEARED

WKPND: RANDOM

S Register: CLEARED

SP (Stack Pointer): Loaded with 6F Hex

B and X Pointers:

UNAFFECTED after RESET with power already applied

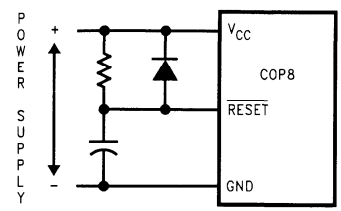
RANDOM after RESET at power-on

RAM:

UNAFFECTED after RESET with power already applied

RANDOM after RESET at power-on

The external RC network shown in *Figure 5* should be used to ensure that the $\overline{\text{RESET}}$ pin is held low until the power supply to the chip stabilizes.



TL/DD/12065–6

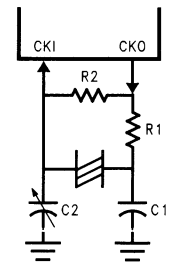
RC > 5 × POWER SUPPLY RISE TIME

FIGURE 5. Recommended Reset Circuit

Oscillator Circuits

The chip can be driven by a clock input on the CKI input pin which can be between DC and 10 MHz. The CKO output clock is on pin G7 (crystal configuration), The CKI input frequency is divided down by 10 to produce the instruction cycle clock (t_C).

Figure 6 shows the Crystal diagram



TL/DD/12065–7

FIGURE 6. Crystal Diagram

CRYSTAL OSCILLATOR

CKI and CKO can be connected to make a closed loop crystal (or resonator) controlled oscillator.

Oscillator Circuits (Continued)

Table I shows the component values required for various standard crystal values.

TABLE I. Crystal Oscillator Configuration, T_A = 25°C

R1 (kΩ)	R2 (MΩ)	C1 (pF)	C2 (pF)	CKI Freq (MHz)	Conditions
0	1	30	30–36	10	V_{CC} = 5V
0	1	30	30–36	4	V_{CC} = 5V
0	1	200	100–150	0.455	V_{CC} = 5V

Control Registers

CNTRL Register (Address X'00EE)

The Timer1 (T1) and MICROWIRE/PLUS control register contains the following bits:

SL1 & SL0 Select the MICROWIRE/PLUS clock divide by (00 = 2, 01 = 4, 1x = 8)

IEDG External interrupt edge polarity select (0 = Rising edge, 1 = Falling edge)

MSEL Selects G5 and G4 as MICROWIRE/PLUS signals SK and SO respectively

T1C0 Timer T1 Start/Stop control in timer modes 1 and 2

T1 Underflow Interrupt Pending Flag in timer mode 3

T1C1 Timer T1 mode control bit

T1C2 Timer T1 mode control bit

T1C3 Timer T1 mode control bit

T1C3	T1C2	T1C1	T1C0	MSEL	IEDG	SL1	SL0

Bit 7 Bit 0

PSW Register (Address X'00EF)

The PSW register contains the following select bits:

GIE Global interrupt enable (enables interrupts)

EXEN Enable external interrupt

BUSY MICROWIRE/PLUS busy shifting flag

EXPND External interrupt pending

T1ENA Timer T1 Interrupt Enable for Timer Underflow or T1A Input capture edge

T1PNDA Timer T1 Interrupt Pending Flag (Autoreload RA in mode 1, T1 Underflow in Mode 2, T1A capture edge in mode 3)

C Carry Flag

HC Half Carry Flag

HC	C	T1PNDA	T1ENA	EXPND	BUSY	EXEN	GIE

Bit 7 Bit 0

The Half-Carry flag is also affected by all the instructions that affect the Carry flag. The SC (Set Carry) and RC (Reset Carry) instructions will respectively set or clear both the carry flags. In addition to the SC and RC instructions, ADC, SUBC, RRC and RLC instructions affect the Carry and Half Carry flags.

ICNTRL Register (Address X'00E8)

The ICNTRL register contains the following bits:

T1ENB Timer T1 Interrupt Enable for T1B Input capture edge

T1PNDB Timer T1 Interrupt Pending Flag for T1B capture edge

μWEN Enable MICROWIRE/PLUS interrupt

μWPND MICROWIRE/PLUS interrupt pending

T0EN Timer T0 Interrupt Enable (Bit 12 toggle)

T0PND Timer T0 Interrupt pending

LPEN L Port Interrupt Enable (Multi-Input Wake up/Interrupt)

Bit 7 could be used as a flag

Unused	LPEN	T0PND	T0EN	WPND	WEN	T1PNDB	T1ENB

Bit 7 Bit 0

T2CNTRL Register (Address X'00C6)

The T2CNTRL register contains the following bits:

T2ENB Timer T2 Interrupt Enable for T2B Input capture edge

T2PNDB Timer T2 Interrupt Pending Flag for T2B capture edge

T2ENA Timer T2 Interrupt Enable for Timer Underflow or T2A Input capture edge

T2PNDA Timer T2 Interrupt Pending Flag (Auto reload RA in mode 1, T2 Underflow in mode 2, T2A capture edge in mode 3)

T2C0 Timer T2 Start/Stop control in timer modes 1 and 2 Timer T2 Underflow Interrupt Pending Flag in timer mode 3

T2C1 Timer T2 mode control bit

T2C2 Timer T2 mode control bit

T2C3 Timer T2 mode control bit

T2C3	T2C2	T2C1	T2C0	T2PNDA	T2ENA	T2PNDB	T2ENB

Bit 7 Bit 0

Timers

The device contains a very versatile set of timers (T0, T1, T2). All timers and associated autoreload/capture registers power up containing random data.

TIMER T0 (IDLE TIMER)

The device supports applications that require maintaining real time and low power with the IDLE mode. This IDLE mode support is furnished by the IDLE timer T0, which is a 16-bit timer. The Timer T0 runs continuously at the fixed rate of the instruction cycle clock, t_c. The user cannot read or write to the IDLE Timer T0, which is a count down timer.

The Timer T0 supports the following functions:

- Exit out of the Idle Mode (See Idle Mode description)
- Start up delay out of the HALT mode

The IDLE Timer T0 can generate an interrupt when the thirteenth bit toggles. This toggle is latched into the T0PND pending flag, and will occur every 4 ms at the maximum clock frequency (t_c = 1 μs). A control flag T0EN allows the interrupt from the thirteenth bit of Timer T0 to be enabled or disabled. Setting T0EN will enable the interrupt, while resetting it will disable the interrupt.

TIMER T1 AND TIMER T2

The device has a set of two powerful timer/counter blocks, T1 and T2. The associated features and functioning of a timer block are described by referring to the timer block Tx. Since the two timer blocks, T1 and T2 are identical, all comments are equally applicable to either of the two timer blocks.

Each timer block consists of a 16-bit timer, Tx, and two supporting 16-bit autoreload/capture registers, RxA and RxB. Each timer block has two pins associated with it, TxA and TxB. The pin TxA supports I/O required by the timer block, while the pin TxB is an input to the timer block. The powerful and flexible timer block allows the device to easily perform all timer functions with minimal software overhead. The timer block has three operating modes: Processor Independent PWM mode, External Event Counter mode, and Input Capture mode.

The control bits TxC3, TxC2, and TxC1 allow selection of the different modes of operation.

Mode 1. Processor Independent PWM Mode

As the name suggests, this mode allows the device to generate a PWM signal with very minimal user intervention. The user only has to define the parameters of the PWM signal (ON time and OFF time). Once begun, the timer block will continuously generate the PWM signal completely independent of the microcontroller. The user software services the timer block only when the PWM parameters require updating.

In this mode the timer Tx counts down at a fixed rate of tc. Upon every underflow the timer is alternately reloaded with the contents of supporting registers, RxA and RxB. The very first underflow of the timer causes the timer to reload from the register RxA. Subsequent underflows cause the timer to be reloaded from the registers alternately beginning with the register RxB.

The Tx Timer control bits, TxC3, TxC2 and TxC1 set up the timer for PWM mode operation.

Figure 7 shows a block diagram of the timer in PWM mode.

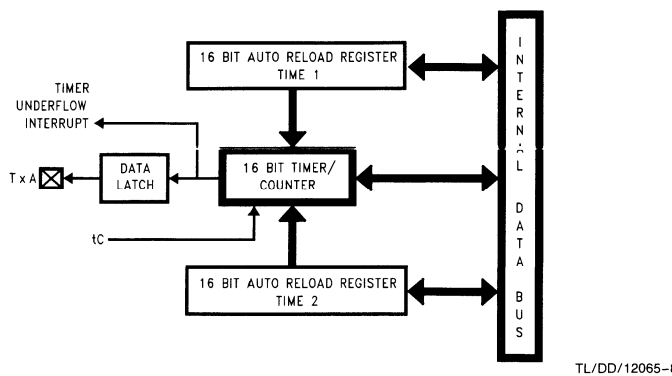
The underflows can be programmed to toggle the TxA output pin. The underflows can also be programmed to generate interrupts.

Underflows from the timer are alternately latched into two pending flags, TxPNDA and TxPNDB. The user must reset these pending flags under software control. Two control enable flags, TxENA and TxENB, allow the interrupts from the timer underflow to be enabled or disabled. Setting the timer enable flag TxENA will cause an interrupt when a timer underflow causes the RxA register to be reloaded into the timer. Setting the timer enable flag TxENB will cause an interrupt when a timer underflow causes the RxB register to be reloaded into the timer. Resetting the timer enable flags will disable the associated interrupts.

Either or both of the timer underflow interrupts may be enabled. This gives the user the flexibility of interrupting once per PWM period on either the rising or falling edge of the PWM output. Alternatively, the user may choose to interrupt on both edges of the PWM output.

Mode 2. External Event Counter Mode

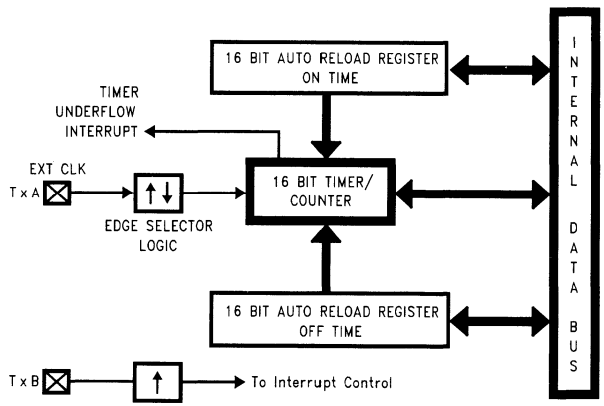
This mode is quite similar to the processor independent PWM mode described above. The main difference is that the timer, Tx, is clocked by the input signal from the TxA pin. The Tx timer control bits, TxC3, TxC2 and TxC1 allow the timer to be clocked either on a positive or negative edge from the TxA pin. Underflows from the timer are latched into the TxPNDA pending flag. Setting the TxENA control flag will cause an interrupt when the timer underflows.



TL/DD/12065–8

FIGURE 7. Timer in PWM Mode

Timers (Continued)



TL/DD/12065–9

FIGURE 8. Timer in External Event Counter Mode

In this mode the input pin TxB can be used as an independent positive edge sensitive interrupt input if the TxENB control flag is set. The occurrence of a positive edge on the TxB input pin is latched into the TxPNDB flag.

Figure 8 shows a block diagram of the timer in External Event Counter mode.

Note: The PWM output is not available in this mode since the TxA pin is being used as the counter input clock.

Mode 3. Input Capture Mode

The device can precisely measure external frequencies or time external events by placing the timer block, Tx, in the input capture mode.

In this mode, the timer Tx is constantly running at the fixed t_C rate. The two registers, RxA and RxB, act as capture registers. Each register acts in conjunction with a pin. The register RxA acts in conjunction with the TxA pin and the register RxB acts in conjunction with the TxB pin.

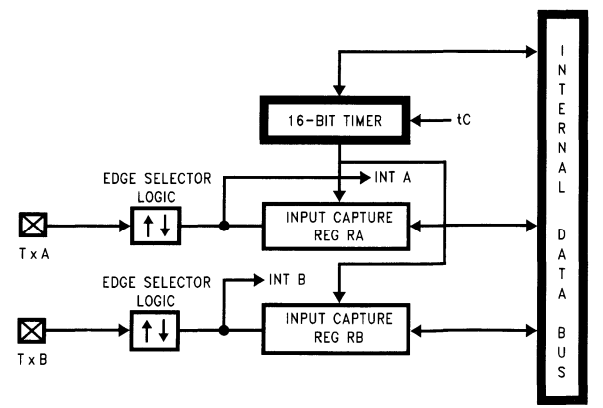
The timer value gets copied over into the register when a trigger event occurs on its corresponding pin. Control bits, TxC3, TxC2 and TxC1, allow the trigger events to be specified either as a positive or a negative edge. The trigger condition for each input pin can be specified independently.

The trigger conditions can also be programmed to generate interrupts. The occurrence of the specified trigger condition on the TxA and TxB pins will be respectively latched into the pending flags, TxPNDA and TxPNDB.

The control flag TxENA allows the interrupt on TxA to be either enabled or disabled. Setting the TxENA flag enables interrupts to be generated when the selected trigger condition occurs on the TxA pin. Similarly, the flag TxENB controls the interrupts from the TxB pin.

Underflows from the timer can also be programmed to generate interrupts. Underflows are latched into the timer TxC0 pending flag (the TxC0 control bit serves as the timer underflow interrupt pending flag in the Input Capture mode). Consequently, the TxC0 control bit should be reset when entering the Input Capture mode. The timer underflow interrupt is enabled with the TxENA control flag. When a TxA interrupt occurs in the Input Capture mode, the user must check both the TxPNDA and TxC0 pending flags in order to determine whether a TxA input capture or a timer underflow (or both) caused the interrupt.

Figure 9 shows a block diagram of the timer in Input Capture mode.



TL/DD/12065–10

FIGURE 9. Timer in Input Capture Mode

Timers (Continued)

TIMER CONTROL FLAGS

The timers T1 and T2 have identical control structures. The control bits and their functions are summarized below.

TxC0 Timer Start/Stop control in Modes 1 and 2 (Processor Independent PWM and External Event Counter), where 1 = Start, 0 = Stop Timer Underflow Interrupt Pending Flag in Mode 3 (Input Capture)

TxPNDA Timer Interrupt Pending Flag

TxPNDB Timer Interrupt Pending Flag

TxENA Timer Interrupt Enable Flag

TxENB Timer Interrupt Enable Flag

1 = Timer Interrupt Enabled

0 = Timer Interrupt Disabled

TxC3 Timer mode control

TxC2 Timer mode control

TxC1 Timer mode control

The timer mode control bits (TxC3, TxC2 and TxC1) are detailed below:

Capture Timer

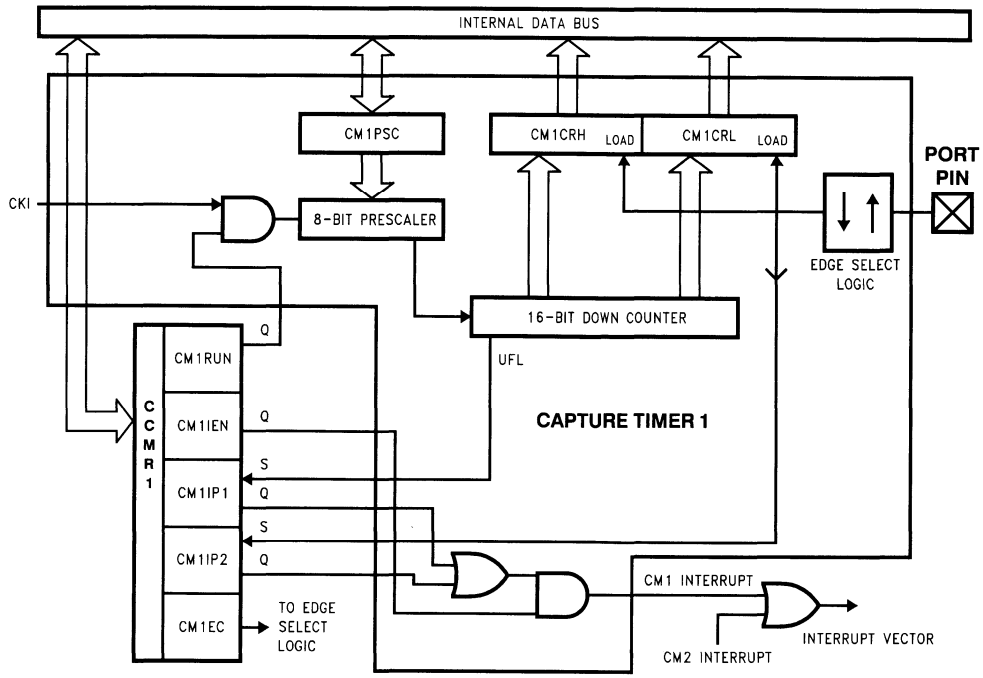
This device contains two independent capture timers, Capture Timer 1 and Capture Timer 2. Each capture timer contains an 8-bit programmable prescaler register, a 16-bit down counter, a 16-bit input capture register, and capture edge select logic. The 16-bit down counter is clocked at a specific frequency determined by the value loaded into the prnscaler register. A selected positive or negative edge transition on the capture input causes the contents of the down counter to be latched into the capture register. The values captured in the registers reflect the elapsed time between two positive or two negative transitions on the capture input. The time between a positive and negative edge (a pulse width) may be measured if the selected capture edge is switched after the first edge is captured. Each capture timer may be stopped/started under software control, and each capture timer may be configured to interrupt the microcontroller on an underflow or input capture.

Figure 10 shows the capture timer 1 block diagram.

TABLE II. Timer Mode Control

TxC3	TxC2	TxC1	Timer Mode	Interrupt A Source	Interrupt B Source	Timer Counts On
0	0	0	MODE 2 (External Event Counter)	Timer Underflow	Positive TxB Edge	TxA Positive Edge
0	0	1	MODE 2 (External Event Counter)	Timer Underflow	Positive TxB Edge	TxA Negative Edge
1	0	1	MODE 1 (PWM) TxA Toggle	Autoreload RA	Autoreload RB	t_C
1	0	0	MODE 1 (PWM) No TxA Toggle	Autoreload RA	Autoreload RB	t_C
0	1	0	MODE 3 (Capture) Captures: TxA Positive Edge TxB Positive Edge	Positive TxA Edge or Timer Underflow	Positive TxB Edge	t_C
1	1	0	MODE 3 (Capture) Captures: TxA Positive Edge TxB Negative Edge	Positive TxA Edge or Timer Underflow	Negative TxB Edge	t_C
0	1	1	MODE 3 (Capture) Captures: TxA Negative Edge TxB Positive Edge	Negative TxA Edge or Timer Underflow	Positive TxB Edge	t_C
1	1	1	MODE 3 (Capture) Captures: TxA Negative Edge TxB Negative Edge	Negative TxA Edge or Timer Underflow	Negative TxB Edge	t_C

Timers (Continued)



TL/DD/12065–11

FIGURE 10. Capture Timer 1 Block Diagram

The registers shown in the block diagram include those for Capture Timer 1 (CM1), as well as, the capture timer 1 control register. These registers are read/writable (with the exception of the capture registers, which are read-only) and may be accessed through the data memory address/data bus. The registers are designated as:

CM1PSC Capture Timer 1 Prescaler (8-bit)

CM1CRL Capture Timer 1 Capture Register (Low-byte), read-only

CM1CRH Capture Timer 1 Capture Register (High-byte), read-only

CM2PSC Capture Timer 2 Prescaler (8-bit)

CM2CRL Capture Timer 2 Capture Register (Low-byte), read-only

CM2CRH Capture Timer 2 Capture Register (High-byte), read-only

CCMR1 Control Register for Capture Timer 1

CCMR2 Control Register for Capture Timer 2

CONTROL REGISTER BITS

The control bits for Capture Timer 1 (CM1) and Capture Timer 2 (CM2) are contained in CCMR1 and CCMR2.

The CCMR1 Register Bits are:

CM1RUN CM1 start/stop control bit (1 = start; 0 = stop)

CM1IEN CM1 interrupt enable control bit (1 = enable IRQ)

CM1IP1 CM1 interrupt pending bit 1 (1 = CM1 underflowed)

CM1IP2 CM1 interrupt pending bit 2 (1 = CM1 captured)

CM1EC Select the active edge for capture on CM1 (0 = rising, 1 = falling)

CM1TM CM1 test mode control bit (1 = special test path in test mode. This bit is reserved during normal operation, and must never be set to one.)

CM1 TM	un- used	un- used	CM1 EC	CM1 IP2	CM1 IP1	CM1 IEN	CM1 RUN

Bit 7 Bit 0

All interrupt pending bits must be reset by software.

Timers (Continued)

The CCMR2 Register Bits are:

CM2RUN CM2 start/stop control bit (1 start; 0 = stop)

CM2IEN CM2 interrupt enable control bit (1 = enable IRQ)

CM2IP1 CM2 interrupt pending bit 1 (1 = CM2 underflowed)

CM2IP2 CM2 interrupt pending bit 2 (1 = CM2 captured)

CM2EC Select the active edge for capture on CM2 (0 = rising, 1 = falling)

CM2TM CM2 test mode control bit (1 = special test path in test mode. This bit is reserved during normal operation, and must never be set to one.)

CM2 TM	un- used	un- used	CM2 EC	CM2 IP2	CM2 IP1	CM2 IEN	CM RUN

Bit 7 Bit 0

All interrupt pending bits must be reset by software.

FUNCTIONAL DESCRIPTION

The capture timer is used to determine the time between events, where an event is simply a selected edge transition on the capture input. The resolution of the time measurement is dependent on the frequency at which the down counter is clocked. The value loaded into the prescaler controls this frequency.

The prescaler is clocked by CKI, while the down counter is clocked on every underflow of the prescaler. This means the prescaler simply divides the CKI clock before it is fed into the down counter. The prescaler register must be loaded with a value corresponding to the CKI divisor needed to produce the desired down counter clock. The appropriate prescaler value can be determined using the following equation:

Down Counter Clock Frequency = CKI/(CMxPSC + 1)

The capture input signal is set up by configuring the port pin associated with the capture timer as an input. The edge select bit for the capture input is then set or reset according to the desired transition. If the pin is configured as an input, the appropriate external transition will cause a capture. If the pin is configured as an output, toggling the data register bit will cause a capture. If interrupts are used, the capture timer interrupt pending bits are cleared and the capture timer interrupt enable bit is set. Both interrupt sources, down counter underflow and input capture edge, are enabled/disabled with the same CMxIEN bit. The GIE bit must also be set to enable interrupts. The interrupt signals from the two capture timers are gated to a single 16-bit interrupt vector located at addresses 0xE6 and 0xE7.

The capture timer is started by writing a "1" to the capture timer start/stop bit. Setting this bit also enables the port pin to be the capture input to the capture timer. The internal prescaler is loaded with the contents of the prescaler register, and begins counting down. Setting the start/stop bit also loads the down counter with 0FFFF Hex. The prescaler is clocked by CKI. An underflow of the prescaler decrements the 16-bit down counter, and reloads the value from the prescaler register into the prescaler. Each additional underflow of the prescaler decrements the down counter, and reloads the prescaler from the prescaler register.

If a selected edge transition on the input capture pin occurs, the contents of the down counter are immediately latched into the capture register, the down counter is re-initialized to 0FFFF Hex, and the capture input pending flag is set. The prescaler counter is not loaded. (In order for an input transition to be guaranteed recognized, the signal on the capture input pin must have a low pulse width and a high pulse width of at least one CKI period.) If interrupts are enabled, the capture timer generates an interrupt. The prescaler and down counter continue to operate until a reset condition occurs or the capture timer start/stop bit is reset. The user must process capture interrupts faster than the capture input frequency, otherwise input captures may be lost or erroneous values may be read.

If the down counter underflows (changes state from 0000 to FFFF) before a capture input is detected, the underflow interrupt pending flag is set. If interrupts are enabled, the capture timer generates an interrupt.

The capture timer may be stopped at any time under software control by resetting the capture timer start/stop bit. A capture may occur before the start/stop bit is physically cleared, due to the fully asynchronous nature of the input capture signal. The user must ensure that the software handles this situation correctly. If the user wishes to process this capture and interrupts are being used, the capture timer interrupts should not be disabled prior to stopping the timer. If interrupts are not being used, the user should poll the capture timer pending bits after stopping the timer. If the user wishes to ignore this capture and interrupts are being used, the capture timer interrupt service routine should check that the timer is still running prior to processing capture interrupts. If the user is polling the pending flags, these flags should be cleared after the timer is stopped. The contents of the prescaler and down counter remain unchanged while the capture timer is stopped. The capture edge detect logic is disabled, and no capture takes place even if an external capture signal occurs. The capture timer may be restarted under software control by writing a "1" to the start/stop bit. This causes the prescaler and down counter to be re-initialized. The prescaler is loaded from the prescaler register, and the down counter is loaded with 0FFFF Hex.

RESET STATE

A reset signal applied to the counter block during normal operation has the following effects:

- Clear CCMR1 register
- Clear CCMR2 register
- CM1PSC, CMICRL, CM1CRH, CM2PSC, CM2CRL and CM2CRH are unaffected. (At power-on, the contents of these registers are undefined.)

The bi-directional port pins are initialized during reset as HI-Z inputs. Setting the start/stop bits connects the pins to the capture timers.

Timers (Continued)

INITIALIZATION

The user should perform the following initialization prior to starting the capture timer:

1. Reset the CMxRUN bit
2. Configure the corresponding Port bits as inputs
3. Set the edge control bits CMxEC
4. Reset CMxIP1 (CMxIP1 = 0)
5. Reset CMxIP2 (CMxIP2 = 0)
6. Load the 8-bit prescaler register CMxPSC with the desired value (from 0 to 255)
7. Set CMxIEN (if interrupts are to be used)
8. Set the Global Interrupt Enable (GIE) bit (if interrupts are to be used)
9. Set CMxRUN bit to start the capture timer

WARNING

In order to avoid erroneous interrupts, the capture timer interrupts must be disabled prior to setting/resetting the capture edge control bits (CMxEC). In addition, after selecting the interrupt edge, the pending flags must be reset before the capture interrupts are enabled or re-enabled. If the initialization sequence outlined above is followed each time the user alters the edge control bits, the user is guaranteed to avoid erroneous interrupts.

Pulse Train Generators

This device contains four independent pulse train generators. Each individual generator is controlled by a corresponding 16-bit counter. Each counter has a 16-bit prescaler and a 16-bit count register. Each counter may be configured to output a selected number of 50% duty cycle pulses. The contents of the prescaler determine the width of the output pulses, and the value of the count register determines the number of pulses. Each counter may be stopped/started under software control, and each counter may be configured to interrupt the microcontroller on an underflow.

Figure 11 shows the pulse train generator 1 block diagram.

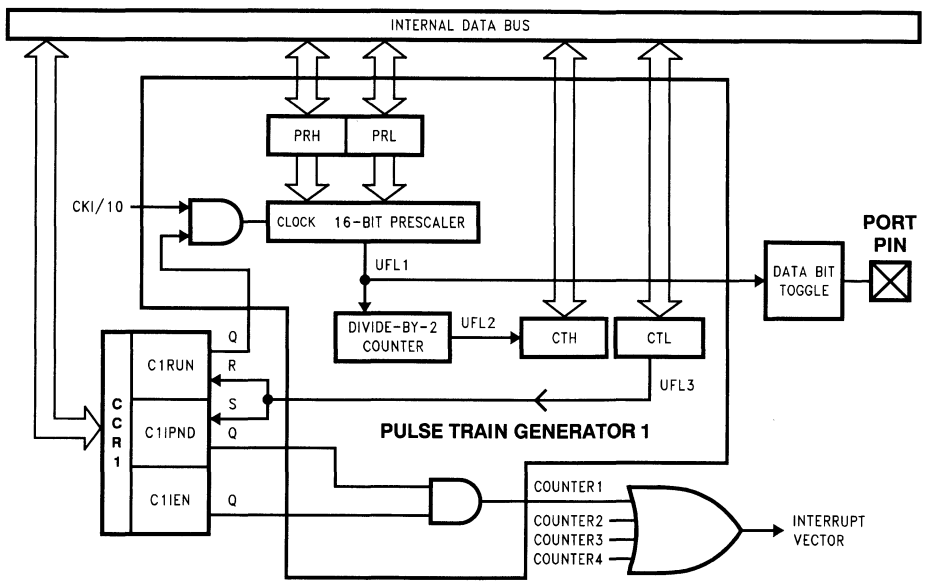


FIGURE 11. Pulse Train Generator 1 Block Diagram

Pulse Train Generators (Continued)

The four 8-bit registers shown in each individual counter in the block diagram constitute a 16-bit prescaler and a 16-bit count register. These registers are all read/writable and may be accessed through the data memory address/data bus. The registers are designated as:

CxPRL Low-byte of the Prescaler

CxPRH High-byte of the Prescaler

CxCTL Low-byte of the Count Register

CxCTH High-byte of the Count Register

CONTROL REGISTER BITS

The control bits for Counter 1 and Counter 2 are contained in the CCR1 register. The CCR1 Register bits are:

C1RUN COUNTER1 start/stop control bit (1 = start; 0 = stop)

C1IEN COUNTER1 interrupt enable control bit (1 = enable IRQ)

C1IPND COUNTER1 interrupt pending bit (1 counter 1 underflowed)

C1TM COUNTER1 test mode control bit (1 = special test path in test mode. This bit is reserved during normal operation, and must never be set to one.)

C2RUN COUNTER2 start/stop control bit (1 = start; 0 = stop)

C2IEN COUNTER2 interrupt enable control bit (1 = enable IRQ)

C2IPND COUNTER2 interrupt pending bit (1 = counter 2 underflowed)

C2TM COUNTER2 test mode control bit (1 = special test path. This bit is reserved during normal operation, and must never be set to one.)

All interrupt pending bits must be reset by software.

C2TM	C2 IPND	C2 IEN	C2 RUN	C1TM	C1 IPND	C1 IEN	C1 RUN

Bit 7 Bit 0

The control bits for Counter 3 and Counter 4 are contained in the CCR2 register. The CCR2 Register bits are:

C3RUN COUNTER3 start stop control bit (1 = start; 0 = stop)

C3IEN COUNTER3 interrupt enable control bit (1 = enable IRQ)

C3IPND COUNTER3 interrupt pending Bit (1 = counter 3 underflowed)

C3TM COUNTER3 test mode control bit (1 = special test path. This bit is reserved during normal operation, and must never be set to one.)

C4RUN COUNTER4 start/stop control bit (1 = start; 0 = stop)

C4IEN COUNTER4 interrupt enable control bit (1 = enable IRQ)

C4IPND COUNTER4 interrupt pending bit (1 = counter 4 underflowed

C4TM COUNTER4 test mode control bit (1 = special test path. This bit is reserved during normal operation, and must never be set to one.)

C4TM	C4 IPND	C4 IEN	C4 RUN	C3TM	C3 IPND	C3 IEN	C3 RUN

Bit 7 Bit 0

All interrupt pending bits must be reset by software.

FUNCTIONAL DESCRIPTION

The pulse train generator may be used to produce a series of output pulses of a given width. The high/low time of a pulse is determined by the contents of the prescaler. The number of pulses in a series is determined by the contents of the count register.

The prescaler is loaded with a value corresponding to the desired width of the output pulse (t_w). The high time and low time of the output signal are each equal to t_w, therefore the output signal produced has a 50% duty cycle and a period equal to 2 * t_w. The appropriate prescaler value can be determined using the following equation:

$$t_w = [(PRH * 256) + PRL + 1] * t_c$$

Since PRH and PRL are both 8-bit registers, this equation allows a maximum t_w of 65536 t_c and a minimum t_w of one t_c. The internal prescaler is automatically loaded from PRH and PRL when the counter start/stop bit is set.

The count register is loaded with a value corresponding to the desired number of output pulses. The appropriate count value is calculated with the following equation:

$$\text{Number of Pulses} = CTH * 256 + CTL + 1$$

The port pin associated with the counter OUT signal is configured in software as an output, and preset to the desired start logic level. If interrupts are to be used, the counter interrupt pending bit is cleared and the interrupt enable bit is set. The GIE bit must also be set to enable interrupts. The interrupt signals from the four counters are gated to a single interrupt vector located at addresses 0xF0–0xF1.

The counter is started by writing a "1" to the counter start/stop bit. This resets the divide-by-2 counter which produces the clock signal for the counter register from the prescaler underflow (See *Figure 11*). It also reloads the internal prescaler and starts the prescaler counting down on the next rising edge of t_c. The prescaler is clocked on the rising edge of t_c to ensure synchronization. Each subsequent rising edge of t_c causes the prescaler to be decremented. When the prescaler underflows, UFL1 is generated (see *Figure 12*). This signal causes the port pin to toggle. In addition, the internal prescaler is reloaded with the value from the PRH and PRL registers. Each additional underflow of the prescaler causes the port pin to toggle and reloads the internal prescaler.

Every second underflow of the prescaler generates the signal UFL2. (UFL2 occurs at half the frequency of UFL1, or once per output pulse.) This signal, UFL2, decrements the count register. Therefore, the count registers are decremented once per output pulse.

Pulse Train Generators (Continued)

The underflow of the counter register produces the signal UFL3. This signal stops the counter by resetting the counter start/stop bit, and sets the counter interrupt pending flag. If the counter interrupt is enabled, an interrupt occurs.

The counter may be stopped at any time under software control by resetting the counter start/stop bit. The contents of the count register and the output on the associated port pin are frozen. The counter may be restarted under software control by setting the start/stop bit. The internal prescaler is automatically reloaded from PRH and PRL when the counter start/stop bit is set, therefore a full width pulse will be generated before the output is toggled. The user may also choose to alter the logic level on the port pin before restarting. This is done by initializing the associated port pin data register bit. A counter underflow may occur before the start/stop bit is physically cleared by software. The user must ensure that the software handles this situation correctly. If the user wishes to process this underflow and interrupts are being used, the counter interrupts should not be disabled prior to stopping the timer. If interrupts are not being used, the user should poll the counter pending bits after stopping the timer. If the user wishes to ignore this underflow and interrupts are being used, the counter interrupt should be disabled prior to stopping the timer. If the user is polling the pending flags, these flags should be cleared after the timer is stopped.

If the default level of the output pin is high (associated port data register bit is set to "1") and the counter is stopped during a low level, the low level becomes the default level. The software must reinitialize the port pin to a high level before restarting if necessary. The programmer may also have to adjust the counter value (See *Figure 12*).

RESET STATE

A reset signal applied to the pulse train generator block during normal operation has the following effects:

- Counting stops immediately
- Interrupt enable bit is reset to zero
- Counter start/stop bit is reset to zero
- Interrupt pending bit is reset to zero
- Test mode control bit is reset to zero
- PRL, PRH, CTL and CTH are unaffected (At power-on reset, the contents of the prescaler and count register are undefined.)
- Divide-by-2 counter is reset
- The bi-directional port pins are initialized during reset as HI-Z inputs. The appropriate bits must be initialized as outputs, in order to route the Counter OUT signals to the port pins.

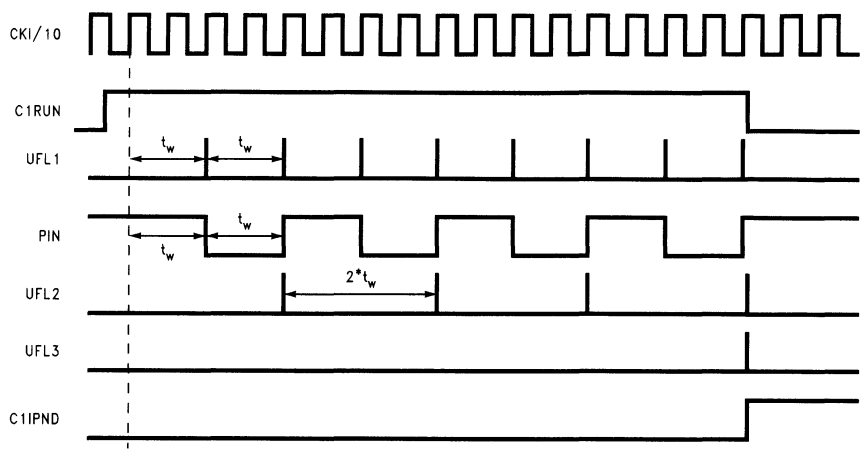
INITIALIZATION

The user should perform the following initialization prior to starting the counter:

1. Load PRL register
2. Load PRH register
3. Load CTL register
4. Load CTH register
5. Reset CxIPND bit
6. Set CxIEN (if interrupt is to be used)
7. Configure the associated port bit as an output (if OUT is to be used)
8. Set the Global Interrupt Enable (GIE) bit (if interrupt is to be used)
9. Set CxRUN bit to start counter

Multiply/Divide

This device contains a multiply/divide block. This block supports a 1 byte x 2 bytes (3 bytes result) multiply or a 3 bytes/2 bytes (2 bytes result) divide operation. The multiply or divide operation is executed by setting control bits located in the multiply/divide control register. The multiply or divide operands must be placed into the appropriate memory mapped locations before the operation is initiated.



TL/DD/12065–13

FIGURE 12. Timing Diagram for PRL = 1, PRH = 0, CTL = 3, CTH = 0

Multiply/Divide (Continued)

TABLE III. Multiply/Divide Registers

Register Name (Address)	Multiplication Assignment		Division Assignment	
	Before Operation	After Operation	Before Operation	After Operation
MDR1 (xx98)	Unused	Unchanged	Low byte of dividend	Low byte of result
MDR2 (xx99)	Multiplier	Low byte of result	Middle byte of dividend	High byte of result
MDR3 (xx9A)		Middle byte of result	High byte of dividend	Undefined
MDR4 (xx9B)	Low byte of multiplicand	High byte of result	Low byte of divisor	Low byte of divisor
MDR5 (xx9C)	High byte of multiplicand	Unchanged	High byte of divisor	High byte of divisor

CONTROL REGISTER BITS

The Multiply/Divide control register (MDCR) is located at address xx9D. It has the following bit assignments:

MULT Start Multiplication Operation (1 = start)

DIV Start Division Operation (1 = start)

DIVOVF Division Overflow (if the result of a division is greater than 16 bits or the user attempted to divide by zero; 1 = error)

Rsvd	Rsvd	Rsvd	Rsvd	Rsvd	DIV OVF	DIV	MULT

Bit 7 Bit 0

After the appropriate MDR registers are loaded, the MULT and DIV start bits are set by the user to start a multiply or divide operation. The division operation has priority, if both bits are set simultaneously. The MULT and DIV bits are BOTH automatically cleared by hardware at the end of a divide or multiply operation. Each division operation causes the DIVOVF flag to be set/reset as appropriate. The DIVOVF flag is cleared following a multiplication operation. DIVOVF is a read-only bit. The MULT and DIV bits are read/writable. Bits 3-7 in MDCR should not be used, as the MULT and DIV operations will change their values.

MULTIPLY/DIVIDE OPERATION

For the multiply operation, the multiplicand is placed at addresses xx9B and xx9C. The multiplier is placed at address xx99. For the divide operation, the dividend is placed at addresses xx98 to xx9A and the divisor is placed at addresses xx9B to xx9C. In both operations, all operands are interpreted as unsigned values. The divide or multiply operation is started by setting the appropriate MDCR bit. If both the MULT and DIV bits are set, the microcontroller performs a divide operation. (The user is not required to read or clear the DIVOVF error bit prior to beginning a new multiply/divide operation. This bit is ignored during subsequent operations. However, the next divide operation will overwrite the error flag as appropriate, and the next multiply operation will clear it.)

The multiply operation requires 1 instruction cycle to complete. The divide operation requires 2 instruction cycles to complete. A divide by zero or a division which produces an overflow requires only 1 instruction cycle to execute. The MDR1 through MDR5 registers and the MDCR register can not be read from or written to during a multiply or divide operation. Any attempt to write into these registers will be ignored. Any attempt to read these registers will return undefined data.

The result of a multiply is placed in addresses xx99-xx9B. The result of a divide is placed in addresses xx98-xx99. If a division by zero is attempted or if the resulting quotient of a divide operation is more than 16 bits long, then the DIVOVF bit is set in the multiply/divide control register. The dividend and the divisor are left unchanged. The divide operation always causes the DIVOVF flag to be set or reset as appropriate. The DIVOVF flag is cleared following a multiply operation.

RESET STATE

A reset signal applied to the device during normal operation has the following affects:

MDCR is cleared, and any operation in progress is stopped.

MDR1 through MDR5 are undefined.

Power Save Modes

The device offers the user two power save modes of operation: HALT and IDLE. In the HALT mode, all microcontroller activities are stopped. In the IDLE mode, the on-board oscillator circuitry and timer T0 are active but all other microcontroller activities are stopped. In either mode, all on-board RAM, registers, I/O states, and timers (with the exception of T0) are unaltered.

HALT MODE

The device can be placed in the HALT mode by writing a "1" to the HALT flag (G7 data bit). All microcontroller activities, including the clock and timers, are stopped. In the HALT mode, the power requirements of the device are minimal and the applied voltage (V_{CC}) may be decreased to V_r (V_r = 2.0V) without altering the state of the machine.

The device supports two different ways of exiting the HALT mode. The first method of exiting the HALT mode is with the Multi-Input Wakeup feature on the L port. The second method of exiting the HALT mode is by pulling the $\overline{\text{RESET}}$ pin low.

Since a crystal or ceramic resonator may be selected as the oscillator, the Wakeup signal is not allowed to start the chip running immediately since crystal oscillators and ceramic resonators have a delayed start up time to reach full amplitude and frequency stability. The IDLE timer is used to generate a fixed delay to ensure that the oscillator has indeed stabilized before allowing instruction execution. In this case, upon detecting a valid Wakeup signal, only the oscillator circuitry is enabled. The IDLE timer is loaded with a value of 256 and is clocked with the t_c instruction cycle clock. The t_c clock is derived by dividing the oscillator clock down by a

Power Save Modes (Continued)

factor of 10. The Schmitt trigger following the CKI inverter on the chip ensures that the IDLE timer is clocked only when the oscillator has a sufficiently large amplitude to meet the Schmitt trigger specifications. This Schmitt trigger is not part of the oscillator closed loop. The startup timeout from the IDLE timer enables the clock signals to be routed to the rest of the chip.

The devices have two mask options associated with the HALT mode. The first mask option enables the HALT mode feature, while the second mask option disables the HALT mode. With the HALT mode enable mask option, the device will enter and exit the HALT mode as described above. With the HALT disable mask option, the device cannot be placed in the HALT mode (writing a "1" to the HALT flag will have no effect, the HALT flag will remain "0").

IDLE MODE

The device is placed in the IDLE mode by writing a "1" to the IDLE flag (G6 data bit). In this mode, all activities, except the associated on-board oscillator circuitry and the IDLE Timer T0, are stopped.

As with the HALT mode, the device can be returned to normal operation with a reset, or with a Multi-Input Wake up from the L Port. Alternately, the microcontroller resumes normal operation from the IDLE mode when the thirteenth bit (representing 4.096 ms at internal clock frequency of 10 MHz, t_c = 1 μs) of the IDLE Timer toggles.

This toggle condition of the thirteenth bit of the IDLE Timer T0 is latched into the T0PND pending flag.

The user has the option of being interrupted with a transition on the thirteenth bit of the IDLE Timer T0. The interrupt can be enabled or disabled via the T0EN control bit. Setting the T0EN flag enables the interrupt and vice versa.

The user can enter the IDLE mode with the Timer T0 interrupt enabled. In this case, when the T0PND bit gets set, the device will first execute the Timer T0 interrupt service routine and then return to the instruction following the "Enter Idle Mode" instruction.

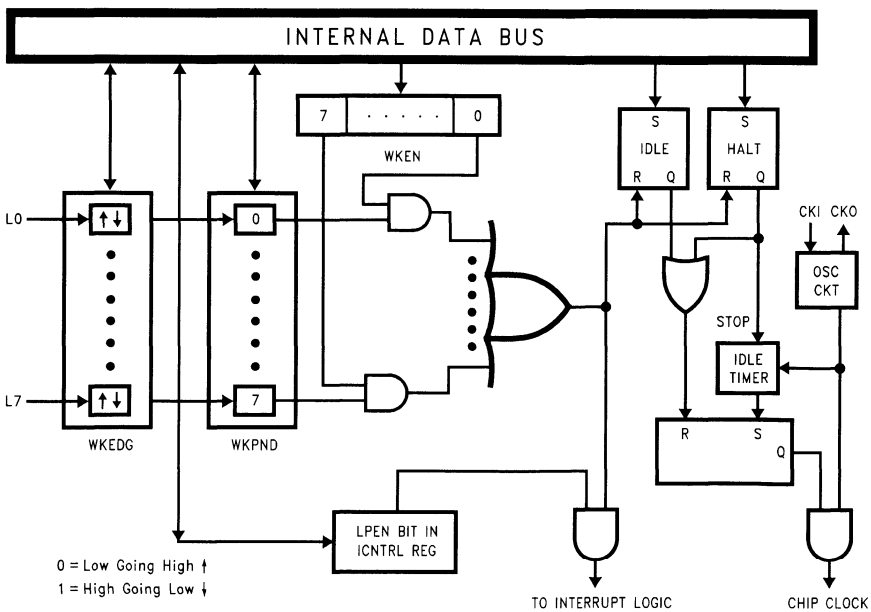
Alternatively, the user can enter the IDLE mode with the IDLE Timer T0 interrupt disabled. In this case, the device will resume normal operation with the instruction immediately following the "Enter IDLE Mode" instruction.

Note: It is necessary to program two NOP instructions following both the set HALT mode and set IDLE mode instructions. These NOP instructions are necessary to allow clock resynchronization following the HALT or IDLE modes.

Multi-Input Wakeup

The Multi-Input Wake Up feature is used to return (wake up) the device from either the HALT or IDLE modes. Alternately Multi-Input Wake Up/Interrupt feature may also be used to generate up to 8 edge selectable external interrupts.

Figure 13 shows the Multi-Input Wake Up logic.



TL/DD/12065–15

FIGURE 13. Multi-Input Wake Up Logic

Multi-Input Wakeup (Continued)

The Multi-Input Wake Up feature utilizes the L Port. The user selects which particular L port bit (or combination of L Port bits) will cause the device to exit the HALT or IDLE modes. The selection is done through the register WKEN. The register WKEN is an 8-bit read/write register, which contains a control bit for every L port bit. Setting a particular WKEN bit enables a Wake Up from the associated L port pin.

The user can select whether the trigger condition on the selected L Port pin is going to be either a positive edge (low to high transition) or a negative edge (high to low transition). This selection is made via the register WKEDG, which is an 8-bit control register with a bit assigned to each L Port pin. Setting the control bit will select the trigger condition to be a negative edge on that particular L Port pin. Resetting the bit selects the trigger condition to be a positive edge. Changing an edge select entails several steps in order to avoid a Wake Up condition as a result of the edge change. First, the associated WKEN bit should be reset, followed by the edge select change in WKEDG. Next, the associated WKPND bit should be cleared, followed by the associated WKEN bit being reenabled.

An example may serve to clarify this procedure. Suppose we wish to change the edge select from positive (low going high) to negative (high going low) for L Port bit 5, where bit 5 has previously been enabled for an input interrupt. The program would be as follows:

```
RBIT 5, WKEN
SBIT 5, WKEDG
RBIT 5, WKPND
SBIT 5, WKEN
```

If the L port bits have been used as outputs and then changed to inputs with Multi-Input Wake Up/Interrupt, a safety procedure should also be followed to avoid wakeup conditions. After the selected L port bits have been changed from output to input but before the associated WKEN bits are enabled, the associated edge select bits in WKEDG should be set or reset for the desired edge selects, followed by the associated WKPND bits being cleared,

This same procedure should be used following reset, since the L port inputs are left floating as a result of reset.

The occurrence of the selected trigger condition for Multi-Input Wake Up is latched into a pending register called WKPND. The respective bits of the WKPND register will be set on the occurrence of the selected trigger edge on the corresponding Port L pin. The user has the responsibility of clearing these pending flags. Since WKPND is a pending register for the occurrence of selected wake up conditions, the device will not enter the HALT mode if any Wake Up bit is both enabled and pending. Consequently, the user must clear the pending flags before attempting to enter the HALT mode.

WKEN, WKPND and WKEDG are all read/write registers, and are cleared at reset.

PORT L INTERRUPTS

Port L provides the user with an additional eight fully selectable, edge sensitive interrupts which are all vectored into the same service subroutine.

The interrupt from Port L shares logic with the wake up circuitry. The register WKEN allows interrupts from Port L to be individually enabled or disabled. The register WKEDG specifies the trigger condition to be either a positive or a negative edge. Finally, the register WKPND latches in the pending trigger conditions.

The GIE (Global Interrupt Enable) bit enables the interrupt function.

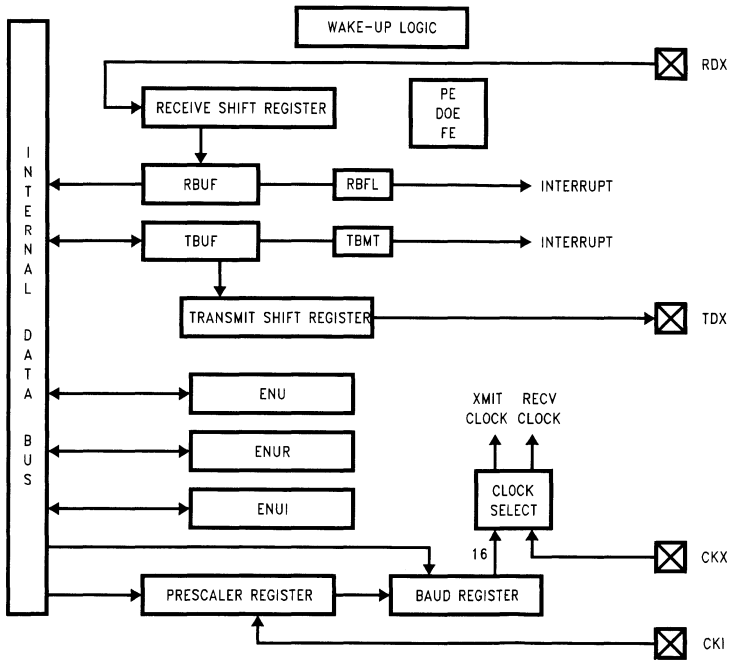
A control flag, LPEN, functions as a global interrupt enable for Port L interrupts. Setting the LPEN flag will enable interrupts and vice versa. A separate global pending flag is not needed since the register WKPND is adequate.

Since Port L is also used for waking the device out of the HALT or IDLE modes, the user can elect to exit the HALT or IDLE modes either with or without the interrupt enabled. If he elects to disable the interrupt, then the device will restart execution from the instruction immediately following the instruction that placed the microcontroller in the HALT or IDLE modes. In the other case, the device will first execute the interrupt service routine and then revert to normal operation. (See HALT MODE for clock option wake up information.)

UART

The device contains a full-duplex software programmable UART. The UART (*Figure 14*) consists of a transmit shift register, a receive shift register and seven addressable registers, as follows: a transmit buffer register (TBUF), a receiver buffer register (RBUF), a UART control and status register (ENU), a UART receive control and status register (ENUR), a UART interrupt and clock source register (ENUI), a prescaler select register (PSR) and baud (BAUD) register. The ENU register contains flags for transmit and receive functions; this register also determines the length of the data frame (7, 8 or 9 bits), the value of the ninth bit in transmission, and parity selection bits. The ENUR register flags framing, data overrun and parity errors while the UART is receiving.

Other functions of the ENUR register include saving the ninth bit received in the data frame, enabling or disabling the UART's attention mode of operation and providing additional receiver/transmitter status information via RCVG and XMTG bits. The determination of an internal or external clock source is done by the ENUI register, as well as selecting the number of stop bits and enabling or disabling transmit and receive interrupts. A control flag in this register can also select the UART mode of operation: asynchronous or synchronous.



TL/DD/12065–16

FIGURE 14. UART Block Diagram

UART (Continued)

UART CONTROL AND STATUS REGISTERS

The operation of the UART is programmed through three registers: ENU, ENUR and ENUI. The function of the individual bits in these registers is as follows:

ENU-UART Control and Status Register (Address at 0BA)

PEN	PSEL1	XBIT9/ PSEL0	CHL1	CHL0	ERR	RBFL	TBMT
0RW	0RW	0RW	0RW	0RW	0R	0R	IR

Bit 7 Bit 0

ENUR-UART Receive Control and Status Register (Address at 0BB)

DOE	FE	PE	SPARE	RBIT9	ATTN	XMTG	RCVG
0RD	0RD	0RD	0RW*	0R	0RW	0R	0R

Bit 7 Bit 0

ENUI-UART Interrupt and Clock Source Register (Address at 0BC)

STP2	STP78	ETDX	SSEL	XRCLK	XTCLK	ERI	ETI
0RW	0RW	0RW	0RW	0RW	0RW	0RW	0RW

Bit 7 Bit 0

* Bit is not used.

0 Bit is cleared on reset.

1 Bit is set to one on reset.

R Bit is read-only; it cannot be written by software.

RW Bit is read/write.

D Bit is cleared on read; when read by software as a one, it is cleared automatically. Writing to the bit does not affect its state.

DESCRIPTION OF UART REGISTER BITS

ENU—UART CONTROL AND STATUS REGISTER

TBMT: This bit is set when the UART transfers a byte of data from the TBUF register into the TSFT register for transmission. It is automatically reset when software writes into the TBUF register.

RBFL: This bit is set when the UART has received a complete character and has copied it into the RBUF register. It is automatically reset when software reads the character from RBUF.

ERR: This bit is a global UART error flag which gets set if any or a combination of the errors (DOE, FE, PE) occur.

CHL1, CHL0: These bits select the character frame format. Parity is not included and is generated/verified by hardware.

CHL1 = 0, CHL0 = 0 The frame contains eight data bits.

CHL1 = 0, CHL0 = 1 The frame continues seven data bits.

CHL1 = 1, CHL0 = 0 The frame continues nine data bits.

CHL1 = 1. CHL0 = 1 Loopback Mode selected. Transmitter output internally looped back to receiver input. Nine bit framing format is used.

XBIT9/PSEL0: Programs the ninth bit for transmission when the UART is operating with nine data bits per frame. For seven or eight data bits per frame, this bit in conjunction with PSEL1 selects parity.

PSEL1, PSEL0: Parity select bits.

PSEL1 = 0, PSEL0 = 0 Odd Parity (if Parity enabled)

PSEL1 = 0, PSEL1 = 1 Odd Parity (if Parity enabled)

PSEL1 = 1, PSEL0 = 0 Mark(1) (if Parity enabled)

PSEL1 = 1, PSEL1 = 1 Space(0) (if Parity enabled)

PEN: This bit enables/disables Parity (7- and 8-bit modes only).

PEN = 0 Parity disabled.

PEN = 1 Parity enabled.

ENUR—UART RECEIVE CONTROL AND STATUS REGISTER

RCVG: This bit is set high whenever a framing error occurs and goes low when RDX goes high.

XMTG: This bit is set to indicate that the UART is transmitting. It gets reset at the end of the last frame (end of last Stop bit).

ATTN: ATTENTION Mode is enabled while this bit is set. This bit is cleared automatically on receiving a character with data bit nine set.

RBIT9: Contains the ninth data bit received when the UART is operating with nine data bits per frame.

SPARE: Reserved for future use.

PE: Flags a Parity Error.

PE = 0 Indicates no Parity Error has been detected since the last time the ENUR register was read.

PE = 1 Indicates the occurrence of a Parity Error.

FE: Flags a Framing Error.

FE = 0 Indicates no Framing Error has been detected since the last time the ENUR register was read.

FE = 1 Indicates the occurrence of a Framing Error.

DOE: Flags a Data Overrun Error.

DOE = 0 Indicates no Data Overrun Error has been detected since the last time the ENUR register was read.

DOE = 1 Indicates the occurrence of a Data Overrun Error.

ENUI—UART INTERRUPT AND CLOCK SOURCE REGISTER

ETI: This bit enables/disables interrupt from the transmitter section.

ETI = 0 Interrupt from the transmitter is disabled.

ETI = 1 Interrupt from the transmitter is enabled.

ERI: This bit enables/disables interrupt from the receiver section.

UART (Continued)

ERI = 0 Interrupt from the receiver is disabled.

ERI = 1 Interrupt from the receiver is enabled.

XTCLK: This bit selects the clock source for the transmitter section.

XTCLK = 0 The clock source is selected through the PSR and BAUD registers.

XTCLK = 1 Signal on CKX (L1) pin is used as the clock.

XRCLK: This bit selects the clock source for the receiver section.

XRCLK = 0 The clock source is selected through the PSR and BAUD registers.

XRCLK = 1 Signal on CKX (L1) pin is used as the clock.

SSEL: UART mode select.

SSEL = 0 Asynchronous Mode.

SSEL = 1 Synchronous Mode.

ETDX: TDX (UART Transmit Pin) is the alternate function assigned to Port L pin L2; it is selected by setting ETDX bit. To simulate line break generation, software should reset ETDX bit and output logic zero to TDX pin through Port L data and configuration registers.

STP78: This bit is set to program the last Stop bit to be 7/8th of a bit in length.

STP2: This bit programs the number of Stop bits to be transmitted.

STP2 = 0 One Stop bit transmitted.

STP2 = 1 Two Stop bits transmitted.

Associated I/O Pins

Data is transmitted on the TDX pin and received on the RDX pin. TDX is the alternate function assigned to Port L pin L2; it is selected by setting ETDX (in the ENUI register) to one. RDX is an inherent function of Port L pin L3, requiring no setup.

The baud rate clock for the UART can be generated on-chip, or can be taken from an external source. Port L pin L1 (CKX) is the external clock I/O pin. The CKX pin can be either an input or an output, as determined by Port L Configuration and Data registers (Bit 1). As an input, it accepts a clock signal which may be selected to drive the transmitter and/or receiver. As an output, it presents the internal Baud Rate Generator output.

UART Operation

The UART has two modes of operation: asynchronous mode and synchronous mode.

ASYNCHRONOUS MODE

This mode is selected by resetting the SSEL (in the ENUI register) bit to zero. The input frequency to the UART is 16 times the baud rate.

The TSFT and TBUF registers double-buffer data for transmission. While TSFT is shifting out the current character on the TDX pin, the TBUF register may be loaded by software with the next byte to be transmitted. When TSFT finishes transmitting the current character the contents of TBUF are transferred to the TSFT register and the Transmit Buffer Empty Flag (TBMT in the ENU register) is set. The TBMT flag is automatically reset by the UART when software loads a new character into the TBUF register. There is also the XMTG bit which is set to indicate that the UART is transmitting. This bit gets reset at the end of the last frame (end of last Stop bit). TBUF is a read/write register.

The RSFT and RBUF registers double-buffer data being received. The UART receiver continually monitors the signal on the RDX pin for a low level to detect the beginning of a Start bit. Upon sensing this low level, it waits for half a bit time and samples again. If the RDX pin is still low, the receiver considers this to be a valid Start bit, and the remaining bits in the character frame are each sampled a single time, at the mid-bit position. Serial data input on the RDX pin is shifted into the RSFT register. Upon receiving the complete character, the contents of the RSFT register are copied into the RBUF register and the Received Buffer Full Flag (RBFL) is set. RBFL is automatically reset when software reads the character from the RBUF register. RBUF is a read only register. There is also the RCVG bit which is set high when a framing error occurs and goes low once RDX goes high. TBMT, XMTG, RBFL and RCVG are read only bits.

SYNCHRONOUS MODE

In this mode data is transferred synchronously with the clock. Data is transmitted on the rising edge and received on the falling edge of the synchronous clock.

This mode is selected by setting SSEL bit in the ENUI register. The input frequency to the UART is the same as the baud rate.

When an external clock input is selected at the CKX pin, data transmit and receive are performed synchronously with this clock through TDX/RDX pins.

If data transmit and receive are selected with the CKX pin as clock output, the device generates the synchronous clock output at the CKX pin. The internal baud rate generator is used to produce the synchronous clock. Data transmit and receive are performed synchronously with this clock.

FRAMING FORMATS

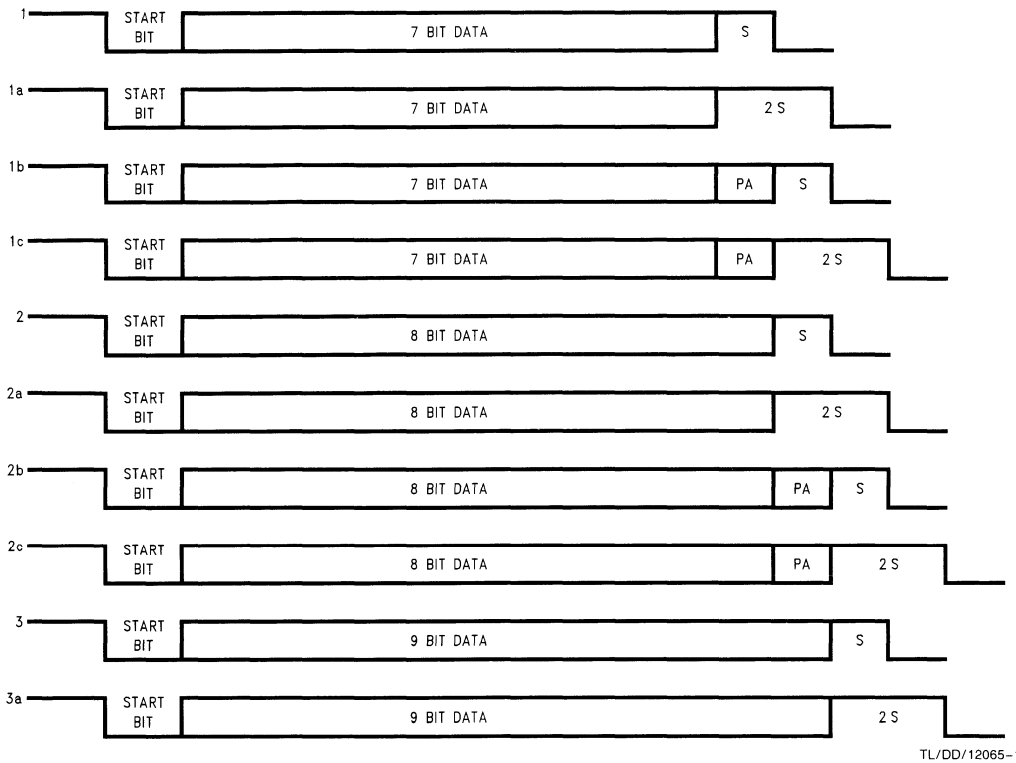
The UART supports several serial framing formats *(Figure 15)*. The format is selected using control bits in the ENU, ENUR and ENUI registers.

The first format (1,1a, 1b, 1c) for data transmission (CHL0 = 1, CHL1 = 0) consists of Start bit, seven Data bits (excluding parity) and 7/8, one or two Stop bits. In applications using parity, the parity bit is generated and verified by hardware.

The second format (CHL0 = 0, CHL1 = 0) consists of one Start bit, eight Data bits (excluding parity) and 7/8, one or two Stop bits. Parity bit is generated and verified by hardware.

The third format for transmission (CHL0 = 0, CHL1 = 1) consists of one Start bit, nine Data bits and 7/8, one or two Stop bits. This format also supports the UART "ATTENTION" feature. When operating in this format, all eight bits of TBUF and RBUF are used for data. The ninth data bit is transmitted and received using two bits in the ENU and ENUR registers, called XBIT9 and RBIT9. RBIT9 is a read only bit. Parity is not generated or verified in this mode.

UART Operation (Continued)



TL/DD/12065–17

FIGURE 15. Framing Formats

For any of the above framing formats, the last Stop bit can be programmed to be 7/8th of a bit in length. If two Stop bits are selected and the 7/8th bit is set (selected), the second Stop bit will be 7/8th of a bit in length.

The parity is enabled/disabled by PEN bit located in the ENU register. Parity is selected for 7- and 8-bit modes only. If parity is enabled (PEN = 1), the parity selection is then performed by PSEL0 and PSEL1 bits located in the ENU register.

Note that the XBIT9/PSEL0 bit located in the ENU register serves two mutually exclusive functions. This bit programs the ninth bit for transmission when the UART is operating with nine data bits per frame. There is no parity selection in this framing format. For other framing formats XBIT9 is not needed and the bit is PSEL0 used in conjunction with PSEL1 to select parity.

The frame formats for the receiver differ from the transmitter in the number of Stop bits required. The receiver only requires one Stop bit in a frame, regardless of the setting of the Stop bit selection bits in the control register. Note that an implicit assumption is made for full duplex UART operation that the framing formats are the same for the transmitter and receiver.

UART INTERRUPTS

The UART is capable of generating interrupts. Interrupts are generated on Receive Buffer Full and Transmit Buffer Empty. Both interrupts have individual interrupt vectors. Two bytes of program memory space are reserved for each interrupt vector. The two vectors are located at addresses 0xEC to 0xEF Hex in the program memory space. The interrupts can be individually enabled or disabled using Enable Transmit Interrupt (ETI) and Enable Receive Interrupt (ERI) bits in the ENUI register.

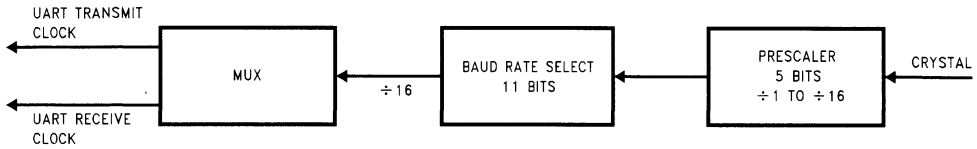
The interrupt from the Transmitter is set pending, and remains pending, as long as both the TBMT and ETI bits are set. To remove this interrupt, software must either clear the ETI bit or write to the TBUF register (thus clearing the TBMT bit).

The interrupt from the receiver is set pending, and remains pending, as long as both the RBFL and ERI bits are set. To remove this interrupt, software must either clear the ERI bit or read from the RBUF register (thus clearing the RBFL bit).

Baud Clock Generation

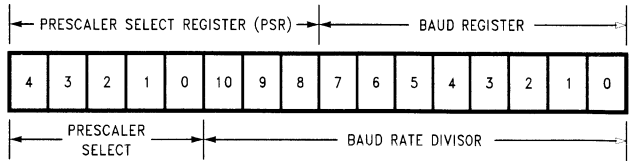
The clock inputs to the transmitter and receiver sections of the UART can be individually selected to come either from an external source at the CKX pin (port L, pin L1) or from a source selected in the PSR and BAUD registers. Internally, the basic baud clock is created from the oscillator frequency through a two-stage divider chain consisting of a 1–16 (increments of 0.5) prescaler and an 11-bit binary counter *(Figure 16)*. The divide factors are specified through two read/write registers shown in *Figure 17*. Note that the 11-bit Baud Rate Divisor spills over into the Prescaler Select Register (PSR). PSR is cleared upon reset.

Baud Clock Generation (Continued)



TL/DD/12065–18

FIGURE 16. UART BAUD Clock Generation



TL/DD/12065–19

FIGURE 17. UART BAUD Clock Divisor Registers

Baud Clock Generation (Continued)

As shown in Table V, a Prescaler Factor of 0 corresponds to NO CLOCK. This condition is the UART power down mode where the UART clock is turned off for power saving purpose. The user must also turn the UART clock off when a different baud rate is chosen.

The correspondences between the 5-bit Prescaler Select and Prescaler factors are shown in Table V. There are many ways to calculate the two divisor factors, but one particularly effective method would be to achieve a 1.8432 MHz frequency coming out of the first stage. The 1.8432 MHz prescaler output is then used to drive the software programmable baud rate counter to create a 16x clock for the following baud rates: 110, 134.5, 150, 300, 600, 1200, 1800, 2400, 3600, 4800, 7200, 9600, 19200 and 38400 (Table IV). Other baud rates may be created by using appropriate divisors. The 16x clock is then divided by 16 to provide the rate for the serial shift registers of the transmitter and receivers.

TABLE IV. Baud Rate Divisors (1.8432 MHz Prescaler Output)

Baud Rate	Baud Rate Divisor − 1 (N-1)
110 (110.03)	1046
134.5 (134.58)	855
150	767
300	383
600	191
1200	95
1800	63
2400	47
3600	31
4800	23
7200	15
9600	11
19200	5
38400	2

Note: The entries in Table IV assume a prescaler output of 1.8432 MHz. In asynchronous mode the baud rate could be as high as 625k.

TABLE V. Prescaler Factors

Prescaler Select	Prescaler Factor
00000	NO CLOCK
00001	1
00010	1.5
00011	2
00100	2.5
00101	3
00110	3.5
00111	4
01000	4.5
01001	5
01010	5.5
01011	6
01100	6.5
01101	7
01110	7.5
01111	8
10000	8.5
10001	9
10010	9.5
10011	10
10100	10.5
10101	11
10110	11.5
10111	12
11000	12.5
11001	13
11010	13.5
11011	14
11100	14.5
11101	15
11110	15.5
11111	16

Baud Clock Generation (Continued)

As an example, considering Asynchronous Mode and a CKI clock of 4.608 MHz, the prescaler factor selected is:

$$4.608/1.8432 = 2.5$$

The 2.5 entry is available in Table V. The 1.8432 MHz prescaler output is then used with proper Baud Rate Divisor (Table V) to obtain different baud rates. For a baud rate of 19200 e.g., the entry in Table IV is 5.

$$N - 1 = 5 \text{ (N} - 1 \text{ is the value from Table IV)}$$

$$N = 6 \text{ (N is the Baud Rate Divisor)}$$

$$\text{Baud Rate} = 1.8432 \text{ MHz}/(16 \times 6) = 19200$$

The divide by 16 is performed because in the asynchronous mode, the input frequency to the UART is 16 times the baud rate. The equation to calculate baud rates is given below.

The actual Baud Rate may be found from:

$$BR = Fc/(16 \times N \times P)$$

Where:

BR is the Baud Rate

Fc is the CKI frequency

N is the Baud Rate Divisor (Table IV).

P is the Prescaler Divide Factor selected by the value in the Prescaler Select Register (Table V)

Note: In the Synchronous Mode, the divisor 16 is replaced by two.

Example:

Asynchronous Mode:

Crystal Frequency = 5 MHz

Desired baud rate = 9600

Using the above equation N × P can be calculated first.

$$N \times P = (5 \times 106)/(16 \times 9600) = 32.552$$

Now 32.552 is divided by each Prescaler Factor (Table V) to obtain a value closest to an integer. This factor happens to be 6.5 (P = 6.5).

$$N = 32.552/6.5 = 5.008 \text{ (N} = 5)$$

The programmed value (from Table IV) should be 4 (N − 1).

Using the above values calculated for N and P:

$$BR = (5 \times 106)/(16 \times 5 \times 6.5) = 9615.384$$

$$\% \text{ error} = (9615.385 - 9600)/9600 = 0.16$$

Effect of HALT/IDLE

The UART logic is reinitialized when either the HALT or IDLE modes are entered. This reinitialization sets the TBMT flag and resets all read only bits in the UART control and status registers. Read/Write bits remain unchanged. The Transmit Buffer (TBUF) is not affected, but the Transmit Shift register (TSFT) bits are set to one. The receiver registers RBUF and RSFT are not affected.

The device will exit from the HALT/IDLE modes when the Start bit of a character is detected at the RDX (L3) pin. This feature is obtained by using the Multi-Input Wakeup scheme provided on the device.

Before entering the HALT or IDLE modes the user program must select the Wakeup source to be on the RDX pin. This selection is done by setting bit 3 of WKEN (Wakeup Enable) register. The Wakeup trigger condition is then selected to be high to low transition. This is done via the WKEDG register (Bit 3 is "one").

If the device is halted and crystal oscillator is used, the Wake Up signal will not start the chip running immediately because of the finite start up time requirement of the crystal oscillator. The idle timer (T0) generates a fixed (256 t_C) delay to ensure that the oscillator has indeed stabilized before allowing the device to execute code. The user has to consider this delay when data transfer is expected immediately after exiting the HALT mode.

Diagnostic

Bits CHARL0 and CHARL1 in the ENU register provide a loopback feature for diagnostic testing of the UART. When these bits are set to one, the following occur: The receiver input pin (RDX) is internally connected to the transmitter output pin (TDX); the output of the Transmitter Shift Register is "looped back" into the Receive Shift Register input. In this mode, data that is transmitted is immediately received. This feature allows the processor to verify the transmit and receive data paths of the UART.

Note that the framing format for this mode is the nine bit format; one Start bit, nine data bits, and 7/8, one or two Stop bits. Parity is not generated or verified in this mode.

Attention Mode

The UART Receiver section supports an alternate mode of operation, referred to as ATTENTION Mode. This mode of operation is selected by the ATTN bit in the ENUR register. The data format for transmission must also be selected as having nine Data bits and either 7/8, one or two Stop bits.

The ATTENTION mode of operation is intended for use in networking the device with other processors. Typically in such environments the messages consists of device addresses, indicating which of several destinations should receive them, and the actual data. This Mode supports a scheme in which addresses are flagged by having the ninth bit of the data field set to a 1. If the ninth bit is reset to a zero the byte is a Data byte.

While in ATTENTION mode, the UART monitors the communication flow, but ignores all characters until an address character is received. Upon receiving an address character, the UART signals that the character is ready by setting the RBFL flag, which in turn interrupts the processor if UART Receiver interrupts are enabled. The ATTN bit is also cleared automatically at this point, so that data characters as well as address characters are recognized. Software examines the contents of the RBUF and responds by deciding either to accept the subsequent data stream (by leaving the ATTN bit reset) or to wait until the next address character is seen (by setting the ATTN bit again).

Operation of the UART Transmitter is not affected by selection of this Mode. The value of the ninth bit to be transmitted is programmed by setting XBIT9 appropriately. The value of the ninth bit received is obtained by reading RBIT9. Since this bit is located in ENUR register where the error flags reside, a bit operation on it will reset the error flags.

Interrupts

The devices supports a vectored interrupt scheme. It supports a total of fourteen interrupt sources. Table VI lists all the possible device interrupt sources, their arbitration rankings and the memory locations reserved for the interrupt vector for each source.

Two bytes of program memory space are reserved for each interrupt source. All interrupt sources except the software interrupt are maskable. Each of the maskable interrupts have an Enable bit and one or more Pending bits. A maskable interrupt is active it its associated enable and pending bits are set. If GIE = 1 and an interrupt is active, then the processor will be interrupted as soon as it is ready to start executing an instruction except if the above conditions happen during the Software Trap service routine. This exception is described in the Software Trap sub-section.

The interruption process is accomplished with the INTR instruction (opcode 00), which is jammed inside the Instruction Register and replaces the opcode about to be executed. The following steps are performed for every interrupt:

1. The GIE (Global Interrupt Enable) bit is reset.
2. The address of the instruction about to be executed is pushed into the stack.
3. The PC (Program Counter) branches to address 00FF. This procedure takes 7 t_C cycles to execute.

At this time, since GIE = 0, other maskable interrupts are disabled. The user is now free to do whatever context switching is required by saving the context of the machine in the stack with PUSH instructions. The user would then program a VIS (Vector Interrupt Select) instruction in order to branch to the interrupt service routine of the highest priority interrupt enabled and pending at the time of the VIS. Note that this is not necessarily the interrupt that caused the branch to address location 00FF Hex prior to the context switching.

Thus, if an interrupt with a higher rank than the one which caused the interruption becomes active before the decision of which interrupt to service is made by the VIS, then the interrupt with the higher rank will override any lower ones and will be acknowledged. The lower priority interrupt(s) are still pending, however, and will cause another interrupt immediately following the completion of the interrupt service routine associated with the higher priority interrupt just serviced. This lower priority interrupt will occur immediately following the RETI (Return from Interrupt) instruction at the end of the interrupt service routine just completed.

Inside the interrupt service routine, the associated pending bit has to be cleared by software. The RETI (Return from Interrupt) instruction at the end of the interrupt service routine will set the GIE (Global Interrupt Enable) bit, allowing the processor to be interrupted again if another interrupt is active and pending.

The VIS instruction looks at all the active interrupts at the time it is executed and performs an indirect jump to the beginning of the service routine of the one with the highest rank.

The addresses of the different interrupt service routines, called vectors, are chosen by the user and stored in ROM in a table starting at 01E0 (assuming that VIS is located between 00FF and 01DF). The vectors are 15-bit wide and therefore occupy 2 ROM locations.

TABLE VI. Interrupt Vector Table

ARBITRATION RANKING	SOURCE DESCRIPTION		VECTOR* ADDRESS (Hi-Low Byte)
(1) Highest	Software		0yFE–0yFF
(2)	Reserved		0yFC–0yFD
(3)	External	G0	0yFA-0yFB
(4)	Timer T0	Underflow	0yF8–0yF9
(5)	Timer T1	T1A/Underflow	0yF6–0yF7
(6)	Timer T1	T1B	0yF4-0yF5
(7)	Microwire/Plus	Busy Low	0yF2–0yF3
(8)	Counters		0yF0–0yF1
(9)	UART	Receive	0yEE–0yEF
(10)	UART	Transmit	0yEC–0yED
(11)	Timer T2	T2A/Underflow	0yEA–0yEB
(12)	Timer T2	T2B	0yE8–0yE9
(13)	Capture Timer 1 and 2		0yE6–0yE7
(14)	Unused		0yE4–0yE5
(15)	Port L/Wakeup		0yE2–0yE3
(16) Lowest	Default VIS	Reserved	0yE0–0yE1

* y is a variable which represents the VIS block. VIS and the vector table must be located in the same 256-byte block except if VIS is located at the last address of a block, In this case, the table must be in the next block.

Interrupts (Continued)

VIS and the vector table must be located in the same 256-byte block (0y00 to 0yFF) except if VIS is located at the last address of a block. In this case, the table must be in the next block. The vector table cannot be inserted in the first 256-byte block (y ≠ 0).

The vector of the maskable interrupt with the lowest rank is located at 0yE0 (Hi-Order byte) and 0yE1 (Lo-Order byte) and so forth in increasing rank number. The vector of the maskable interrupt with the highest rank is located at 0yFA (Hi-Order byte) and 0yFB (Lo-Order byte).

The Software Trap has the highest rank and its vector is located at 0yFE and 0yFF.

If, by accident, a VIS gets executed and no interrupt is active, then the PC (Program Counter) will branch to a vector located at 0yE0–0yE1.

Warning:

A Default VIS interrupt handler routine must be present. As a minimum, this handler should confirm that the GIE bit is cleared (this indicates that the interrupt sequence has been taken), take care of any required housekeeping, restore context and return. Some sort of Warm Restart procedure should be implemented. These events can occur without any error on the part of the system designer or programmer.

Note: There is always the possibility of an interrupt occurring during an instruction which is attempting to reset the GIE bit or any other interrupt enable bit. If this occurs when a single cycle instruction is being used to reset the interrupt enable bit, the interrupt enable bit will be reset but an interrupt may still occur. This is because interrupt processing is started at the same time as the interrupt bit is being reset. To avoid this scenario, the user should always use a two-, three- or four-cycle instruction to reset interrupt enable bits.

Figure 18 shows the Interrupt block diagram.

SOFTWARE TRAP

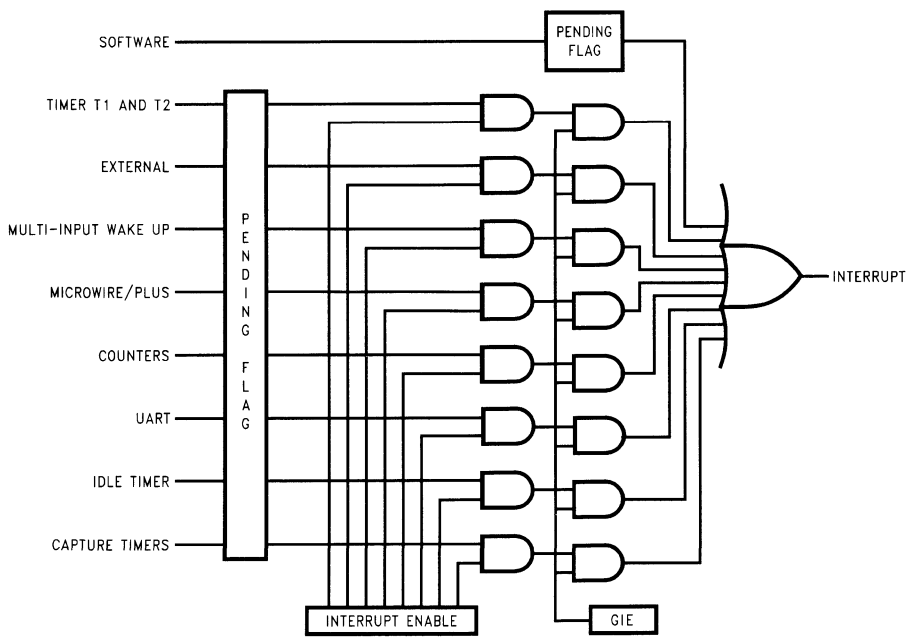
The Software Trap (ST) is a special kind of non-maskable interrupt which occurs when the INTR instruction (used to acknowledge interrupts) is fetched from ROM and placed inside the instruction register. This may happen when the PC is pointing beyond the available ROM address space or when the stack is over-popped.

When an ST occurs, the user can re-initialize the stack pointer and do a recovery procedure (similar to reset, but not necessarily containing all of the same initialization procedures) before restarting.

The occurrence of an ST is latched into the ST pending bit. The GIE bit is not affected and the ST pending bit (**not accessible by the user**) is used to inhibit other interrupts and to direct the program to the ST service routine with the VIS instruction. The RPND instruction is used to clear the software interrupt pending bit. This pending bit is also cleared on reset.

The ST has the highest rank among all interrupts.

Nothing (except another ST) can interrupt an ST being serviced.



TL/DD/12065–20

FIGURE 18. Interrupt Block Diagram

Detection of Illegal Conditions

The device can detect various illegal conditions resulting from coding errors, transient noise, power supply voltage drops, runaway programs, etc.

Reading of undefined ROM gets zeroes. The opcode for software interrupt is 00. If the program fetches instructions from undefined ROM, this will force a software interrupt, thus signaling that an illegal condition has occurred.

The subroutine stack grows down for each call (jump to subroutine), interrupt, or PUSH, and grows up for each return or POP. The stack pointer is initialized to RAM location 06F Hex during reset. Consequently, if there are more returns than calls, the stack pointer will point to addresses 070 and 071 Hex (which are undefined RAM). Undefined RAM from addresses 070 to 07F (Segment 0), 140 to 17F (Segment 1), and all other segments (i.e., Segments 3... etc.) is read as all 1's, which in turn will cause the program to return to address 7FFF Hex. This is an undefined ROM location and the instruction fetched (all 0's) from this location will generate a software interrupt signaling an illegal condition.

Thus, the chip can detect the following illegal conditions:

1. Executing from undefined ROM
2. Over "POP"ing the stack by having more returns than calls.

When the software interrupt occurs, the user can re-initialize the stack pointer and do a recovery procedure before restarting (this recovery program is probably similar to that following reset, but might not contain the same program initialization procedures). The recovery program should reset the software interrupt pending bit using the RPND instruction.

MICROWIRE/PLUS

MICROWIRE/PLUS is a serial synchronous communications interface. The MICROWIRE/PLUS capability enables the device to interface with any of National Semiconductor's MICROWIRE peripherals (i.e., A/D converters, display drivers, E2PROMs etc.) and with other microcontrollers which support the MICROWIRE interface. It consists of an 8-bit serial shift register (SIO) with serial data input (SI), serial data output (SO) and serial shift clock (SK). *Figure 19* shows a block diagram of the MICROWIRE/PLUS logic.

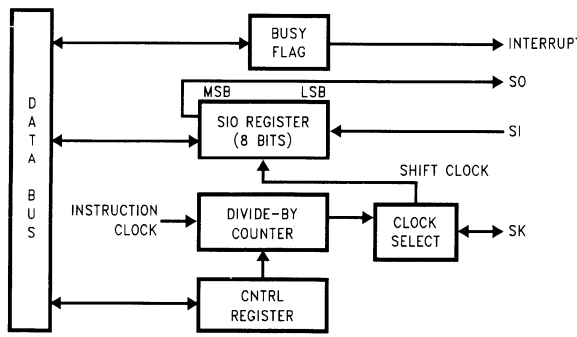
The shift clock can be selected from either an internal source or an external source. Operating the MICROWIRE/PLUS arrangement with the internal clock source is called the Master mode of operation. Similarly, operating the MICROWIRE/PLUS arrangement with an external shift clock is called the Slave mode of operation.

The CNTRL register is used to configure and control the MICROWIRE/PLUS mode. To use the MICROWIRE/PLUS, the MSEL bit in the CNTRL register is set to one. In the master mode, the SK clock rate is selected by the two bits, SL0 and SL1, in the CNTRL register. Table VII details the different clock rates that may be selected.

TABLE VII. MICROWIRE/PLUS Master Mode Clock Select

SL1	SL0	SK Period
0	0	$2 \times t_c$
0	1	$4 \times t_c$
1	x	$8 \times t_c$

Where t_c is the instruction cycle clock



TL/DD/12065-21

FIGURE 19. MICROWIRE/PLUS Block Diagram

MICROWIRE/PLUS (Continued)

MICROWIRE/PLUS OPERATION

Setting the BUSY bit in the PSW register causes the MICROWIRE/PLUS to start shifting the data. It gets reset when eight data bits have been shifted. The user may reset the BUSY bit by software to allow less than 8 bits to shift. If enabled, an interrupt is generated when eight data bits have been shifted. The device may enter the MICROWIRE/PLUS mode either as a Master or as a Slave. *Figure 20* shows how two devices, microcontrollers and several peripherals may be interconnected using the MICROWIRE/PLUS arrangements.

Warning:

The SIO register should only be loaded when the SK clock is low. Loading the SIO register while the SK clock is high will result in undefined data in the SIO register. SK clock is normally low when not shifting.

Setting the BUSY flag when the input SK clock is high in the MICROWIRE/PLUS slave mode may cause the current SK clock for the SIO shift register to be narrow. For safety, the BUSY flag should only be set when the input SK clock is low.

MICROWIRE/PLUS Master Mode Operation

In the MICROWIRE/PLUS Master mode of operation the shift clock (SK) is generated internally by the device. The MICROWIRE Master always initiates all data exchanges. The MSEL bit in the CNTRL register must be set to enable the SO and SK functions onto the G Port. The SO and SK pins must also be selected as outputs by setting appropriate bits in the Port G configuration register. Table VIII summarizes the bit settings required for Master mode of operation.

MICROWIRE/PLUS Slave Mode Operation

In the MICROWIRE/PLUS Slave mode of operation the SK clock is generated by an external source. Setting the MSEL bit in the CNTRL register enables the SO and SK functions onto the G Port. The SK pin must be selected as an input and the SO pin is selected as an output pin by setting and resetting the appropriate bits in the Port G configuration register. Table VIII summarizes the settings required to enter the Slave mode of operation.

TABLE VIII. MICROWIRE Mode Settings

G4 (SO) Config. Bit	G5 (SK) Config. Bit	G4 Fun.	G5 Fun.	Operation
1	1	SO	Int. SK	MICROWIRE/PLUS Master
0	1	TRI-STATE	Int. SK	MICROWIRE/PLUS Master
1	0	SO	Ext. SK	MICROWIRE/PLUS Slave
0	0	TRI-STATE	Ext. SK	MICROWIRE/PLUS Slave

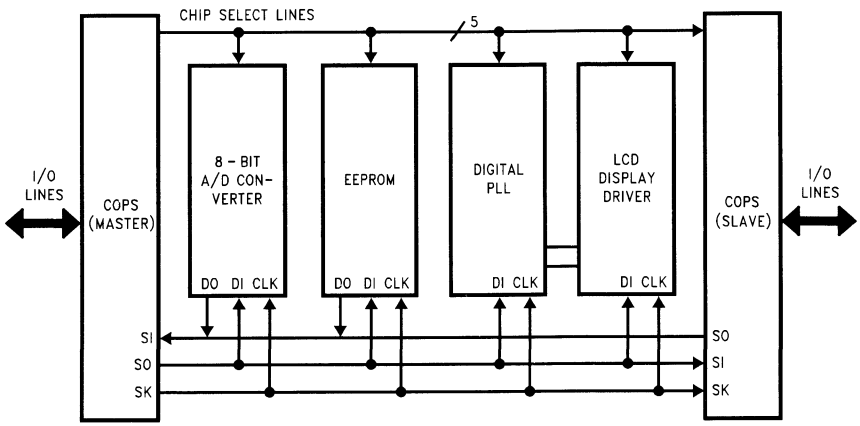
This table assumes that the control flag MSEL is set.

The user must set the BUSY flag immediately upon entering the Slave mode. This will ensure that all data bits sent by the Master will be shifted properly. After eight clock pulses the BUSY flag will be cleared and the sequence may be repeated.

Alternate SK Phase Operation

The device allows either the normal SK clock or an alternate phase SK clock to shift data in and out of the SIO register. in both the modes the SK is normally low. In the normal mode data is shifted in on the rising edge of the SK clock and the data is shifted out on the falling edge of the SK clock. The SIO register is shifted on each falling edge of the SK clock. In the alternate SK phase operation, data is shifted in on the falling edge of the SK clock and shifted out on the rising edge of the SK clock.

A control flag, SKSEL, allows either the normal SK clock or the alternate SK clock to be selected. Resetting SKSEL causes the MICROWIRE/PLUS logic to be clocked from the normal SK signal. Setting the SKSEL flag selects the alternate SK clock. The SKSEL is mapped into the G6 configuration bit. The SKSEL flag will power up in the reset condition, selecting the normal SK signal.



TL/DD/12065-22

FIGURE 20. MICROWIRE/PLUS Application

Memory Map

All RAM, ports and registers (except A and PC) are mapped into data memory address space.

ADDRESS S/ADD REG	CONTENTS
0000 to 006F	112 On-Chip RAM Bytes
0070 to 007F	Unused RAM Address Space (reads as all 1's)
xx80 to xx8F	Unused RAM Address Space (reads undefined data)
xx90	Port E Data Register
xx91	Port E Configuration Register
xx92	Port E Input Pins (read only)
xx93	Reserved
xx94	Port F Data Register
xx95	Port F Configuration Register
xx96	Port F Input Pins (read only)
xx97	Reserved
xx98	Dividend or Result Byte (MDR1)
xx99	Dividend/Multiplier or Result Byte (MDR2)
xx9A	Dividend/Result Byte or Undefined (MDR3)
xx9B	Divisor/Multiplicand or Result Byte (MDR4)
xx9C	Divisor or Multiplicand Byte(MDR5)
xx9D	Multiply/Divide Control Register (MDCR)
xx9E	Counter Control 1 Register (CCR1)
xx9F	Counter Control 2 Register (CCR2)
xxA0	Counter 1 Prescaler Lower Byte (C1PRL)
xxA1	Counter 1 Prescaler Upper Byte (C1PRH)
xxA2	Counter 1 Count Register Lower Byte (C1CTL)
xxA3	Counter 1 Count Register Upper Byte (C1CTH)
xxA4	Counter 2 Prescaler Lower Byte (C2PRL)
xxA5	Counter 2 Prescaler Upper Byte (C2PRH)
xxA6	Counter 2 Count Register Lower Byte (C2CTL)
xxA7	Counter 2 Count Register Upper Byte (C2CTH)
xxA8	Counter 3 Prescaler Lower Byte (C3PRL)
xxA9	Counter 3 Prescaler Upper Byte (C3PRH)
xxAA	Counter 3 Count Register Lower Byte (C3CTL)
xxAB	Counter 3 Count Register Upper Byte (C3CTH)
xxAC	Counter 4 Prescaler Lower Byte (C4PRL)
xxAD	Counter 4 Prescaler Upper Byte (C4PRH)
xxAE	Counter 4 Count Register Lower Byte (C4CTL)
xxAF	Counter 4 Count Register Upper Byte (C4CTH)
xxB0	Capture Timer 1 Prescaler Register (CM1 PSC)
xxB1	Capture Timer 1 Lower Byte (CM1CRL) Read-Only
xxB2	Capture Timer 1 Upper Byte (CM1CRH) Read-Only
xxB3	Capture Timer 2 Prescaler Register (CM2PSC)
xxB4	Capture Timer 2 Lower Byte (CM2CRL) Read-Only
xxB5	Capture Timer 2 Upper Byte (CM2CRH) Read-Only
xxB6	Capture Timer 1 Control Register (CCMR1)
xxB7	Capture Timer 2 Control Register (CCMR2)
xxB8	UART Transmit Buffer (TBUF)
xxB9	UART Receive Buffer (RBUF)
xxBA	UART Control and Status Register (ENU)

Memory Map (Continued)

ADDRESS S/ADD REG	CONTENTS
xxBB	UART Receive Control and Status Register (ENUR)
xxBC	UART Interrupt and Clock Source Register (ENUI)
xxBD	UART Baud Register (BAUD)
xxBE	UART Prescaler Select Register (PSR)
xxBF	Reserved for UART
xxC0	Timer T2 Lower Byte
xxC1	Timer T2 Upper Byte
xxC2	Timer T2 Autoload Register T2RA Lower Byte
xxC3	Timer T2 Autoload Register T2RA Upper Byte
xxC4	Timer T2 Autoload Register T2RB Lower Byte
xxC5	Timer T2 Autoload Register T2RB Upper Byte
xxC6	Timer T2 Control Register
xxC7	Reserved
xxC8	MIWU Edge Select Register (WKEDG)
xxC9	MIWU Enable Register (WKEN)
xxCA	MIWU Pending Register (WKPND)
xxCB	Reserved
xxCC	Reserved
xxCD to xxCF	Reserved
xxD0	Port L Data Register
xxD1	Port L Configuration Register
xxD2	Port L Input Pins (Read Only)
xxD3	Reserved for Port L
xxD4	Port G Data Register
xxD5	Port G Configuration Register
xxD6	Port G Input Pins (Read Only)
xxD7	Port I Input Pins (Read Only)
xxD8	Port C Data Register
xxD9	Port C Configuration Register
xxDA	Port C Input Pins (Read Only)
xxDB	Reserved for Port C
xxDC	Port D
xxDD to xxDF	Reserved for Port D
xxE0 to xxE5	Reserved for EE Control Registers
xxE6	Timer T1 Autoload Register T1RB Lower Byte
xxE7	Timer T1 Autoload Register T1RB Upper Byte
xxE8	ICNTRL Register
xxE9	MICROWIRE Shift Register
xxEA	Timer T1 Lower Byte
xxEB	Timer T1 Upper Byte
xxEC	Timer T1 Autoload Register T1RA Lower Byte
xxED	Timer T1 Autoload Register T1RA Upper Byte
xxEE	CNTRL Control Register
xxEF	PSW Register
xxF0 to xxFB	On-chip RAM Mapped as Registers
xxFC	X Register
xxFD	SP Register
xxFE	B Register
xxFF	S Register
0100 to 017F 0200 to 027F 0300 to 037F	On Chip RAM Bytes (384 Bytes)

Reading memory locations 0070H-007FH (Segment 0) will return all ones. Reading unused memory locations between 0080H-00F0 Hex (Segment 0) will return undefined data. Reading memory locations from other segments (i.e., segment 4, segment 5, etc.) will return all ones.

Memory Map (Continued)

ADDRESSING MODES

There are ten addressing modes, six for operand addressing and four for transfer of control.

OPERAND ADDRESSING MODES

Register Indirect

This is the "normal" addressing mode. The operand is the data memory addressed by the B pointer or X pointer.

Register Indirect (with auto post Increment or decrement of pointer)

This addressing mode is used with the LD and X instructions. The operand is the data memory addressed by the B pointer or X pointer. This is a register indirect mode that automatically post increments or decrements the B or X register after executing the instruction.

Direct

The instruction contains an 8-bit address field that directly points to the data memory for the operand.

Immediate

The instruction contains an 8-bit immediate field as the operand.

Short Immediate

This addressing mode is used with the Load B Immediate instruction. The instruction contains a 4-bit immediate field as the operand.

Indirect

This addressing mode is used with the LAID instruction. The contents of the accumulator are used as a partial address (lower 8 bits of PC) for accessing a data operand from the program memory.

TRANSFER OF CONTROL ADDRESSING MODES

Relative

This mode is used for the JP instruction, with the instruction field being added to the program counter to get the new program location. JP has a range from −31 to +32 to allow a 1-byte relative jump (JP + 1 is implemented by a NOP instruction). There are no "pages" when using JP, since all 15 bits of PC are used.

Absolute

This mode is used with the JMP and JSR instructions, with the instruction field of 12 bits replacing the lower 12 bits of the program counter (PC). This allows jumping to any location in the current 4k program memory segment.

Absolute Long

This mode is used with the JMPL and JSRL instructions, with the instruction field of 15 bits replacing the entire 15 bits of the program counter (PC). This allows jumping to any location up to 32k in the program memory space.

Indirect

This mode is used with the JID instruction. The contents of the accumulator are used as a partial address (lower 8 bits of PC) for accessing a location in the program memory. The contents of this program memory location serve as a partial address (lower 8 bits of PC) for the jump to the next instruction.

Note: The VIS is a special case of the Indirect Transfer of Control addressing mode, where the double byte vector associated with the interrupt is transferred from adjacent addresses in the program memory into the program counter (PC) in order to jump to the associated interrupt service routine.

Instruction Set

Register and Symbol Definition

Registers	
A	8-Bit Accumulator Register
B	8-Bit Address Register
X	8-Bit Address Register
SP	8-Bit Stack Pointer Register
PC	15-Bit Program Counter Register
PU	Upper 7 Bits of PC
PL	Lower 8 Bits of PC
C	1 Bit of PSW Register for Carry
HC	1 Bit of PSW Register for Half Carry
GIE	1 Bit of PSW Register for Global Interrupt Enable
VU	Interrupt Vector Upper Byte
VL	Interrupt Vector Lower Byte

Symbols	
[B]	Memory Indirectly Addressed by B Register
[X]	Memory Indirectly Addressed by X Register
MD	Direct Addressed Memory
Mem	Direct Addressed Memory or [B]
Meml	Direct Addressed Memory or [B] or Immediate Data
Imm	8-Bit Immediate Data
Reg	Register Memory: Addresses F0 to FF (Includes B, X and SP)
Bit	Bit Number (0 to 7)
←	Loaded with
⟷	Exchanged with

INSTRUCTION SET

ADD	A,Meml	ADD	A ← A + Meml
ADC	A,Meml	ADD with Carry	A ← A + Meml + C, C ← Carry, HC ← Half Carry
SUBC	A,Meml	Subtract with Carry	A ← A − Meml + C, C ← Carry, HC ← Half Carry
AND	A,Meml	Logical AND	A ← A and $\overline{\text{Meml}}$
ANDSZ	A,Imm	Logical AND Immed., Skip if Zero	Skip next if (A and Imm) = 0
OR	A,Meml	Logical OR	A ← A or Meml
XOR	A,Meml	Logical EXclusive OR	A ← A xor Meml
IFEQ	MD,Imm	IF EQual	Compare MD and Imm, Do next if MD = Imm
IFEQ	A,Meml	IF EQual	Compare A and Meml, Do next if A = Meml
IFNE	A,Meml	IF Not Equal	Compare A and Meml, Do next if A ≠ Meml
IFGT	A,Meml	IF Greater Than	Compare A and Meml, Do next if A > Meml
IFBNE	#	IF B Not Equal	Do next if lower 4 bits of B ≠ Imm
DRSZ	Reg	Decrement Reg., Skip if Zero	Reg ← Reg − 1, Skip if Reg = 0
SBIT	#,Mem	Set BIT	1 to bit, Mem (bit = 0 to 7 immediate)
RBIT	#,Mem	Reset BIT	0 to bit, Mem
IFBIT	#,Mem	IF BIT	If bit #, A or Mem is true do next instruction
RPND		Reset PeNDing Flag	Reset Software Interrupt Pending Flag
X	A,Mem	EXchange A with Memory	A ⟷ Mem
X	A,[X]	EXchange A with Memory [X]	A ⟷ [X]
LD	A,Meml	LoaD A with Memory	A ← Meml
LD	A,[X]	LoaD A with Memory [X]	A ← [X]
LD	B, Imm	LoaD B with Immed.	B ← Imm
LD	Mem, Imm	LoaD Memory Immed.	Mem ← Imm
LD	Reg, Imm	LoaD Register Memory Immed.	Reg ← Imm
X	A, [B±]	EXchange A with Memory [B]	A ⟷ [B], (B ← B ± 1)
X	A, [X±]	EXchange A with Memory [X]	A ⟷ [X], (X ← X ± 1)
LD	A, [B±]	LoaD A with Memory [B]	A ← [B], (B ← B ± 1)
LD	A, [X±]	LoaD A with Memory [X]	A ← [X], (X ← X ± 1)
LD	[B±],Imm	LoaD Memory [B] Immed.	[B] ← Imm, (B ← B ± 1)
CLR	A	CLeaR A	A ← 0
INC	A	INCrement A	A ← A + 1
DEC	A	DECrement A	A ← A − 1
LAID		Load A InDirect from ROM	A ← ROM (PU, A)
DCOR	A	Decimal CORrect A	A ← BCD correction of A (follows ADC, SUBC)
RRC	A	Rotate A Right thru C	C → A7 → . . . → A0 → C
RLC	A	Rotate A Left thru C	C ← A7 ← . . . ← A0 ← C
SWAP	A	SWAP nibbles of A	A7 . . . A4 ⟷ A3 . . . A0
SC		Set C	C ← 1, HC ← 1
RC		Reset C	C ← 0, HC ← 0
IFC		IF C	If C is true, do next instruction
IFNC		IF Not C	If C is not true, do next instruction
POP	A	POP the stack into A	SP ← SP + 1, A ← [SP]
PUSH	A	PUSH A onto the stack	[SP] ← A, SP ← SP − 1
VIS		Vector to Interrupt Service Routine	PU ← [VU], PL ← [VL]
JMPL	Addr.	Jump absolute Long	PC ← ii (ii = 15 bits, 0 to 32k)
JMP	Addr.	Jump absolute	PC9 . . . 0 ← i (i = 12 bits)
JP	Disp.	Jump relative short	PC ← PC + r (r is −31 to +32, except 1)
JSRL	Addr.	Jump SubRoutine Long	[SP] ← PL, [SP − 1] ← PU, SP − 2, PC ← ii
JSR	Addr	Jump SubRoutine	[SP] ← PL, [SP − 1] ← PU, SP − 2, PC9 . . . 0 ← i
JID		Jump InDirect	PL ← ROM (PU, A)
RET		RETurn from subroutine	SP + 2, PL ← [SP], PU ← [SP − 1]
RETSK		RETurn and SKip	SP + 2, PL ← [SP], PU ← [SP − 1], skip next instruction
RETI		RETurn from Interrupt	SP + 2, PL ← [SP], PU ← [SP − 1], GIE ← 1
INTR		Generate an Interrupt	[SP] ← PL, [SP − 1] ← PU, SP − 2, PC ← 0FF
NOP		No OPeration	PC ← PC + 1

Instruction Execution Time

Most instructions are single byte (with immediate addressing mode instructions taking two bytes).

Most single byte instructions take one cycle time to execute.

Skipped instructions require x number of cycles to be skipped, where x equals the number of bytes in the skipped instruction opcode.

See the BYTES and CYCLES per INSTRUCTION table for details.

Bytes and Cycles per Instruction

The following table shows the number of bytes and cycles for each instruction in the format of byte/cycle.

Arithmetic and Logic Instructions

	[B]	Direct	Immed.
ADD	1/1	3/4	2/2
ADC	1/1	3/4	2/2
SUBC	1/1	3/4	2/2
AND	1/1	3/4	2/2
OR	1/1	3/4	2/2
XOR	1/1	3/4	2/2
IFEQ	1/1	3/4	2/2
IFGT	1/1	3/4	2/2
IFBNE	1/1		
DRSZ		1/3	
SBIT	1/1	3/4	
RBIT	1/1	3/4	
IFBIT	1/1	3/4	

Instructions Using A & C

CLRA	1/1
INCA	1/1
DECA	1/1
LAID	1/3
DCORA	1/1
RRCA	1/1
RLCA	1/1
SWAPA	1/1
SC	1/1
RC	1/1
IFC	1/1
IFNC	1/1
PUSHA	1/3
POPA	1/3
ANDSZ	2/2

Transfer of Control Instructions

JMPL	3/4
JMP	2/3
JP	1/3
JSRL	3/5
JSR	2/5
JID	1/3
VIS	1/5
RET	1/5
RETSK	1/5
RETI	1/5
INTR	1/7
NOP	1/1

RPND	1/1

Memory Transfer Instructions

	Register Indirect		Direct	Immed.	Register Indirect Auto Incr. and Decr.		
	[B]	[X]			[B+, B−]	[X+, X−]	
X A, *	1/1	1/3	2/3		1/2	1/3	
LD A, *	1/1	1/3	2/3	2/2	1/2	1/3	
LD B, Imm				1/1			(IF B < 16)
LD B, Imm				2/2			(IF B > 15)
LD Mem, Imm	2/2		3/3		2/2		
LD Reg, Imm			2/3				
IFEQ MD, Imm			3/3				

Note: * = > Memory location addressed by B or X or directly.

Mask Options

The mask programmable options are shown below. The options are programmed at the same time as the ROM pattern submission.

OPTION 1: CLOCK CONFIGURATION

= 1 Crystal Oscillator (CKI/10)

G7 (CKO) is clock generator output to crystal/resonator with CKI being the clock input

OPTION 2: HALT

= 1 Enable HALT mode

= 2 Disable HALT mode

OPTION 3: BONDING OPTIONS

= 1 68 Pins PLCC

The chip can be driven by a clock input on the CKI input pin which can be between DC and 10 MHz. The CKO output clock is on pin G7 (if clock option = 1 has been selected). The CKI input frequency is divided down by 10 to produce the instruction cycle clock ($1/t_c$).

Opcode Table

Upper Nibble																Lower Nibble
F	E	D	C	B	A	9	8	7	6	5	4	3	2	1	0	
JP −15	JP −31	LD 0F0, #i	DRSZ 0F0	RRCA	RC	ADC A, #i	ADC A,[B]	IFBIT 0,[B]	ANDSZ A, #i	LD B, #0F	IFBNE 0	JSR x000–x0FF	JMP x000–x0FF	JP +17	INTR	0
JP −14	JP −30	LD 0F1, #i	DRSZ 0F1	*	SC	SUBC A, #i	SUBC A,[B]	IFBIT 1,[B]	*	LD B, #0E	IFBNE 1	JSR x100–x1FF	JMP x100–x1FF	JP +18	JP + 2	1
JP −13	JP −29	LD 0F2, #i	DRSZ 0F2	X A, [X+]	X A, [B+]	IFEQ A, #i	IFEQ A,[B]	IFBIT 2,[B]	*	LD B, #0D	IFBNE 2	JSR x200–x2FF	JMP x200–x2FF	JP +19	JP + 3	2
JP −12	JP −28	LD 0F3, #i	DRSZ 0F3	X A, [X−]	X A, [B−]	IFGT A, #i	IFGT A,[B]	IFBIT 3,[B]	*	LD B, #0C	IFBNE 3	JSR x300–x3FF	JMP x300–x3FF	JP +20	JP + 4	3
JP −11	JP −27	LD 0F4, #i	DRSZ 0F4	VIS	LAID	ADD A, #i	ADD A,[B]	IFBIT 4,[B]	CLRA	LD B, #0B	IFBNE 4	JSR x400–x4FF	JMP x400–x4FF	JP +21	JP + 5	4
JP −10	JP −26	LD 0F5, #i	DRSZ 0F5	RPND	JID	AND A, #i	AND A,[B]	IFBIT 5,[B]	SWAPA	LD B, #0A	IFBNE 5	JSR x500–x5FF	JMP x500–x5FF	JP +22	JP + 6	5
JP −9	JP −25	LD 0F6, #i	DRSZ 0F6	X A,[X]	X A,[B]	XOR A, #i	XOR A,[B]	IFBIT 6,[B]	DCORA	LD B, #09	IFBNE 6	JSR x600–x6FF	JMP x600–x6FF	JP +23	JP + 7	6
JP −8	JP −24	LD 0F7, #i	DRSZ 0F7	*	*	OR A, #i	OR A,[B]	IFBIT 7,[B]	PUSHA	LD B, #08	IFBNE 7	JSR x700–x7FF	JMP x700–x7FF	JP +24	JP + 8	7
JP −7	JP −23	LD 0F8, #i	DRSZ 0F8	NOP	RLCA	LD A, #i	IFC	SBIT 0,[B]	RBIT 0,[B]	LD B, #07	IFBNE 8	JSR x800–x8FF	JMP x800–x8FF	JP +25	JP + 9	8
JP −6	JP −22	LD 0F9, #i	DRSZ 0F9	IFNE A,[B]	IFEQ Md, #i	IFNE A, #i	IFNC	SBIT 1,[B]	RBIT 1,[B]	LD B, #06	IFBNE 9	JSR x900–x9FF	JMP x900–x9FF	JP +26	JP + 10	9
JP −5	JP −21	LD 0FA, #i	DRSZ 0FA	LD A, [X+]	LD A, [B+]	LD [B+], #i	INCA	SBIT 2,[B]	RBIT 2,[B]	LD B, #05	IFBNE 0A	JSR xA00–xAFF	JMP xA00–xAFF	JP +27	JP + 11	A
JP −4	JP −20	LD 0FB, #i	DRSZ 0FB	LD A, [X−]	LD A, [B−]	LD [B−], #i	DECA	SBIT 3,[B]	RBIT 3,[B]	LD B, #04	IFBNE 0B	JSR xB00–xBFF	JMP xB00–xBFF	JP +28	JP + 12	B
JP −3	JP −19	LD 0FC, #i	DRSZ 0FC	LD Md, #i	JMPL	X A,Md	POPA	SBIT 4,[B]	RBIT 4,[B]	LD B, #03	IFBNE 0C	JSR xC00–xCFF	JMP xC00–xCFF	JP +29	JP + 13	C
JP −2	JP −18	LD 0FD, #i	DRSZ 0FD	DIR	JSRL	LD A,Md	RETSK	SBIT 5,[B]	RBIT 5,[B]	LD B, #02	IFBNE 0D	JSR xD00–xDFF	JMP xD00–xDFF	JP +30	JP + 14	D
JP −1	JP −17	LD 0FE, #i	DRSZ 0FE	LD A,[X]	LD A,[B]	LD [B], #i	RET	SBIT 6,[B]	RBIT 6,[B]	LD B, #01	IFBNE 0E	JSR xE00–xEFF	JMP xE00–xEFF	JP +31	JP + 15	E
JP −0	JP −16	LD 0FF, #i	DRSZ 0FF	*	*	LD B, #i	RETI	SBIT 7,[B]	RBIT 7,[B]	LD B, #00	IFBNE 0F	JSR xF00–xFFF	JMP xF00–xFFF	JP +32	JP + 16	F

Where,

i is the immediate data

Md is a directly addressed memory location

* is an unused opcode

Note: The opcode 60 Hex is also the opcode for IFBIT #i,A.

Development Support

SUMMARY

- iceMASTER: IM-COP8/400—Full feature in-circuit emulation for all COP8 products. A full set of COP8 Basic and Feature Family device and package specific probes are available.
- COP8 Debug Module: Moderate cost in-circuit emulation and development programming unit.
- COP8 Evaluation and Programming Unit: EPU-COP888GG—low cost In-circuit simulation and development programming unit.
- Assembler: COP8-DEV-IBMA. A DOS installable cross development Assembler, Linker, Librarian and Utility Software Development Tool Kit.
- C Compiler: COP8C. A DOS installable cross development Software Tool Kit.
- OTP/EPROM Programmer Support: Covering needs from engineering prototype, pilot production to full production environments.

iceMASTER (IM) IN-CIRCUIT EMULATION

The iceMASTER IM-COP8/400 is a full feature, PC based, in-circuit emulation tool developed and marketed by MetaLink Corporation to support the whole COP8 family of products. National is a resale vendor for these products.

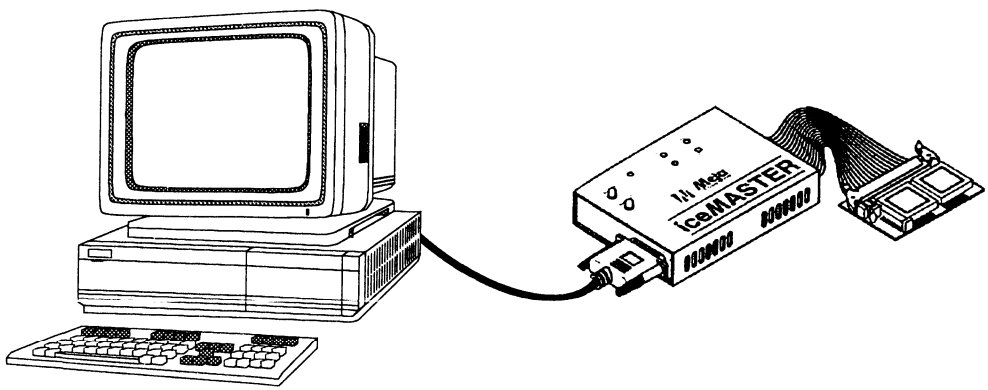
See *Figure 21* for configuration.

The iceMASTER IM-COP8/400 with its device specific COP8 Probe provides a rich feature set for developing, testing and maintaining product:

- Real-time in-circuit emulation; full 2.4V–5.5V operation range, full DC-10 MHz clock. Chip options are programmable or jumper selectable.
- Direct connection to application board by package compatible socket or surface mount assembly.
- Full 32 kbytes of loadable programming space that overlays (replaces) the on-chip ROM or EPROM. On-chip RAM and I/O blocks are used directly or recreated on the probe as necessary.
- Full 4k frame synchronous trace memory. Address, instruction, and 8 unspecified, circuit connectable trace lines. Display can be HLL source (e.g., C source), assembly or mixed.
- A full 64k hardware configurable break, trace on, trace oft control, and pass count increment events.
- Tool set integrated interactive symbolic debugger—supports both assembler (COFF) and C Compiler (.COD) linked object formats.
- Real time peformance profiling analysis; selectable bucket definition.
- Watch windows, content updated automatically at each execution break.
- Instruction by instruction memory/register changes displayed on source window when in single step operation.
- Single base unit and debugger software reconfigurable to support the entire COP8 family; only the probe personality needs to change. Debugger software is processor customized, and reconfigured from a master model file.
- Processor specific symbolic display of registers and bit level assignments, configured from master model file.
- Halt/Idle mode notification.
- On-line HELP customized to specific processor using master model file.
- Includes a copy of COP8-DEV-IBMA assembler and linker SDK.

IM Order Information

Base Unit	
IM-COP8/400-1	iceMASTER base unit, 110V power supply
IM-COP8/400-2	iceMASTER base unit, 220V power supply
iceMASTER Probe	
MHW-888GW68PWPC	68 PLCC



TL/DD/12065–31

FIGURE 21. COP8 iceMASTER Environment

Development Support (Continued)

iceMASTER DEBUG MODULE (DM)

The iceMASTER Debug Module is a PC based, combination in-circuit emulation tool and COP8 based OTP/EPROM programming tool developed and marketed by MetaLink Corporation to support the whole COP8 family of products. National is a resale vendor for these products.

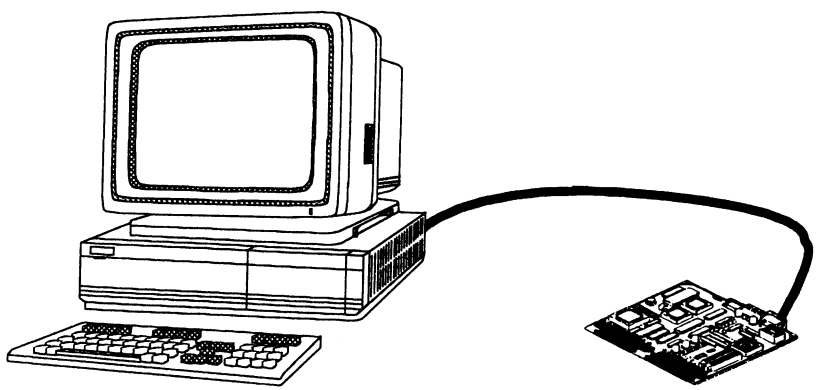
See *Figure 22* for configuration.

The iceMASTER Debug Module is a moderate cost development tool. It has the capability of in-circuit emulation for a specific COP8 microcontroller and in addition serves as a programming tool for COP8 OTP and EPROM product families. Summary of features is as follows:

- Real-time in-circuit emulation; full operating voltage range operation, full DC-10 MHz clock.
- All processor I/O pins can be cabled to an application development board with package compatible cable to socket and surface mount assembly.
- Full 32 kbytes of loadable programming space that overlays (replaces) the on-chip ROM or EPROM. On-chip RAM and I/O blocks are used directly or recreated as necessary.
- 100 frames of synchronous trace memory. The display can be HLL source (C source), assembly or mixed. The most recent history prior to a break is available in the trace memory.
- Configured break points; uses INTR instruction which is modestly intrusive.
- Software—only supported features are selectable.
- Tool set integrated interactive symbolic debugger—supports both assembler (COFF) and C Compiler (.COD) SDK linked object formats.
- Instruction by instruction memory/register changes displayed when in single step operation.
- Debugger software is processor customized, and reconfigured from a master model file.
- Processor specific symbolic display of registers and bit level assignments, configured from master model file.
- Halt/Idle mode notification.
- Programming menu supports full product line of programmable OTP and EPROM COP8 products. Program data is taken directly from the overlay RAM.
- Programming of 44 PLCC and 68 PLCC parts requires external programming adapters.
- Includes wallmount power supply.
- On-board V_{PP} generator from 5V input or connection to external supply supported. Requires V_{PP} level adjustment per the family programming specification (correct level is provided on an on-screen pop-down display).
- On-line HELP customized to specific processor using master model file.
- Includes a copy of COP8-DEV-IBMA assembler and linker SDK.

DM Order Information

Debug Module Unit	
COP8-DM/888GW	
Cable Adapters	
DM-COP8/68P	68 PLCC



TL/DD/12065–32

FIGURE 22. COP8-DM Environment

Development Support (Continued)

COP8 ASSEMBLER/LINKER SOFTWARE DEVELOPMENT TOOL KIT

National Semiconductor offers a relocateable COP8 macro cross assembler, linker, librarian and utility software development tool kit. Features are summarized as follows:

- Basic and Feature Family instruction set by "device" type.
- Nested macro capability.
- Extensive set of assembler directives.
- Supported on PC/DOS platform.
- Generates National standard COFF output files.
- Integrated Linker and Librarian.
- Integrated utilities to generate ROM code file outputs.
- DUMPCOFF utility.

This product is integrated as a part of MetaLink tools as a development kit, fully supported by the MetaLink debugger. It may be ordered separately or it is bundled with the MetaLink products at no additional cost.

Order-Information

Assembler SDK	
COP8-DEV-IBMA	Assembler SDK on installable 3.5" PC/DOS Floppy Disk Drive format. Periodic upgrades and most recent version is available on National's BBS and Internet.

COP8 C COMPILER

A C Compiler is developed and marketed by Byte Craft Limited. The COP8C compiler is a fully integrated development tool specifically designed to support the compact embedded configuration of the COP8 family of products. Features are summarized as follows:

- ANSI C with some restrictions and extensions that optimize development for the COP8 embedded application.
- BITS data type extension. Register declaration #pragma with direct bit level definitions.
- C language support for interrupt routines.
- Expert system, rule based code generation and optimization.
- Performs consistency checks against the architectural definitions of the target COP8 device.
- Generates program memory code.
- Supports linking of compiled object or COP8 assembled object formats.
- Global optimization of linked code.
- Symbolic debug load format fully source level supported by the MetaLink debugger.

SINGLE CHIP OTP/EMULATOR SUPPORT

The COP8 family is supported by single chip OTP emulators. For detailed information refer to the emulator specific datasheet and the emulator selection table below:

OTP Emulator Ordering Information

Device Number	Clock Option	Package	Emulates
COP87L88GWV-XE	Crystal/ HALT En	68 PLCC	COP888GW

INDUSTRY WIDE OTP/EPROM PROGRAMMING SUPPORT

Programming support, in addition to the MetaLink development tools, is provided by a full range of independent approved vendors to meet the needs from the engineering laboratory to full production.

Approved List

Manufacturer	North America	Europe	Asia
BP Microsystems	(800) 225-2102 (713) 688-4600 Fax: (713) 688-0920	+49-8152-4183 +49-8856-932616	+852-234-16611 +852-2710-8121
Data I/O	(800) 426-1045 (206) 881-6444 Fax: (206) 882-1043	+44-0734-440011	Call North America
HI-LO	(510) 623-8860	Call Asia	+886-2-764-0215 Fax: +886-2-756-6403
ICE Technology	(800) 624-8949 (919) 430-7915	+44-1226-767404 Fax: 0-1226-370-434	
MetaLink	(800) 638-2423 (602) 926-0797 Fax: (602) 693-0681	+49-80 9156 96-0 Fax: +49-80 9123 86	+852-737-1800
Systems General	(408) 263-6667	+41-1-9450300	+886-2-917-3005 Fax: +886-2-911-1283
Needhams	(916) 924-8037 Fax: (916) 924-8065		

Development Support (Continued)

AVAILABLE LITERATURE

For more information, please see the COP8 Basic Family User's Manual, Literature Number 620895, COP8 Feature Family User's Manual, Literature Number 620897 and National's Family of 8-bit Microcontrollers COP8 Selection Guide, Literature Number 630009.

DIAL-A-HELPER SERVICE

Dial-A-Helper is a service provided by the Microcontroller Applications group. The Dial-A-Helper is an Electronic Information System that may be accessed as a Bulletin Board System (BBS) via data modem, as an FTP site on the Internet via standard FTP client application or as an FTP site on the Internet using a standard Internet browser such as Netscape or Mosaic.

The Dial-A-Helper system provides access to an automated information storage and retrieval system. The system capabilities include a MESSAGE SECTION (electronic mail, when accessed as a BBS) for communications to and from the Microcontroller Applications Group and a FILE SECTION which consists of several file areas where valuable application software and utilities could be found.

DIAL-A-HELPER BBS via a Standard Modem

Modem: CANADA/U.S.: (800) NSC-MICRO
(800) 672-6427
EUROPE: (+49) 0-8141-351332
Baud: 14.4k
Set-Up: Length: 8-Bit
Parity: None
Stop Bit: 1
Operation: 24 Hours, 7 Days

DIAL-A-HELPER via FTP

ftp nscmicro.nsc.com
user: anonymous
password: username@yourhost.site.domain

DIAL-A-HELPER via a WorldWide Web Browser

ftp://nscmicro.nsc.com

National Semiconductor on the WorldWide Web

See us on the WorldWide Web at: http://www.national.com

CUSTOMER RESPONSE CENTER

Complete product information and technical support is available from National's customer response centers.

CANADA/ U.S.:	Tel:		(800) 272-9959
	email:		support@tevm2.nsc.com
EUROPE:	email:		europe.support@nsc.com
	Deutsch	Tel:	+49 (0) 180-530 85 85
	English	Tel:	+49 (0) 180-532 78 32
	Français	Tel:	+49 (0) 180-532 93 58
	Italiano	Tel:	+49 (0) 180-534 16 80
JAPAN:	Tel:		+81-043-299-2309
S.E. ASIA:	Beijing	Tel:	(+86) 10-6856-8601
	Shanghai	Tel:	(+86) 21-6415-4092
	Hong Kong	Tel:	(+852) 2737-1600
	Korea	Tel:	(+82) 2-3771-6909
	Malaysia	Tel:	(+60-4) 644-9061
	Singapore	Tel:	(+65) 255-2226
	Taiwan	Tel:	+886-2-521-3288
AUSTRALIA:	Tel:		(+61) 3-9558-9999
INDIA:	Tel:		(+91) 80-559-9467

Section 6

MICROWIRE/PLUS™ Peripherals

Section 6 Contents

MICROWIRE/PLUS: Serial Interface 6-3
ADC0811 8-Bit Serial I/O A/D Converter with 11-Channel Multiplexer 6-7
ADC08031/ADC08032/ADC08034/ADC08038 8-Bit High Speed Serial I/O A/D Converters with Multiplexer Options, Voltage Reference and Track/Hold Functions 6-8
ADC0852/ADC0854 Multiplexed Comparator with 8-Bit Reference Divider 6-9
ADC1038 10-Bit Serial I/O A/D Converters with Analog Multiplexer and Track/Hold Function 6-10
ADC12038 Self-Calibrating 12-Bit Plus Sign Serial I/O A/D Converters with MUX and Sample/Hold 6-11
COP472-3 Liquid Crystal Display Controller 6-12
MM5450/MM5451 LED Display Drivers 6-18
MM5483 Liquid Crystal Display Driver 6-19
MM5484 16-Segment LED Display Drivers 6-20
MM5486 LED Display Driver 6-21
MM58241 High Voltage Display Driver 6-22
MM58341 High Voltage Display Driver 6-23
MM58342 High Voltage Display Driver 6-24
NM93C13/NM93C14 256-/1024-Bit Serial EEPROM 6-25
NM93C06/NM93C46/NM93C56/NM93C66 256-/1024-/2048-/4096-Bit Serial EEPROM (MICROWIRE Bus Interface) 6-26
NM93C46A 1024-Bit Serial Interface, Standard Voltage CMOS EEPROM 6-27
NM93C56A 2048-Bit Serial Interface, Standard Voltage CMOS EEPROM (MICROWIRE Bus Interface) 6-28
NM93C66A 4096-Bit Serial Interface, Standard Voltage CMOS EEPROM (MICROWIRE Bus Interface) 6-29
NM93C86A 16,384-Bit Serial Interface CMOS EEPROM (MICROWIRE Synchronous Bus) 6-30
NM93CS06/NM93CS46/NM93CS56/NM93CS66 (MICROWIRE Bus Interface) 256-/1024-/2048-/4096-Bit Serial EEPROM with Data Protect and Sequential Read 6-31
LMC1982 Digitally-Controlled Stereo Tone and Volume Circuit with Two Selectable Stereo Inputs 6-32
LMC1983 Digitally-Controlled Stereo Tone and Volume Circuit with Three Selectable Stereo Inputs 6-33
LMC1992 Digitally-Controlled Stereo Tone and Volume Circuit with Four-Channel Input-Selector 6-34
LMC835 Digitally-Controlled Graphic Equalizer 6-35
LM1971 μPot Digitally Controlled 62 dB Audio Attenuator with Mute 6-36
LM1972 μPot 2-Channel 78 dB Audio Attenuator with Mute 6-37
LM1973 μPot 3-Channel 76 dB Audio Attenuator with Mute 6-38

National Semiconductor

MICROWIRE/PLUS™ Serial Interface

National's MICROWIRE/PLUS provides for high-speed, serial communications in a simple 3-wire implementation.

MICROWIRE/PLUS consists of an 8-bit serial shift register (SIO), serial data input (SI), serial data output (SO), and a serial shift clock (SK).

The shift clock in MICROWIRE/PLUS can be internal or external, the interface can be designated as either bus master or slave, giving it the flexibility necessary for distributed and multiprocessing applications.

With its simple 3-wire interface, MICROWIRE/PLUS can connect a variety of nodes in a serial-communication network.

This simple 3-wire design also helps increase system reliability while reducing system size and development time.

Because the COP8 family has memory-mapped architectures, the contents of the SIO register can be accessed through standard memory-addressing instructions.

The control register (CNTRL) is used to configure and control the mode and operation of the interface through user-selectable bits that program the internal shift rate. This greatly increases the flexibility of the interface.

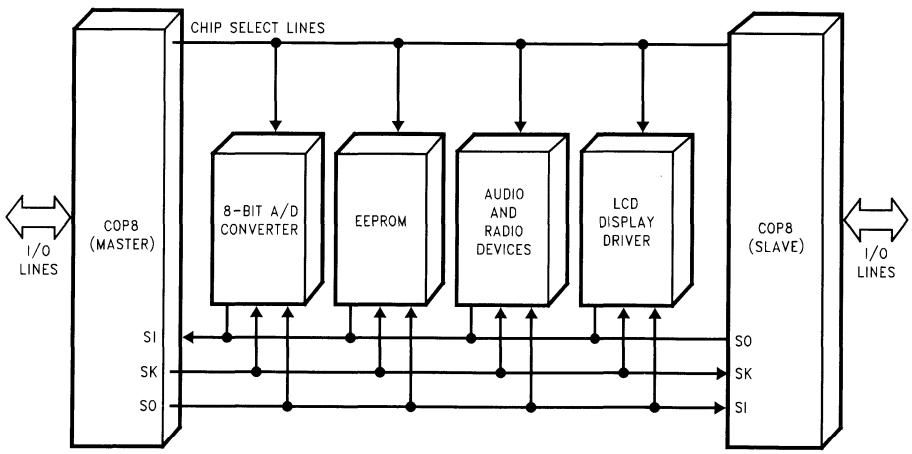
MICROWIRE/PLUS can also provide additional I/O capability by connecting, for example, external 8-bit parallel-to-serial shift registers to 8-bit serial-to-parallel shift registers.

And it can interface a wide variety of peripherals:

- Memory (CMOS RAM and EEPROM)
- A/D converters
- Timers/counters
- Digital phase locked-loops
- Telecom peripherals
- Vacuum fluorescent display drivers
- LED display drivers
- LCD display drivers

Both MICROWIRE and MICROWIRE/PLUS give all the members of National's microcontroller families the flexibility and design-ease to implement a solution quickly, simply, and cost-effectively.

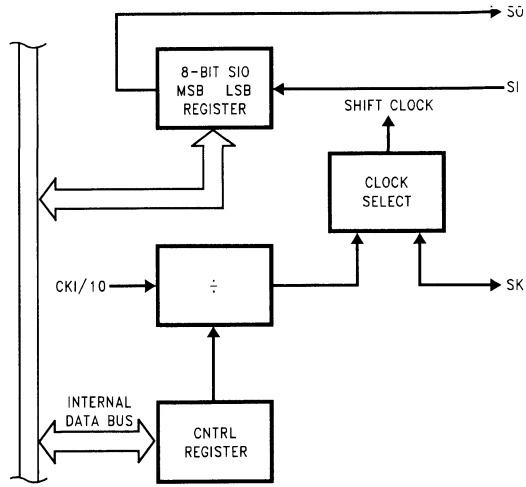
MICROWIRE/PLUS System Block



TL/XX/0074–1

6

MICROWIRE/PLUS Block Diagram



TL/XX/0074–2

MICROWIRE/PLUS Peripherals

Part Number	Description	Databook
A/D CONVERTERS AND COMPARATORS		
ADC0811	11 Channel 8-Bit A/D Converter with Multiplexer	Data Aquisition
ADC08031	1 Channel 8-Bit A/D Converter with Multiplexer	Data Aquisition
ADC08038	8 Channel 8-Bit A/D Converter with Multiplexer	Data Aquisition
ADC08032	2 Channel 8-Bit A/D Converter with Multiplexer	Data Aquisition
ADC08034	4 Channel 8-Bit A/D Converter with Multiplexer	Data Aquisition
ADC0852	Multiplexed Comparator with 8-Bit Reference Divider	Data Aquisition
ADC0854	Multiplexed Comparator with 8-Bit Reference Divider	Data Aquisition
ADC1038	8 Channel 10-Bit A/D Converter with MUX	Data Aquisition
ADC12038	8 Channel 12-Bit A/D Converter with MUX	Data Aquisition
DISPLAY DRIVERS		
COP472-3	3 x 12 Multiplexed Expandable LCD Display Driver	Microcontroller
MM5450	35 Output LED Display Driver	Application Specific Analog Products
MM5451	34 Output LED Display Driver	Application Specific Analog Products
MM5483	31 Segment LCD Display Driver	Application Specific Analog Products
MM5484	16 Segment LED Display Driver	Application Specific Analog Products
MM5486	33 Output LED Display Driver	Application Specific Analog Products
MM58241	32 Output High Voltage Display Driver	Application Specific Analog Products
MM58341	32 Output High Voltage Display Driver	Application Specific Analog Products
MM58342	20 Output High Voltage Display Driver	Application Specific Analog Products
MEMORY DEVICES		
NM93C13	16 x 16 CMOS EEPROM	Memory
NM93C14	64 x 16 CMOS EEPROM	Memory
NM93C06	16 x 16 CMOS EEPROM	Memory
NM93C46	64 x 16 CMOS EEPROM	Memory
NM93C56	128 x 16 CMOS EEPROM	Memory
NM93C66	256 x 16 CMOS EEPROM	Memory
NM93C46A	64 x 16 or 128 x 8 CMOS EEPROM	Memory
NM93C56A	128 x 16 or 256 x 8 CMOS EEPROM	Memory
NM93C66A	256 x 16 or 512 x 8 CMOS EEPROM	Memory
NM93C86A	512 x 16 or 1024 x 8 CMOS EEPROM	Memory
NM93CS06	16 x 16 CMOS EEPROM with Write Protect	Memory
NM93CS46	64 x 16 CMOS EEPROM with Write Protect	Memory
NM93CS56	128 x 16 CMOS EEPROM with Write Protect	Memory
NM93CS66	256 x 16 CMOS EEPROM with Write Protect	Memory

Note: The low voltage (2.7V–5.5V) version is available for all parts excluding the NM93C13 and NM93C14.

MICROWIRE/PLUS Peripherals (Continued)

Part Number	Description	Databook
AUDIO AND RADIO DEVICES		
LMC1982	Stereo Volume/Tone Circuit with two Selectable Stereo Inputs	Application Specific Analog Products
LMC1983	Stereo Volume/Tone Circuit with three Selectable Stereo Inputs	Application Specific Analog Products
LMC1992	Stereo Volume/Tone with Four-Channel Input Selector	Application Specific Analog Products
LMC835	7 Band Graphic Equalizer	Application Specific Analog Products
LM1971/72/73	1/2/3 Channel μPot 62/78/76 dB Audio, Attenuator with Mute	Application Specific Analog Products

National Semiconductor

ADC0811 8-Bit Serial I/O A/D Converter With 11-Channel Multiplexer

General Description

The ADC0811 is an 8-Bit successive approximation A/D converter with simultaneous serial I/O. The serial input controls an analog multiplexer which selects from 11 input channels or an internal half scale test voltage.

An input sample-and-hold is implemented by a capacitive reference ladder and sampled data comparator. This allows the input signal to vary during the conversion cycle.

Separate serial I/O and conversion clock inputs are provided to facilitate the interface to various microprocessors.

Features

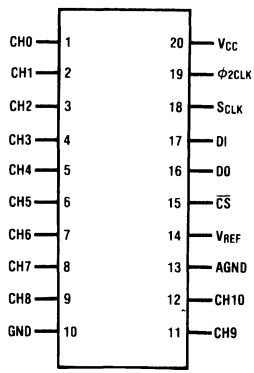
- Separate asynchronous converter clock and serial data I/O clock.
- 11-Channel multiplexer with 4-Bit serial address logic.
- Built-in sample and hold function.
- Ratiometric or absolute voltage referencing.
- No zero or full-scale adjust required.
- Internally addressable test voltage.
- 0V to 5V input range with single 5V power supply.
- TTL/MOS input/output compatible.
- 0.3" standard width 20-pin dip or 20-pin molded chip carrier

Key Specifications

■ Resolution	8-Bits
■ Total unadjusted error	± ½LSB and ± 1LSB
■ Single supply	$5V_{DC}$
■ Low Power	15 mW
■ Conversion Time	32 μS

Connection Diagrams

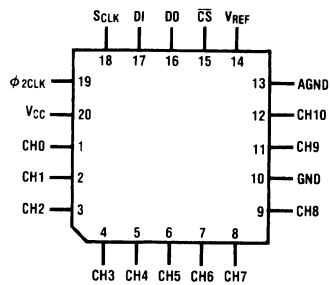
Dual-In-Line Package



TL/H/5587–1

Top View

Molded Chip Carrier (PCC) Package

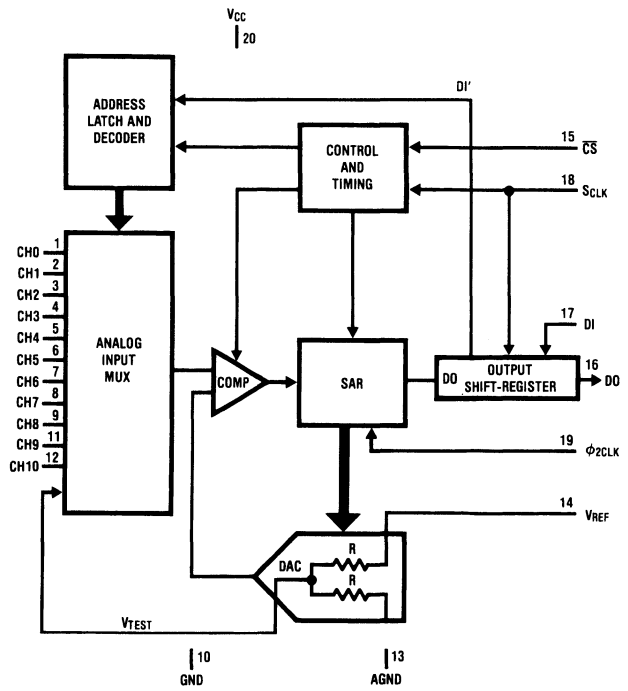


TL/H/5587–2

Top View

Order Number ADC0811J,N,V
See NS Packages J20A, N20A, V20A
Use Ordering Information

Functional Diagram



TL/H/5587–3

National Semiconductor

ADC08031/ADC08032/ADC08034/ADC08038 8-Bit High-Speed Serial I/O A/D Converters with Multiplexer Options, Voltage Reference, and Track/Hold Function

General Description

The ADC08031/ADC08032/ADC08034/ADC08038 are 8-bit successive approximation A/D converters with serial I/O and configurable input multiplexers with up to 8 channels. The serial I/O is configured to comply with the NSC MICROWIRE™ serial data exchange standard for easy interface to the COPS™ family of controllers, and can easily interface with standard shift registers or microprocessors.

The ADC08034 and ADC08038 provide a 2.6V band-gap derived reference. For devices offering guaranteed voltage reference performance over temperature see ADC08131, ADC08134 and ADC08138.

A track/hold function allows the analog voltage at the positive input to vary during the actual A/D conversion.

The analog inputs can be configured to operate in various combinations of single-ended, differential, or pseudo-differential modes. In addition, input voltage spans as small as 1V can be accommodated.

Applications

- Digitizing automotive sensors
- Process control monitoring
- Remote sensing in noisy environments
- Instrumentation
- Test systems
- Embedded diagnostics

Features

- Serial digital data link requires few I/O pins
- Analog input track/hold function
- 2-, 4-, or 8-channel input multiplexer options with address logic
- 0V to 5V analog input range with single 5V power supply
- No zero or full scale adjustment required
- TTL/CMOS input/output compatible
- On chip 2.6V band-gap reference
- 0.3" standard width 8-, 14-, or 20-pin DIP package
- 14-, 20-pin small-outline packages

Key Specifications

- Resolution — 8 bits
- Conversion time (f_C = 1 MHz) — 8μs (max)
- Power dissipation — 20mW (max)
- Single supply — $5V_{DC}$ (±5%)
- Total unadjusted error — ±½ LSB and ±1LSB
- No missing codes over temperature

Ordering Information

Industrial ($-40°C \leq T_A \leq +85°C$)	Package
ADC08031BIN, ADC08031CIN	N08E
ADC08032BIN, ADC08032CIN	N08E
ADC08034BIN, ADC08034CIN	N14A
ADC08038BIN, ADC08038CIN	N20A
ADC08031BIWM, ADC08031CIWM, ADC08032BIWM, ADC08032CIWM, ADC08034BIWM, ADC08034CIWM	M14B
ADC08038BIWM, ADC08038CIWM	M20B

National Semiconductor

ADC0852/ADC0854
Multiplexed Comparator with 8-Bit Reference Divider

General Description

The ADC0852 and ADC0854 are CMOS devices that combine a versatile analog input multiplexer, voltage comparator, and an 8-bit DAC which provides the comparator's threshold voltage (V_{TH}). The comparator provides a "1-bit" output as a result of a comparison between the analog input and the DAC's output. This allows for easy implementation of set-point, on-off or "bang-bang" control systems with several advantages over previous devices.

The ADC0854 has a 4 input multiplexer that can be software configured for single ended, pseudo-differential, and full-differential modes of operation. In addition the DAC's reference input is brought out to allow for reduction of the span.

The ADC0852 has a two input multiplexer that can be configured as 2 single-ended or 1 differential input pair. The DAC reference input is internally tied to V_{CC}.

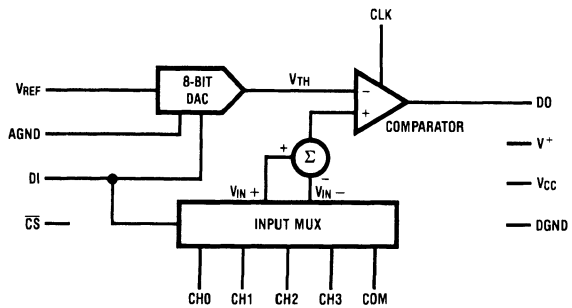
The multiplexer and 8-bit DAC are programmed via a serial data input word. Once programmed the output is updated once each clock cycle up to a maximum clock rate of 400 kHz.

Features

- 2 or 4 channel multiplexer
- Differential or Single-ended input, software controlled
- Serial digital data interface
- 256 programmable reference voltage levels
- Continuous comparison after programming
- Fixed, ratiometric, or reduced span reference capability (ADC 0854)

Key Specifications

- Accuracy, ± ½ LSB or ± 1 LSB of Reference (0.2%)
- Single 5V power supply
- Low Power, 15 mW

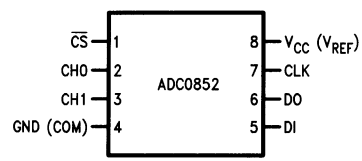


TL/H/5521–1

FIGURE 1. ADC0854 Simplified Block Diagram (ADC0852 has 2 input channels, COM tied to GND, V_{REF} tied to V_{CC}, V+ omitted, and one GND connection)

2 Channel and 4 Channel Pin Out

ADC0852 2-CHANNEL MUX
Dual-In-Line Package



TL/H/5521–10

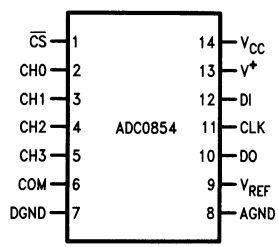
Top View

AGND and COM internally connected to GND

V_{REF} internally connected to V_{CC}

Order Number ADC0852
See NS Package Number N08E

ADC0854 4-CHANNEL MUX
Dual-In-Line Package



TL/H/5521–11

Top View

Order Number ADC0854
See NS Package Number N14A

6

National Semiconductor

ADC1031/ADC1034/ADC1038 10-Bit Serial I/O A/D Converters with Analog Multiplexer and Track/Hold Function

General Description

The ADC1031, ADC1034 and ADC1038 are 10-bit successive approximation A/D converters with serial I/O. The serial input, for the ADC1034 and ADC1038, controls a single-ended analog multiplexer that selects one of 4 input channels (ADC1034) or one of 8 input channels (ADC1038). The ADC1034 and ADC1038 serial output data can be configured into a left- or right-justified format.

An input track/hold is implemented by a capacitive reference ladder and sampled-data comparator. This allows the analog input to vary during the A/D conversion cycle.

Separate serial I/O and conversion clock inputs are provided to facilitate the interface to various microprocessors.

Applications

- Engine control
- Process control
- Instrumentation
- Test equipment

Features

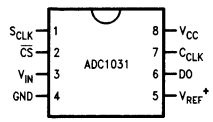
- Serial I/O (MICROWIRE™ compatible)
- Separate asynchronous converter clock and serial data I/O clock
- Analog input track/hold function
- Ratiometric or absolute voltage referencing
- No zero or full scale adjustment required
- 0V to 5V analog input range with single 5V power supply
- TTL/MOS input/output compatible
- No missing codes

Key Specifications

■ Resolution	10 bits
■ Total unadjusted error	±1 LSB (max)
■ Single supply	5V ±5%
■ Power dissipation	20 mW (max)
■ Max. conversion time (f_C = 3 MHz)	13.7 μs (max)
■ Serial data exchange time (f_S = 1 MHz)	10 μs (max)

Connection Diagrams

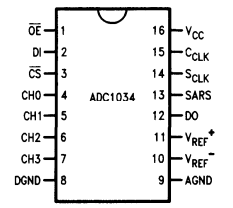
Dual-In-Line and SO Packages



TL/H/10556–4

Top View

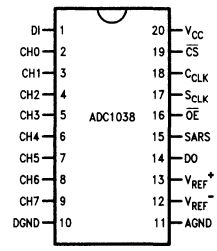
ADC1031 In NS Package N08E



TL/H/10556–3

Top View

ADC1034 In NS Packages J16A, M16B or N16E



TL/H/10556–2

Top View

ADC1038 In NS Packages J20A, M20B or N20A

Ordering Information

Industrial −40°C ≤ T_A ≤ +85°C	Package
ADC1031CIN	N08E
ADC1034CIN	N16E
ADC1034CIWM	M16B
ADC1038CIN	N20A
ADC1038CIWM	M20B
Military −55°C ≤ T_A ≤ +125°C	**Package**
ADC1034CMJ	J16A
ADC1038CMJ	J20A

National Semiconductor

ADC12H030/ADC12H032/ADC12H034/ADC12H038, ADC12030/ADC12032/ADC12034/ADC12038 Self-Calibrating 12-Bit Plus Sign Serial I/O A/D Converters with MUX and Sample/Hold

General Description

The ADC12030, and ADC12H030 families are 12-bit plus sign successive approximation A/D converters with serial I/O and configurable input multiplexers. The ADC12032/ADC12H032, ADC12034/ADC12H034 and ADC12038/ADC12H038 have 2, 4 and 8 channel multiplexers, respectively. The differential multiplexer outputs and A/D inputs are available on the MUXOUT1, MUXOUT2, A/DIN1 and A/DIN2 pins. The ADC12030/ADC12H030 has a two channel multiplexer with the multiplexer outputs and A/D inputs internally connected. The ADC12030 family is tested with a 5 MHz clock, while the ADC12H030 family is tested with an 8 MHz clock. On request, these A/Ds go through a self calibration process that adjusts linearity, zero and full-scale errors to less than ±1 LSB each.

The analog inputs can be configured to operate in various combinations of single-ended, differential, or pseudo-differential modes. A fully differential unipolar analog input range (0V to +5V) can be accommodated with a single +5V supply. In the differential modes, valid outputs are obtained even when the negative inputs are greater than the positive because of the 12-bit plus sign output data format.

The serial I/O is configured to comply with the NSC MICROWIRE™. For complementary voltage references see the LM4040, LM4041 or LM9140.

Applications

- Medical instruments
- Process control systems
- Test equipment

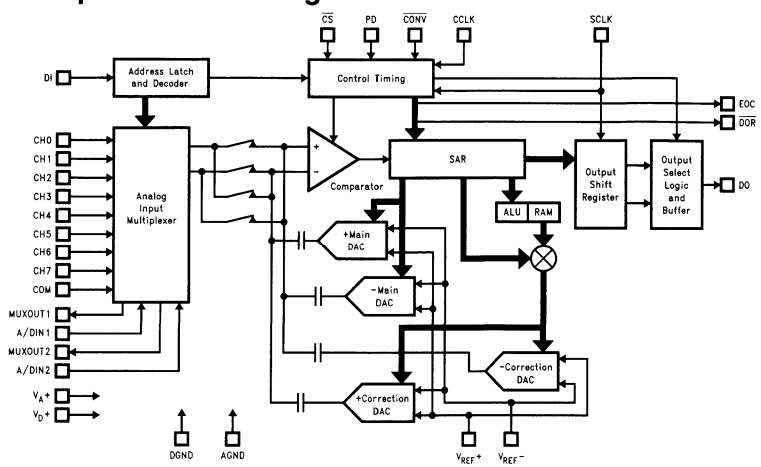
Features

- Serial I/O (MICROWIRE Compatible)
- 2, 4, or 8 channel differential or single-ended multiplexer
- Analog input sample/hold function
- Power down mode
- Variable resolution and conversion rate
- Programmable acquisition time
- Variable digital output word length and format
- No zero or full scale adjustment required
- Fully tested and guaranteed with a 4.096V reference
- 0V to 5V analog input range with single 5V power supply
- No Missing Codes over temperature

Key Specifications

■ Resolution	12-bit plus sign
■ 12-bit plus sign conversion time	
— ADC12H030 family	5.5 μs (max)
— ADC12030 family	8.8 μs (max)
■ 12-bit plus sign throughput time	
— ADC12H030 family	8.6 μs (max)
— ADC12030 family	14 μs (max)
■ Integral linearity error	±1 LSB (max)
■ Single supply	5V ±10%
■ Power dissipation	33 mW (max)
— Power down	100 μW (typ)

ADC12038 Simplified Block Diagram



TL/H/11354–1

National Semiconductor

COP472-3 Liquid Crystal Display Controller

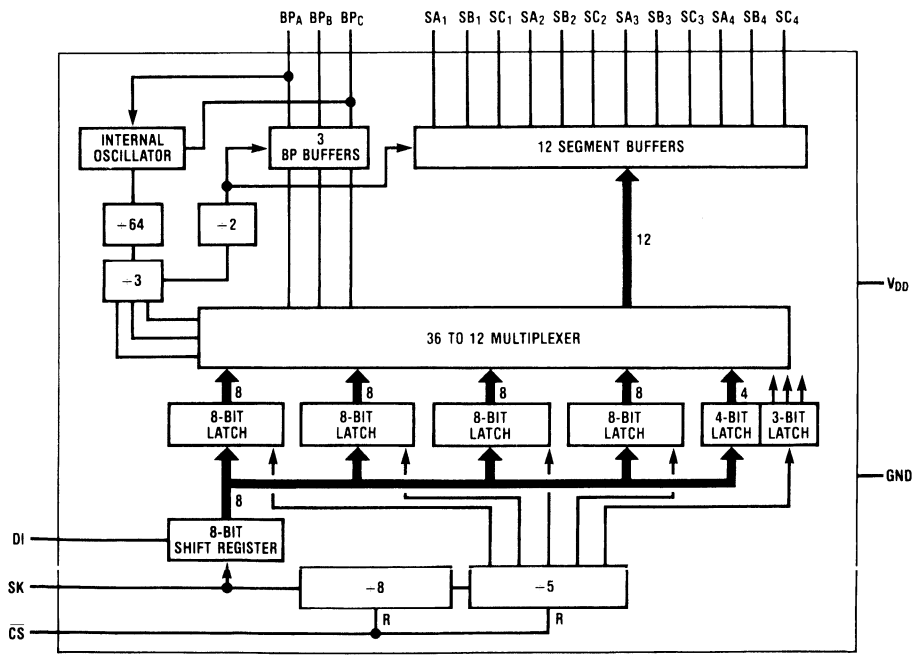
General Description

The COP472–3 Liquid Crystal Display (LCD) Controller is a peripheral member of the COPS™ family, fabricated using CMOS technology. The COP472-3 drives a multiplexed liquid crystal display directly. Data is loaded serially and is held in internal latches. The COP472-3 contains an on-chip oscillator and generates all the multi-level waveforms for backplanes and segment outputs on a triplex display. One COP472-3 can drive 36 segments multiplexed as 3 x 12 (4½ digit display). Two COP472-3 devices can be used together to drive 72 segments (3 x 24) which could be an 8½ digit display.

Features

- Direct interface to TRIPLEX LCD
- Low power dissipation (100 μW typ.)
- Low cost
- Compatible with all COPS processors
- Needs no refresh from processor
- On-chip oscillator and latches
- Expandable to longer displays
- Operates from display voltage
- MICROWIRE™ compatible serial I/O
- 20-pin Dual-In-Line package and 20-pin SO

Block Diagram



TL/DD/6932–1

Absolute Maximum Ratings

Voltage at CS, DI, SK pins	−0.3V to +9.5V	Storage Temperature	−65°C to +150°C
Voltage at all other Pins	−0.3V to V_{DD}+0.3V	Lead Temp. (Soldering, 10 Seconds)	300°C
Operating Temperature Range	0°C to 70°C		

DC Electrical Characteristics

GND = 0V, V_{DD} = 3.0V to 5.5V, T_A = 0°C to 70°C (depends on display characteristics)

Parameter	Conditions	Min	Max	Units
Power Supply Voltage, V_{DD}		3.0	5.5	Volts
Power Supply Current, I_{DD} (Note 1)	V_{DD}=5.5V		250	μA
	V_{DD}=3V		100	μA
Input Levels				
DI, SK, CS				
V_{IL}			0.8	Volts
V_{IH}		0.7 V_{DD}	9.5	Volts
BPA (as Osc. in)				
V_{IL}			0.6	Volts
V_{IH}		V_{DD}−0.6	V_{DD}	Volts
Output Levels, BPC (as Osc. Out)				
V_{OL}			0.4	Volts
V_{OH}		V_{DD}−0.4	V_{DD}	Volts
Backplane Outputs (BPA, BPB, BPC)				
$V_{BPA, BPB, BPC}$ ON	During	$V_{DD}-\Delta V$	V_{DD}	Volts
$V_{BPA, BPB, BPC}$ OFF	BP+ Time	$\frac{1}{3} V_{DD}-\Delta V$	$\frac{1}{3} V_{DD}+\Delta V$	Volts
$V_{BPA, BPB, BPC}$ ON	During	0	ΔV	Volts
$V_{BPA, BPB, BPC}$ OFF	BP− Time	$\frac{2}{3} V_{DD}-\Delta V$	$\frac{2}{3} V_{DD}+\Delta V$	Volts
Segment Outputs ($SA_1 \sim SA_4$)				
V_{SEG} ON	During	0	ΔV	Volts
V_{SEG} OFF	BP+ Time	$\frac{2}{3} V_{DD}-\Delta V$	$\frac{2}{3} V_{DD}+\Delta V$	Volts
V_{SEG} ON	During	$V_{DD}-\Delta V$	V_{DD}	Volts
V_{SEG} OFF	BP− Time	$\frac{1}{3} V_{DD}-\Delta V$	$\frac{1}{3} V_{DD}+\Delta V$	Volts
Internal Oscillator Frequency		15	80	kHz
Frame Time (Int. Osc. ÷ 192)		2.4	12.8	ms
Scan Frequency ($1/T_{SCAN}$)		39	208	Hz
SK Clock Frequency		4	250	kHz
SK Width		1.7		μs
DI				
Data Setup, t_{SETUP}		1.0		μs
Data Hold, t_{HOLD}		100		ns
$\overline{CS}$				
t_{SETUP}		1.0		μs
t_{HOLD}		1.0		μs
Output Loading Capacitance			100	pF

Note 1: Power supply current is measured in stand-alone mode with all outputs open and all inputs at V_{DD}.

Note 2: $\Delta V = 0.05 V_{DD}$.

Absolute Maximum Ratings

If Military/Aerospace specified devices are required, please contact the National Semiconductor Sales Office/Distributors for availability and specifications.

Voltage at CS, DI, SK Pins	−0.3V to +9.5V
Voltage at All Other Pins	−0.3V to V_{DD} +0.3V
Operating Temperature Range	−40°C to +85°C
Storage Temperature	−65°C to +150°C
Lead Temperature (Soldering, 10 seconds)	300°C

DC Electrical Characteristics

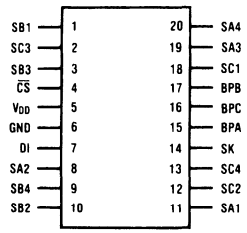
GND = 0V, V_{DD} = 3.0V to 5.5V, T_A = −40°C to +85°C (depends on display characteristics)

Parameter	Conditions	Min	Max	Units
Power Supply Voltage, V_{DD}		3.0	5.5	Volts
Power Supply Current, I_{DD} (Note 1)	V_{DD} = 5.5V		300	μA
	V_{DD} = 3V		120	μA
Input Levels				
DI, SK, CS				
V_{IL}			0.8	Volts
V_{IH}		0.7 V_{DD}	9.5	Volts
BPA (as Osc. In)				
V_{IL}			0.6	Volts
V_{IH}		V_{DD} − 0.6	V_{DD}	Volts
Output Levels, BPC (as Osc. Out)				
V_{OL}			0.4	Volts
V_{OH}		V_{DD} − 0.4	V_{DD}	Volts
Backplane Outputs (BPA, BPB, BPC)				
$V_{BPA, BPB, BPC}$ ON	During	$V_{DD} - \Delta V$	V_{DD}	Volts
$V_{BPA, BPB, BPC}$ OFF	BP+ Time	$\frac{1}{3} V_{DD} - \Delta V$	$\frac{1}{3} V_{DD} + \Delta V$	Volts
$V_{BPA, BPB, BPC}$ ON	During	0	ΔV	Volts
$V_{BPA, BPB, BPC}$ OFF	BP− Time	$\frac{2}{3} V_{DD} - \Delta V$	$\frac{2}{3} V_{DD} + \Delta V$	Volts
Segment Outputs (SA_1 ~ SA_4)				
V_{SEG} ON	During	0	ΔV	Volts
V_{SEG} OFF	BP+ Time	$\frac{2}{3} V_{DD} - \Delta V$	$\frac{2}{3} V_{DD} + \Delta V$	Volts
V_{SEG} ON	During	$V_{DD} - \Delta V$	V_{DD}	Volts
V_{SEG} OFF	BP− Time	$\frac{1}{3} V_{DD} - \Delta V$	$\frac{1}{3} V_{DD} + \Delta V$	Volts
Internal Oscillator Frequency		15	80	kHz
Frame Time (Int. Osc. ÷ 192)		2.4	12.8	ms
Scan Frequency ($1/T_{SCAN}$)		39	208	Hz
SK Clock Frequency		4	250	kHz
SK Width		1.7		μs
DI				
Data Setup, t_{SETUP}		1.0		μs
Data Hold, t_{HOLD}		100		ns
$\overline{CS}$				
t_{SETUP}		1.0		μs
t_{HOLD}		1.0		μs
Output Loading Capacitance			100	pF

Note 1: Power supply current is measured in stand-alone mode with all outputs open and all inputs at V_{DD}.

Note 2: $\Delta V = 0.05\ V_{DD}$.

Dual-In-Line Package



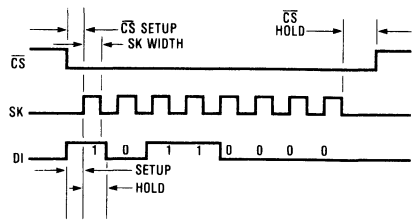
Top View

TL/DD/6932-2

Pin	Description
$\overline{CS}$	Chip select
V_{DD}	Power supply (display voltage)
GND	Ground
DI	Serial data input
SK	Serial clock input
BP_A	Display backplane A (or oscillator in)
BP_B	Display backplane B
BP_C	Display backplane C (or oscillator out)
SA1 ~ SC4	12 multiplexed outputs

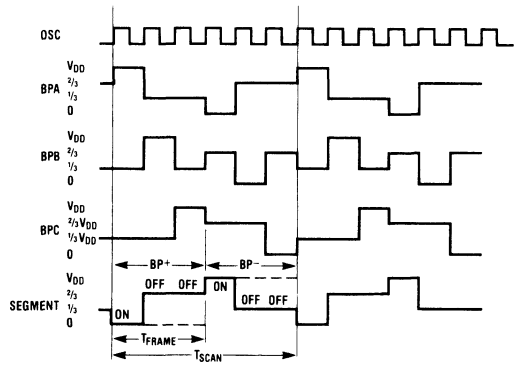
Order Number COP472MW-3 or COP472N-3
See NS Package Number M20A or N20A

FIGURE 2. Connection Diagram



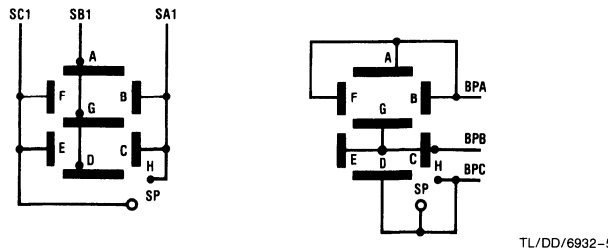
TL/DD/6932-3

FIGURE 3. Serial Load Timing Diagram



TL/DD/6932-4

FIGURE 4. Backplane and Segment Waveforms



TL/DD/6932-5

FIGURE 5. Typical Display Internal Connections
Epson LD-370

Functional Description

The COP472-3 drives 36 bits of display information organized as twelve segments and three backplanes. The COP472-3 requires 40 information bits: 36 data and 4 control. The function of each control bit is described below. Display information format is a function of the LCD interconnections. A typical segment/backplane configuration is illustrated in *Figure 5*, with this configuration the COP472-3 will drive 4 digits of 9 segments.

To adapt the COP472-3 to any LCD display configuration, the segment/backplane multiplex scheme is illustrated in Table I.

Two or more COP472-3 chips can be cascaded to drive additional segments. There is no limit to the number of COP472-3's that can be used as long as the output loading capacitance does not exceed specification.

TABLE I. COP472-3 Segment/Backplane Multiplex Scheme

Bit Number	Segment, Backplane	Data to Numeric Display	
1	SA1, BPC	SH	
2	SB1, BPB	SG	
3	SC1, BPA	SF	
4	SC1, BPB	SE	Digit 1
5	SB1, BPC	SD	
6	SA1, BPB	SC	
7	SA1, BPA	SB	
8	SB1, BPA	SA	
9	SA2, BPC	SH	
10	SB2, BPB	SG	
11	SC2, BPA	SF	
12	SC2, BPB	SE	Digit 2
13	SB2, BPC	SD	
14	SA2, BPB	SC	
15	SA2, BPA	SB	
16	SB2, BPA	SA	
17	SA3, BPC	SH	
18	SB3, BPB	SG	
19	SC3, BPA	SF	
20	SC3, BPB	SE	Digit 3
21	SB3, BPC	SD	
22	SA3, BPB	SC	
23	SA3, BPA	SB	
24	SB3, BPA	SA	
25	SA4, BPC	SH	
26	SB4, BPB	SG	
27	SC4, BPA	SF	
28	SC4, BPB	SE	Digit 4
29	SB4, BPC	SD	
30	SA4, BPB	SC	
31	SA4, BPA	SB	
32	SB4, BPA	SA	
33	SC1, BPC	SPA	Digit 1
34	SC2, BPC	SP2	Digit 2
35	SC3, BPC	SP3	Digit 3
36	SC4, BPC	SP4	Digit 4
37	not used		
38	Q6		
39	Q7		
40	SYNC		

SEGMENT DATA BITS

Data is loaded in serially, in sets of eight bits. Each set of segment data is in the following format:

SA	SB	SC	SD	SE	SF	SG	SH

Data is shifted into an eight bit shift register. The first bit of the data is for segment H, digit 1. The eighth bit is segment A, digit 1. A set of eight bits is shifted in and then loaded into the digit one latches. The second set of 8 bits is loaded into digit two latches. The third set into digit three latches, and the fourth set is loaded into digit four latches.

CONTROL BITS

The fifth set of 8 data bits contains special segment data and control data in the following format:

SYNC	Q7	Q6	X	SP4	SP3	SP2	SP1

The first four bits shifted in contain the special character segment data. The fifth bit is not used. The sixth and seventh bits program the COP472-3 as a stand alone LCD driver or as a master or slave for cascading COP472-3's. BPC of the master is connected to BPA of each slave. The following table summarizes the function of bits six and seven:

Q7	Q6	Function	BPC Output	BPA Output
1	1	Slave	Backplane Output	Oscillator Input
0	1	Stand Alone	Backplane Output	Backplane Output
1	0	Not Used	Internal Osc. Output	Oscillator Input
0	0	Master	Internal Osc. Output	Backplane Output

The eighth bit is used to synchronize two COP472-3's to drive an 8½-digit display.

LOADING SEQUENCE TO DRIVE A 4½-DIGIT DISPLAY

Steps:

1. Turn $\overline{CE}$ low.
2. Clock in 8 bits of data for digit 1.
3. Clock in 8 bits of data for digit 2.
4. Clock in 8 bits of data for digit 3.
5. Clock in 8 bits of data for digit 4.
6. Clock in 8 bits of data for special segment and control function of BPC and BPA.

0	0	1	1	SP4	SP3	SP2	SP1

7. Turn $\overline{CS}$ high.

Note: $\overline{CS}$ may be turned high after any step. For example to load only 2 digits of data, do steps 1, 2, 3, and 7.

$\overline{CS}$ must make a high to low transition before loading data in order to reset internal counters.

LOADING SEQUENCE TO DRIVE AN 8½-DIGIT DISPLAY

Two or more COP472-3's may be connected together to drive additional segments. An eight digit multiplexed display is shown in *Figure 7*. The following is the loading sequence to drive an eight digit display using two COP472-3's. The right chip is the master and the left the slave.

Steps:

1. Turn $\overline{CS}$ low on both COP472-3's.
2. Shift in 32 bits of data for the slave's four digits.
3. Shift in 4 bits of special segment data: a zero and three ones.

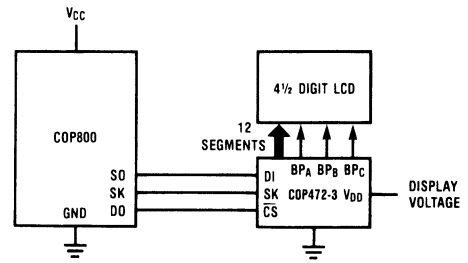
1	1	1	0	SP4	SP3	SP2	SP1

   This synchronizes both the chips and BPA is oscillator input. Both chips are now stopped.
4. Turn CS high to both chips.
5. Turn CS low to master COP472-3.
6. Shift in 32 bits of data for the master's 4 digits.
7. Shift in four bits of special segment data, a one and three zeros.

0	0	0	1	SP4	SP3	SP2	SP1

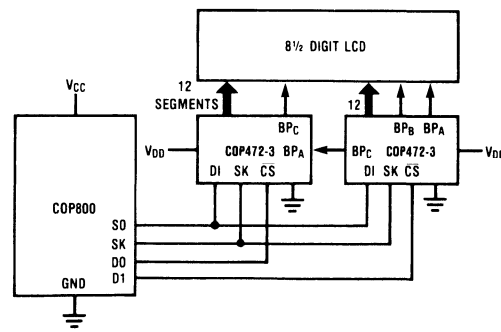
   This sets the master COP472-3 to BPA as a normal backplane output and BPC as oscillator output. Now both the chips start and run off the same oscillator.
8. Turn $\overline{CS}$ high.

The chips are now synchronized and driving 8 digits of display. To load new data simply load each chip separately in the normal manner, keeping the correct status bits to each COP472-3 (0110 or 0001).



TL/DD/6932-6

FIGURE 6. System Diagram – 4½ Digit Display



TL/DD/6932-7

FIGURE 7. System Diagram – 8½ Digit Display

National Semiconductor

MM5450/MM5451 LED Display Drivers

General Description

The MM5450 and MM5451 are monolithic MOS integrated circuits utilizing N-channel metal-gate low threshold, enhancement mode, and ion-implanted depletion mode devices. They are available in 40-pin molded or cavity dual-in-line packages. The MM5450/MM5451 is designed to drive common anode-separate cathode LED displays. A single pin controls the LED display brightness by setting a reference current through a variable resistor connected to V_{DD}.

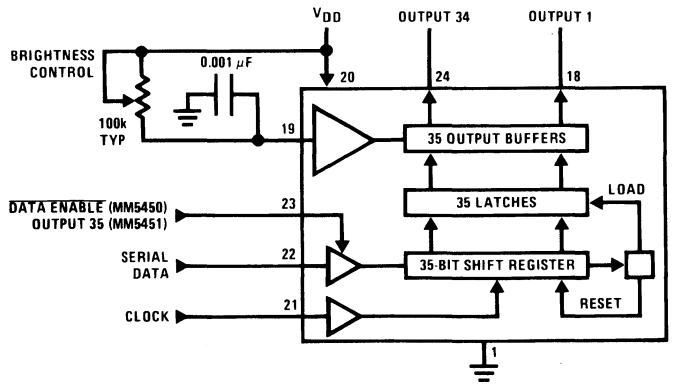
Applications

- COPS™ or microprocessor displays
- Industrial control indicator
- Relay driver
- Digital clock, thermometer, counter, voltmeter
- Instrumentation readouts

Features

- Continuous brightness control
- Serial data input
- No load signal required
- Enable (on MM5450)
- Wide power supply operation
- TTL compatibility
- 34 or 35 outputs, 15 mA sink capability
- Alphanumeric capability
- θ_{JA} DIP — Board = 49°C/W, Socket = 54°C/W

Block Diagram



TL/F/6136–1

FIGURE 1

National Semiconductor

MM5483 Liquid Crystal Display Driver

General Description

The MM5483 is a monolithic integrated circuit utilizing CMOS metal-gate low-threshold enhancement mode devices. It is available in a 40-pin molded package. The chip can drive up to 31 segments of LCD and can be cascaded to increase this number. This chip is capable of driving a 4½-digit 7-segment display with minimal interface between the display and the data source.

The MM5483 stores the display data in latches after it is latched in, and holds the data until another load pulse is received

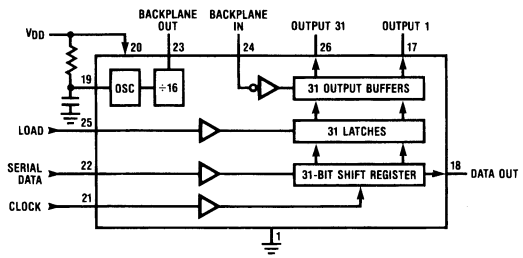
Features

- Serial data input
- Serial data output
- Wide power supply operation
- TTL compatibility
- 31 segment outputs
- Alphanumeric and bar graph capability
- Cascade capability

Applications

- COPS™ or microprocessor displays
- Industrial control indicator
- Digital clock, thermometer, counter, voltmeter
- Instrumentation readouts
- Remote displays

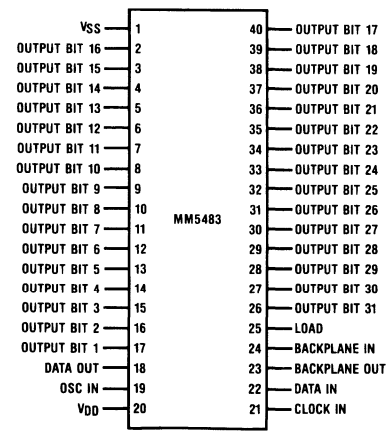
Block and Connection Diagrams



TL/F/6140–1

FIGURE 1

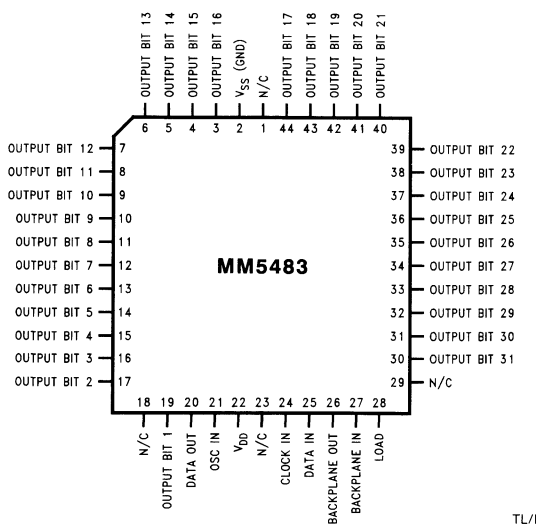
Dual-In-Line Package



TL/F/6140–2

Top View

FIGURE 2
Order Number MM5483MS or MM5483N
See SSOP Package Number MS40A
See NS Package Number N40A



TL/F/6140–7

Order Number MM5483V
See NS Package Number V44A

National Semiconductor

MM5484 16-Segment LED Display Driver

General Description

The MM5484 is a low threshold N-channel metal gate circuit using low threshold enhancement and ion implanted depletion devices. The MM5484 is available in a 22-pin molded package and is capable of driving 16 LED segments. The MM5484 is designed to drive common anode separate cathode LED displays.

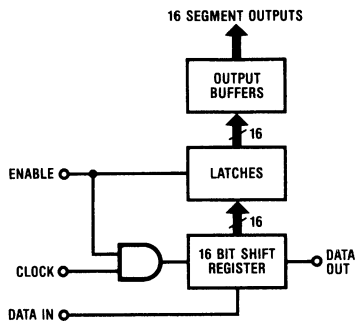
Features

- Serial data input
- Wide power supply operation
- 16 output, 15 mA sink capability
- MM5484 is cascadeable
- TTL compatibility
- No load signal required
- Non multiplex display
- 2½ digit capability—MM5484

Applications

- COPS™ or microprocessor displays
- Instrumentation readouts
- Industrial control indicator
- Relay driver

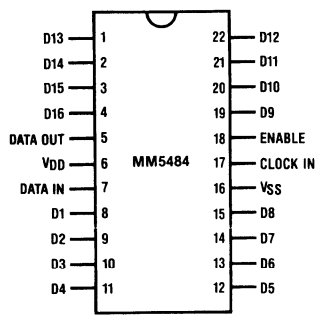
Block and Connection Diagrams



TL/F/6141–1

FIGURE 1. MM5484

Dual-In-Line Package

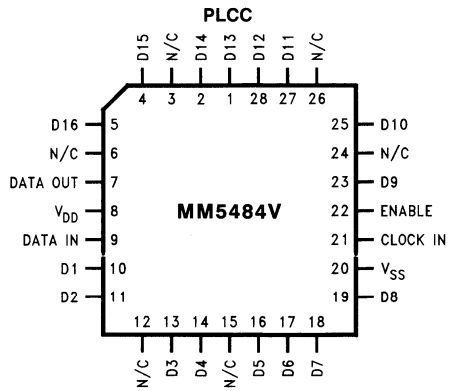


TL/F/6141–3

Top View

Order Number MM5484N
See NS Package Number N22A

PLCC



TL/F/6141–6

Top View

Order Number MM5484V
See NS Package Number V28A

National Semiconductor

MM5486 LED Display Driver

General Description

The MM5486 is a monolithic MOS integrated circuit utilizing N-channel metal-gate low-threshold, enhancement mode and ion-implanted depletion mode devices. It is available in a 40-pin molded dual-in-line package. The MM5486 is designed to drive common anode-separate cathode LED displays. A single pin controls the LED display brightness by setting a reference current through a variable resistor connected to V_{DD}.

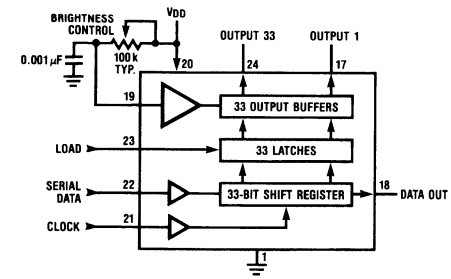
Features

- Continuous brightness control
- Serial data input/outut
- External load input
- Cascaded operation capability
- Wide power supply operation
- TTL compatibility
- 33 outputs, 15 mA sink capability
- Alphanumeric capability

Applications

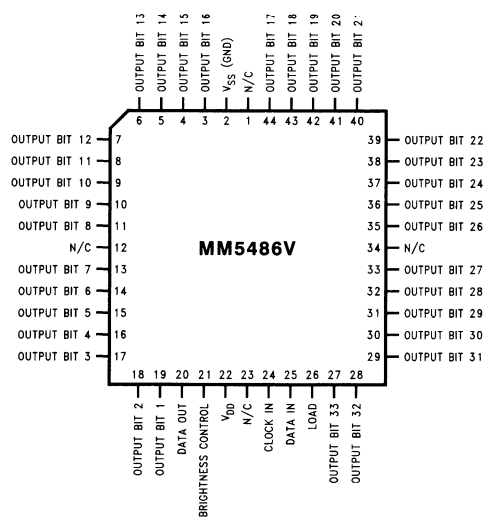
- COPS™ or microprocessor displays
- Industrial control indicator
- Relay driver
- Digital clock, thermometer, counter, voltmeter
- Instrumentation readouts

Block and Connection Diagrams



TL/F/6142–1

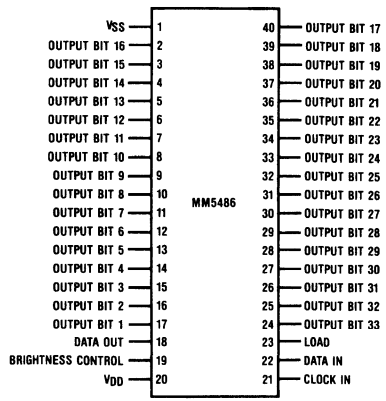
FIGURE 1



TL/F/6142–13

Order Number MM5486V
See NS Package Number V44A

Dual-In-Line Package



TL/F/6142–2

Top View

Order Number MM5486N
See NS Package Number N40A

FIGURE 2

National Semiconductor

MM58241 High Voltage Display Driver

General Description

The MM58241 is a monolithic MOS integrated circuit utilizing CMOS metal gate low threshold P- and N-channel devices. It is available both in 40-pin molded dual-in-line packages or as dice. The MM58241 is particularly suited for driving high voltage (60V max) vacuum fluorescent (VF) displays (e.g., a 32-digit alphanumeric or dot matrix display).

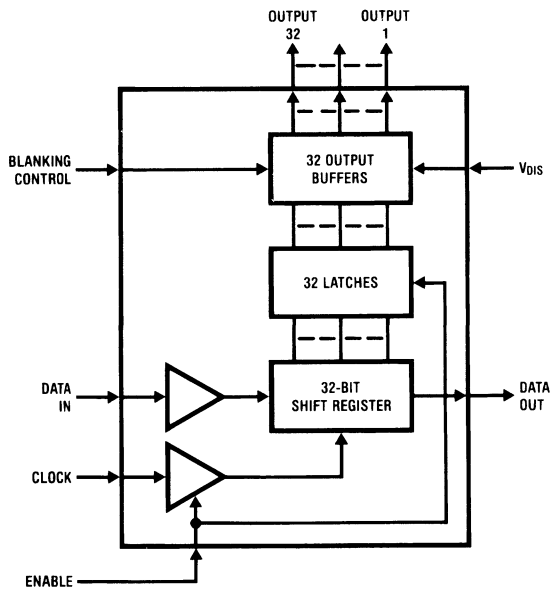
Applications

- COPS™ or microprocessor-driven displays
- Instrumentation readouts
- Industrial control indicator
- Digital clock, thermostat, counter, voltmeter
- Word processor text displays
- Automotive dashboards

Features

- Direct interface to high voltage display
- Serial data input
- No external resistors required
- Wide display power supply operation
- LSTTL compatible inputs
- Software compatible with NS display driver family
- Compatible with alphanumeric or dot matrix displays
- Display blanking control input
- Simple to cascade

Block Diagram



TL/F/5600–1

FIGURE 1

National Semiconductor

MM58341 High Voltage Display Driver

General Description

The MM58341 is a monolithic MOS integrated circuit utilizing CMOS metal gate low threshold P and N-channel devices. It is available both in 40-pin molded dual-in-line packages or as dice. The MM58341 is particularly suited for driving high voltage (35V max) vacuum fluorescent (VF) displays, (e.g., a 32-digit alphanumeric or dot matrix display).

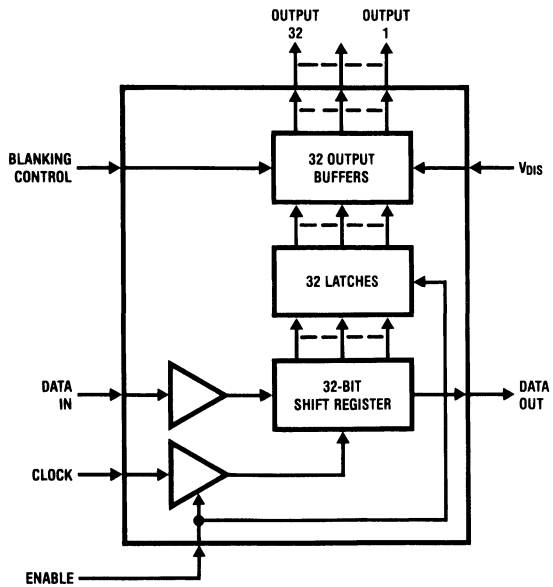
Applications

- COPS™ or microprocessor-driven displays
- Instrumentation readouts
- Industrial control indicator
- Digital clock, thermostat, counter, voltmeter
- Word processor text displays
- Automotive dashboards

Features

- Direct interface to high voltage display
- Serial data input
- No external resistors required
- Wide display power supply operation
- LSTTL compatible inputs
- Software compatible with NS display driver family
- Compatible with alphanumeric or dot matrix displays
- Display blanking control input
- Simple to cascade

Block Diagram



TL/F/5603–1

FIGURE 1

National Semiconductor

MM58342 High Voltage Display Driver

General Description

The MM58342 is a monolithic MOS integrated circuit utilizing CMOS metal gate low threshold P- and N-channel devices. It is available both in 28-pin molded dual-in-line packages or as dice. The MM58342 is particularly suited for driving high voltage (35V max) vacuum fluorescent (VF) displays (e.g., a 20-digit alphanumeric or dot matrix display).

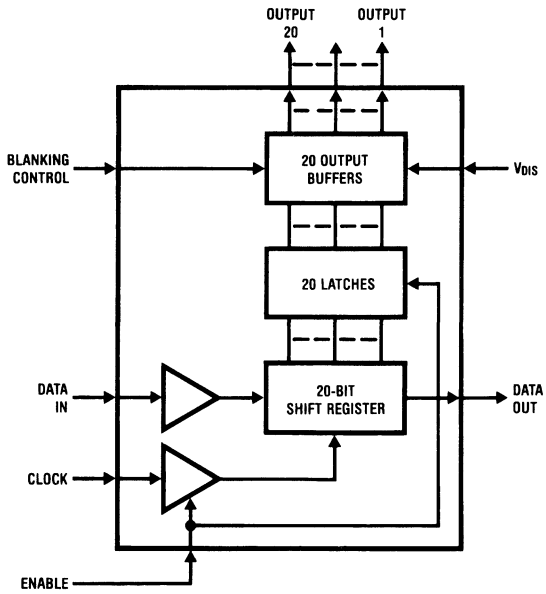
Applications

- COPS™ or microprocessor-driven displays
- Instrumentation readouts
- Industrial control indicator
- Digital clock, thermostat, counter, voltmeter
- Word processor text displays
- Automotive dashboards

Features

- Direct interface to high voltage display
- Serial data input
- No external resistors required
- Wide display power supply operation
- LSTTL compatible inputs
- Software compatible with NS display driver family
- Compatible with alphanumeric or dot matrix displays
- Display blanking control input
- Simple to cascade

Block Diagram



TL/F/7925–1

FIGURE 1

National Semiconductor

NM93C13/C14
256-/1024-Bit Serial EEPROM

General Description

The NM93C13/C14 is 256/1024, respectively, bits of CMOS electrically erasable memory divided into 16/64 16-bit registers. They are fabricated using National Semiconductor's floating-gate CMOS process for high speed, high reliability and low power. The NM93C13/C14 is available in an 8-pin SO package to save board space.

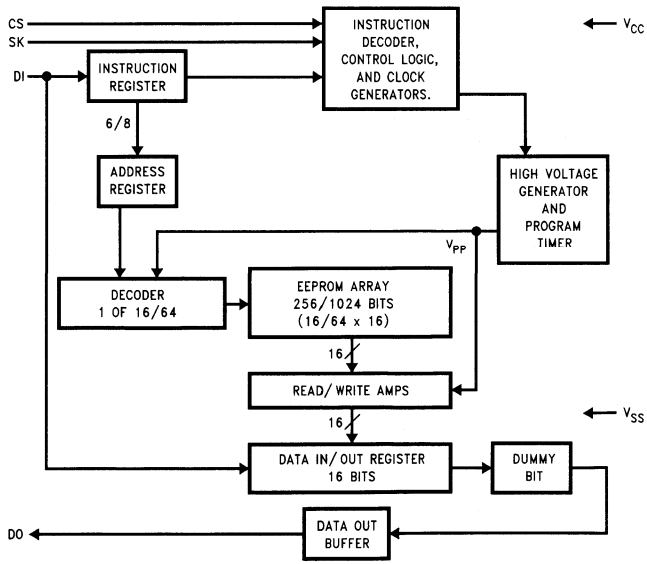
The serial interface of the NM93C13/C14 is MICROWIRE™ compatible for simple interface to standard microcontrollers and microprocessors. There are 7 instructions: Read, Erase/Write Enable, Erase, Erase All, Write, Write All, and Erase/Write Disable.

All programming cycles are completely self-timed for simplified operation. The ready/busy status is available on the DO pin to indicate the completion of a programming cycle.

Features

- Typical active current 400 μA; Typical standby current 25 μA
- Reliable CMOS floating gate technology
- 4.5V to 5.5V operation in all modes
- MICROWIRE compatible serial I/O
- Self-timed programming cycle
- Device status indication during programming mode
- 15 years data retention
- Endurance: 100,000 read/write cycles minimum
- Packages available: 8-pin DIP, 8-pin SO

Block Diagram



TL/D/11291–1

6

National Semiconductor

NM93C06/C46/C56/C66
256-/1024-/2048-/4096-Bit Serial EEPROM (MICROWIRE™ Bus Interface)

General Description

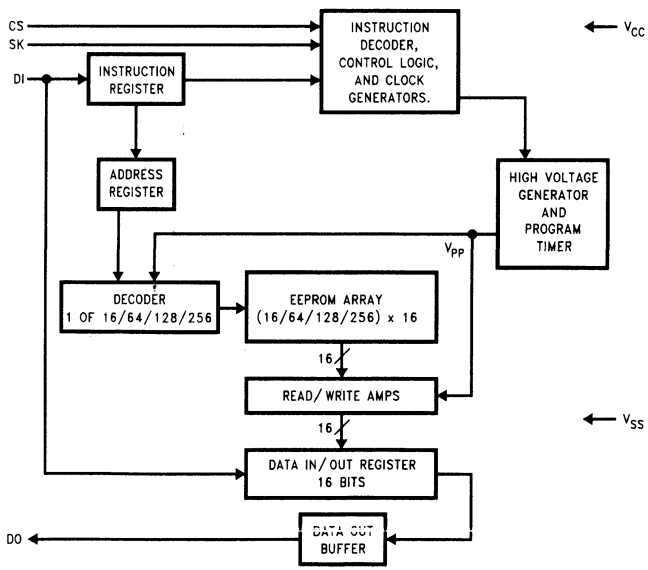
The NM93C06/C46/C56/C66 devices are 256/1024/2048/4096 bits, respectively, of CMOS non-volatile electrically erasable memory divided into 16/64/128/256 16-bit registers. They are fabricated using National Semiconductor's floating-gate CMOS process for high reliability and low power consumption. These memory devices are available in both SO and TSSOP packages for small space considerations.

The EEPROM Interfacing is MICROWIRE compatible for simple interface to standard microcontrollers and microprocessors. There are 7 instructions that control these devices: Read, Erase/Write Enable, Erase, Erase All, Write, Write All, and Erase/Write Disable. The ready/busy status is available on the DO pin during programming.

Features

- Device status during programming mode
- Typical active current of 200 μA; Typical standby current of 10 μA
- No erase required before write
- Reliable CMOS floating gate technology
- 4.5V to 5.5V operation in all modes
- MICROWIRE compatible serial I/O
- Self-timed programming cycle
- 40 years data retention
- Endurance: 10^6 data changes
- Packages available: 8-pin SO, 8-pin DIP, 8-pin TSSOP

Block Diagram



TL/D/10751–1

National Semiconductor

NM93C46A
1,024-Bit Serial Interface, Standard Voltage CMOS EEPROM (MICROWIRE™ Synchronous Bus)

General Description

The NM93C46A is 1,024 bits of CMOS nonvolatile, electrically erasable memory available as either 64 16-bit registers or 128 8-bit registers. The user organization is determined by the status of the ORG input. The memory device is fabricated using National Semiconductor's floating gate CMOS process for high reliability, high endurance, and low power consumption. The NM93C46A is available in both 8-pin SO and TSSOP packages for space considerations.

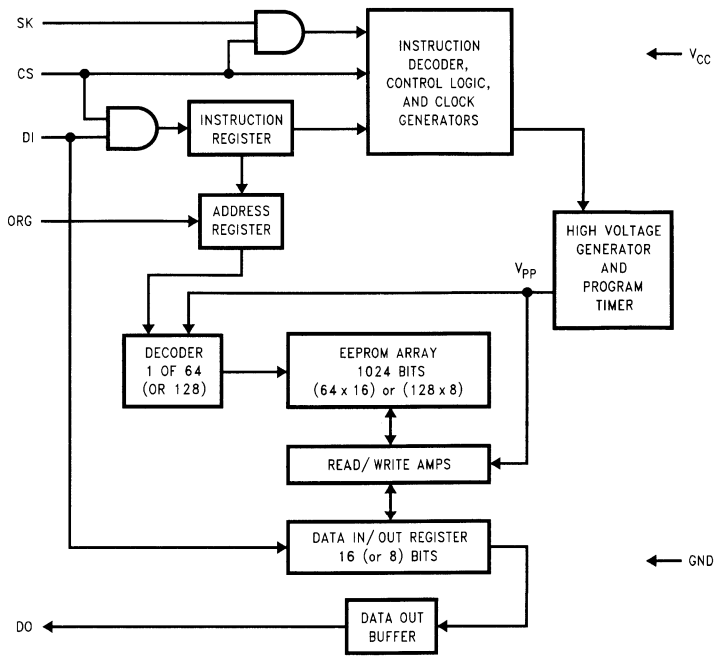
The EEPROM is MICROWIRE compatible for simple interfacing to a wide variety of microcontrollers and microprocessors. There are 7 instructions that operate the NM93C46A: Read, Erase/Write Enable, Erase, Write, Erase/Write Disable, Write All, and Erase All.

The NM93C46A defaults to the 64 x 16 configuration if the ORG pin (Pin 6) is left floating, as it is internally pulled up to V_{CC}.

Features

- 4.5V to 5.5V operation in all modes
- Typical active current of 200 μA; typical standby current of 10 μA
- Self-timed programming cycle
- Device status indication during programming mode
- No erase required before write
- Reliable CMOS floating gate technology
- MICROWIRE compatible serial I/O
- 40 years data retention
- Endurance: 10^6 data changes
- Packages available: 8-pin TSSOP, 8-pin SO, 8-pin DIP

Block Diagram



TL/D/11042–1

National Semiconductor

NM93C56A
2,048-Bit Serial Interface, Standard Voltage CMOS EEPROM (MICROWIRE™ Synchronous Bus)

General Description

The NM93C56A is 2,048 bits of CMOS non-volatile, electrically erasable memory user organized as either 128 16-bit registers or 256 8-bit registers. The user organization is determined by the status of the ORG input. The memory device is fabricated using National Semiconductor's floating gate CMOS process for high reliability, high endurance and low power consumption. The NM93C56A is available in both 8-pin SO and TSSOP packages for space considerations.

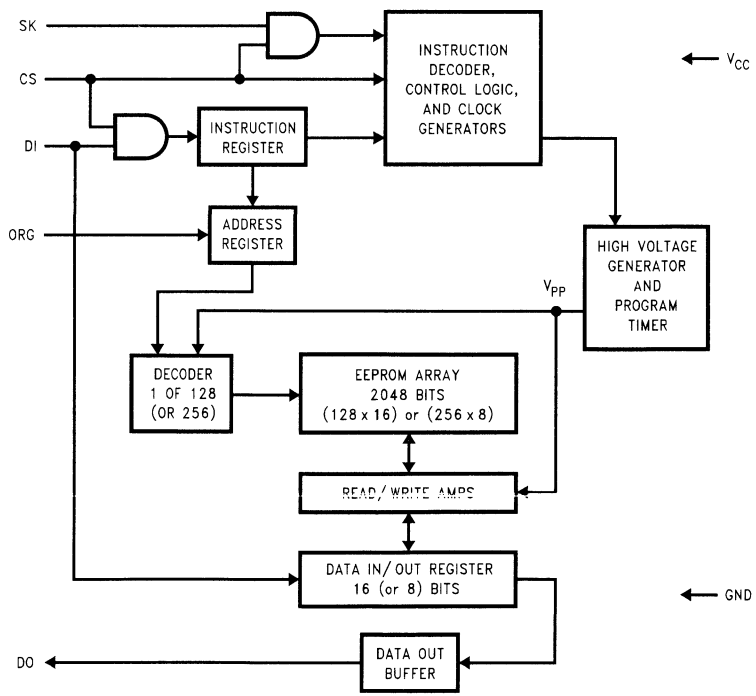
The EEPROM is MICROWIRE compatible for simple interfacing to a wide variety of microcontrollers and microprocessors. There are 7 instructions that operate the NM93C56A: Read, Erase/Write Enable, Erase, Write, Erase/Write Disable, Write All, and Erase All.

The NM93C56A defaults to the 128 x 16 configuration if the ORG pin (Pin 6) is left floating, as it is internally pulled up to V_{CC}.

Features

- 4.5V to 5.5V operation in all modes
- Typical active current of 200 μA; typical standby current of 10 μA
- Self-timed programming cycle
- Device status indication during programming mode
- No erase required before write
- Reliable CMOS floating gate technology
- MICROWIRE compatible serial I/O
- 40 years data retention
- Endurance: 10^6 data changes
- Packages available: 8-Pin TSSOP, 8-pin SO, 8-pin DIP

Block Diagram



TL/D/12509–1

National Semiconductor

NM93C66A
4,096-Bit Serial Interface, Standard Voltage CMOS EEPROM (MICROWIRE™ Synchronous Bus)

General Description

The NM93C66A is 4,096 bits of CMOS non-volatile, electrically erasable memory available user organized as either 256 16-bit registers or 512 8-bit registers. The user organization is determined by the status of the ORG input. The memory device is fabricated using National Semiconductor's floating gate CMOS process for high reliability, high endurance and low power consumption. The NM93C66A is available in both 8-pin SO and TSOP packages for space considerations.

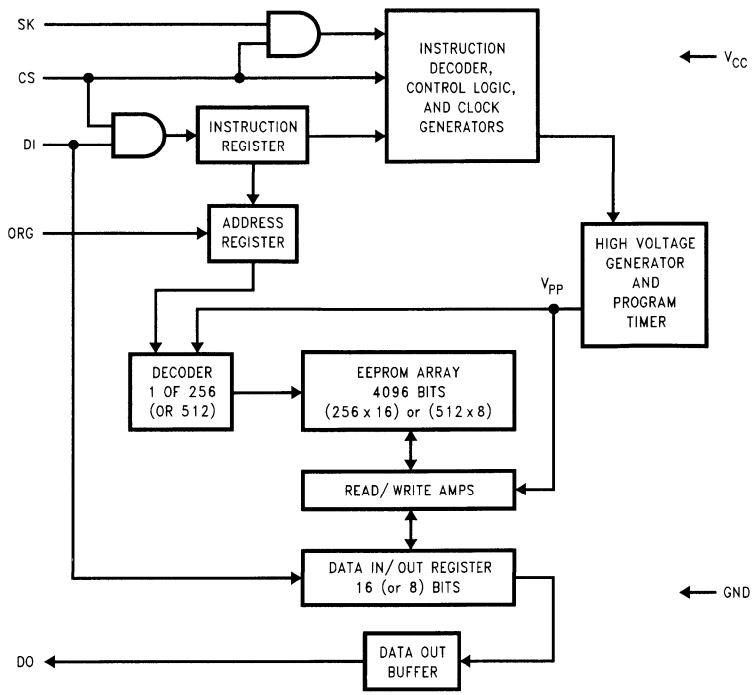
The EEPROM is MICROWIRE compatible for simple interfacing to a wide variety of microcontrollers and microprocessors. There are 7 instructions that operate the NM93C66A: Read, Erase/Write Enable, Erase, Write, Erase/Write Disable, Write All, and Erase All.

The NM93C66A defaults to the 256 x 16 configuration if the ORG pin (Pin 6) is left floating, as it is internally pulled up to V_{CC}.

Features

- 4.5V to 5.5V operation in all modes
- Typical active current of 200 μA; typical standby current of 10 μA
- Self-timed programming cycle
- Device status indication during programming mode
- No erase required before write
- Reliable CMOS floating gate technology
- MICROWIRE compatible serial I/O
- 40 years data retention
- Endurance: 10^6 data changes
- Packages available: 8-pin TSSOP, 8-pin SO, 8-pin DIP

Block Diagram



TL/D/12510–1

National Semiconductor

NM93C86A
16,384-Bit Serial Interface, Standard Voltage CMOS EEPROM (MICROWIRE™ Synchronous Bus)

General Description

The NM93C86A is 16,384 bits of CMOS nonvolatile, electrically erasable memory available in user organized as either 1024 16-bit registers or 2048 8-bit registers. The user organization is determined by the status of the ORG input. The memory device is fabricated using National Semiconductor's floating gate CMOS process for high reliability, high endurance and low power consumption. The NM93C86A is available in an 8-pin SO package for space considerations.

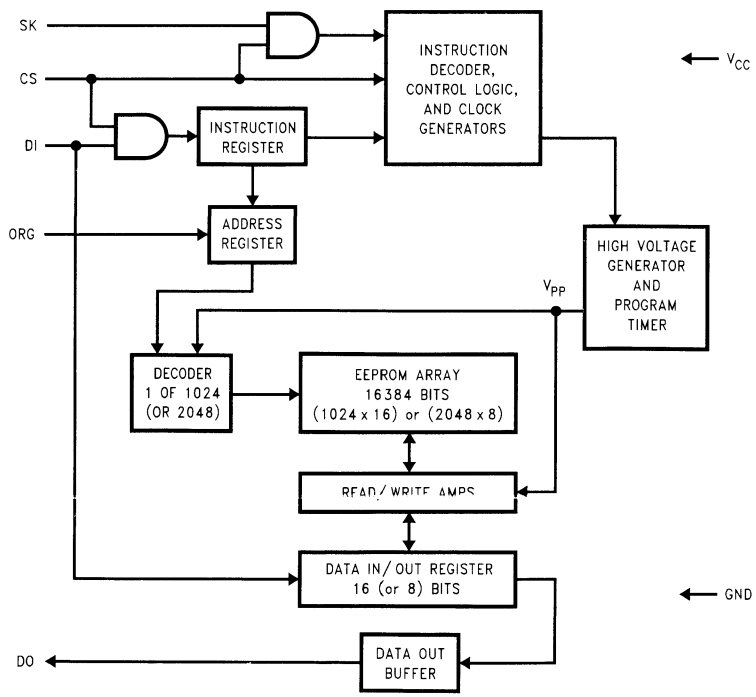
The EEPROM is MICROWIRE™ compatible for simple interfacing to a wide variety of microcontrollers and microprocessors. There are 7 instructions that operate the NM93C86A: Read, Erase/Write Enable, Erase, Write, Erase/Write Disable, Write All, and Erase All.

The NM93C86A defaults to the 1024 x 16 configuration if the ORG pin (Pin 6) is left floating, as it is internally pulled up to V_{CC}.

Features

- 4.5V to 5.5V operation in all modes
- Typical active current of 200 μA; typical standby current of 10 μA
- Device status indication during programming mode
- No erase required before write
- Reliable CMOS floating gate technology
- MICROWIRE™ compatible serial I/O
- Self-timed programming cycle
- 40 years data retention
- Endurance: 10^6 data changes
- Packages available: 8-pin SO, 8-pin DIP

Block Diagram



TL/D/11254–12

National Semiconductor

NM93CS06/CS46/CS56/CS66 (MICROWIRE™ Bus Interface) 256-/1024-/2048-/4096-Bit Serial EEPROM with Data Protect and Sequential Read

General Description

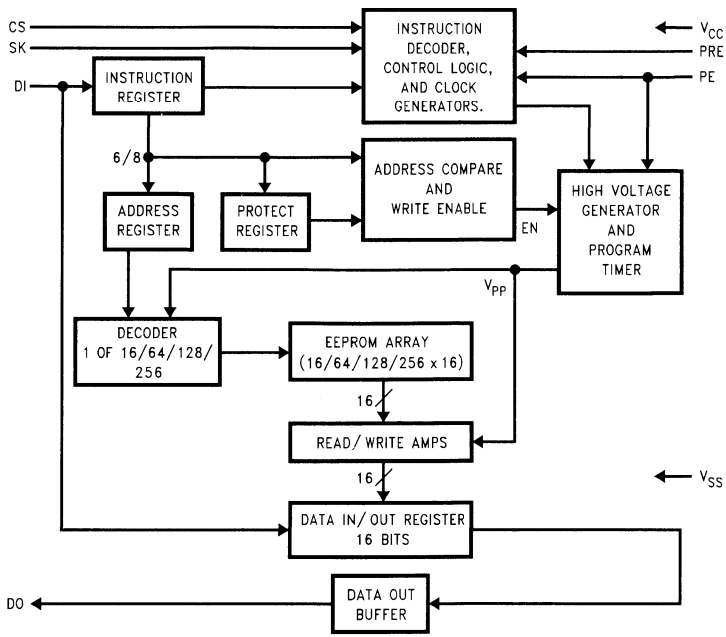
The NM93CS06/CS46/CS56/CS66 devices are 256/1024/2048/4096 bits, respectively, of CMOS non-volatile electrically erasable memory divided into 16/64/128/256 16-bit registers. Selected registers can be protected against data modification by programming the Protect Register with the address of the first register to be protected against data modification (all registers greater than, or equal to, the selected address are then protected from further change). Additionally, this address can be "locked" into the device, making all future attempts to change data impossible. These devices are fabricated using National Semiconductor floating-gate CMOS process for high reliability, high endurance and low power consumption. The NM93CSXX Family is offered in both SO and TSSOP packages for small space considerations.

The EEPROM interfacing is MICROWIRE compatible providing simple interfacing to standard microcontrollers and microprocessors. There are a total of 10 instructions, 5 which operate on the EEPROM memory, and 5 which operate on the Protect Register. The memory instructions are READ, WRITE, WRITE ALL, WRITE ENABLE, and WRITE DISABLE. The Protect register instructions are PRREAD, PRWRITE, PRENABLE, PRCLEAR, and PRDISABLE.

Features

- Write protection in a user defined section of memory
- Sequential register read
- Typical active current of 200 μA and standby current of 10 μA
- No erase required before write
- Reliable CMOS floating gate technology
- MICROWIRE compatible serial I/O
- Self timed write cycle
- Device status during programming mode
- 40 year data retention
- Endurance: 10^6 data changes
- 4.5V to 5.5V operation in all modes of operation
- Packages available: 8-pin SO, 8-pin DIP, 8-pin TSSOP

Block Diagram



TL/D/10750–1

National Semiconductor

LMC1982
Digitally-Controlled Stereo Tone and Volume Circuit with Two Selectable Stereo Inputs

General Description

The LMC1982 is a monolithic integrated circuit that provides volume, balance, tone (bass and treble), enhanced stereo, and loudness controls and selection between two pairs of stereo inputs. These functions are digitally controlled through a three-wire communication interface. There are two digital inputs for easy interface to other audio peripherals such as stereo decoders. The LMC1982 is designed for line level input signals (300 mV–2V) and has a maximum gain of −0.5 dB. Volume is set at minimum and tone controls are flat when supply voltage is first applied.

Low noise and distortion result from using analog switches and poly-silicon resistor networks in the signal path.

Additional tone control can be achieved using the LMC835 stereo 7-band graphic equalizer connected to the LMC1982's SELECT OUT/SELECT IN external processor loop.

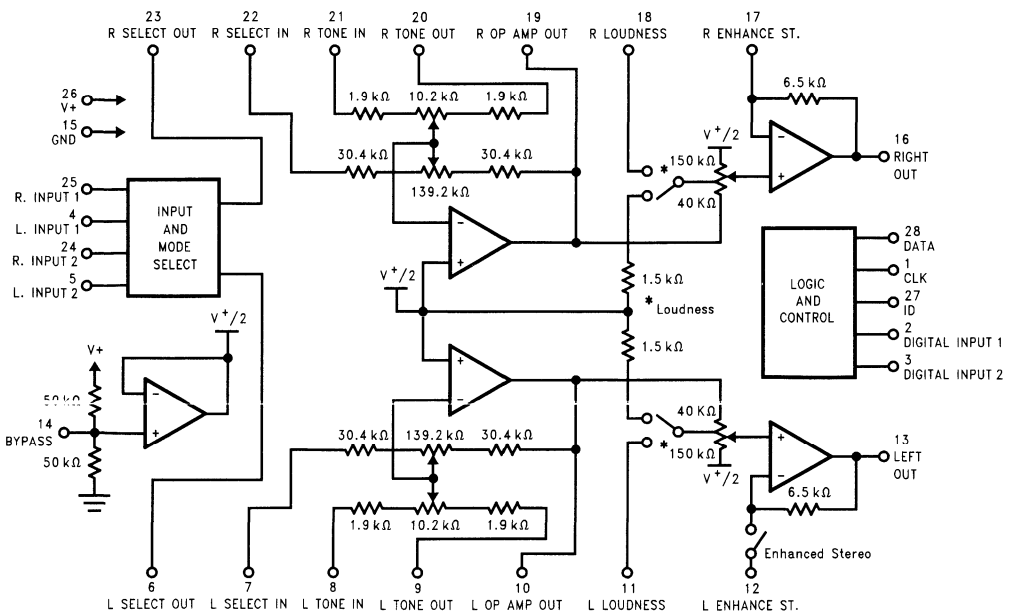
Features

- Low noise and distortion
- Two pairs of stereo inputs
- Enhanced stereo function
- Loudness compensation
- 40 position 2 dB/step volume attenuator plus mute
- Independent left and right volume controls
- Low noise-suitable for use with DNR® and Dolby® noise reduction
- External processor loop
- Signal handling suitable for compact discs
- Pop-free switching
- Serially programmable: INTERMETAL bus (IM) interface
- 6V to 12V single supply operation
- 28 Pin DIP or PLCC package

Applications

- Stereo television
- Music reproduction systems
- Sound reinforcement systems
- Electronic music (MIDI)
- Personal computer audio control

Block and Connection Diagrams



TL/H/11028–1

National Semiconductor

LMC1983
Digitally-Controlled Stereo Tone and Volume Circuit with Three Selectable Stereo Inputs

General Description

The LMC1983 is a monolithic integrated circuit that provides volume, balance, tone (bass and treble), loudness controls and selection between three pairs of stereo inputs. These functions are digitally controlled through a three-wire communication interface. There are two digital inputs for easy interface to other audio peripherals such as stereo decoders. The LMC1983 is designed for line level input signals (300 mV–2V) and has a maximum gain of −0.5 dB. Volume is set at minimum and tone controls are flat when supply voltage is first applied.

Low noise and distortion result from using analog switches and poly-silicon resistor networks in the signal path.

Additional tone control can be achieved using the LMC835 stereo 7-band graphic equalizer connected to the LMC1983's SELECT OUT/SELECT IN external processor loop.

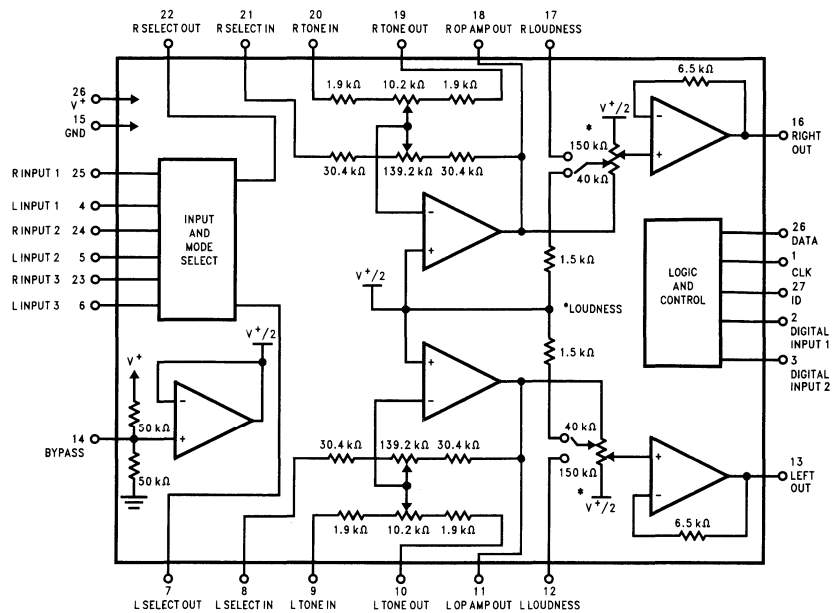
Features

- Low noise and distortion
- Three pairs of stereo inputs
- Loudness compensation
- 40 position 2 dB/step volume attenuator plus mute
- Independent left and right volume controls
- Low noise-suitable for use with DNR® and Dolby® noise reduction
- External processor loop
- Signal handling suitable for compact discs
- Pop-free switching
- Serially programmable: INTERMETAL bus (IM) interface
- 6V to 12V single supply operation
- 28 Pin DIP or PLCC Package

Applications

- Stereo television
- Music reproduction systems
- Sound reinforcement systems
- Electronic music (MIDI)
- Personal computer audio control

Block Diagram



TL/H/11279–1

National Semiconductor

LMC1992 Digitally-Controlled Stereo Tone and Volume Circuit with Four-Channel Input-Selector

General Description

The LMC1992 is a monolithic integrated circuit that provides four stereo inputs, bass and treble tone controls, and volume, balance, and front-rear fader controls. These functions are digitally controlled through a three-wire communication interface. All of the LMC1992s functions are achieved with only three external capacitors per channel. It is designed for line level input signals (300 mV – 2V) and has a maximum gain of 0 dB.

The internal design is optimized for external capacitors having values of 0.1 μF or less. This allows the use of chip capacitors for coupling and tone control functions.

Low noise and distortion result from using analog switches and thin-film silicon-chromium resistor networks in the signal path.

Volume and fader are at minimum and tone controls are flat when supply voltage is first applied.

Additional tone control can be achieved using the LMC835 stereo 7-band graphic equalizer connected to the LMC1992's select-out/select-in external processor loop.

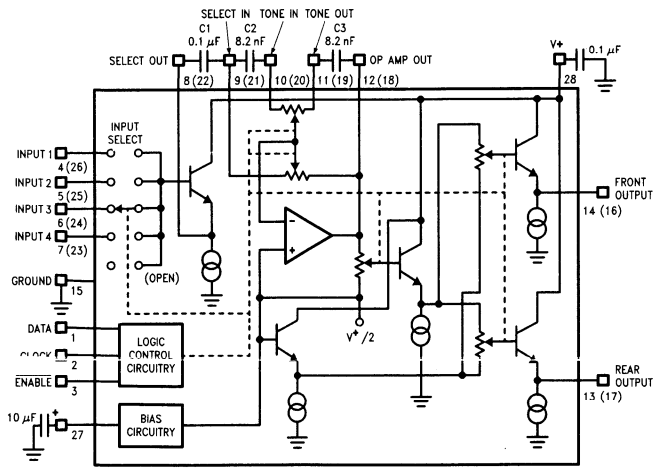
Features

- Low noise and distortion
- Four stereo inputs
- 40 volume levels including mute
- 20 fader levels
- All attenuators have a 2 dB of attenuation per step
- Front/back fade control
- External processor loop
- Only three external components per channel
- Serial programmable: standard MICROWIRE™ interface
- Single supply operation: 6V to 12V supply voltage
- Protection address (similar to DS8906)
- DC-coupled inputs
- Single supply operation

Applications

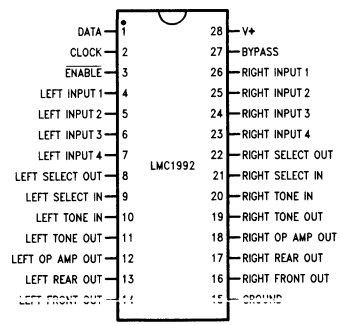
- Automotive audio systems
- Sound reinforcement systems
- Home entertainment—stereo television and music reproduction systems
- Electronic music (MIDI)

Block and Connection Diagrams



TL/H/10789-1

Left channel shown. Pin numbers in parentheses are for the right channel.



TL/H/10789-2

Order Number LMC1992CCN
See NS Package Number N28B

National Semiconductor

LMC835 Digital Controlled Graphic Equalizer

General Description

The LMC835 is a monolithic, digitally-controlled graphic equalizer CMOS LSI for Hi-Fi audio. The LMC835 consists of a Logic section and a Signal Path section made of analog switches and thin-film silicon-chromium resistor networks. The LMC835 is used with external resonator circuits to make a stereo equalizer with seven bands, ±12 dB or ±6 dB gain range and 25 steps each. Only three digital inputs are needed to control the equalization. The LMC835 makes it easy to build a μP-controlled equalizer.

The signal path is designed for very low noise and distortion, resulting in very high performance, compatible with PCM audio.

Features

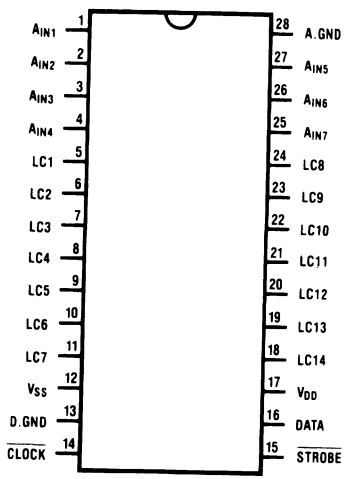
- No volume controls required
- Three-wire interface
- 14 bands, 25 steps each
- ±12 dB or ±6 dB gain ranges
- Low noise and distortion
- TTL, CMOS logic compatible

Applications

- Hi-Fi equalizer
- Receiver
- Car stereo
- Musical instrument
- Tape equalization
- Mixer
- Volume controller

Connection Diagrams

Dual-In-Line Package

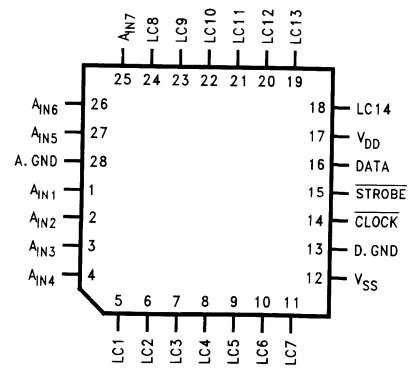


TL/H/6753-1

Top View

Order Number LMC835N
See NS Package N28B

Molded Chip Carrier Package



TL/H/6753-26

Top View

Order Number LMC835V
See NS Package V28A

National Semiconductor

LM1971 *Overture*™ Audio Attenuator Series
Digitally Controlled 62 dB Audio Attenuator with/Mute

General Description

The LM1971 is a digitally controlled single channel audio attenuator fabricated on a CMOS process. Attenuation is variable in 1 dB steps from 0 dB to −62 dB. A mute function disconnects the input from the output, providing over 100 dB of attenuation.

The performance of the device is exhibited by its ability to change attenuation levels without audible clicks or pops. In addition, the LM1971 features a low Total Harmonic Distortion (THD) of 0.0008%, and a Dynamic Range of 115 dB, making it suitable for digital audio needs. The LM1971 is available in both 8-pin plastic DIP or SO packages.

The LM1971 is controlled by a TTL/CMOS compatible 3-wire serial digital interface. The active low LOAD line enables the data input registers while the CLOCK line provides system timing. Its DATA pin receives serial data on the rising edge of each CLOCK pulse, allowing the desired attenuation setting to be selected.

Key Specifications

- Total harmonic distortion 0.0008% (typ)
- Frequency response > 200 kHz (−3 dB) (typ)
- Attenuation range (excluding mute) 62 dB (typ)
- Dynamic range 115 dB (typ)
- Mute attenuation 102 dB (typ)

Features

- 3-wire serial interface
- Mute function
- Click and pop free attenuation changes
- 8-pin plastic DIP and SO packages available

Applications

- Communication systems
- Cellular Phones and Pagers
- Personal computer audio control
- Electronic music (MIDI)
- Sound reinforcement systems
- Audio mixing automation

Typical Application

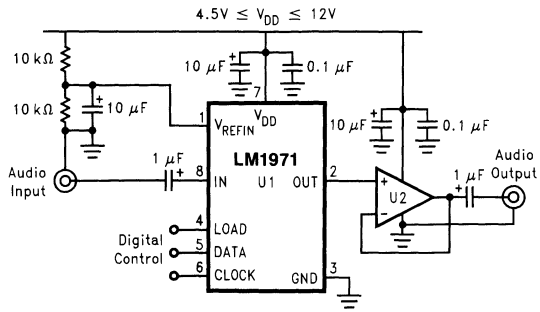
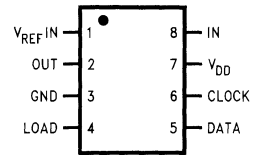


FIGURE 1. Typical Audio Attenuator Application Circuit

Connection Diagram

Dual-In-Line Plastic or Surface Mount Package



Top View

Order Number LM1971M or LM1971N
See NS Package Number M08A or N08E

National Semiconductor

LM1972
μPot™ 2-Channel 78dB Audio Attenuator with Mute

General Description

The LM1972 is a digitally controlled 2-channel 78dB audio attenuator fabricated on a CMOS process. Each channel has attenuation steps of 0.5dB from 0dB–47.5dB, 1.0dB steps from 48dB–78dB, with a mute function attenuating 104dB. Its logarithmic attenuation curve can be customized through software to fit the desired application.

The performance of a μPot is demonstrated through its excellent Signal-to-Noise Ratio, extremely low (THD+N), and high channel separation. Each μPot contains a mute function that disconnects the input signal from the output, providing a minimum attenuation of 96dB. Transitions between any attenuation settings are pop free.

The LM1972's 3-wire serial digital interface is TTL and CMOS compatible; receiving data that selects a channel and the desired attenuation level. The Data-Out pin of the LM1972 allows multiple μPots to be daisy-chained together, reducing the number of enable and data lines to be routed for a given application.

Key Specifications

- Total Harmonic Distortion + Noise 0.003% (max)
- Frequency response 100 kHz (−3dB) (min)
- Attenuation range (excluding mute) 78dB (typ)
- Differential attenuation ±0.25dB (max)
- Signal-to-noise ratio (ref. 4 Vrms) 110dB (min)
- Channel separation 100dB (min)

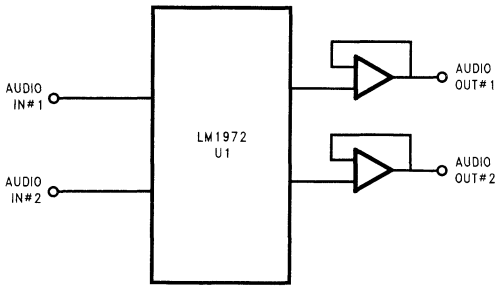
Features

- 3-wire serial interface
- Daisy-chain capability
- 104dB mute attenuation
- Pop and click free attenuation changes

Applications

- Automated studio mixing consoles
- Music reproduction systems
- Sound reinforcement systems
- Electronic music (MIDI)
- Personal computer audio control

Typical Application

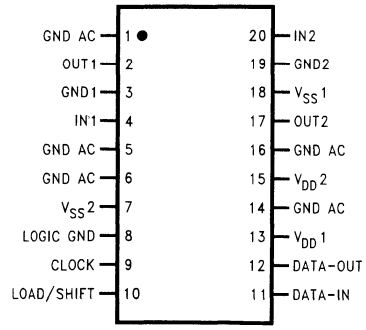


TL/H/11978–1

FIGURE 1. Typical Audio Attenuator Application Circuit

Connection Diagram

Dual-In-Line Plastic or Surface Mount Package



TL/H/11978–2

Top View

Order Number LM1972M or LM1972N
See NS Package Number M20B or N20A

6

National Semiconductor

LM1973
μPot™ 3-Channel 76dB Audio Attenuator with Mute

General Description

The LM1973 is a digitally controlled 3-channel 76dB audio attenuator fabricated on a CMOS process. Each channel has attenuation steps of 0.5dB from 0dB–15.5dB, 1.0dB steps from 16dB–47dB, and 2.0dB steps from 48dB–76dB, with a mute function attenuating 104dB. Its logarithmic attenuation curve can be customized through software to fit the desired application.

The performance of a μPot is demonstrated through its excellent Signal-to-Noise Ratio, extremely low (THD+N), and high channel separation. Each μPot contains a mute function that disconnects the input signal from the output, providing a minimum attenuation of 96dB. Transitions between any attenuation settings are pop free.

The LM1973's 3-wire serial digital interface is TTL and CMOS compatible; receiving data that selects a channel and the desired attenuation level. The Data-Out pin of the LM1973 allows multiple μPots to be daisy-chained together, reducing the number of enable and data lines to be routed for a given application.

Key Specifications

- Total Harmonic Distortion + Noise 0.003% (max)
- Frequency response 100 kHz (−3dB) (min)
- Attenuation range (excluding mute) 76dB (typ)
- Differential attenuation ±0.25dB (max)
- Signal-to-noise ratio (ref. 4 Vrms) 110dB (min)
- Channel separation 110dB (typ)

Features

- 3-wire serial interface
- Daisy-chain capability
- 104dB mute attenuation
- Pop and click free attenuation changes

Applications

- Automated studio mixing consoles
- Music reproduction systems
- Sound reinforcement systems
- Electronic music (MIDI)
- Personal computer audio control

Typical Application

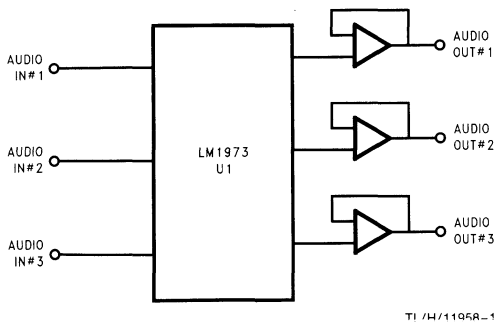
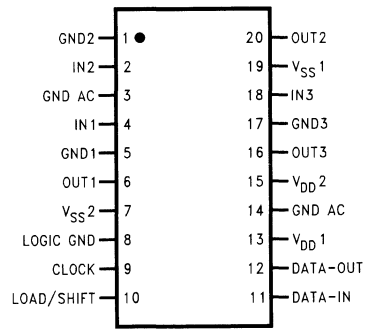


FIGURE 1. Typical Audio Attenuator Application Circuit

Connection Diagram

Dual-In-Line Plastic or Surface Mount Package



Top View

Order Number LM1973M or LM1973N
See NS Package Number M20B or N20A

Section 7

COP8™ Applications

Section 7 Contents

AB-15 Protecting Data in Serial EEPROMs 7-3
AB-22 Automatic Low Cost Thermostat 7-5
AN-521 Dual Tone Multiple Frequency (DTMF) 7-7
AN-579 MICROWIRE/PLUS Serial Interface for COP800 Family 7-16
AN-596 COP800 MathPak 7-28
AN-607 Pulse Width Modulation A/D Conversion Techniques with COP800 Family Microcontrollers 7-64
AN-662 COP800 Based Automated Security/Monitoring System 7-71
AN-663 Sound Effects for the COP800 Family 7-79
AN-666 DTMF Generation with a 3.58 MHz Crystal 7-102
AN-673 2-Way Multiplexed LCD Drive and Low Cost A/D Converter Using V/F Techniques with COP8 Microcontrollers 7-130
AN-681 PC MOUSE Implementation Using COP800 7-149
AN-714 Using COP800 Devices to Control DC Stepper Motors 7-174
AN-734 MF2 Compatible Keyboard with COP8 Microcontrollers 7-184
AN-739 RS-232C Interface with COP800 7-204
AN-755 NM95C12 Flexibility in Industrial Control Applications 7-216
AN-758 Using National's MICROWIRE EEPROM 7-227
AN-794 Using an EEPROM-I²C Interface NM24C02/03/04/05/08/09/16/17 7-238
AN-823 Timekeeping Using a COP800 Microcontroller 7-247
AN-824 8-Channel 8-Bit PWM Controller 7-249
AN-841 Software for Interfacing the COP800 Family Microcontrollers to National's MICROWIRE EEPROMs 7-252
AN-871 Selling National's Write Protected "CS" Series EEPROMs to High Volume OEMs 7-258
AN-936 How to Use the NM93C86A Serial EEPROM as a PC/Laptop Detachable Printer File Memory Card (DPFMC) 7-262
AN-952 Low Cost A/D Conversion Using COP800 7-269
AN-953 LCD Triplex Drive with COP820CJ 7-278
AN-982 COP888GW Features and Applications 7-302
AN-983 Simple, Cost Effective A/D Conversion Using COP888EK 7-320
AN-1042 COP8 Instruction Set Performance Evaluation 7-324
AN-1043 Comparison of COP878x to the Enhanced COP8SAx7 Family—Hardware/Software Considerations 7-366
AN-1044 Comparison of COP82xCJ to the Enhanced COP8SAx7 Family—Hardware/Software Considerations 7-371
AN-1048 Replacing Dedicated Protocol Controllers with Code Efficient and Configurable Microcontrollers—Low Speed CAN Network Applications 7-376
AN-1049 Using CAN Networking for Cost Effective DC Motor Control in Vehicle Body Electronics 7-381
AN-1050 Understanding and Eliminating EMI in Microcontroller Applications 7-385

Protecting Data in Serial EEPROMs

National Semiconductor
Application Brief 15
Paul Lubeck

National offers a broad line of serial interface EEPROMs which share a common set of features:

- Low cost
- Single supply in all modes (+5V ± 10%)
- TTL compatible interface
- MICROWIRE™ compatible interface
- Read-Only mode or read-write mode

This Application Brief will address protecting data in any of National's Serial Interface EEPROMs by using read-only mode.

Whereas EEPROM is non-volatile and does not require V_{CC} to retain data, the problem exists that stored data can be destroyed during power transitions. This is due to either uncontrolled interface signals during power transitions or noise on the power supply lines. There are various hardware design considerations which can help eliminate the problem although the simplest most effective method may be the following programming method.

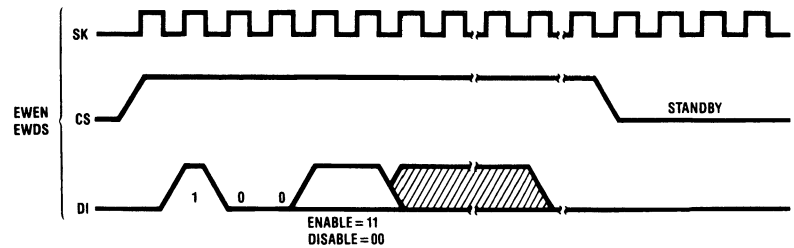
All National Serial EEPROMs, when initially powered up are in the Program Disable Mode*. In this mode, the EEPROM will abort any requested Erase or Write cycles. Prior to Erasing or Writing it is necessary to place the device in the Program Enable Mode†. Following placing the device in the Program Enable Mode, Erase and Write will remain enabled until either executing the Disable instruction or removing V_{CC}. Having V_{CC} unexpectedly removed often results in uncontrolled interface signals which could result in the EEPROM interpreting a programming instruction causing data to be destroyed.

Upon power up the EEPROM will automatically enter the Program Disable Mode. Subsequently the design should incorporate the following to achieve protection of stored data.

1) The device powers up in the read-only mode. However, as a backup, the EWDS instruction should be executed as soon as possible after V_{CC} to the EEPROM is powered up to ensure that it is in the read-only mode.
2) Immediately preceding a programming instruction (ERASE, WRITE, ERAL or WRAL), the EWEN instruction should be executed to enable the device for programming; the EWDS instruction should be executed immediately following the programming instruction to return

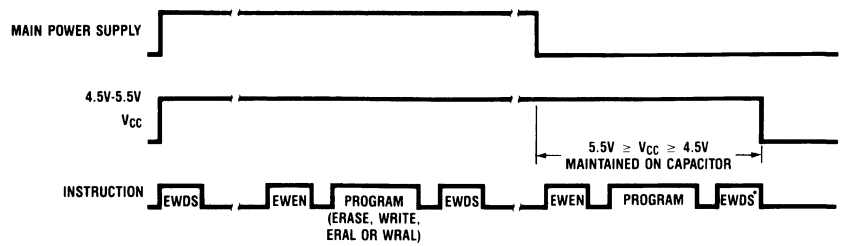
*EWDS or WDS, depending on exact device.

†EWEN or WEN, depending on exact device.



TL/D/7085–1

FIGURE 1. EWEN, EWDS Instruction Timing



TL/D/7085–2

*EWDS must be executed before V_{CC} drops below 4.5V to prevent accidental data loss during subsequent power down and/or power up transients.

FIGURE 2. Typical Instruction Flow for Maximum Data Protection

the device to the read-only mode and protect the stored data from accidental disturb during subsequent power transients or noise.

3) Special care must be taken in designs in which programming instructions are initiated to store data in the EEPROM after the main power supply has gone down. This is usually accomplished by maintaining V_{CC} for the EEPROM and its controller on a capacitor for a sufficient amount of time (approximately 50 ms, depending on the clock rate) to complete these operations. This capacitor must be large enough to maintain V_{CC} between 4.5 and 5.5 volts for the total duration of the store operation, INCLUDING the execution of the EWDS instruction immediately following the last programming instruction. FAILURE TO EXECUTE THE LAST EWDS INSTRUCTION BEFORE V_{CC} DROPS BELOW 4.5 VOLTS MAY CAUSE INADVERTENT DATA DISTURB DURING SUBSEQUENT POWER DOWN AND/OR POWER UP TRANSIENTS.

Automatic Low Cost Thermostat

National Semiconductor
Application Brief 22
Kent Brooten

Ths application brief describes the use of the NMC9346 (64 x 16) serial EEPROM. With the advent of the inexpensive COPS™ family from National Semiconductor, heretofore "expensive" applications can now be realized inexpensively. Such an application is a low cost thermostat. Typical features of such a device are:

1) Ability to interface to local and remote temperature sensors,
2) Ability to hold changeable settings,
3) Digital display of present temperature,
4) Inexpensive in high volume.

CIRCUIT DESCRIPTION

The basis of the thermostat is the COP410 microcontroller. This, with the addition of 2 ADC0854 A/D converters, an NMC9346 EEPROM and some logic for LED display, comprise an extremely versatile, yet low cost, system. The ADC0854 allows 4 channels of temperature sensors, 1 local and 3 remote. Temperature sensors used are LM34 (for readings in °F) or LM35 (for readings in °C).

While there are several possible choices for A/D converters that are MICROWIRE™ compatible, the ADC0854 was chosen because of its "settability". By presetting the "cold" temperature (i.e., when the cooling unit should come on—say 80°F) all the microcomputer has to do is to multiplex the inputs and read the data in line. Similarly, the "hot" A/D can be preset to the temperature where the furnace should come on (e.g., 60°F) and scanned in a like manner. Since the microcomputer is also keeping time of day, selecting an A/D with more "smarts" (as in the ADC0854) the software can be kept manageable and an external real time clock chip is not needed.

The EEPROM (NMC9346) holds the presettable temperature ranges (high and low settings) by day of the week. Since data is in EEPROM rather than in mask ROM, it can be changed.

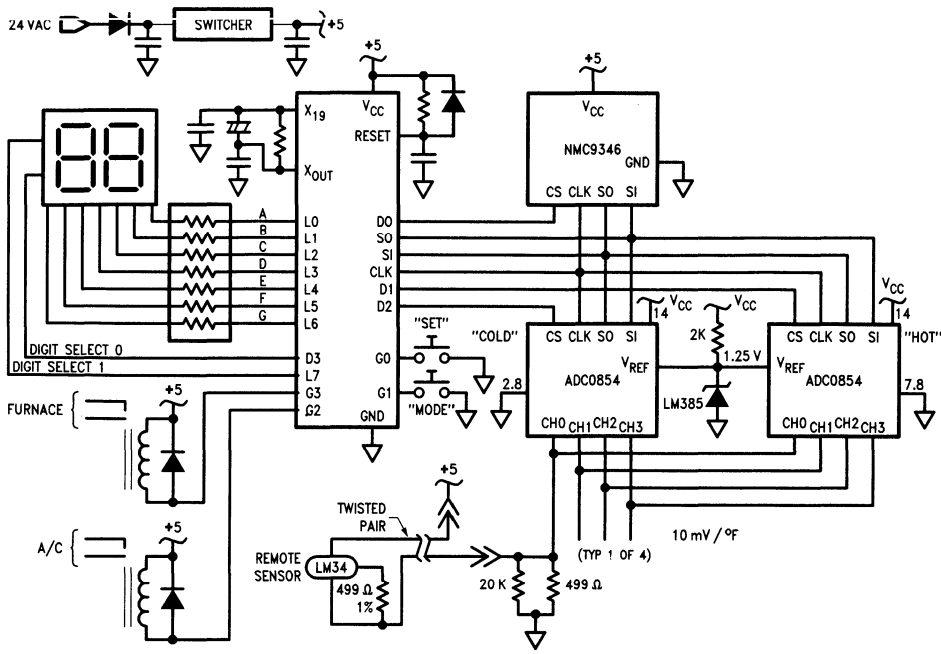
The LED display is multiplexed by the microcomputer. Depending on the type of display selected, external drivers may be necessary.

Input power is typically 24 VAC. Using a linear regulator would cause too much heat to be dissipated, which would upset the local temperature sensors. Thus, a switch mode regulator must be used. Fortunately, National Semiconductor has provided a solution to the problem with the LM3578, a switching regulator in an 8-pin mini-DIP, providing more than enough current for the application, using only a minimum of external components.

SOFTWARE DESCRIPTION

Since a real time clock is implemented in software, all routines must execute the same number of cycles independent of the input. Because of the flexibility of the COPS family instruction set, this is not as difficult a problem as it first appears. Since the EEPROM contains the settings that are periodically sent to the A/D converters, the COPS program merely fetches data from one source and dumps it to another while monitoring the output. Even the SET and MODE keys can be acted upon in a predictable manner IF the software designer carefully plans the program flow BEFORE writing code.

Note: Also see App Brief 15.



TL/D/8647–1

FIGURE 1

Dual Tone Multiple Frequency (DTMF)

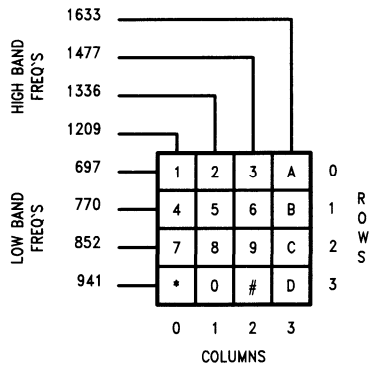
National Semiconductor
Application Note 521
Verne H. Wilson

The DTMF (Dual Tone Multiple Frequency) application is associated with digital telephony, and provides two selected output frequencies (one high band, one low band) for a duration of 100 ms. A benchmark subroutine has been written for the COP820C/840C microcontrollers, and is outlined in detail in this application note. This DTMF subroutine takes 110 bytes of COP820C/840C code, consisting of 78 bytes of program code and 32 bytes of ROM table. The timings in this DTMF subroutine are based on a 20 MHz COP820C/840C clock, giving an instruction cycle time of 1 μs.

The matrix for selecting the high and low band frequencies associated with each key is shown in *Figure 1*. Each key is uniquely referenced by selecting one of the four low band frequencies associated with the matrix rows, coupled with selecting one of the four high band frequencies associated with the matrix columns. The low band frequencies are 697, 770, 852, and 941 Hz, while the high band frequencies are 1209, 1336, 1477, and 1633 Hz. The DTMF subroutine assumes that the key decoding is supplied as a low order hex digit in the accumulator. The COP820C/840C DTMF subroutine will then generate the selected high band and low band frequencies on port G output pins G3 and G2 respectively for a duration of 100 ms.

The COP820C/840C each contain only one timer. The problem is that three different times must be generated to satisfy the DTMF application. These three times are the periods of the two selected frequencies and the 100 ms duration period. Obviously the single timer can be used to generate any one (or possibly two) of the required times, with the program having to generate the other two (or one) times.

The solution to the DTMF problem lies in dividing the 100 ms time duration by the half periods (rounded to the nearest micro second) for each of the eight frequencies, and then examining the respective high band and low band quotients and remainders. The results of these divisions are detailed in Table I. The low band frequency quotients range from 139 to 188, while the high band quotients range from 241 to 326. The observation that only the low band quotients will each fit in a single byte dictates that the high band frequency be produced by the 16 bit (2 byte) COP820C/840C timer running in PWM (Pulse Width Modulation) Mode.



TL/DD/9662–1

FIGURE 1. DTMF Keyboard Matrix

The solution then is to use the program to produce the selected low band frequency as well as keep track of the 100 ms duration. This is achieved by using three programmed register counters R0, R2, and R3, with a backup register R1 to reload the counter R0. These three counters represent the half period, the 100 ms quotient, and the 100 ms remainder associated with each of the four low band frequencies.

The theory of operation in producing the selected low band frequency starts with loading the three counters with values obtained from a ROM table. The half period for the selected frequency is counted out, after which the G2 output bit is toggled. During this half period countout, the quotient counter is decremented. This procedure is repeated until the quotient counter counts out, after which the program branches to the remainder loop. During the remainder loop, the remainder counter counts out to terminate the 100 ms. Following the remainder countout, the G2 and G3 bits are both reset, after which the DTMF subroutine is exited. Great care must be taken in time balancing the half period loop for the selected low band frequency. Furthermore, the toggling of the G2 output bit (achieved with either a set or reset bit instruction) must also be exactly time balanced to maintain the half period time integrity. Local stall loops (consisting of a DRSZ instruction followed by a JP jump back to the DRSZ for a two byte, six instruction cycle loop) are embedded in both the half period and remainder loops. Consequently, the ROM table parameters for the half period and remainder counters are approximately only one sixth of what otherwise might be expected. The program for the half period loop, along with the detailed time balancing of the loop for each of the low band frequencies, is shown in *Figure 2*.

The DTMF subroutine makes use of two 16 byte ROM tables. The first ROM table contains the translation table for the input hex digit into the core vector. The encoding of the hex digit along with the hex digit ROM translation table is shown in Table II. The row and column bits (RR, CC) representing the low band and high band frequencies respectively of the keyboard matrix shown in *Figure 1*, are encoded in

TABLE I. Frequency Half Periods, Quotients, and Remainders

	Freq. Hz	Half Period 0.5P	Half Period in μs	100 ms/0.5P	
				Quotient	Remainder
Low Band Freq.'s	697	717.36	717	139	337
	770	649.35	649	154	54
	852	586.85	587	170	210
	941	531.35	531	188	172
High Band Freq.'s	1209	413.56	414 (256 + 158)	241	226
	1336	374.25	374 (256 + 118)	267	142
	1477	338.52	339 (256 + 83)	294	334
	1633	306.18	306 (256 + 50)	326	244

the two upper and two lower bits of the hex digit respectively. Consequently, the format for the hex digit bits is RRCC, so that the input byte in the accumulator will consist of 0000RRCC. The program changes this value into 1101RRCC before using it in setting up the address for the hex digit ROM translation table.

The core vectors from the hex digit ROM translation table consist of a format of XX00TT00, where the two T (Timer) bits select one of four high band frequencies, while the two X bits select one of four low band frequencies. The core vector is transformed into four different inputs for the second ROM table. This transformation of the core vector is shown in Table III. The core vector transformation produces a timer vector 1100TT00 (T), and three programmed counter vectors for R1, R2, and R3. The formats for the three counter vectors are 1100XX11 (F), 1100XX10 (Q), and 1100XX01 (R) for R1, R2, and R3 respectively. These four vectors produced from the core vector are then used as inputs to the second ROM table. One of these four vectors (the T vector) is a function of the T bits from the core vector, while the other three vectors (F, Q, R) are a function of the X bits. This correlates to only one parameter being needed for the timer (representing the selected high band frequency), while three parameters are needed for the three counters (half period, 100 ms quotient, 100 ms remainder) associated with the low band frequency and 100 ms duration. The frequency parameter ROM translation table, accessed by the T, F, Q, and R vectors, is shown in Table IV.

Program			Bytes/Cycle	Conditional Cycles		Cycles	Total Cycles
	LD	B,#PORTGD	2/3				
	LD	X,#R1	2/3				
LUP1:	LD	A,[X−]	1/3			3	
	IFBIT	2,[B]	1/1			1	
	JP	BYP1	1/3	3	1		
	X	A,[X+]	1/3		3		
	SBIT	2,[B]	1/1		1		
	JP	BYP2	1/3		3		
BYP1:	NOP		1/1	1			
	RBIT	2,[B]	1/1	1			
	X	A,[X+]	1/3	3			
BYP2:	DRSZ	R2	1/3 DECREMENT			3	
	JP	LUP2	1/3 Q COUNT			3	
	JP	FINI	1/3				
LUP2:	DRSZ	R0	1/3 DECREMENT		3	3	
	JP	LUP2	1/3 F COUNT		3	1	
	NOP		1/1			1	
	LD	A,[X]	1/3			3	
	IFEQ	A,#104	2/2			2	
	JP	LUP1	1/3		1	3	31
	NOP		1/1		1		
	IFEQ	A,#93	2/2		2		
BACK:	JP	LUP1	1/3	1	3		35
	JP	BACK	1/3	3			
				3			39

Table IV Frequency	×	Stall Loop	+	Total Cycles	=	Half Period
((114 − 1)		x 6)		+ 39		= 717
((104 − 1)		x 6)		+ 31		= 649
((93 − 1)		x 6)		+ 35		= 587
((83 − 1)		x 6)		+ 39		= 531

FIGURE 2. Time Balancing for Half Period Loop

TABLE II. Hex Digit ROM Translation Table

	0	1	2	3
ROW	697 Hz	770 Hz	852 Hz	941 Hz
COLUMN	1209 Hz	1336 Hz	1477 Hz	1633 Hz

ADDRESS *	DATA (HEX)	KEYBOARD
0xD0	000	1
0xD1	004	2
0xD2	008	3
0xD3	00C	A
0xD4	040	4
0xD5	044	5
0xD6	048	6
0xD7	04C	B
0xD8	080	7
0xD9	084	8
0xDA	088	9
0xDB	08C	C
0xDC	0C0	*
0xDD	0C4	0
0xDE	0C8	#
0xDF	0CC	D

```
* HEX DIGIT IS RRCC,
  WHERE R = ROW #
    AND C = COLUMN #
- - - EXAMPLE: KEY 3 IS ROW #0,
  COLUMN #2, SO HEX DIGIT
  IS 0010 = 2
     RRCC
```

TABLE III. Core Vector Translation

```
CORE VECTOR   -   XX00TT00 - - - - - - - - - -
                                         *
                                        * *
                                       * * *

     TIMER VECTOR            TIMER   T     1100TT00
     HALF PERIOD VECTOR      R1      F     1100XX11
     QUOTIENT VECTOR         R2      Q     1100XX10
     REMAINDER VECTOR        R3      R     1100XX01
```

TABLE IV. Frequency Parameter ROM Translation Table

T - TIMER F - FREQUENCY Q - QUOTIENT R - REMAINDER

ADDRESS	DATA (DEC)	VECTOR
0xC0	158	T
0xC1	53	R
0xC2	140	Q
0xC3	114	F
0xC4	118	T
0xC5	6	R
0xC6	155	Q
0xC7	104	F
0xC8	83	T
0xC9	32	R
0xCA	171	Q
0xCB	93	F
0xCC	50	T
0xCD	25	R
0xCE	189	Q
0xCF	83	F

In summary, the input hex digit selects one of 16 core vectors from the first ROM table. This core vector is then transformed into four other vectors (T, F, Q, R), which in turn are used to select four parameters from the second ROM table. These four parameters are used to load the timer, and the respective half period, quotient, and remainder counters. The first ROM table (representing the hex digit matrix table) is arbitrarily placed starting at ROM location 01D0, and has a reference setup with the ADD A,#0D0 instruction. The second ROM table (representing the frequency parameter table) must be placed starting at ROM location 01C0 (or 0xC0) in order to minimize program size, and has reference setups with the OR A,#0C3 instruction for the F vector and with the OR A,#0C0 instruction for the T vector.

The three parameters associated with the two X bits of the core vector require a multi-level table lookup capability with the LAID instruction. This is achieved with the following section of code in the DTMF subroutine:

```
        LD      B,#R1
        LD      X,#R4
        X       A,[X]
LUP:    LD      A,[X]
        LAID
        X       A,[B+]
        DRSZ    R4
        IFBNE   #4
        JP      LUP
```

This program code loads the F frequency vector into R4, and then decrements the vector each time around the loop. This successive loop decrementation of the R4 vector changes the F vector into the Q vector, and then changes the Q vector into the R vector. This R4 vector is used to access the ROM table with the LAID instruction. The X pointer references the R4 vector, while the B pointer is incremented each time around the loop after it has been used to store away the three selected ROM table parameters (one per loop). These three parameters are stored in sequential RAM locations R1, R2, and R3. The IFBNE test instruction is used to skip out of the loop once the three selected ROM table parameters have been accessed and stored away.

The timer is initialized to a count of 15 so that the first timer underflow and toggling of the G3 output bit (with timer PWM mode and G3 toggle output selected) will occur at the same time as the first toggling of the G2 output bit. The half period counts for the high band frequencies range from 306 to 414, so these values minus 256 are stored in the timer section of the second ROM table. The selected value from this frequency ROM table is then stored in the lower half of the timer autoreload register, while a 1 is stored in the upper half. The timer is selected for PWM output mode and started with the instruction LD [B],#0B0 where the B pointer is selecting the CNTRL register at memory location 0EE.

The DTMF subroutine for the COP820C/840C uses 110 bytes of code, consisting of 78 bytes of program code and 32 bytes of ROM table. A program routine to sequentially call the DTMF subroutine for each of the 16 hex digit inputs is supplied with the listing for the DTMF subroutine.

```
NATIONAL SEMICONDUCTOR CORPORATION          PAGE:     1
 COP800 CROSS ASSEMBLER,REV:B,20 JAN 87
 DTMF

                ;DTMF PROGRAM FOR COP820C/840C          VERNE H. WILSON
                ;                                            5/1/89
                ;DTMF - DUAL TONE MULTIPLE FREQUENCY
                ;
                ;PROGRAM NAME: DTMF.MAC
                ;
                          .TITLE DTMF
                          .CHIP 840
                ;******* THE DTMF SUBROUTINE CONTAINS 110 BYTES *******
                ; ***** THE DTMF SUBROUTINE TIMES OUT IN 100MSEC *****
                ;    ** FROM THE FIRST TOGGLE OF THE G2/G3 OUTPUTS **
                ;       *** BASED ON A 20 MHZ COP820C/840C CLOCK ***
                ;
                ;G PORT IS USED FOR THE TWO OUTPUTS
                ;     -    HIGH BAND (HB) FREQUENCY OUTPUT ON G3
                ;     -    LOW BAND (LB) FREQUENCY OUTPUT ON G2
                ;
                ;TIMER COUNTS OUT
                ;     -    HB FREQUENCIES
                ;
                ;PROGRAM COUNTS OUT
                ;     -    LB FREQUENCIES
                ;     -    100 MSEC DIVIDED BY LB HALF PERIOD QUOTIENT
                ;     -    100 MSEC DIVIDED BY LB HALF PERIOD REMAINDER
                ;
                ;FORMAT FOR THE 16 HEX DIGIT MATRIX VECTOR IS 1101RRCC,
                ;    WHERE  -  RR IS ROW SELECT (LB FREQUENCIES)
                ;           -  CC IS COLUMN SELECT (HB FREQUENCIES)
                ;
                ;FORMAT FOR THE 16 CORE VECTORS FROM THE MATRIX SELECT
                ;    TABLE IS XX00TT00, WHERE  -  TT IS HB SELECT
                ;                                 XX IS LB SELECT
                ;
                ;FREQUENCY VECTORS (HB & LB) FOR FREQ PARAMETER TABLE
                ;    MADE FROM CORE VECTORS
                ;
                ;HB FREQUENCY VECTORS(4) END WITH 00 FOR TIMER COUNTS,
                ;    WHERE VECTOR FORMAT IS 1100TT00
                ;
                ;LB FREQUENCY VECTORS(12) END WITH:
                ;    11 FOR HALF PERIOD LOOP COUNTS,
                ;         WHERE VECTOR FORMAT IS 1100XX11
                ;    10 FOR 100 MSEC DIVIDED BY HALF PERIOD QUOTIENTS,
                ;         WHERE VECTOR FORMAT IS 1100XX10
                ;    01 FOR 100 MSEC DIVIDED BY HALF PERIOD REMAINDERS,
                ;         WHERE VECTOR FORMAT IS 1100XX01
                ;
                ;HEX DIGIT MATRIX TABLE AT HEX 01D* (OPTIONAL LOCATION,
                ;    DEPENDING ON  'ADD  A,#0D0' INST. IMMEDIATE VALUE)
                ;
                ;FREQ PARAMETER TABLE AT HEX 01C*  (REQUIRED LOCATION)
```

TL/DD/9662-2

```
NATIONAL SEMICONDUCTOR CORPORATION          PAGE:     2
 COP800 CROSS ASSEMBLER,REV:B,20 JAN 87
 DTMF


 52                          .FORM
 53                    ;
 54                    ;MAGIC:        CORE VECTOR
 55                    ;                XX00TT00
 56                    ;
 57                    ;    TIMER      T     TT00
 58                    ;    R1         F     XX11
 59                    ;    R2         Q     XX10
 60                    ;    R3         R     XX01
 61                    ;
 62                    ;DECLARATIONS:
 63       00D0            PORTLD = 0D0          ; PORTL DATA REG
 64       00D1            PORTLC = 0D1          ; PORTL CONFIG REG
 65       00D4            PORTGD = 0D4          ; PORTG DATA REG
 66       00D5            PORTGC = 0D5          ; PORTG CONFIG REG
 67       00DC             PORTD = 0DC          ; PORTD REG
 68       00EA           TIMERLO = 0EA          ; TIMER LOW COUNTER
 69       00EE             CNTRL = 0EE          ; CONTROL REG
 70       00EF               PSW = 0EF          ; PROC STATUS WORD
 71       00F0                R0 = 0F0          ; LB FREQ LOOP COUNTER
 72       00F1                R1 = 0F1          ; LB FREQ LOOP COUNT
 73       00F2                R2 = 0F2          ; LB FREQ Q COUNT
 74       00F3                R3 = 0F3          ; LB FREQ R COUNT
 75       00F4                R4 = 0F4          ; LB FREQ TABLE VECTOR
 76                    ;
 77 0000 DD2F    START:      LD        SP,#02F       ; HEX DIGIT MATRIX
 78 0002 BCD1FF              LD        PORTLC,#0FF   ; 1  2  3  A
 79 0005 BCD080              LD        PORTLD,#080   ; 4  5  6  B
 80 0008 DEDC                LD        B,#PORTD      ; 7  8  9  C
 81 000A 9E00                LD        [B],#0        ; *  0  #  D
 82 000C AE      LOOP:       LD        A,[B]         ; DTMF TEST LOOP
 83 000D 3160                JSR       DTMF          ; HEX MATRIX DIGIT
 84 000F DEDC                LD        B,#PORTD      ; TO SUBROUTINE IS
 85 0011 AE                  LD        A,[B]         ; OUTPUT TO PORTD
 86 0012 9405                ADD       A,#5          ; DO WILL TOGGLE
 87 0014 A6                  X         A,[B]         ; FOR EACH CALL OF
 88 0015 6C                  RBIT      4,[B]         ; DTMF SUBROUTINE
 89 0016 9DD0                LD        A,PORTLD      ; PORTL OUTPUTS
 90 0018 A1                  SC                      ; PROVIDE SYNC
 91 0019 B0                  RRC       A             ; OUTPUT ORDER IS
 92 001A 9CD0                X         A,PORTLD      ; 1,5,9,D,4,8,#,A,
 93 001C EF                  JP        LOOP          ; 7,0,3,B,*,2,6,C
 94                    ;
 96                    ;
```

TL/DD/9662-3

```
 97      0160                .=0160
 98                  ;
 99 0160 DED5    DTMF:       LD          B,#PORTGC
100 0162 9B3F               LD          [B-],#03F
101 0164 6B                 RBIT        3,[B]           ; OPTIONAL
102 0165 6A                 RBIT        2,[B]           ; OPTIONAL
103                 ;
104 0166 94D0               ADD         A,#0D0
105 0168 A4                 LAID                        ; DIGIT MATRIX TABLE
106                 ;
107 0169 5F                 LD          B,#0
108 016A A6                 X           A,[B]
109 016B AE                 LD          A,[B]
110 017B 65                 SWAP        A
111 016C 97C3               OR          A,#0C3
112 016E DEF1               LD          B,#R1
113 0170 DCF4               LD          X,#R4
114 0172 B6                 X           A,[X]
115 0173 BE     LUP:        LD          A,[X]
116 0174 A4                 LAID                        ; LB FREQ TABLES
117 0175 A2                 X           A,[B+]          ;   (3 PARAMETERS)
118 0176 C4                 DRSZ        R4
119 0177 44                 IFBNE       #4
120 0178 FA                 JP          LUP
121                 ;
122 0179 5F                 LD          B,#0
123 017A AE                 LD          A,[B]
124 017C 97C0               OR          A,#0C0
125 017E A4                 LAID                        ; HB FREQ TABLE
126 017F DEEA               LD          B,#TIMERLO      ;   (1 PARAMETER)
127 0181 9A0F               LD          [B+],#15
128 0183 9A00               LD          [B+],#0
129 0185 A2                 X           A,[B+]
130 0186 9A01               LD          [B+],#1
131 0188 9EB0               LD          [B],#0B0        ; START TIMER PWM
132                 ;
133 018A DED4               LD          B,#PORTGD
134 018C DCF1               LD          X,#R1
135                 ;
136 018E BB     LUP1:       LD          A,[X-]
137 018F 72                 IFBIT       2,[B]           ; TEST LB OUTPUT
138 0190 03                 JP          BYP1
139 0191 B2                 X           A,[X+]
140 0192 7A                 SBIT        2,[B]           ; SET LB OUTPUT
141 0193 03                 JP          BYP2
142 0194 B8     BYP1:       NOP
143 0195 6A                 RBIT        2,[B]           ; RESET LB OUTPUT
144 0196 B2                 X           A,[X+]
145 0197 C2     BYP2:       DRSZ        R2              ; DECR. QUOT. COUNT
146 0198 01                 JP          LUP2
147 0199 0C                 JP          FINI            ; Q COUNT FINISHED
148                 ;
149 019A C0     LUP2:       DRSZ        R0              ; DECR. F COUNT
150 019B FE                 JP          LUP2            ; LB (HALF PERIOD)
151                 ;
152 019C B8                 NOP                         ; **********
153 019D BE                 LD          A,[X]           ; BALANCE
154 019E 9268               IFEQ        A,#104          ; LB FREQUENCY
155 01A0 ED                 JP          LUP1            ; HALF PERIOD
156                 ;                                   ; RESIDUE
157 01A1 B8                 NOP                         ; DELAY FOR
158 01A2 925D               IFEQ        A,#93           ; EACH OF 4
159 01A4 E9     BACK:       JP          LUP1            ; LB FREQ'S
160 01A5 FE                 JP          BACK            ; **********
161                 ;
162 01A6 C3     FINI:       DRSZ        R3              ; DECR. REM. COUNT
163 01A7 FE                 JP          FINI            ; R CNT NOT FINISHED
164                 ;
165 01A8 BDEE6C             RBIT        4,CNTRL         ; STOP TIMER
166 01AB 6B                 RBIT        3,[B]           ; CLR HB OUTPUT
167 01AC 6A                 RBIT        2,[B]           ; CLR LB OUTPUT
168                 ;
169 01AD 8E                 RET
170                 ;
```

TL/DD/9662–4

```
NATIONAL SEMICONDUCTOR CORPORATION          PAGE:      4
 COP800 CROSS ASSEMBLER,REV:B,20 JAN 87
 DTMF


 171                                .FORM
 172                    ;
 173                    ;FREQUENCY AND 100MSEC PARAMETER TABLE
 174         01C0                   .=01C0
 175                    ;
 176  01C0  9E                      .BYTE      158              ; T
 177  01C1  35                      .BYTE       53              ; R
 178  01C2  8C                      .BYTE      140              ; Q
 179  01C3  72                      .BYTE      114              ; F
 180  01C4  76                      .BYTE      118              ; T
 181  01C5  06                      .BYTE        6              ; R
 182  01C6  9B                      .BYTE      155              ; Q
 183  01C7  68                      .BYTE      104              ; F
 184  01C8  53                      .BYTE       83              ; T
 185  01C9  20                      .BYTE       32              ; R
 186  01CA  AB                      .BYTE      171              ; Q
 187  01CB  5D                      .BYTE       93              ; F
 188  01CC  32                      .BYTE       50              ; T
 189  01CD  19                      .BYTE       25              ; R
 190  01CE  BD                      .BYTE      189              ; Q
 191  01CF  53                      .BYTE       83              ; F
 192                    ;
 193                    ;DIGIT MATRIX TABLE
 194         01D0                   .=01D0
 195                    ;                                                  ROW   COL
 196  01D0  00                      .BYTE      000              ; 1         0     0
 197  01D1  04                      .BYTE      004              ; 2         0     1
 198  01D2  08                      .BYTE      008              ; 3         0     2
 199  01D3  0C                      .BYTE      00C              ; A         0     3
 200  01D4  40                      .BYTE      040              ; 4         1     0
 201  01D5  44                      .BYTE      044              ; 5         1     1
 202  01D6  48                      .BYTE      048              ; 6         1     2
 203  01D7  4C                      .BYTE      04C              ; B         1     3
 204  01D8  80                      .BYTE      080              ; 7         2     0
 205  01D9  84                      .BYTE      084              ; 8         2     1
 206  01DA  88                      .BYTE      088              ; 9         2     2
 207  01DB  8C                      .BYTE      08C              ; C         2     3
 208  01DC  C0                      .BYTE      0C0              ; *         3     0
 209  01DD  C4                      .BYTE      0C4              ; 0         3     1
 210  01DE  C8                      .BYTE      0C8              ; #         3     2
 211  01DF  CC                      .BYTE      0CC              ; D         3     3
 212                    ;
 213                    .END
```

TL/DD/9662-5

```
NATIONAL SEMICONDUCTOR CORPORATION          PAGE:     5
 COP800 CROSS ASSEMBLER,REV:B,20 JAN 87
 DTMF

SYMBOL TABLE

 B       00FE          BACK    01A4          BYP1    0194          BYP2    0197
 CNTRL   00EE          DTMF    0160          FINI    01A6          LOOP    000C
 LUP     0174          LUP1    018E          LUP2    019A          PORTD   00DC
 PORTGC  00D5          PORTGD  00D4          PORTLC  00D1          PORTLD  00D0
 PSW     00EF  *       R0      00F0          R1      00F1          R2      00F2
 R3      00F3          R4      00F4          SP      00FD          START   0000  *
 TIMERL  00EA          X       00FC


MACRO TABLE


   NO WARNING LINES

   NO ERROR LINES

  139  ROM BYTES USED

SOURCE CHECKSUM = 99A7
OBJECT CHECKSUM = 03E1

INPUT   FILE C:DTMF.MAC
LISTING FILE C:DTMF.PRN
OBJECT  FILE C:DTMF.LM
```

TL/DD/9662–6

The code listed in this App Note is available on Dial-A-Helper.

Dial-A-Helper is a service provided by the Microcontroller Applications Group. The Dial-A-Helper system provides access to an automated information storage and retrieval system that may be accessed over standard dial-up telephone lines 24 hours a day. The system capabilities include a MESSAGE SECTION (electronic mail) for communicating to and from the Microcontroller Applications Group and a FILE SECTION mode that can be used to search out and retrieve application data about NSC Microcontrollers. The minimum system requirement is a dumb terminal, 300 or 1200 baud modem, and a telephone.

With a communications package and a PC, the code detailed in this App Note can be down loaded from the FILE SECTION to disk for later use. The Dial-A-Helper telephone lines are:

Modem (408) 739-1162
Voice (408) 721-5582

For Additional Information, Please Contact Factory

MICROWIRE/PLUS™ Serial Interface for COP800 Family

National Semiconductor
Application Note 579
Ramesh Sivakolundu
Sunder Velamuri

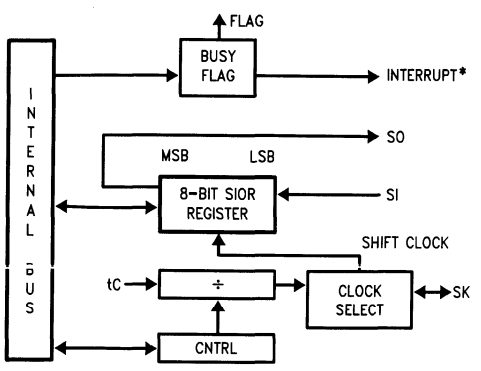
INTRODUCTION

National Semiconductor's COP800 family of full-feature, cost-effective microcontrollers use a new 8-bit single chip core architecture fabricated with M²CMOS process technology. These high performance microcontrollers provide efficient system solutions with a versatile instruction set and high functionality.

The COP800 family of microcontrollers feature the MICROWIRE/PLUS mode of serial communication. MICROWIRE/PLUS is an enhancement of the MICROWIRE™ synchronous serial communications scheme, originally implemented on the COP400 family of microcontrollers. The MICROWIRE/PLUS interface on the COP800 family of microcontrollers enables easy I/O expansion and interfacing to several COPS peripheral devices (A/D converters, EEPROMs, Display drivers etc.), and interfacing with other microcontrollers which support MICROWIRE/PLUS or SPI* modes of serial interface.

MICROWIRE/PLUS DEFINITION

MICROWIRE/PLUS is a versatile three wire, SI (serial input), SO (serial output), and SK (serial clock), bidirectional serial synchronous communication scheme where the COP800 is either the Master providing the Shift Clock (SK) or a slave accepting an external Shift Clock (SK). The COP800 MICROWIRE/PLUS system block diagram is shown in *Figure 1*. The MICROWIRE/PLUS serial interface utilizes an 8-bit memory mapped MICROWIRE/PLUS serial shift register, SIOR, clocked by the SK signal. As the name suggests, the SIOR register serves as the shift register for serial transfers. SI, the serial input line to the COP800 microcontroller, is the shift register input. SO, the shift register output, is the serial output to external devices. SK is the serial synchronous clock. Data is clocked into and out of the peripheral devices with the SK clock. The SO, SK and SI are mapped as alternate functions on pins 4, 5, and 6 respectively of the 8-bit bidirectional G Port.



*only in COP888XX series

FIGURE 1. MICROWIRE/PLUS Block Diagram

MICROWIRE/PLUS OPERATION

In MICROWIRE/PLUS serial interface, the input data on the SI pin is shifted high order first into the Least Significant Bit (LSB) of the 8-bit SIOR shift register. The output data is shifted out high order first from the Most Significant Bit (MSB) of the shift register onto the SO pin. The SIOR register is clocked on the falling edge of the SK clock signal. The input data on the SI pin is shifted into the LSB of the SIOR register on the rising edge of the SK clock. The MSB of the SIOR register is shifted out to the SO pin on the falling edge of the SK clock signal. The SK clock signal is generated internally by the COP800 for the master mode of MICROWIRE/PLUS operation. In the slave mode, the SK clock is generated by an external device (which acts as the master) and is input to the COP800.

The MSEL (MICROWIRE Select) flag in the CNTRL register is used to enable MICROWIRE/PLUS operation. Setting the MSEL flag enables the gating of the MICROWIRE/PLUS interface signals through the G port. Pins G4, G5, and G6 of the G port are used for the signals SO, SK and SI, respectively. It should be noted that the G port configuration register must be set up appropriately for MICROWIRE/PLUS operation. Table I illustrates the G-port configurations. In the master mode of MICROWIRE/PLUS operation, G4 and G5 need to be selected as outputs for SO and SK signals. Alternatively, in the slave mode of operation, G5 needs to be configured as an input for the external SK. The SI signal is a dedicated input on G6 and therefore no further setup is required.

TABLE I. G Port Configurations

G4 (SO) Config. Bit	G5 (SK) Config Bit.	G4 Fun.	G5 Fun.	Operation
1	1	SO	Int. SK	MICROWIRE Master
0	1	TRI-STATE	Int. SK	MICROWIRE Master
1	0	SO	Ext. SK	MICROWIRE Slave
0	0	TRI-STATE	Ext. SK	MICROWIRE Slave

The SL1 and SL0 (S1 and S0 in COP820C and COP840C) bits of the CNTRL register are used to select the clock division factor (2, 4, or 8) for SK clock generation in MICROWIRE/PLUS master mode operation. A clock select table for these bits of the CNTRL register along with the CNTRL register is shown in Table II. The counter associated with

the master mode clock division factor is cleared when the MICROWIRE/PLUS BUSY flag is low. The clock division factor is relative to the instruction cycle frequency. For example, if the COP800 is operating with an internal clock of 1 MHz, the SK clock rate would be 500 kHz, 250 kHz, or 125 kHz for SL1 and SL0 values of 00, 01 and 10 (or 11) respectively.

TABLE II

CNTRL Register (Address X'00EE)

The Timer1 (T1) and MICROWIRE control register contains the following bits:

SL1 & SL0 Select the MICROWIRE clock divide by (00 = 2, 01 = 4, 1X = 8)

IEDG External Interrupt Edge Polarity Select (0 = Rising Edge, 1 = Falling Edge)

MSEL Selects G5 and G4 as MICROWIRE Signals SK and SO Respectively

T1C0 Timer T1 Start/Stop Control in Timer Modes 1 and 2
Timer T1 Underflow Interrupt Pending Flag in Timer Mode 3

T1C1 Timer T1 Mode Control Bit

T1C2 Timer T1 Mode Control Bit

T1C3 Timer T1 Mode Control Bit

T1C3	T1C2	T1C1	T1C0	MSEL	IEDG	SL1	SL0

Bit 7 Bit 0

SL1	SL0	SK
0	0	$2 \times t_C$
0	1	$4 \times t_C$
1	x	$8 \times t_C$

Where t_C is the instruction cycle clock

MICROWIRE/PLUS MASTER MODE OPERATION

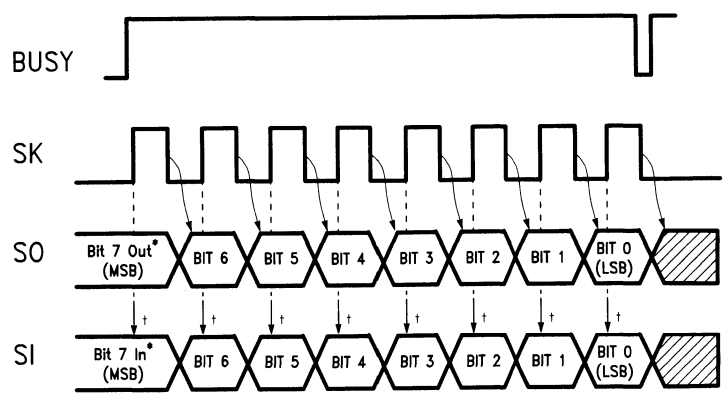
In the MICROWIRE/PLUS master mode, the BUSY flag of PSW (Processor Status Word) is used to control the shifting of the MICROWIRE/PLUS 8-bit shift register. Setting the BUSY flag causes the SIOR register to shift out 8 bits of data from SO at the high order end of the shift register. During the same time, 8 new bits of data from SI are shifted into the low order end of the SIOR register. The BUSY flag is automatically reset after the 8 bits of data have been shifted *(Figure 2)*. The COP888XX series of microcontrollers provide a vectored maskable interrupt when the BUSY goes low indicating the end of an 8-bit shift. Input data is clocked into the SIOR register from the SI pin with the rising edge of the SK clock, while the MSB of the SIOR is shifted onto the SO pin with the falling edge of the SK clock. The user may reset the BUSY bit by software to allow less than 8 bits to shift. However, the user should ensure that the software BUSY resets only occurs when the SK clock is low, in order to avoid a narrow SK terminal clock.

MICROWIRE/PLUS SLAVE MODE OPERATION

In the MICROWIRE/PLUS Slave mode of operation the SK clock is generated by an external source. Setting the MSEL bit in the CNTRL register enables the SO and SK functions onto the G Port. The SK pin must be configured as an input and the SO pin configured as an output by resetting and setting the appropriate bits in the Port G configuration register. The user must set the BUSY flag immediately upon entering the Slave mode. After eight clock pulses the Busy flag will be cleared and the sequence may be repeated. However, in the Slave mode the COP888 series does not shift data if the BUSY flag is reset, whereas the COP820C and COP840C continues to shift regardless of the BUSY flag, if the SK clock is active.

MICROWIRE/PLUS ALTERNATE SK MODE

The COP888XX series of microcontrollers also allow an additional Alternate SK Phase Operation. In the normal mode data is shifted in on the rising edge of the SK clock and data is shifted out on the falling edge of the SK clock *(Figure 2)*. The SIOR register is shifted on each falling edge of the SK clock. In the alternate SK phase operation, data is shifted in on the falling edge of the SK clock and data is shifted out on the rising edge of the SK clock *(Figure 3)*.

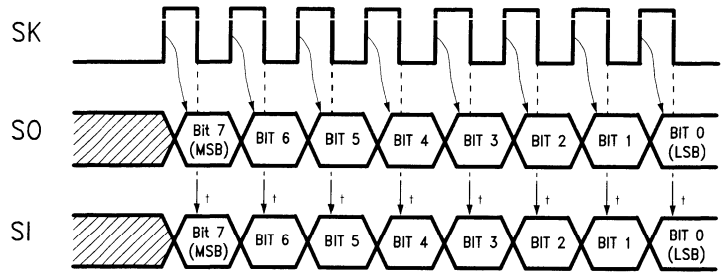


TL/DD/10252–2

*This bit becomes valid immediately after loading the SIOR register of the transmitting device.

†Arrows indicate points at which SI is sampled.

FIGURE 2. MICROWIRE/PLUS Timing



↑ Arrows indicates points at which SI is sampled.

FIGURE 3. Alternate Phase SK Clock Timing

A control flag, SKSEL, allows either the normal SK clock or alternate SK clock to be selected. Resetting SKSEL selects the normal SK clock and setting SKSEL selects the alternate SK clock for the MICROWIRE/PLUS logic. The SKSEL flag is mapped into the G6 configuration bit. The SKSEL flag is reset after power up, selecting the normal SK clock signal. The alternate mode facilitates the usage of the MICROWIRE/PLUS protocol for serial data transfer between peripheral devices which are not compatible with the normal SK clock operation, i.e., shifting data out on the falling edge of the SK clock and shifting in data on the rising edge of the SK clock.

MICROWIRE/PLUS SAMPLE PROTOCOL

This section gives a sample MICROWIRE/PLUS protocol using a COP888CL and COP840C. The slave mode operating procedure for this sample protocol is explained, and a timing illustration of the protocol is provided.

1. The MSEL bit in the CNTRL register is set to enable MICROWIRE; G0 ($\overline{CS}$) and G5 (SK) are configured as inputs and G4 (SO) as an output. G6 (SI) is always an input.
2. Chip Select line ($\overline{CS}$) from master device is connected to G0 of the slave device. An active-low level on $\overline{CS}$ line causes the slave to interrupt.
3. From the high-to-low transistion on the $\overline{CS}$ line, there is no data transfer on the MICROWIRE until time "T" (See *Figure 4*).
4. The master initiates data transfer on the MICROWIRE by turning on the SK clock.
5. A series of data transfers take place between the master and slave devices.
6. The master pulls the $\overline{CS}$ line high to end the MICROWIRE operation. The slave device returns to normal mode of operation.

SLAVE MODE OPERATING PROCEDURE

1. The MSEL bit in the CNTRL register is set to enable MICROWIRE; G0 ($\overline{CS}$) and G5 (SK) are configured as inputs and G4 (SO) as an output. G6 (SI) is always an input.
2. Normal mode of operation until interrupted by $\overline{CS}$ going low.
3. Set the BUSY flag and load SIOR register with the data to be sent out on SO. (The shift register shifts 8 bits of data from SO at the high order end of the shift register. During the same time, 8 new bits of data from SI are loaded into the low order end of the shift register.)
4. Wait for the BUSY flag to reset. (The BUSY flag is automatically reset after 8 bits of data have been shifted).
5. If data is being read in, the user should save contents of the SIOR register.
6. The prearranged set of data transfers are performed.
7. Repeat steps 3 through 6. The user must ensure steps 3 through 6 are performed in time "t" (See *Figure 4*) as agreed upon in the protocol.

DIFFERENCES BETWEEN COP888 AND COP820/COP840

The COP888 series MICROWIRE/PLUS feature differs from that of the COP820/COP840 in some respects. The COP888 series can be configured to interrupt the processor after the completion of a MICROWIRE/PLUS operation indicated by the BUSY flag going low. The COP888 series supports a vectored interrupt scheme. Two bytes of program memory space are reserved for each interrupt source. The user would do any required context switching and then program a VIS (Vector Interrupt Select) instruction in order to branch to the interrupt service routine of the highest priority interrupt enabled and pending at the time of the VIS instruction. The addresses of the different interrupt service routines are chosen by the user and stored in ROM in a table starting at 0yE0 where "y" depends on the 256 byte block (0y00 to 0yFF) in which the VIS instruction is located. The vector address for the MICROWIRE/PLUS interrupt is 0yF2-0yF3.

Secondly, the COP888 series supports the alternate SK phase mode of MICROWIRE/PLUS operation. This feature facilitates the usage of the MICROWIRE/PLUS protocol for serial data transfer between peripheral devices which are not compatible with the normal SK clock operation, i.e., shifting data out on the falling edge of SK clock and shifting in data on the rising edge of the SK clock.

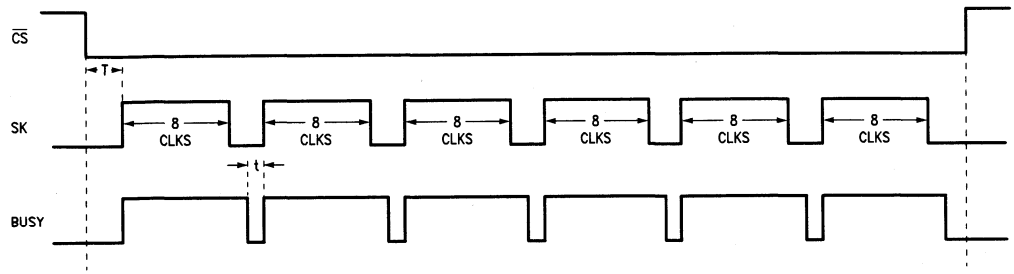


FIGURE 4. MICROWIRE/PLUS Sample Protocol Timing Diagram

INTERFACE CONSIDERATIONS

To preserve the integrity of data exchange using MICROWIRE/PLUS, two aspects have to be considered:

1. Serial data exchange timing.
2. Fan-out/fan-in requirements.

Theoretically, infinite devices can access the same interface and be uniquely enabled sequentially in time. In practice, however, the actual number of devices that can access the same serial interface depends on the following: System data transfer rate, system supply requirement, capacitive loading on SK and SO outputs, the fan-in requirements of the logic families or discrete devices to be interfaced.

HARDWARE INTERFACE

For proper data transfer to occur the output should be able to switch between a HIGH level and a LOW level in a predetermined amount of time. The transfer is strictly synchronous and the timing is related to the MICROWIRE/PLUS system clock (SK). For example, if a COPS controller outputs a value at the falling edge of the clock and is latched in by the peripheral device at the rising edge, then the following relationship has to be satisifed:

$$t_{DELAY} + t_{SETUP} \leq t_{CK}$$

where t_{CK} is the time from data output starts to switch to data being latched into the peripheral chip, t_{SETUP} is the setup time for the peripheral device where the data has to be at a valid level, and t_{DELAY} is the time for the output to read the valid level. t_{CK} is related to the system clock provided by the SK pin of the COPS controller and can be increased by increasing the COPS instruction cycle time.

Besides the timing requirements, system supply and fan-out/fan-in requirements also have to be considered when interfacing with MICROWIRE/PLUS. To drive multi-devices on the same MICROWIRE/PLUS, the output drivers of the controller need to source and sink the total maximum leakage current of all the inputs connected to it and keep the signal level within the valid logic "1" and "0" input voltage levels. Thus, if devices of different types are connected to the same serial interface, output driver of the controller must satisfy all the input requirements of each device. Similarly, devices with TRI-STATE® outputs, when connected to the SI input, must satisfy the minimum valid input level of the controller and the maximum TRI-STATE® leakage current of all outputs.

So, for devices that have incompatible input levels or source/sink requirements, external pull-up resistors or buffers are necessary to provide level-shifting or driving.

TABLE III

Features		Part Number							
		DS890XX	**MM545X**	**COP470**	**COP472**	**ADC83X (COP430)**	**COP498/499**	**COP452L**	**NMC9306 (COP494)**
GENERAL									
Chip Function		AM/PM PLL	LED Display Driver	VF Display Driver	LCD Display Driver	A/D	RAM & Timer	Frequency Generator	E^2PROM
Process		ECL	NMOS	PMOS	CMOS	CMOS	CMOS	NMOS	NMOS
V_{CC} Range		4.75V–5.25V	4.5V–11V	−9.5V to −4.5V	3.0V–5.5V	4.5V–0.3V	2.4V–5.5V	4.5V–6.3V	4.5V–5.5V
Pinout		20	40	20	20	8/14/20	14/8	14	14
HARDWARE INTERFACE									
Min V_{IH}/Max V_{IL}		2.1V/0.7V	2.2V/0.8V	−1.5V/−4.0V	0.7 V_{CC}/0.8V	2.0V/0.8V	0.8 V_{CC}/0.4 V_{CC}	2.0V/0.8V	2.0V/0.8V
SK Clock Range		0–625 kHz	0–500 kHz	0–250 kHz	4–250 kHz	10–200 kHz	4–250 kHz	25–250 kHz	0–250 kHz
Write Data DI	Setup Min	0.3 μs	0.3 μs	1.0 μs	1.0 μs	0.2 μs	0.4 μs	800 ns	0.4 μs
	Hold Min	0.8 μs	(Note 3)	50 ns	100 ns (Note 1)	0.2 μs	0.4 μs	1.0 μs	0.4 μs
Read Data Prop Delay		(Note 4)	(Note 3)	(Note 3)	(Note 3)	(Note 3)	2 μs (Note 2)	1 μs (Note 2)	2.0 μs
Chip Enable	Setup	0.275 μs	0.4 μs	1.0 μs Min	1 μs (Note 1)	0.2 μs	0.2 μs (Note 1)	(Note 3)	0.2 μs
	HOLD	0.300 μs	(Note 3)	1.0 μs Min	1 μs (Note 2)	0.2 μs	0 (Note 2)	(Note 3)	0
Max Frequency Range	AM	8 MHz	(Note 3)	(Note 3)	(Note 3)	(Note 3)	(Note 3)	(Note 3)	(Note 3)
	FM	120 MHz	(Note 3)	(Note 3)	(Note 3)	(Note 3)	(Note 3)	(Note 3)	(Note 3)
Max Osc. Freq.		(Note 3)	(Note 3)	250 kHz	(Note 3)	(Note 3)	2.1 MHz (−21) 32 kHz (−15)	256–2100 kHz (−4) 64–525 kHz (−2)	(Note 3)
SOFT									
Serial I/O Protocol		11D1–D20	1D1–D35	8 Bits At a Time	b1–b40	1xxx	1yyxxD6–D0 Start Bit	1yxxxx	1AA–DD
Instruction/ Address Word		None	None	None	None	(Note 4)	(Note 4)	(Note 4)	(Note 4)

Note 1: Reference to SK rising edge.

Note 2: Reference to SK falling edge.

Note 3: Not defined.

Note 4: See data sheet for different modes of operation.

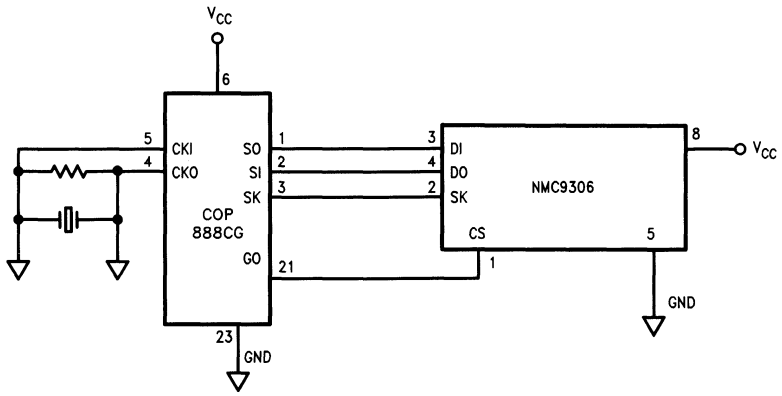
TYPICAL APPLICATIONS

A whole family of off-the shelf devices exist that are directly compatible with MICROWIRE/PLUS protocol. This allows direct interface with the COP800 family of microcontrollers. Table III provides a summary of the existing devices, their function and specification.

NMC9306-COP888CG INTERFACE

The pin connection involved in interfacing an NMC9306 (COP494), a 256 bit E²PROM, with the COP888CG microcontroller is shown in *Figure 5*. Some notes on the NMC9306 interface requirements are:

1. The SK clock frequency should be in the 0 kHz–250 kHz range.
2. $\overline{CS}$ low period following an Erase/Write instruction must not exceed 30 ms maximum. It should be set at typical or minimum specification of 10 ms.
3. The start bit on DI must be set by a "0" to "1" transition following a $\overline{CS}$ enable ("0" to "1") when executing any instruction. One $\overline{CS}$ enable transition can only execute one instruction.
4. In the read mode, following an instruction and data train, the DI can be a "don't care", while the data is being outputted, i.e., for the next 17 bits or clocks. The same is true for other instructions after the instrution and data has been fed in.
5. The data out train starts with a dummy bit 0 and is terminated by chip deselect. Any extra SK cycle after 16 bits is not essential.
   If $\overline{CS}$ is held on after all 16 of the data bits have been outputed, the DO will output the state of DI until another $\overline{CS}$ LO to HI transition starts a new instruction cycle.
6. After a read cycle, the $\overline{CS}$ must be brought low for one SK clock cycle before another instruction cycle starts.



TL/DD/10252–5

FIGURE 5. NMC9306-COP888CG Interface

Instruction Set

Commands	Start Bit	Opcode	Address	Comments
READ	1	0000	A3A2A1A0	Read Register 0–15
WRITE	1	1000	A3A2A1A0	Write Register 0–15
ERASE	1	0100	A3A2A1A0	Erase Register 0–15
EWEN	1	1100	00 01	Write/Erase Enable
ENDS	1	1100	00 10	Write/Erase Disable
***WRAL	1	1100	01 00	Write All Registers
ERAL	1	1100	01 01	Read All Registers

Where A3A2A1A0 corresponds to one of the sixteen 16-bit registers.

All commands, data in, and data out are shifted in/out on the rising edge of the SK clock.

Write/Erase is then done by pulsing $\overline{CS}$ low for 10 ms.

All instructions are initiated by a LO–HI transition on $\overline{CS}$ followed by a LO–HI transition on DI.

READ— After read command is shifted in DI becomes don't care and data can be read out on data out, starting with dummy bit zero.

WRITE— Write command shifted in followed by data in (16 bits) the$\overline{CS}$ pulsed low for 10 ms minimum.

ERASE/ERASE ALL— Command shifted in followed by $\overline{CS}$ low.

WRITE ALL— Pulsing $\overline{CS}$ low for 10 ms.

ENABLE/DISABLE— Command shifted in.

A detailed explanation of the E²PROM timing diagrams, instruction set and the various considerations could be found in the NMC9306 data sheet. A source listing of the software to interface the NMC9306 with the COP888CG is provided.

SOURCE LISTING

```
.INCLD COP888.INC
;
;This program provides in the form of subroutines, the ability to erase,enable, disable, read and write to the COP494 EEPROM.
;
;
SNDBUF = 0                  ;CONTAINS THE COMMAND BYTE TO BE WRITTEN TO COP494
RDATL  = 1                  ;LOWER BYTE OF THE COP494 REGISTER DATA READ
RDATH  = 2                  ;UPPER BYTE OF THE COP494 REGISTER DATA READ
WDATL  = 3                  ;LOWER BYTE OF THE DATA TO BE WRITTEN TO COP494
                            ;REGISTER
WDATH  = 4                  ;UPPER BYTE OF THE DATA TO BE WRITTEN TO COP494
                            ;REGISTER
ADRESS = 5                  ;THE LOWER 4-BITS OF THIS LOCATION CONTAIN THE
                            ;ADDRESS
                            ;OF THE COP494 REGISTER TO BE READ/WRITTEN
FLAGS  = 6                  ;USED FOR SETTING UP FLAGS
                            ;
                            ; FLAG VALUE     ACTION
                            ; --------       ------
                            ;  00            ERASE,ENABLE,DISABLE,ERASE ALL
                            ;  01            READ CONTENTS OF COP494 REGISTER
                            ;  03            WRITE TO COP494 REGISTER
                            ;  OTHERS        ILLEGAL COMBINATION
DLYH   = 0F0
DLYL   = 0F1
;
;THE INTERFACE BETWEEN THE COP888CG AND THE COP494 (256-BIT EEPROM) CONSISTS OF FOUR LINES. THE
;G0 (CHIP SELECT LINE), G4 (SERIAL OUT SO), G5 (SERIAL CLOCK SK) ;AND G6 (SERIAL IN SI).
;
;    INITIALIZATION
;
                LD          PORTGC,#031         ;Setup G0,G4,G5 as outputs
                LD          PORTGD,#00          ;Initialize G data reg to zero
                LD          CNTROL,#08          ;Enable MSEL, select MW rate of 2tc
                LD          B,#PSW
                LD          X,#SIOR
;
;THIS ROUTINE ERASES THE MEMORY LOCATION POINTED TO BY THE ADDRESS CONTAINED IN THE LOCATION
;"ADRESS". THE LOWER NIBBLE OF "ADRESS" CONTAINS THE COP494 REGISTER ADDRESS AND THE UPPER NIBBLE
;SHOULD BE SET TO ZERO.
;
ERASE:          LD          A,ADRESS
                OR          A,#0C0
                X           A,SNDBUF
                LD          FLAGS,#0
                JSR         INIT
                RET
;
;THIS ROUTINE ENABLES PROGRAMMING OF THE COP494. PROGRAMMING MUST BE PRECEDED ONCE BY A
;PROGRAMMING ENABLE (EWEN).
;
EWEN:           LD          SNDBUF,#030
```

TL/DD/10252-6

```
                LD      FLAGS,#0
                JSR     INIT
                RET
;
;THIS ROUTINE DISABLES PROGRAMMING OF THE COP494.
;
EWDS:           LD      SNDBUF,#0
                LD      FLAGS,#0
                JSR     INIT
                RET
;
;THIS ROUTINE ERASES ALL REGISTERS OF THE COP494.
;
ERAL:           LD      SNDBUF,#020
                LD      FLAGS,#0
                JSR     INIT
                RET
;
;THIS ROUTINE READS THE CONTENTS OF THE COP494 REGISTER. THE COP494 ADDRESS IS SPECIFIED IN THE
;LOWER NIBBLE OF LOCATION "ADRESS". THE UPPER NIBBLE SHOULD BE SET TO ZERO. THE 16-BIT CONTENTS OF
;THE COP494 REGISTER ARE STORED IN RDATL AND RDATH.
;
READ:           LD      A,ADRESS
                OR      A,#080
                X       A,SNDBUF
                LD      FLAGS,#1
                JSR     INIT
                RET
;
;THIS ROUTINE WRITES A 16-BIT VALUE STORED IN WDATL AND WDATH TO THE COP494 REGISTER WHOSE ADDRESS
;IS CONTAINED IN THE LOWER NIBBLE OF THE LOCATION "ADRESS". THE UPPER NIBBLE OF ADDRESS LOCATION
;SHOULD BE SET TO ZERO.
;
WRITE:          LD      A,ADRESS
                OR      A,#040
                X       A,SNDBUF
                LD      FLAGS,#3
                JSR     INIT
                RET
;
;THIS ROUTINE SENDS OUT THE START BIT AND THE COMMAND BYTE. IT ALSO DECIPHERS THE CONTENTS OF THE
;FLAG LOCATION AND TAKES A DECISION REGARDING WRITE, READ OR RETURN TO THE CALLING ROUTINE.
;
INIT:           SBIT    0,PORTGD        ;SET CHIP SELECT HIGH
                LD      SIOR,#001       ;LOAD SIOR WITH START BIT
                SBIT    BUSY,[B]        ;SEND OUT THE START BIT
PUNT1:          IFBIT   BUSY,[B]
                JP      PUNT1
                LD      A,SNDBUF
                X       A,[X]           ;LOAD SIOR WITH COMMAND BYTE
                SBIT    BUSY,[B]        ;SEND OUT COMMAND BYTE
PUNT2:          IFBIT   BUSY,[B]
                JP      PUNT2
                IFBIT   0,FLAGS         ;ANY FURTHER PROCESSING ?
```

TL/DD/10252–7

```
                JP          NOTDON          ;YES
                RBIT        0,PORTGD        ;NO, RESET CS AND RETURN
                RET
;
NOTDON:         IFBIT       1,FLAGS         ;READ OR WRITE?
                JP          WR494           ;JUMP TO WRITE ROUTINE
                LD          SIOR,#000       ;NO, READ COP494
                SBIT        BUSY,PSW        ;DUMMY CLOCK TO READ ZERO
                RBIT        BUSY,[B]
                SBIT        BUSY,[B]
PUNT3:          IFBIT       BUSY,[B]
                JP          PUNT3
                X           A,[X]
                SBIT        BUSY,[B]
                X           A,RDATH
PUNT4:          IFBIT       BUSY,[B]
                JP          PUNT4
                LD          A,[X]
                X           A,RDATL
                RBIT        0,PORTGD
                RET
;
WR494:          LD          A,WDATH
                X           A,[X]
                SBIT        BUSY,[B]
PUNT5:          IFBIT       BUSY,[B]
                JP          PUNT5
                LD          A,WDATL
                X           A,[X]
                SBIT        BUSY,[B]
PUNT6:          IFBIT       BUSY,[B]
                JP          PUNT6
                RBIT        0,PORTGD
                JSR         TOUT
                RET
;
;ROUTINE TO GENERATE DELAY FOR WRITE
;
TOUT:           LD          DLYH,#00A
WAIT:           LD          DLYL,#0FF
WAIT1:          DRSZ        DLYL
                JP          WAIT1
                DRSZ        DLYH
                JP          WAIT
                RET
                .END
```

TL/DD/10252–8

COP472-COP820 Interface

The pin connection required for interfacing COP472-3 Liquid Crystal Display (LCD) Controller with COP820C microcontroller is shown in *Figure 6*. The COP472-3 drives a multiplexed liquid crystal display directly. Data is loaded serially and is held in internal latches. One COP472-3 can drive 36 segments and two or more COP472-3's can be cascaded to drive additional segments as long as the output loading capacitance does not exceed specifications.

The COP472-3 requires 40 information bits: 36 data and 4 control. The function of each control bit is described briefly. Data is loaded in serially, in sets of eight bits. Each set of segment data is in the following format:

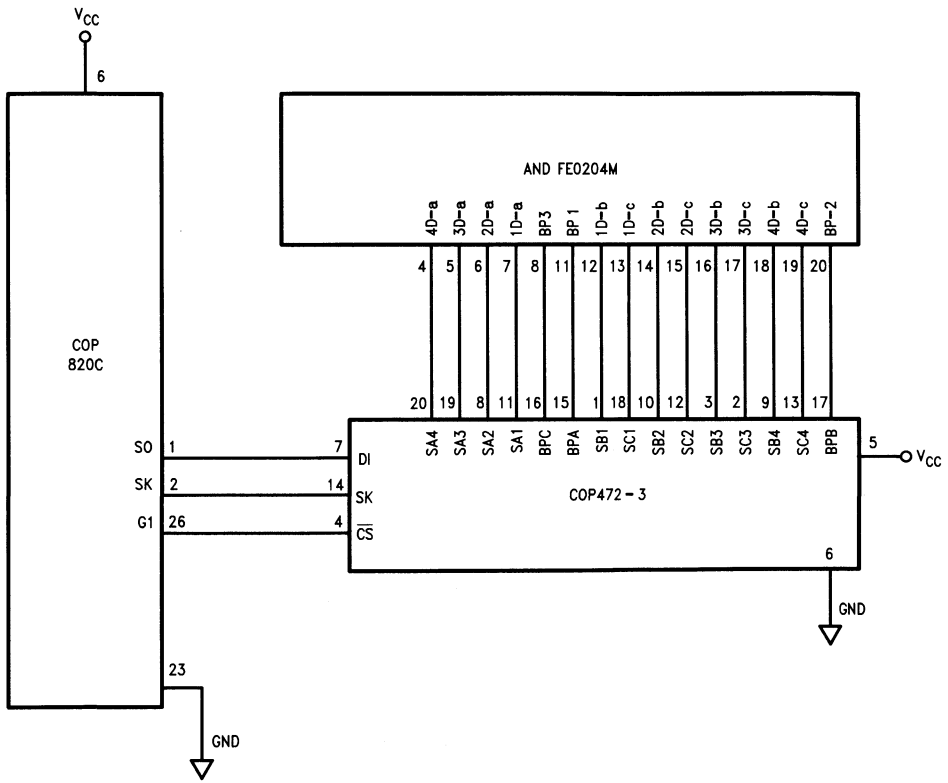
SA	SB	SC	SD	SE	SF	SG	SH

Data is shifted into an eight bit shift register. The first bit of data is for segment H, digit 1, and the eight bit is for segment A, digit 1. A set of eight bits are shifted in and then loaded into the digit one latches. The second, third, and fourth set is then loaded sequentially. The fifth set of data bits contain special segment data and control data in the following format:

SYNC	Q7	Q6	X	SP4	SP3	SP2	SP1

The first four bits shifted in contain the special character segment data. The fifth bit is not used. The sixth and seventh bits program the COP472-3 as a stand alone LCD driver or as a master or slave for cascading COP472-3's. The Table IV summarizes the function of bits six and seven.

The eight bit is used to synchronize two COP472-3's to drive an $8\frac{1}{2}$ digit display. A detailed explanation of the various timing diagrams, loading sequence and segment/backplane multiplex scheme can be found in the data sheets of COP472-3. The source listing of the software used in the interface is provided.



TL/DD/10252-12

FIGURE 6. COP472–COP820C Interface

SOURCE LISTING

```
;THIS PROGRAM DISPLAYS FOUR DIGITS OF THE RAM SPECIFIED BY; THE ADDRESS POINTER "HEAD" ON A 4 DIGIT 3
;DECIMAL POINT (MULTIPLEXED) LCD DISPLAY. THE DATA STREAM IS SENT OUT SERIALLY THROUGH THE
;MICROWIRE/PLUS INTERFACE TO THE COP472 LCD DISPLAY DRIVER. NOTE: THE RAM CONTENTS SHOULD BE
;BETWEEN "0" AND "F".
;
                .TITLE      LCD
;
```

TL/DD/10252–9

```
;
                .CHIP       820
;
;
                PORTGD      = 0D4           ;PORT G DATA REGISTER
                PORTGC      = 0D5           ;PORT G CONFIGURATION
;
;
                SIO         = 0E9           ;MICROWIRE SHIFT REGISTER
;
;
                PSW         = 0EF           ;PSW REGISTER
                CNTRL       = 0EE           ;CNTRL REGISTER
;
;
                CONTRL      = 04            ;MEMORY LOCATION FOR THE
;                                           ;COP472 CONTROL WORD
                HEAD        = 00            ;STARTING MEMORY LOC FOR
                                            ;DATA TO BE DISPLAYED
                MEMSTR      = 05            ;STARTING MEMORY LOC FOR
                                            ;STORING SEGMENT DATA
                MEMEND      = 08            ;MEMORY LOC FOR LAST
                                            ;SEGMENT DATA
;
;
START:          LD          CNTRL,#08       ;SET MSEL BIT IN CNTRL
                LD          PORTGC,#032     ;SET G5,G4& G1 AS OUTPUTS
                LD          CONTRL,#0FC     ;SET COP472 IN STAND ALONE MODE
;                                           ;
;                                           ;
;
;THIS ROUTINE  GETS THE SEGMENT DATA FOR RAM DIGITS POINTED BY B REGISTER AND STORES IN RAM MEMORY
;POINTED BY X REGISTER
;
;
AGAIN:          LD          B,#HEAD         ;POINTER TO START ADDRESS
                LD          X,#MEMSTR       ;POINTER TO STORE ADDRESS
NEXDIG:         LD          A,[B+]          ;LOAD A WITH RAM DIGIT AND
                                            ;INCREMENT B POINTER
                ADD         A,#0F0          ;ADD OFFSET TO THE DIGIT
                LAID                        ;LOOKUP SEGMENT DATA TO A
                X           A,[X+]          ;STORE IN MEMORY
                IFBNE       #04             ;CHECK FOR END OF FOUR
                                            ;DIGITS AND REPEAT
                JP          NEXDIG          ;IF NECESSARY
;
;
; THIS ROUTINE DISPLAYS THE CONTENTS OF FOUR MEMORY LOCATION
; ON THE LCD DISPLAY.
;
;
DSP:            LD          B,#MEMEND       ;LOAD THE START ADDRESS
                RBIT        1,PORTGD        ;BIT G1 IS USED TO SELECT
                                            ;COP472 (PIN 4)
```

TL/DD/10252–10

```
REPEAT:     LD      A,[B-]          ;SEGMENT DATA TO A
            X       A,SIO           ;LOAD THE SIO REGISTER
            SBIT    #2,PSW          ;SET BUSY BIT IN PSW
WAIT:       IFBIT   #2,PSW          ;WAIT TILL SHIFTING IS
            JP      WAIT            ;COMPLETE
            IFBNE   #04             ;CHECK FOR END OF FOUR
            JP      REPEAT          ;DIGITS AND REPEAT
            SBIT    1,PORTGD        ;DESELECT COP472
LOOP:       JP      LOOP            ;DONE DISPLAYING
;
;
; STORE THE LOOKUP TABLE FOR SEGMENT DATA IN ROM LOCATION 0F0
;
;
            .=0F0
;
            .BYTE   03F,006,05B,04F ;DATA FOR 0,1,2,3
            .BYTE   066,06D,07D,07  ;DATA FOR 4,5,6,7
            .BYTE   07F,067,077,07C ;DATA FOR 8,9,A,B
            .BYTE   039,05E,079,071 ;DATA FOR C,D,E,F
;
;
            .END
```

TL/DD/10252–11

The code listed in this App Note is available on Dial-A-Helper.

Dial-A-Helper is a service provided by the Microcontroller Applications Group. The Dial-A-Helper system provides access to an automated information storage and retrieval system that may be accessed over standard dial-up telephone lines 24 hours a day. The system capabilities include a MESSAGE SECTION (electronic mail) for communicating to and from the Microcontroller Applications Group and a FILE SECTION mode that can be used to search out and retrieve application data about NSC Microcontrollers. The minimum system requirement is a dumb terminal, 300 or 1200 baud modem, and a telephone.

With a communications package and a PC, the code detailed in this App Note can be downloaded from the FILE SECTION to disk for later use. The Dial-A-Helper telephone lines are:

Modem (408) 739-1162
Voice (408) 721-5582

For Additional Information, Please Contact Factory

COP800 MathPak

National Semiconductor
Application Note 596
Verne H. Wilson

OVERVIEW

This application note discusses the various arithmetic operations for National Semiconductor's COP800 family of 8-bit microcontrollers. These arithmetic operations include both binary and BCD (Binary Coded Decimal) operation. The four basic arithmetic operations (add, subtract, multiply, divide) are outlined in detail, with several examples shown for both binary and BCD addition and subtraction. Multiplication, division, and BCD conversion algorithms are also provided. Both BCD to binary and binary to BCD conversion subroutines are included, as well as the various multiplication and division subroutines.

Four sets of optimal subroutines are provided for

1. Multiplication
2. Division
3. Decimal (Packed BCD) to binary conversion
4. Binary to decimal (Packed BCD) conversion

One class of subroutines is optimized for minimal COP800 program code, while the second class is optimized for minimal execution time in order to optimize throughput time.

This application note is organized in four different sections. The first section outlines various addition and subtraction routines, including both binary and BCD (Binary Coded Decimal). The second section outlines the multiplication algorithm and provides several optimal multiply subroutines for 1, 2, 3, and 4 byte operation. The third section outlines the division algorithm and provides several optimal division subroutines for 1, 2, 3, and 4 byte operation. The fourth section outlines both the decimal (Packed BCD) to binary and binary to decimal (Packed BCD) conversion algorithms. This section provides several optimal subroutines for these BCD conversions.

The COP800 arithmetic instructions include the Add (ADD), Add with Carry (ADC), Subtract with Carry (SUBC), Increment (INCR), Decrement (DECR), Decimal Correct (DCOR), Clear Accumulator (ACC), Set Carry (SC), and Reset Carry (RC). The shift and rotate instructions, which include the Rotate Right through Carry (RRC) and the Swap Accumulator Nibbles (SWAP), may also be considered as arithmetic instruction variations. The RRC instruction is instrumental in writing a fast multiply routine.

1.0 BINARY AND BCD ADDITION AND SUBTRACTION

In subtraction, a borrow is represented by the absence of a carry and vice versa. Consequently, the carry flag needs to be set (no borrow) before a subtraction, just as the carry flag is reset before an addition. The ADD instruction does not use the carry flag as an input, nor does it change the carry flag. It should also be noted that both the carry and half carry flags (bits 6 and 7, respectively, of the PSW control register) are cleared with reset, and remain unchanged with the ADD, INC, DEC, DCOR, CLR and SWAP instructions. The DCOR instruction uses both the carry and half carry flags. The SC instruction sets both the carry and half carry flags, while the RC instruction resets both these flags.

The following program examples illustrate additions and subtractions of 4-byte data fields in both binary and BCD (Binary Coded Decimal). The four bytes from data memory locations 24 through 27 are added to or subtracted from the four bytes in data memory locations 16 through 19. The results replace the data in memory locations 24 through 27.

These operations are performed both in Binary and BCD. It should be noted that the BCD pre-conditioning of Adding (ADD) the hex 66 is only necessary with the BCD addition, not with the BCD subtraction. The (Binary Coded Decimal) DCOR (Decimal Correct) instruction uses both the carry and half carry flags as inputs, but does not change the carry and half carry flags. Also note that the #12 with the IFBNE instruction represents 28 − 16, since the IFBNE operand is modulo 16 (remainder when divided by 16).

BINARY ADDITION:

```
        LD          X,#16           ;  NO LEADING ZERO
        LD          B,#24           ;     INDICATES DECIMAL
        RC                          ;  RESET CARRY TO START
LOOP:   LD          A,[X+]          ;  [X] TO ACC
        ADC         A,[B]           ;  ADD [B] TO ACC
        X           A,[B+]          ;  RESULT TO [B]
        IFBNE       #12             ;  IF STILL IN DATA FIELD
        JP          LOOP            ;     JUMP BACK TO REPEAT LOOP
        IFC                         ;  IF TERMINAL CARRY,
        JP          OVFLOW          ;     JUMP TO OVERFLOW
```

BINARY SUBTRACTION:

```
        LD          X,#010          ;  LEADING ZERO
        LD          B,#018          ;     INDICATES HEX
        SC                          ;  RESET BORROW TO START
LOOP:   LD          A,[X+]          ;  [X] TO ACC
        SUBC        A,[B]           ;  SUBTRACT [B] FROM ACC
        X           A,[B+]          ;  RESULT TO [B]
        IFBNE       #12             ;  IF STILL IN DATA FIELD
        JP          LOOP            ;     JUMP BACK TO REPEAT LOOP
        IFNC                        ;  IF TERMINAL BORROW,
        JP          NEGRSLT         ;     JUMP TO NEGATIVE RESULT
```

BCD ADDITION:

```
        LD          X,#010          ;  LEADING ZERO
        LD          B,#018          ;     INDICATES HEX
        RC                          ;  RESET CARRY TO START
LOOP:   LD          A,[X+]          ;  [X] TO ACC
        ADD         A,#066          ;  ADD HEX 66 TO ACC
        ADC         A,[B]           ;  ADD [B] TO ACC
        DCOR        A               ;  DECIMAL CORRECT RESULT
        X           A,[B+]          ;  RESULT TO [B]
        IFBNE       #12             ;  IF STILL IN DATA FIELD
        JP          LOOP            ;     JUMP BACK TO REPEAT LOOP
        IFC                         ;  IF TERMINAL CARRY
        JP          OVFLOW          ;     JUMP TO OVERFLOW
```

BCD SUBTRACTION:

```
        LD          X,#16           ;  NO LEADING ZERO
        LD          B,#24           ;     INDICATES DECIMAL
        C
LOOP:   LD          A,[X+]          ;  [X] TO ACC
        SUBC        A,[B]           ;  SUBTRACT [B] FROM ACC
        DCOR        A               ;  DECIMAL CORRECT RESULT
        X           A,[B+]          ;  RESULT TO [B]
        IFBNE       #12             ;  IF STILL IN DATA FIELD
        JP          LOOP            ;     JUMP BACK TO REPEAT LOOP
        IFNC                        ;  IF TERMINAL BORROW
        JP          NEGRSLT         ;     JUMP TO NEGATIVE RESULT
```

The astute observer will notice that these previous additions and subtractions are not "adding machine" type arithmetic operations in that the result replaces the second operand rather than the first. The following program examples illustrate "adding machine" type operation where the result replaces the first operand. With subtraction, this entails the result replacing the minuend rather than the subtrahend. Note that the B and X pointers are now reversed.

BINARY ADDITION:

```
        LD          B,#16           ;  B POINTER AT FIRST OPERAND
        LD          X,#24           ;  X POINTER AT SECOND OPERAND
        RC                          ;  RESET CARRY TO START
LOOP:   LD          A,[X+]          ;  [X] TO ACC
        ADC         A,[B]           ;  ADD [B] TO ACC
        X           A,[B+]          ;  RESULT TO [B]
        IFBNE       #4              ;  IF STILL IN DATA FIELD
        JP          LOOP            ;     JUMP BACK TO REPEAT LOOP
        IFC                         ;  IF TERMINAL CARRY
        JP          OVFLOW          ;     JUMP TO OVERFLOW
```

BINARY SUBTRACTION:

```
        LD          B,#010          ;  B POINTER AT FIRST OPERAND
        LD          X,018           ;  X POINTER AT SECOND OPERAND
        SC                          ;  RESET BORROW TO START
LOOP:   LD          A,[X+]          ;  [X] TO ACC
        X           A,[B]           ;  EXCHANGE [B] AND ACC
        SUBC        A,[B]           ;  SUBTRACT [B] FROM ACC
        X           A,[B+]          ;  RESULT TO [B]
        IFBNE       #4              ;  IF STILL IN DATA FIELD
        JP          LOOP            ;     JUMP BACK TO REPEAT LOOP
        IFNC                        ;  IF TERMINAL BORROW
        JP          NEGRSLT         ;     JUMP TO NEGATIVE RESULT
```

BCD ADDITION:

```
        LD          B,#010          ;  B POINTER AT FIRST OPERAND
        LD          X,#018          ;  X POINTER AT SECOND OPERAND
        RC                          ;  RESET CARRY TO START
LOOP:   LD          A,[X+]          ;  [X] TO ACC
        ADD         A,#066          ;  ADD HEX66 TO ACC
        ADC         A,[B]           ;  ADD [B] TO ACC
        DCOR        A               ;  DECIMAL CORRECT RESULT
        X           A,[B+]          ;  RESULT TO [B]
        IFBNE       #4              ;  IF STILL IN DATA FIELD
        JP          LOOP            ;     JUMP BACK TO REPEAT LOOP
        IFC         ;               ;  IF TERMINAL CARRY
        JP          OVFLOW          ;     JUMP TO OVERFLOW
```

BCD SUBTRACTION:

```
        LD          B,#16           ;  B POINTER AT FIRST OPERAND
        LD          X,#24           ;  X POINTER AT SECOND OPERAND
        SC                          ;  RESET BORROW TO START
LOOP:   LD          A,[X+]          ;  [X] TO ACC
        X           A,[B]           ;  EXCHANGE [B] AND ACC
        SUBC        A,[B]           ;  SUBTRACT [B] FROM ACC
        DCOR        A               ;  DECIMAL CORRECT RESULT
        X           A,[B+]          ;  RESULT TO [B]
        IFBNE       #4              ;  IF STILL IN DATA FIELD
        JP          LOOP            ;     JUMP BACK TO REPEAT LOOP
        IFNC                        ;  IF TERMINAL BORROW
        JP          NEGRSLT         ;     JUMP TO NEGATIVE RESULT
```

Let us now consider a hybrid arithmetic example, where we wish to add five successive bytes of a data table in ROM program memory to a two byte sum, and then subtract the SUM result from a two byte total TOT. Let us further assume that the ROM table is located starting at program memory address 0401, while SUM and TOT are at RAM data memory locations [1, 0] and [3, 2] respectively, and that we wish to encode the program as a subroutine.

ROM Table:

. = 0401
. Byte 102
. Byte 41
. Byte 31
. Byte 26
. Byte 5

ROM Table Accessed Top Down

SUMLO = 0
SUMHI = 1
TOTLO = 2
TOTHI = 3

```
ARITH1: LD      X,#5            ; SET UP ROM TABLE POINTER
        LD      B,#0            ; SET UP SUM POINTER
LOOP:   RC                      ; RESET CARRY TO START ADDITION
        LD      A,X             ; ROM POINTER TO ACC
        LAID                    ; TABLE VALUE FROM ROM TO ACC
        ADC     A,[B]           ; ADD SUMLO TO ACC
        X       A,[B+]          ; RESULT TO SUMLO
        CLR     A               ; CLEAR ACC
        ADC     A,[B]           ; ADD SUMHI TO ACC
        X       A,[B-]          ; RESULT TO SUMHI
        DRSZ    X               ; DECR AND TEST ROM PTR FOR ZERO
        JP      LOOP            ; JUMP BACK TO REPEAT LOOP
                                ;    IF X PTR NOT ZERO
        SC                      ; RESET BORROW TO START SUBTRACTION
        LD      B,#2            ; SET UP TOT POINTER
LUP:    LD      A,[X+]          ; SUBTRAHEND (SUM) TO ACC
        X       A,[B]           ; REVERSE OPERANDS
        SUBC    A,[B]           ;    FOR SUBTRACTION
        X       A,[B+]          ; RESULT TO TOT
        IFBNE   #4              ; IF STILL IN TOT FIELD
        JP      LUP             ;    JUMP BACK TO REPEAT LUP
        RET                     ; RETURN FROM SUBROUTINE
```

2.0 MULTIPLICATION

The COP800 multiplications are all based on starting the multiplier in the low order end of the double length product space. The high end of the double length product space is initially cleared, and then the double length product is shifted right one bit. The bit shifted out from the low order end represents the low order bit of the multiplier. If this bit is a "1", the multiplicand is added to the high end of the double length product space. The entire shifting process and the conditional addition of the multiplicand to the upper end of the double length product is then repeated. The number of shift cycles is equal to the number of bit positions in the multiplier plus one extra shift cycle. This extra terminal shift cycle is necessary to correctly align the resultant product.

Note that an M byte multiplicand multiplied by an N byte multiplier will result in an M + N byte double length product. However, these multiplication subroutines will only use 2M + N + 1 bytes of RAM memory space, since the multiplier initially occupies the low order end of the double length product. The one extra byte is necessary for the shift counter CNTR.

The minimal code (28 byte) general multiplication subroutine is shown with two different examples, MY2448 and MY4824. Both examples multiply 24 bits by 48 bits. The MY2448 subroutine uses the 48-bit operand as the multiplier, and consequently uses minimal RAM as well as minimal program code. The MY4824 subroutine uses the 24-bit operand as the multiplier, and consequently executes considerably faster than the minimal RAM MY2448 subroutine.

MPY88
- 8 by 8 Multiplication Subroutine
- 19 Bytes
- 180 Instruction Cycles
- Minimum Code

MLT88
- Fast 8 by 8 Multiplication Subroutine
- 42 Bytes
- 145 Instruction Cycles

VFM88
- Very Fast 8 by 8 Multiply Subroutine
- 96 Bytes
- 116 Instruction Cycles

MPY168
- Fast 16 by 8 Multiplication Subroutine
- 36 Bytes
- 230 Instruction Cycles Average
- 254 Instruction Cycles Maximum

MPY816 (or MPY824, MPY832)
- 8 by 16 (or 24, 32) Multiply Subroutine
- 22 Bytes
- 589 (or 1065, 1669) Instruction Cycles Average
- 597 (or 1077, 1685) Instruction Cycles Maximum
- Minimum Code, Minimum RAM
- Extendable Routine for MPY8XX by Changing Parameters, with Number of Bytes (22) Remaining a Constant

MPY248
- Fast 24 by 8 Multiplication Subroutine
- 47 Bytes
- 289 Instruction Cycles Average
- 333 Instruction Cycles Maximum

MX1616
- Fast 16 by 16 Multiplication Subroutine
- 39 Bytes
- 498 Instruction Cycles Average
- 546 Instruction Cycles Maximum

MP1616
- 16 by 16 Multiplicand Subroutine
- 29 Bytes
- 759 Instruction Cycles Average
- 807 Instruction Cycles Maximum
- Almost Minimum Code

MY1616 (or MY1624, MY1632)
- 28 Bytes
- 16 by 16 (or 24, 32) Multiply Subroutine
- 861 (or 1473, 2213) Inst. Cycles Average
- 1029 (or 1725, 2549) Inst. Cycles Maximum
- Minimum Code, Minimum RAM
- Extendable Routne for MY16XX by Changing Parameters, with Number of Bytes (28) Remaining a Constant

Minimal general multiplication subroutine for any number of bytes in multiplicand and multiplier
- 28 Bytes
- Minimum Code
- MY2448 Used as First Example, with Minimum RAM and 4713 Instruction Cycles Average 5457 Instruction Cycles Maximum
- MY4824 Used as Second Example, with Non Minimal RAM and 2751 Instruction Cycles Average 3483 Instruction Cycles Maximum

MPY88—8 BY 8 MULTIPLICATION SUBROUTINE

```
          MINIMUM CODE
          19 BYTES
          180 INSTRUCTION CYCLES
          MULTIPLICAND IN [0]          (ICAND)
          MULTIPLIER IN [1]            (IER)
          PRODUCT IN [2,1]             (PROD)
MPY88:    LD          CNTR,#9          ;  LD CNTR WITH LENGTH OF
          RC                           ;     MULTIPLIER FIELD + 1
          LD          B,#2
          CLR         A                ;  CLEAR UPPER PRODUCT
M88LUP:   RRC         A                ;  RIGHT SHIFT
          X           A,[B-]           ;     UPPER PRODUCT
          LD          A,[B]
          RRC         A                ;  RIGHT SHIFT LOWER
          X           A,[B-]           ;     PRODUCT/MULTIPLIER
          CLR         A                ;  CLR ACC AND TEST LOW
          IFC                          ;     ORDER MULTIPLER BIT
          LD          A,[B]            ;  MULTIPLICAND TO ACC IF
          RC                           ;     LOW ORDER BIT = 1
          LD          B,#2             ;  ADD MULTIPLICAND TO
          ADC         A,[B]            ;     UPPER PRODUCT
          DRSZ        CNTR             ;  DECREMENT AND TEST
          JP          M88LUP           ;     CNTR FOR ZERO
          RET                          ;  RETURN FROM SUBROUTINE
```

MLT88—FAST 8 BY 8 MULTIPLICATION SUBROUTINE

```
          42 BYTES
          145 INSTRUCTION CYCLES
          MULTIPLICAND IN [0]          (ICAND)
          MULTIPLIER IN [1]            (IER)
          PRODUCT IN [2,1]             (PROD)
MLT88:    LD          CNTR,#3          ;  LOAD CNTR WITH
          RC                           ;   1/3 OF LENGTH OF
          LD          B,#2             ;   (MULTIPLIER FIELD + 1)
          CLR         A                ;  CLEAR UPPER PRODUCT
;
ML88LP:   RRC         A                ;  RIGHT SHIFT   ***
          X           A,[B-]           ;     UPPER PRODUCT
          LD          A,[B]
          RRC         A                ;  RIGHT SHIFT LOWER
          X           A,[B-]           ;     PRODUCT/MULTIPLIER
          CLR         A                ;  CLR ACC AND TEST LOW
          IFC                          ;     ORDER MULTIPLIER BIT
          LD          A,[B]            ;  MULTIPLICAND TO ACC IF
          RC                           ;     LOW ORDER BIT = 1
          LD          B,#2             ;  ADD MULTIPLICAND TO
          ADC         A,[B]            ;     UPPER PRODUCT ***
;
          RRC         A                ;  REPEAT THE ABOVE
          X           A,[B-]           ;     11 BYTE
          LD          A,[B]            ;     13 INSTRUCTION
          RRC         A                ;     CYCLE PROGRAM
          X           A,[B-]           ;     SECTION (WITH
          CLR         A                ;     THE *** DELIMITERS)
          IFC                          ;     TWICE MORE FOR A
          LD          A,[B]            ;     TOTAL OF THREE TIMES
          RC
          LD          B,#2
          ADC         A,[B]            ;  END OF SECOND REPEAT
;
          RRC         A                ;  START OF THIRD REPEAT
          X           A,[B-]
          LD          A,[B]
          RRC         A
          X           A,[B-]
          CLR         A
          IFC
          LD          A,[B]
          RC
          LD          B,#2
          ADC         A,[B]            ;  END OF THIRD REPEAT
;
          DRSZ        CNTR             ;  DECREMENT AND TEST
          JMP         ML88LP           ;   CNTR FOR ZERO
          RET                          ;  RETURN FROM SUBROUTINE
```

VFM88—VERY FAST 8 BY 8 MULTIPLY SUBROUTINE

```
          96 BYTES
          116 INSTRUCTION CYCLES

          MULTIPLICAND IN [0]          (ICAND)
          MULTIPLIER IN [1]            (IER)
          PRODUCT IN [2,1]             (PROD)
VFM88:    RC
          LD          B,#2
          LD          [B-],#0          ;  CLEAR UPPER PRODUCT
          LD          A,[B]
          RRC         A                ;  RIGHT SHIFT LOWER
          X           A,[B-]           ;   PRODUCT/MULTIPLIER
          CLR         A                ;  CLR ACC AND TEST LOW
          IFC                          ;     ORDER MULTIPLIER BIT
          LD          A,[B]            ;  MULTIPLICAND TO ACC IF
          RC                           ;     LOW ORDER BIT = 1
          LD          B,#2             ;  ADD MULTIPLICAND TO
          ADC         A,[B]            ;     UPPER PRODUCT
;
          RRC         A                ;  RIGHT SHIFT   ***
          X           A,[B-]           ;     UPPER PRODUCT
          LD          A,[B]
          RRC         A                ;  RIGHT SHIFT LOWER
          X           A,[B-]           ;     PRODUCT/MULTIPLIER
          CLR         A                ;  CLR ACC AND TEST LOW
          IFC                          ;     ORDER MULTIPLIER BIT
          LD          A,[B]            ;  MULTIPLICAND TO ACC IF
          RC                           ;     LOW ORDER BIT = 1
          LD          B,#2             ;  ADD MULTIPLICAND TO
          ADC         A,[B]            ;     UPPER PRODUCT ***
;
;    THE ABOVE 11 BYTE, 13 INSTRUCTION CYCLE SECTION WITH THE ***
;    DELIMITERS REPRESENTS THE PROCESSING FOR ONE MULTIPLIER BIT.
;
;
;         ---                         ;  REPEAT THE
;                                     ;  ABOVE SECTION
;         ---                         ;  SIX MORE TIMES,
;                                     ;  FOR A TOTAL
;         ---                         ;  OF SEVEN TIMES
;
          RRC         A                ;  RIGHT SHIFT
          X           A,[B-]           ;  UPPER PRODUCT
          LD          A,[B]
          RRC         A                ;  RIGHT SHIFT LOWER
          X           A,[B]            ;     PRODUCT/MULTIPLIER
          RET                          ;  RETURN FROM SUBROUTINE
;
;
;
```

MPY168—FAST 16 BY 8 MULTIPLICATION SUBROUTINE

```
          30 BYTES
          230 INSTRUCTION CYCLES AVERAGE
          254 INSTRUCTION CYCLES MAXIMUM

          MULTIPLICAND IN [1,0]        (ICAND)
          MULTIPLIER IN [2]            (IER)
          PRODUCT IN [4,3,2]           (PROD)
MPY168:   LD           CNTR,#9         ; LD CNTR WITH LENGTH OF
          RC                           ;  MULTIPLIER FIELD + 1
          LD           B,#4
          LD           [B-],#0         ; CLEAR
          LD           [B-],#0         ;    UPPER PRODUCT
          JP           MP168S
M168LP:   RRC          A               ; RIGHT SHIFT UPPER
          X            A,[B-]          ;  BYTE OF PRODUCT
          LD           A,[B]
          RRC          A               ; RIGHT SHIFT MIDDLE
          X            A,[B-]          ;  BYTE OF PRODUCT
MP168S:   LD           A,[B]
          RRC          A               ; RIGHT SHIFT LOWER
          X            A,[B]           ;    PRODUCT/MULTIPLIER
          IFNC                         ; TEST LOWER BIT
          JP           MP168T          ;    OF MULTIPLIER
          RC                           ; CLEAR CARRY
          LD           B,#0            ; LOWER BYTE OF
          LD           A,[B]           ;    MULTIPLICAND TO ACC
          LD           B,#3            ; ADD LOWER BYTE OF
          ADC          A,[B]           ;    MULTIPLICAND TO
          X            A,[B]           ;    MIDDLE BYTE OF PROD
          LD           B,#1            ; UPPER BYTE OF
          LD           A,[B]           ;    MULTIPLICAND TO ACC
          LD           B,#4            ; ADD UPPER BYTE OF ICAND
          ADC          A,[B]           ;    TO UPPER BYTE OF PROD
          DRSZ         CNTR            ; DECREMENT CNTR AND JUMP
          JP           M168LP          ;    BACK TO LOOP; CNTR
                                       ;    CANNOT EQUAL ZERO
MP168T:   LD           B,#4            ; HIGH ORDER PRODUCT
          LD           A,[B]           ;    BYTE TO ACC
          DRSZ         CNTR            ; DECREMENT AND TEST IF
          JP           M168LP          ;    CNTR EQUAL TO ZERO
          RET                          ; RETURN FROM SUBROUTINE
```

MPY816—(OR MPY824, MPY832) 8 BY 16 (OR 24, 32) MULTIPLY SUBROUTINE

```
          MINIMUM CODE, MINIMUM RAM
          22 BYTES
          589 (OR 1065, 1669) INSTR. CYCLES AVERAGE
          597 (OR 1077, 1685) INSTR. CYCLES MAXIMUM
          EXTENDABLE ROUTINE FOR MPY8XX BY CHANGING
           PARAMETERS, WITH NUMBER OF BYTES (22)
           REMAINING A CONSTANT.

          MULTIPLICAND IN [0]              (ICAND)
          MULTIPLIER IN [2,1] FOR 16 BIT (IER)
                       OR [3,2,1] for 24 BIT
                       OR [4,3,2,1] for 32 BIT
          PRODUCT IN [3,2,1] FOR 16 BIT    (PROD)
                       OR [4,3,2,1] FOR 24 BIT
                       OR [5,4,3,2,1] FOR 32 BIT


MPY816:   LD           CNTR,#17          ; LD CNTR WITH LENGTH OF
                                         ;  MULTIPLIER FIELD + 1
                                         ;  #17 FOR MPY816 16 BIT
                                         ;  (#25 FOR MPY824 24 BIT)
                                         ;  (#33 FOR MPY832 32 BIT)
          RC
          LD           B,#3              ; #3 FOR MPY816
                                         ; (#4 FOR MPY824)
                                         ; (#5 FOR MPY832)
          LD           [B-],#0           ; CLEAR UPPER PRODUCT
M8XXLP:   LD           A,[B]             ; FIVE INSTRUCTION
M8XXL:    RRC          A                 ;    PROGRAM LOOP TO
          X            A,[B-]            ;    RIGHT SHIFT
          IFBNE        #0                ;    PRODUCT/MULTIPLIER
          JP           M8XXLP            ; LOOP JUMP BACK
          CLR          A                 ; CLR ACC AND TEST LOW
          IFNC                           ;    ORDER MULTIPLIER BIT
          JP           M8XXT             ; JP IF LOW ORDER BIT = 0
          RC
          LD           B,#0
          LD           A,[B]             ; MULTIPLICAND TO ACC
M8XXT:    LD           B,#3              ; #3 FOR MPY816
                                         ; (#4 FOR MPY824)
                                         ; (#5 FOR MPY832)
          ADC          A,[B]             ; ADD MULTIPLICAND TO
                                         ;    UPPER BYTE OF PRODUCT
          DRSZ         CNTR              ; DECREMENT AND TEST
          JP           M8XXL             ;    CNTR FOR ZERO
          RET                            ; RETURN FROM SUBROUTINE
```

MPY248—FAST 24 BY 8 MULTIPLICATION SUBROUTINE

```
          47 BYTES
          289 INSTRUCTION CYCLES AVERAGE
          333 INSTRUCTION CYCLES MAXIMUM
          MULTIPLICAND IN [2,1,0]      (ICAND)
          MULTIPLIER IN [3]            (IER)
          PRODUCT IN [6,5,4,3]         (PROD)

MPY248:   LD           CNTR,#9         ; LD CNTR WITH LENGTH OF
          RC                           ;  MULTIPLIER FIELD + 1
          LD           B,#6
          LD           [B-],#0         ; CLEAR THREE
          LD           [B-],#0         ;    UPPER BYTES
          LD           [B-],#0         ;    OF PRODUCT
          JP           MP248S          ; JUMP TO START
M248LP:   RRC          A               ; RIGHT SHIFT HIGH
          X            A,[B-]          ;    ORDER PRODUCT BYTE
          LD           A,[B]
          RRC          A               ; RIGHT SHIFT NEXT LOWER
          X            A,[B-]          ;    ORDER PRODUCT BYTE
          LD           A,[B]
          RRC          A               ; RIGHT SHIFT NEXT LOWER
          X            A,[B-]          ;    ORDER PRODUCT BYTE
MP248S:   LD           A,[B]
          RRC          A               ; RIGHT SHIFT LOW ORDER
          X            A,[B]           ;    PRODUCT/MULTIPLIER
          IFNC                         ; TEST LOW ORDER
          JP           MP248T          ;    MULTIPLIER BIT
          RC
          LD           B,#0            ; LOAD ACC WITH LOW ORDER
          LD           A,[B]           ;    MULTIPLICAND BYTE
          LD           B,#4            ; ADD LOW ORDER ICAND
          ADC          A,[B]           ;    BYTE TO NEXT TO LOW
          X            A,[B]           ;    ORDER PRODUCT BYTE
          LD           B,#1            ; LOAD ACC WTIH MIDDLE
          LD           A,[B]           ;    MULTIPLICAND BYTE
          LD           B,#5            ; ADD MIDDLE ICAND BYTE
          ADC          A,[B]           ;    TO NEXT TO HIGH ORDER
          X            A,[B]           ;    MULTIPLICAND BYTE
          LD           B,#2            ; LOAD ACC WITH HIGH ORDER
          LD           A,[B]           ;    MULTIPLICAND BYTE
          LD           B,#6            ; ADD HIGH ORDER ICAND BYTE
          ADC          A,[B]           ;    TO HIGH ORDER PROD BYTE
          DRSZ         CNTR            ; DECREMENT CNTR AND JUMP
          JP           M248LP          ;    BACK TO LOOP; CNTR
                                       ;    CANNOT EQUAL ZERO
MP248T:   LD           B,#6            ; HIGH ORDER PRODUCT
          LD           A,[B]           ;    BYTE TO ACC
          DRSZ         CNTR            ; DECREMENT AND TEST
          JMP          M248LP          ;    CNTR FOR ZERO
          RET                          ; RETURN FROM SUBROUTINE
```

MX1616—FAST 16 BY 16 MULTIPLICATION SUBROUTINE

```
          39 BYTES
          498 INSTRUCTION CYCLES AVERAGE
          546 INSTRUCTION CYCLES AVERAGE

          MULTIPLICAND IN [1,0]     (ICAND)
          MULTIPLIER IN [3,2]       (IER)
          PRODUCT IN [5,4,3,2]      (PROD)

MX1616:   LD          CNTR,#17      ; LD CNTR WITH LENGTH OF
          RC                        ;    MULTIPLIER FIELD + 1
          LD          B,#5
          LD          [B-],#0       ; CLEAR UPPER TWO
          LD          [B-],#0       ;    PRODUCT BYTES
          JP          MXSTRT        ; JUMP TO START
MX1616L:  RRC         A             ; RIGHT SHIFT
          X           A,[B-]        ;  UPPER PRODUCT BYTE
          LD          A,[B]
          RRC         A             ; RIGHT SHIFT NEXT LOWER
          X           A,[B-]        ;    PRODUCT BYTE
MXSTRT:   LD          A,[B]
          RRC         A             ; RIGHT SHIFT PRODUCT
          X           A,[B-]        ;    UPPER MULTIPLIER BYTE
          LD          A,[B]
          RRC         A             ; RIGHT SHIFT PRODUCT
          X           A,[B]         ;    LOWER MULTIPLIER BYTE
          IFNC                      ; TEST LOW ORDER
          JP          MX1616T       ;    MULTIPLIER BIT
          RC
          LD          B,#0          ; LOAD ACC WITH LOWER
          LD          A,[B]         ;    MULTIPLICAND BYTE
          LD          B,#4          ; ADD LOWER ICAND BYTE
          ADC         A,[B]         ;    TO NEXT TO HIGH
          X           A,[B]         ;    ORDER PRODUCT BYTE
          LD          B,#1          ; LOAD ACC WITH UPPER
          LD          A,[B]         ;    MULTIPLICAND BYTE
          LD          B,#5          ; ADD UPPER ICAND BYTE TO
          ADC         A,[B]         ;    HIGH ORDER PRODUCT
          DRSZ        CNTR          ; DECREMENT CNTR AND JUMP
          JP          MX1616L       ;    BACK TO LOOP; CNTR
                                    ;    CANNOT EQUAL ZERO
MX1616T:  LD          B,#5          ; HIGH ORDER PRODUCT
          LD          A,[B]         ;    BYTE TO ACC
          DRSZ        CNTR          ; DECREMENT AND TEST
          JP          MX1616L       ;  CNTR FOR ZERO
          RET                       ; RETURN FROM SUBROUTINE
```

MP1616—16 BY 16 MULTIPLICATION SUBROUTINE

```
          MINIMUM CODE
          29 BYTES
          759 INSTRUCTION CYCLES AVERAGE
          807 INSTRUCTION CYCLES MAXIMUM]
          MULTIPLICAND IN [1,0]      (ICAND)
          MULTIPLIER IN [3,2]        (IER)
          PRODUCT IN [5,4,3,2]       (PROD)

MP1616:   LD          CNTR,#17     ; LD CNTR WITH LENGTH OF
          RC                       ;    MULTIPLIER FIELD + 1
          LD          B,#5
          LD          [B-],#0      ; CLEAR UPPER TWO
          LD          [B-],#0      ;    PRODUCT BYTES
M1616X:   LD          A,[B]        ; FIVE INSTRUCTION
M1616L:   RRC         A            ;    PROGRAM LOOP TO
          X           A,[B-]       ;    RIGHT SHIFT
          IFBNE       #1           ;    PRODUCT/MULTIPLIER.
          JP          M1616X       ;    LOOP JUMP BACK
          CLR         A            ; CLEAR ACC
          IFNC                     ; TEST LOW ORDER
          JP          M1616T       ;    MULTIPLIER BIT
          RC
          LD          B,#0         ; LOAD ACC WITH LOWER
          LD          A,[B]        ;    MULTIPLICAND BYTE
          LD          B,#4         ; ADD LOWER ICAND BYTE
          ADC         A,[B]        ;    TO NEXT TO LOW
          X           A,[B]        ;    ORDER PRODUCT BYTE
          LD          B,#1         ; LOAD ACC WITH UPPER
          LD          A,[B]        ;    MULTIPLICAND BYTE
M1616T:   LD          B,#5         ; ADD UPPER ICAND BYTE TO
          ADC         A,[B]        ;    HIGH ORDER PRODUCT
          DRSZ        CNTR         ; DECREMENT AND TEST
          JP          M1616L       ;    CNTR EQUAL TO ZERO
          RET                      ; RETURN FROM SUBROUTINE
```

MY1616 (OR MY1624, MY1632)—16 BY 16 (OR 24, 32) MULTIPLY SUBROUTINE

```
        MINIMUM CODE, MINIMUM RAM
        28 BYTES
        861 (OR 1473, 2213) INST. CYCLES AVERAGE
        1029 (OR 1725,1473) INST. CYCLES MAXIMUM
        EXTENDABLE ROUTINE FOR MY16XX BY CHANGING
          PARAMETERS, WITH NUMBER OF BYTES (28)
          REMAINING A CONSTANT
        MULTIPLICAND IN [1,0]                (ICAND)
        MULTIPLIER IN [3,2] FOR 16 BIT       (IER)
                   OR [4,3,2] FOR 24 BIT
                   OR [5,4,3,2] FOR 32 BIT
        PRODUCT IN [5,4,3,2] FOR 16 BIT      (PROD)
                OR [6,5,4,3,2] FOR 24 BIT
                OR [7,6,5,4,3,2] FOR 32 BIT

MY1616: LD          CNTR,#17     ; LD CNTR WITH LENGTH OF
                                 ;    MULTIPLIER FIELD + 1
                                 ; #17 FOR MY1616
                                 ; (#25 FOR MY1624)
                                 ; (#33 FOR MY1632)
        LD          B,#5         ; #5 FOR MY1616
                                 ; (#6 FOR MY1624)
                                 ; (#7 FOR MY1632)
        LD          [B-],#0      ; CLEAR UPPER TWO
        LD          [B-],#0      ;    PRODUCT BYTES
        RC
MY16XS: LD          A,[B]        ; FIVE INSTRUCTION
        RRC         A            ;    PROGRAM LOOP TO
        X           A,[B-]       ;    RIGHT SHIFT
        IFBNE       #1           ;    PRODUCT/MULTIPLIER
        JP          M16XS        ;    LOOP JUMP BACK
        IFNC                     ; TEST LOW ORDER
        JP          MY16XT       ;    MULTIPLIER BIT
        RC
        LD          B,#4         ; #4 FOR MY1616
                                 ; (#5 FOR MY1624)
                                 ; (#6 FOR MY1632)
        LD          X,#0         ; LOAD ACC WITH
MY16XL: LD          A,[X+]       ;    MULTIPLICAND BYTES
        ADC         A,[B]        ; ADD MULTIPLICAND TO
        X           A,[B+]       ;    HI TWO PROD. BYTES
        IFBNE       #2           ; LOOP BACK FOR SECOND
        JP          MY16XL       ;    MULTIPLICAND BYTE
MY16XT: LD          B,#5         ; #5 FOR MY1616
                                 ; (#6 FOR MY1624)
                                 ; (#7 FOR MY1632)
        DRSZ        CNTR         ; DECREMENT AND TEST
        JP          MY16XS       ;    CNTR EQUAL TO ZERO
        RET                      ; RETURN FROM INTERRUPT
;
```

MY2448—MINIMAL GENERAL MULTIPLICATION SUBROUTINE (28 BYTES)

```
   ANY NUMBER OF BYTES IN MULTIPLICAND
   AND MULTIPLIER
FIRST EXAMPLE:     (MY2448)
   24 BY 48 MULTIPLICATION SUBROUTINE
        --28 BYTES
        --MINIMAL CODE, MINIMAL RAM
        --4713 INSTRUCTION CYCLES AVERAGE
        --5457 INSTRUCTION CYCLES MAXIMUM
   MULTIPLICAND IN [2,1,0]                    (ICAND)
   MULTIPLIER IN [8,7,6,5,4,3]                (IER)
   PRODUCT IN [11,10,9,8,7,6,5,4,3]           (PROD)

SECOND EXAMPLE: (MY4824)
   48 BY 24 MULTIPLICATION SUBROUTINE
        --28 BYTES
        --MINIMAL CODE, NON MINIMAL RAM
        --2751 INSTRUCTION CYCLES AVERAGE
        --3483 INSTRUCTION CYCLES MAXIMUM
   MULTIPLICAND IN [5,4,3,2,1,0]              (ICAND)
   MULTIPLIER IN [8,7,6]                      (IER)
   PRODUCT IN [14,13,12,11,10,9,8,7,6]        (PROD)



MY2448:   ; (OR MY4824)
          LD          CNTR, #49   ; LD CNTR WITH LENGTH OF
                                  ;    MULTIPLIER FIELD + 1
                                  ; #49 FOR MY2448
                                  ; (#25 FOR MY4824)
          LD          B,#11       ; TOP OF PROD TO B PTR
                                  ; #11 FOR MY2448
                                  ; (#14 FOR MY4824)
CLRLUP:   LD          [B-],#0     ; CLR UNTIL TOP OF IER
          IFBNE       #8          ; #8 FOR BOTH MY2448
          JP          CLRLUP      ;    AND MY4824
          RC                      ; INITIALIZE CARRY
SHFTLP:   LD          A,[B]       ; RIGHT SHIFT PRODUCT
          ADC         A,[B]       ;    AND MULTIPLIER
          X           A,[B-]      ;    UNTIL TOP OF ICAND
          IFBNE       #2          ; #2 FOR MY2448
          JP          SHFTLP      ; (#5 FOR MY4824)
          IFNC                    ; TEST LOW ORDER
          JP          MYTEST      ;    MULTIPLIER BIT
          LD          B,#9        ; TOP OF IER + 1 TO B PTR
          LD          X,#0        ; START OF ICAND TO X PTR
          RC
ADDLUP:   LD          A,[X+]      ; ADD MULTIPLICAND TO TOP
          ADC         A,[B]       ;    OF PRODUCT ABOVE
          X           A,[B+]      ;    MULTIPLIER UNTIL TOP
          IFBNE       #12         ;    OF PRODUCT + 1
          JP          ADDLUP      ; #12 FOR MY2448
                                  ; (#15 FOR MY4824)
MYTEST:   LD          B,#11       ; TOP OF PROD TO B PTR
                                  ; #11 FOR MY2448
                                  ; (#14 FOR MY4824)
          DRSZ        CNTR        ; DECREMENT AND TEST
          JP          SHFTLP      ;    CNTR FOR ZERO
          RET                     ; RETURN FROM SUBROUTINE
```

3.0 DIVISION

The COP 800 divisions are all based on shifting the dividend left up into a test field equal in length to the number of bytes in the divisor. The divisor is resident immediately above this test field. After each shift cycle of the dividend into the test field, a trial subtraction is made of the test field minus the divisor. If the divisor is found equal to or less than the contents of the test field, then the divisor is subtracted from the test field and a 1's quotient digit is recorded by setting the low order bit of the dividend field. The dividend and test field left shift cycle is then repeated. The number of left shift cycles is equal to the number of bit positions in the dividend. The quotient from the division is formed in the dividend field, while the remainder from the division is resident in the test field.

Note that an M byte dividend divided by an N byte divisor will result in an M byte quotient and an N byte remainder.

These division algorithms will use M + 2N + 1 bytes of RAM memory space, since the test field is equal to the length of the divisor. The one extra byte is necessary for the shift counter CNTR.

In special cases where the dividend has an upper bound and the divisor has a lower bound, the upper bytes of the dividend may be used as the test field. One example is shown (DV2815), where a 28 bit dividend is divided by a 15-bit divisor. The dividend is less than 2**28 (upper nibble of high order byte is zero), while the divisor is greater than 2**12 (4096) and less than 2**15 (32768). In this case, the upper limit for the quotient is 2**28/2**12, which indicates a 16-bit quotient (2**16) and a 15-bit remainder. Consequently, the upper two bytes of the dividend may be used as the test field for the remainder, since the divisor is greater than the test field (upper two bytes of the 28-bit dividend) initially.

The minimal code (40 byte) general division subroutine is shown with the example DV3224, which divides a 32 bit dividend by a 24 bit divisor.

DIV88 — 8 by 8 Division Subroutine
- — 24 Bytes
- — 201 Instruction Cycles Average
- — 209 Instruction Cycles Maximum
- Minimum code

DV88 — Fast 8 by 8 Division Subroutine
- — 28 Bytes
- — 194 Instruction Cycles Average
- — 202 Instruction Cycles Maximum

FDV88 — Very Fast 8 by 8 Division Subroutine
- — 131 Bytes
- — 146 Instruction Cycles Average
- — 159 Instruction Cycles Maximum

DIV168 (or DIV248, DIV328)
- — 16 (or 24, 32) by 8 Division Subroutine
- — 26 Bytes
- — 649 (or 1161, 1801) Instruction Cycles Average
- — 681 (or 1209,1865) Instruction Cycles Maximum
- — Minimum Code
- — Extendable Routine for DIVXX8 by Changing Parameters, with Number of Bytes (26) Remaining a Constant

FDV168 — Fast 16 by 8 Division Subroutine
- — 35 Bytes
- — 481 Instruction Cycles Average
- — 490 Instruction Cycles Maximum

FDV248 — Fast 24 by 8 Division Subroutine
- — 38 Bytes
- — 813 Instruction Cycles Average
- — 826 Instruction Cycles Maximum

FDV328 — Fast 32 by 8 Division Subroutine
- — 42 Bytes
- — 1209 Instruction Cycles Average
- — 1226 Instructions Maximum

Divide by 16 Subroutines:

DV1616 — 16 by 16 Division Subroutine
- — 34 Bytes
- — 979 Instruction Cycles Average
- — 1067 Instruction Cycles Maximum
- — Minimum Code

DV2416 (or DV3216)
- — 24 (or 32) by 16 Division Subroutine
- — 39 Bytes
- — 1694 (or 2410) Inst. Cycles Average
- — 1886 (or 2766) Inst. Cycles Maximum
- — Minimum code
- — Extendable Routine for DVXX16 by Changing Parameters, with Number of Bytes (39) Remaining a Constant

DX1616 — Fast 16 by 16 Division Subroutine
- — 53 Bytes
- — 638 Instruction Cycles Average
- — 678 Instruction Cycles Maximum

DV2815 — Fast 28 by 15 Division Subroutine, Where the Dividend is Less Than 2**28 and the Divisor is Greater than 2**12 (4096) and Less than 2**15 (32768)
- — 43 Bytes
- — 640 Instruction Cycles Average
- — 696 Instruction Cycles Maximum

DX3216 — Fast 32 by 16 Division Subroutine
- — 70 Bytes
- — 1511 Instruction Cycles Average
- — 1591 Instruction Cycles Maximum

Minimal General Division Subroutine for any Number of Bytes in Dividend and Divisor
- — 40 Bytes
- — Minimal Code
- — DV3224 Used as Example, with 3879 Instruction Cycles Average 4535 Instruction Cycles Maximum

DIV88—8 BY 8 DIVISION SUBROUTINE

```
          MINIMUM CODE
          24 BYTES
          201 INSTRUCTION CYCLES AVERAGE
          209 INSTRUCTION CYCLES MAXIMUM
          DIVIDEND     IN [0]      (DD)
          DIVISOR IN [2]           (DR)
          QUOTIENT IN [0]          (QUOT)
          REMAINDER IN [1]         (TEST FIELD)

DIV88:    LD           CNTR,#8     ; LOAD CNTR WITH LENGTH
          LD           B,#1        ;    OF DIVIDEND FIELD
          LD           [B],#0      ; CLEAR TEST FIELD
DIV88S    RC
          LD           B,#0
          LD           A,[B]
          ADC          A,[B]       ; LEFT SHIFT DIVIDEND
          X            A,[B+]
          LD           A,[B]
          ADC          A,[B]       ; LEFT SHIFT TEST FIELD
          X            A,[B]
          LD           A,[B+]      ; TEST FIELD TO ACC
          SC                       ; TEST SUBTRACT DIVISOR
          SUBC         A,[B]       ;    FROM TEST FIELD
          IFNC                     ; TEST IF BORROW
          JP           DIV88B      ;    FROM SUBTRACTION
          LD           B,#1        ; SUBTRACTION RESULT
          X            A,[B-]      ;    TO TEST FIELD
          SBIT         0,[B]       ; SET QUOTIENT BIT
DIV88B:   DRSZ         CNTR        ; DECREMENT AND TEST
          JP           DIV88S      ;    CNTR FOR ZERO
          RET                      ; RETURN FROM SUBROUTINE
```

DV88—FAST 8 BY 8 DIVISION SUBROUTINE

```
          28 BYTES
          194 INSTRUCTION CYCLES AVERAGE
          202 INSTRUCTION CYCLES MAXIMUM
          DIVIDEND     IN [0]       (DD)
          DIVISOR IN [2]            (DR)
          QUOTIENT IN [0]           (QUOT)
          REMAINDER IN [1]          (TEST FIELD)
DV88:     LD           CNTR,#8      ;  LOAD CNTR WITH LENGTH
          LD           B,#1         ;   OF DIVIDEND FIELD
          LD           [B-],#0      ;  CLEAR TEST FIELD
          RC
DV88S:    LD           A,[B]
          ADC          A,[B]        ;  LEFT SHIFT DIVIDEND
          X            A,[B+]
          LD           A,[B]
          ADC          A,[B]        ;  LEFT SHIFT TEST FIELD
          X            A,[B]
          LD           A,[B+]       ;  TEST FIELD TO ACC
          SC                        ;  TEST SUBTRACT DIVISOR
          SUBC         A,[B]        ;     FROM TEST FIELD
          IFNC                      ;  TEST IF BORROW
          JP           DV88B        ;     FROM SUBTRACTION
          LD           B,#1         ;  SUBTRACTION RESULT
          X            A,[B-]       ;  TO TEST FIELD
          SBIT         0,[B]        ;  SET QUOTIENT BIT
          RC
          DRSZ         CNTR         ;  DECREMENT AND TEST
          JP           DV88S        ;     CNTR FOR ZERO
          RET                       ;  RETURN FROM SUBROUTINE
DV88B:    LD           B,#0
          DRSZ         CNTR         ;  DECREMENT AND TEST
          JP           DV88S        ;     CNTR FOR ZERO
          RET                       ;  RETURN FROM SUBROUTINE
```

FDV88—VERY FAST 8 BY 8 DIVISION SUBROUTINE

```
          131 BYTES
          146 INSTRUCTION CYCLES AVERAGE
          159 INSTRUCTION CYCLES MAXIMUM
          DIVIDEND IN [0]           (DD)
          DIVISOR IN [2]            (DR)
          QUOTIENT IN [0]           (QUOT)
          REMAINDER IN [1]          (TEST FIELD)
FDV88:    LD          B,#1
          LD          [B-],#0       ;  CLEAR TEST FIELD
          RC
          LD          A,[B]
          ADC         A,[B]         ;  LEFT SHIFT DIVIDEND
          X           A,[B+]
          LD          A,[B]
          ADC         A,[B]         ;  LEFT SHIFT TEST FIELD
          X           A,[B]
          LD          A,[B+]        ;  TEST FIELD TO ACC
          SC                        ;  TEST SUBTRACT DIVISOR
          SUBC        A,[B]         ;     FROM TEST FIELD
          IFNC                      ;  TEST IF BORROW
          JP          DVBP1         ;     FROM SUBTRACTION
          LD          B,#1          ;  SUBTRACTION RESULT
          X           A,[B-]        ;     TO TEST FIELD
          SBIT        0,[B]         ;  SET QUOTIENT BIT
          RC
DVBP1:    LD          B,#0          ;  THIS 16 BYTE SECTION
          LD          A,[B]         ;  OF PROGRAM CODE
          ADC         A,[B]         ;  CONTAINS
          X           A,[B+]        ;  16 INSTRUCTIONS,
          LD          A,[B]         ;  AND REPRESENTS THE
          ADC         A,[B]         ;  PROCESSING FOR THE
          X           A,[B]         ;  GENERATION OF
          LD          A,[B+]        ;  1 QUOTIENT BIT.
          SC                        ;
          SUBC        A,[B]         ;  THE PROGRAM CODE
          IFNC                      ;  EXECUTION TIMES IS 16
          JP          DVBP2         ;  INSTRUCTION CYCLES
          LD          B,#1          ;  FOR A 0'S QUOTIENT BIT
          X           A,[B-]        ;  AND 19 INSTRUCTION
          SBIT        0,[B]         ;  CYCLES FOR A 1'S
          RC                        ;  QUOTIENT BIT.
;         ---                       ;
DVBP2:    LD          B,#0          ;  REPEAT THE ABOVE
;         ---                       ;
;DVBP3:
;         ---                       ;SECTION OF CODE FIVE
;DVBP4:
;         ---                       ;MORE TIMES FOR A
;DVBP5:
;         ---                       ;TOTAL OF SIX TIMES
;DVBP6:
;         ---                       ;
;
DVBP7:    LD          B,#0
          LD          A,[B]
          ADC         A,[B]         ;  LEFT SHIFT DIVIDEND
          X           A,[B+]
          LD          A,[B]
          ADC         A,[B]         ;  LEFT SHIFT TEST FIELD
          X           A,[B]
          LD          A,[B+]        ;  TEST FIELD TO ACC
          SC                        ;  TEST SUBTRACT DIVISOR
          SUBC        A,[B]         ;     FROM TEST FIELD
          IFNC                      ;  TEST BORROW FROM SUBC
          RET                       ;  RETURN FROM SUBROUTINE
          LD          B,#1          ;  SUBTRACTION RESULT
          X           A,[B-]        ;     TO TEST FIELD
          SBIT        0,[B]         ;  SET QUOTIENT BIT
          RET                       ;  RETURN FROM SUBROUTINE
```

DIV168—16 (OR 24, 32) BY 8 DIVISION SUBROUTINE

```
          MINIMUM CODE
          26 BYTES
          649 (or 1161,1801) INST. CYCLES AVERAGE
          681 (or 1209,1865) INST. CYCLES MAXIMUM
          EXTENDABLE ROUTINE FOR DIVXX8 BY CHANGING
          PARAMETERS, WITH NUMBER OF BYTES (26)
          REMAINING A CONSTANT
          DIVIDEND IN [1,0] FOR 16 BIT              (DD)
                   OR [2,1,0] FOR 24 BIT
                   OR [3,2,1,0] FOR 32 BIT
          DIVISOR IN [3] FOR 16 BIT                 (DR)
                   OR [4] FOR 24 BIT
                   OR [5] FOR 32 BIT
          QUOTIENT IN [1,0] FOR 16 BIT              (QUOT)
                   OR [2,1,0] FOR 24 BIT
                   OR [3,2,1,0] FOR 32 BIT
          REMAINDER IN [2] FOR 16 BIT               (TEST FIELD)
                   OR [3] FOR 24 BIT
                   OR [4] FOR 32 BIT


DIV168:   LD       CNTR,#16      ; LOAD CNTR WITH LENGTH
                                 ;    OF DIVIDEND FIELD
                                 ; #16 FOR DIV168
                                 ; (#24 FOR DIV248)
                                 ; (#32 FOR DIV328)
          LD       B,#2          ; (#3 FOR DIV168)
                                 ; (#3 FOR DIV248)
                                 ; (#4 FOR DIV328)
          LD       [B],#0        ; CLEAR TEST FIELD
DVXX8L:   RC
          LD       B,#0
DXX8LP:   LD       A,[B]         ; LEFT SHIFT DIVIDEND
          ADC      A,[B]         ;    AND TEST FIELD
          X        A,[B+]
          IFBNE    #3            ; #3 FOR DIV168
          JP       DXX8LP        ; (#4 FOR DIV248)
                                 ; (#5 FOR DIV328)
          LD       A,[B-]        ; DIVISOR TO ACCUMULATOR
          IFC                    ; TEST IF BIT SHIFTED OUT
          JP       DVXX8S        ;    OF TEST FIELD***
          IFGT     A,[B]         ; TEST DIVISOR GREATER
          JP       DVXX8T        ;    THAN REMAINDER
          SC                     ;
DVXX8S:   X        A,[B]         ; REMAINDER TO ACC
          SUBC     A,[B]         ; SUBTRACT DIVISOR
          X        A,[B]         ;    FROM REMAINDER
          LD       B,#0
          SBIT     0,[B]         ; SET QUOTIENT BIT
DVXX8T:   DRSZ     CNTR          ; DECREMENT AND TEST
          JP       DVXX8L        ;    CNTR FOR ZERO
          RET                    ; RETURN FROM SUBROUTINE


;
;
;   ***   SPECIAL CASE FOR DIVISION WHERE NUMBER OF BYTES
;         IN DIVIDEND IS GREATER THAN NUMBER OF BYTES IN DIVISOR, AND
;         DIVISOR CONTAINS A HIGH ORDER 1'S BIT. THE SHIFTED DIVIDEND
;         MAY CONTAIN A HIGH ORDER 1'S BIT IN THE TEST FIELD AND
;         YET BE SMALLER THAN THE DIVISOR SO THAT NO SUBTRACTION
;         OCCURS. iN THIS CASE A 1'S BIT WILL BE SHIFTED OUT OF
;         THE TEST FIELD AND AN OVERRIDE SUBTRACTION MUST BE PERFORMED
```

FDV168—FAST 16 BY 8 DIVISION SUBROUTINE

```
          35 BYTES
          481 INSTRUCTION CYCLES AVERAGE
          490 INSTRUCTION CYCLES MAXIMUM
          DIVIDEND IN [1,0]          (DD)
          DIVISOR IN [3]             (DR)
          QUOTIENT IN [1,0]          (QUOT)
          REMAINDER IN [2]           (TEST FIELD)

FDV168:   LD         CNTR,#16        ;  LOAD CNTR WITH LENGTH
          LD         B,#3            ;     OF DIVIDEND FIELD
          LD         [B],#0          ;  CLEAR TEST FIELD
FD168S:   LD         B,#0
FD168L:   RC
          LD         A,[B]
          ADC        A,[B]           ;  LEFT SHIFT DIVIDEND LO
          X          A,[B+]
          LD         A,[B]
          ADC        A,[B]           ;  LEFT SHIFT DIVIDEND HI
          X          A,[B+]
          LD         A,[B]
          ADC        A,[B]           ;  LEFT SHIFT TEST FIELD
          X          A,[B]
          LD         A,[B+]          ;  TEST FIELD TO ACC
          IFC                        ;  TEST IF BIT SHIFTED OUT
          JP         FD168B          ;     OF TEST FIELD***
          SC                         ;  TEST SUBTRACT DIVISOR
          SUBC       A,[B]           ;     FROM TEST FIELD
          IFNC                       ;  TEST IF BORROW
          JP         FD168T          ;     FROM SUBTRACTION
FD168R:   LD         B,#2            ;  SUBTRACTION RESULT
          X          A,[B]           ;     TO TEST FIELD
          LD         B,#0
          SBIT       0,[B]           ;  SET QUOTIENT BIT
          DRSZ       CNTR            ;  DECREMENT AND TEST
          JP         FD168L          ;     CNTR FOR ZERO
          RET                        ;  RETURN FROM SUBROUTINE
FD168T:   DRSZ       CNTR            ;  DECREMENT AND TEST
          JP         FD168S          ;     CNTR FOR ZERO
          RET                        ;  RETURN FROM SUBROUTINE
FD168B:   SUBC       A,[B]           ;  SUBTRACT DIVISOR FROM
          JP         FD168R          ;     TEST FIELD***
```

FDV248—FAST 24 BY 8 DIVISION SUBROUTINE

```
          38 BYTES
          813 INSTRUCTION CYCLES AVERAGE
          826 INSTRUCTION CYCLES MAXIMUM
          DIVIDEND IN [2,1,0]       (DD)
          DIVISOR IN [4]            (DR)
          QUOTIENT IN [2,1,0]       (QUOT)
          REMAINDER IN [3]          (TEST FIELD)

FDV248:   LD          CNTR,#24      ; LOAD CNTR WITH LENGTH
          LD          B,#4          ;    OF DIVIDEND FIELD
          LD          [B],#0        ; CLEAR TEST FIELD
FD248S:   LD          B,#0
FD248L:   RC
          LD          A,[B]
          ADC         A,[B]         ; LEFT SHIFT DIVIDEND LO
          X           A,[B+]
          LD          A,[B]
          ADC         A,[B]         ; LEFT SHIFT DIVIDEND MID
          X           A,[B+]
          LD          A,[B]
          ADC         A,[B]         ; LEFT SHIFT DIVIDEND HI
          X           A,[B+]
          LD          A,[B]
          ADC         A,[B]         ; LEFT SHIFT TEST FIELD
          X           A,[B]
          LD          A,[B+]
          IFC                       ; TEST IF BIT SHIFTED OUT
          JP          FD248B        ;    OF TEST FIELD ***
          SC                        ; TEST SUBTRACT DIVISOR
          SUBC        A,[B]         ;    FROM TEST FIELD
          IFNC                      ; TEST IF BORROW
          JP          FD248T        ;    FROM SUBTRACTION
FD248R:   LD          B,#3          ; SUBTRACTION RESULT
          X           A,[B]         ;    TO TEST FIELD
          LD          B,#0
          SBIT        0,[B]         ; SET QUOTIENT BIT
          DRSZ        CNTR          ; DECREMENT AND TEST
          JP          FD248L        ;    CNTR FOR ZERO
          RET                       ; RETURN FROM SUBROUTINE
FD248T:   DRSZ        CNTR          ; DECREMENT AND TEST
          JP          FD248S        ;    CNTR FOR ZERO
          RET                       ; RETURN FROM SUBROUTINE
FD248B:   SUBC        A,[B]         ; SUBTRACT DIVISOR FROM
          JP          FD248R        ;    TEST FIELD ***
```

DV1616—16 (OR 24, 32) BY 16 DIVISION SUBROUTINE

```
          MINIMUM CODE
          34 BYTES
          979 (OR 1655,2459) INSTRUCTION CYCLES AVERAGE
          1067 (OR 1787,2635) INSTRUCTION CYCLES MAXIMUM
          DIVIDEND IN [1,0]          (DD)
          DIVISOR IN [5,4]           (DR)
          QUOTIENT IN [1,0]          (QUOT)
          REMAINDER IN [3,2]         (TEST FIELD)

DV1616:   LD          CNTR,#16       ; LOAD CNTR WITH LENGTH
                                     ;    OF DIVIDEND FIELD
          LD          B,#3
          LD          [B-],#0        ; CLEAR
          LD          [B],#0         ;    TEST FIELD
DV616S:   RC
          LD          X,#2           ; INITIALIZE X POINTER
          LD          B,#0           ; INITIALIZE B POINTER
DV616L:   LD          A,[B]          ; LEFT SHIFT DIVIDEND
          ADC         A,[B]          ;    AND TEST FIELD
          X           A,[B+]
          IFBNE       #4
          JP          DV616L
          SC                         ; RESET BORROW
          LD          A,[X+]         ; TEST FIELD LO TO ACC
          SUBC        A,[B]          ; SUBT DR LO FROM REM LO
          LD          A,[X]          ; TEST FIELD HI TO ACC
          LD          B,#5
          SUBC        A,[B]          ; SUBT DR HI FROM REM HI
          IFNC                       ; TEST IF BORROW
          JP          DV616T         ;    FROM SUBTRACTION
          X           A,[X-]         ; SUBT RESULT HI TO REM HI
          LD          A,[X]          ; TEST FIELD LO TO ACC
          LD          B,#4
          SUBC        A,[B]          ; SUBT DR LO FROM REM LO
          X           A,[X]          ; RESULT LO TO REM LO
          LD          B,#0
          SBIT        0,[B]          ; SET QUOTIENT BIT
DV616T:   DRSZ        CNTR           ; DECREMENT AND TEST
          JP          DV616S         ;    CNTR FOR ZERO
          RET                        ; RETURN FROM SUBROUTINE
```

DX1616—FAST 16 BY 16 DIVISION SUBROUTINE

```
          53 BYTES
          638 INSTRUCTION CYCLES AVERAGE
          678 INSTRUCTION CYCLES MAXIMUM
          DIVIDEND IN [1,0]          (DD)
          DIVISOR IN [5,4]           (DR)
          QUOTIENT IN [1,0]          (QUOT)
          REMAINDER IN [3,2]         (TEST FIELD)


DX1616:   LD          CNTR,#16     ;  LOAD CNTR WITH LENGTH
          LD          B,#5         ;     OF DIVIDEND FIELD
          LD          A,[B]        ;  REPLACE DIVISOR WITH
          XOR         A,#0FF       ;     1'S COMPLEMENT OF
          X           A,[B-]       ;     DIVISOR TO ALLOW
          LD          A,[B]        ;     OPTIONAL ADDITION OF
          XOR         A,#0FF       ;     DIVISOR'S COMPLEMENT
          X           A,[B-]       ;     IN MAIN PROG. LOOP
          LD          [B-],#0      ;  CLEAR
          LD          [B],#0       ;     TEST FIELD
DX616S:   LD          B,#0
DX616L:   RC
          LD          A,[B]
          ADC         A,[B]        ;  LEFT SHIFT DIVIDEND LO
          X           A,[B+]
          LD          A,[B]
          ADC         A,[B]        ;  LEFT SHIFT DIVIDEND HI
          X           A,[B+]
          LD          A,[B]
          ADC         A,[B]        ;  LEFT SHIFT TEST FIELD LO
          X           A,[B+]
          LD          A,[B]
          ADC         A,[B]        ;  LEFT SHIFT TEST FIELD HI
          X           A,[B+]
          SC
          LD          A,[B]        ;  DIVISORX (DRX) LO TO ACC
          LD          B,#2         ;     (1'S COMPLEMENT)
          ADC         A,[B]        ;  ADD REM LO TO DRX LO
          LD          B,#5
          LD          A,[B]        ;  DIVISORX (DRX) HI TO ACC
          LD          B,#3         ;     (1'S COMPLEMENT)
          ADC         A,[B]        ;  ADD REM HI TO DRX HI
          IFNC                     ;  TEST IF NO CARRY FROM
          JP          DX616T       ;     1'S COMPL.ADDITION
          X           A,[B+]       ;  RESULT TO REM HI
          LD          A,[B]        ;  DRX LO TO ACCUMULATOR
          LD          B,#2
          ADC         A,[B]        ;  ADD REM LO TO DRX LO
          X           A,[B]        ;  RESULT TO REM LO
          LD          B,#0
          SBIT        0,[B]        ;  SET QUOTIENT BIT
          DRSZ        CNTR         ;  DECREMENT AND TEST
          JP          DX616L       ;     CNTR FOR ZERO
          RET                      ;  RETURN FROM SUBROUTINE
DX616T:   DRSZ        CNTR         ;  DECREMENT AND TEST
          JMP         DX616S       ;     CNTR FOR ZERO
          RET                      ;  RETURN FROM SUBROUTINE
```

DV2815—FAST 28 BY 15 DIVISION SUBROUTINE

```
          WHERE THE DIVIDEND IS LESS THAN 2**28
          AND THE DIVISOR IS GREATER THAN 2**12 (4096) AND LESS THAN 2**15 (32768)
          43 BYTES
          640 INSTRUCTION CYCLES AVERAGE
          696 INSTRUCTION CYCLES MAXIMUM
          DIVIDEND IN [3,2,1,0]       (DD)
          DIVISOR IN [5,4]            (DR)
          QUOTIENT IN [1,0]           (QUOT)
          REMAINDER IN [3,2]          (TEST FIELD)

DV2815:   LD          CNTR,#16      ; LOAD CNTR WITH LENGTH OF QUOTIENT FIELD
D2815S:   LD          B,#0
D2815L:   RC
          LD          A,[B]
          ADC         A,[B]         ; LEFT SHIFT LOWER
          X           A,[B+]        ;    BYTE OF DIVIDEND
          LD          A,[B]
          ADC         A,[B]         ; LEFT SHIFT NEXT HIGHER
          X           A,[B+]        ;    BYTE OF DIVIDEND
          LD          A,[B]
          ADC         A,[B]         ; LEFT SHIFT NEXT HIGHER
          X           A,[B+]        ;    BYTE OF DIVIDEND
          LD          A,[B]
          ADC         A,[B]         ; LEFT SHIFT UPPER
          X           A,[B-]        ;    BYTE OF DIVIDEND
          ***
          NOTE THAT WITH A 16 BIT DIVISOR (DIV 2816) SUBROUTINE, A TEST FOR A HIGH
          ORDER BIT SHIFTED OUT OF THE TEST FIELD WOULD BE NECESSARY AT THIS POINT.
          IFC
          JP          SUBTRMD       ; SUBTRACT REM MINUS DR
          THE PRESENCE OF THIS CARRY WOULD REQUIRE THAT THE DIVISOR BE SUBTRACTED
          FROM THE REMAINDER AS SHOWN WITH THE DIV168*** SUBROUTINE.
          LD          A,[B]         ; REM LOWER BYTE TO ACC
          SC                        ; TEST SUBTRACT LOWER
          LD          B,#4          ;    BYTE OF DR FROM
          SUBC        A,[B]         ;    LOWER BYTE OF REM
          LD          B,#3          ; TEST SUBTRACT UPPER
          LD          A,[B]         ;    BYTE OF DIVISOR
          LD          B,#5          ;    FROM UPPER BYTE
          SUBC        A,[B]         ;    OF REMAINDER
          IFNC                      ; TEST IF BORROW
          JP          D2815T        ;    FROM SUBTRACTION
          LD          B,#3          ; UPPER BYTE OF RESULT
          X           A,[B+]        ;    TO UPPER BYTE OF REM
          LD          A,[B]         ; DR LOWER BYTE TO ACC
          LD          B,#2          ; SUBTRACT LOWER BYTE
          X           A,[B]         ;    OF DIVISOR FROM
          SUBC        A,[B]         ;    LOWER BYTE OF
          X           A,[B]         ;    REMAINDER
          LD          B,#0
          SBIT        0,[B]         ; SET QUOTIENT BIT
          DRSZ        CNTR          ; DECREMENT AND TEST
          JMP         D2815L        ;    CNTR FOR ZERO
          RET                       ; RETURN FROM SUBROUTINE
D2815T:   DRSZ        CNTR          ; DECREMENT AND TEST
          JMP         D2815S        ;    CNTR FOR ZERO
          RET                       ; RETURN FROM SUBROUTINE
```

DX3216—FAST 32 BY 16 DIVISION SUBROUTINE

```
          70 BYTES
          1510 INSTRUCTION CYCLES AVERAGE
          1590 INSTRUCTION CYCLES MAXIMUM
          DIVIDEND IN [3,2,1,0]        (DD)
          DIVISOR IN [7,6]             (DR)
          QUOTIENT IN [3,2,1,0]        (QUOT)
          REMAINDER IN [5,4]           (TEST FIELD)

DX3216:   LD           CNTR,#32        ; LOAD CNTR WITH LENGTH
          LD           B,#7            ;    OF DIVIDEND FIELD
          LD           A,[B]           ; REPLACE DIVISOR WITH
          XOR          A,#0FF          ;    1'S COMPLEMENT OF
          X            A,[B-]          ;    DIVISOR TO ALLOW
          LD           A,[B]           ;    OPTIONAL ADDITION OF
          XOR          A,#0FF          ;    DIVISOR'S COMPLEMENT
          X            A,[B-]          ; IN MAIN PROG. LOOP
          LD           [B-],#0         ; CLEAR
          LD           [B],#0          ;   TEST FIELD
DX326S:   LD           B,#0
DX326L:   RC
          LD           A,[B]
          ADC          A,[B]           ; LEFT SHIFT DIVIDEND LO
          X            A,[B+]
          LD           A,[B]
          ADC          A,[B]           ; LEFT SHIFT NEXT HIGHER
          X            A,[B+]          ;    DIVIDEND BYTE
          LD           A,[B]
          ADC          A,[B+]          ; LEFT SHIFT NEXT HIGHER
          X            A,[B+]          ;    DIVIDEND BYTE
          LD           A,[B]
          ADC          A,[B]           ; LEFT SHIFT DIVIDEND HI
          X            A,[B+]
          LD           A,[B]
          ADC          A,[B]           ; LEFT SHIFT TST FIELD LO
          X            A,[B+]
          LD           A,[B]
          ADC          A,[B]           ; LEFT SHIFT TST FIELD HI
          X            A,[B+]
          IFC                          ; **TEST IF BIT SHIFTED
          JP           DX326B          ; ** OUT OF TEST FIELD
          SC
          LD           A,[B]           ; DVSORX (DRX) LO TO ACC
          LD           B,#4            ;    (1'S COMPLEMENT)
          ADC          A,[B]           ; ADD REM LO TO DRX LO
          LD           B,#7
          LD           A,[B]           ; DVSORX (DRX) HI TO ACC
          LD           B,#5            ;    (1'S COMPLEMENT)
          ADC          A,[B]           ; ADD REM HI TO DRX HI
          IFNC                         ; TEST IF NO CARRY FROM
          JP           DX326T          ;    1'S COMPL. ADDITION
          X            A,[B+]          ; RESULT TO REM NI
          LD           A,[B]           ; DRX LO TO ACCUMULATOR
          LD           B,#4
DX326R:   ADC          A,[B]           ; ADD REM LO TO DRX LO
                                       ; ** ADD REM HI TO DRX HI
          X            A,[B]           ; RESULT TO REM LO
                                       ; ** RESULT TO REM HI
LD                     B,#0
          SBIT         0,[B]           ; SET QUOTIENT BIT
          DRSZ         CNTR            ; DECREMENT AND TEST
          JMP          DX326L          ;    CNTR FOR ZERO
          RET                          ; RETURN FROM SUBROUTINE
DX326T:   DRSZ         CNTR            ; DECREMENT AND TEST
          JMP          DX326S          ;    CNTR FOR ZERO
          RET                          ; RETURN FROM SUBROUTINE
DX326B:   LD           A,[B]           ; ** REM LO TO ACC
          LD           B,#6            ; ** B PTR TO DRX LO
          ADC          A,[B]           ; ** ADD DRX LO TO REM LO
          X            A,[B]           ; ** RESULT TO REM LO
          LD           B,#7            ; **
          LD           A,[B]           ; ** DRX HI TO ACC
          LD           B,#5            ; ** B PTR TO REM HI
          JP           DX36R           ; **
   **     THESE INSTRUCTIONS UNNECESSARY IF DIVISOR
          LESS THAN 2**15 (DX3215 SUBROUTINE)
```

MINIMAL GENERAL DIVISION SUBROUTINE (40 BYTES)

```
          ANY NUMBER OF BYTES IN DIVIDEND AND DIVISOR
          DV3224 SERVES AS EXAMPLE
          32 BY 24 DIVISION SUBROUTINE
                       --40 BYTES
                       --MINIMAL CODE
                       --3879 INSTRUCTION CYCLES AVERAGE
                       --4535 INSTRUCTION CYCLES MAXIMUM


          DIVIDEND IN [3,2,1,0]        (DD)
          DIVISOR IN [9,8,7]           (DR)
          QUOTIENT IN [3,2,1,0]        (QUOT)
          REMAINDER IN [6,5,4]         (TEST FIELD)

DV3224:   LD          CNTR,#32        ; LOAD CNTR WITH LENGTH
          LD          B,#6            ;    OF DIVIDEND FIELD
CLRLUP:   LD          [B-],#0         ; CLEAR TEST FIELD
          IFBNE       #3              ; TOP OF DIVIDEND FIELD
          JP          CLRLUP
DVSHFT:   RC
          LD          B,#0
SHFTLP:   LD          A,[B]
          ADC         A,[B]           ; LEFT SHIFT DIVIDEND
          X           A,[B+]          ;    AND TEST FIELD
          IFBNE       #7              ; BOTTOM OF DR FIELD
          JP          SHFTLP
          IFC                         ; TEST IF BIT SHIFTED
          JP          DVSUBT          ; *** OUT OF TEST FIELD
          SC                          ; RESET BORROW
          LD          X,#4
TSTLUP:   LD          A,[X+]          ; TEST SUBTRACT DIVISOR
          SUBC        A,[B]           ;    FROM TEST FIELD
          LD          A,[B+]          ; INCREMENT B POINTER
          IFBNE       #10             ; TOP OF DIVISOR + 1
          JP          TSTLUP
          IFNC                        ; TEST IF BORROW
          JP          DVTEST          ;    FROM SUBTRACTION
          LD          B,#7
DVSUBT:   LD          X,#4
SUBTLP:   LD          A,[X]           ; SUBTRACT DIVISOR
          SUBC        A,[B]           ;    FROM REMAINDER
          X           A,[X+]          ;    IN TEST FIELD
          LD          A,[B+]          ; INCREMENT B POINTER
          IFBNE       #10             ; TOP OF DIVISOR + 1
          JP          SUBTLP
          LD          B,#0
          SBIT        0,[B]           ; SET QUOTIENT BIT
DVTEST:   DRSZ        CNTR            ; DECREMENT AND TEST
          JP          DVSHFT          ;    CNTR FOR ZERO
          RET                         ; RETURN FROM SUBROUTINE
```

4.0 DECIMAL (PACKED BCD)/BINARY CONVERSION

Subroutines For Two Byte Conversion:

DECBIN — Decimal (Packed BCD) to Binary
— 24 Bytes ***
— 1030 Instruction Cycles

FDTOB — Fast Decimal (Packaged BCD) to Binary
— 76 Bytes
— 92 Instruction Cycles

BINDEC — Binary to Decimal (Packed BCD)
— 25 Bytes ***
— 856 Instruction Cycles

FBTOD — Fast Binary to Decimal (Packed BCD)
— 59 Bytes
— 334 Instruction Cycles

VFBTOD — Very Fast Binary to Decimal (Packed BCD)
— 189 Bytes
— 144 Instruction Cycles Average
— 208 Instruction Cycles Maximum

***These subroutines extendable to multiple byte conversion by simply changing parameters within subroutine as shown, with number of bytes in subroutine remaining constant.

DECBIN—Decimal (Packed BCD) to Binary

This 24 byte subroutine represents very minimal code for translating a packed BCD decimal number of any length to binary.

ALGORITHM:

The binary result is resident just below the packed BCD decimal number. During each cycle of the algorithm, the decimal operand and the binary result are shifted right one bit position, with the low order bit of the decimal operand shifting down into the high order bit position of the binary field. The residual decimal operand is then tested for a high order bit in each of its nibbles. A three is subtracted from each nibble in the BCD operand space that is found to contain a high order bit equal to one. (This process effectively right shifts the BCD operand one bit position, and then corrects the result to BCD format.) The entire cycle is then repeated, with the total number of cycles being equal to the number of bit positions in the decimal field.

16 Bit: Binary IN [1,0]
Packed BCD in [3, 2]

24 Bit: Binary in [2, 1, 0]
Packed BCD in [5, 4, 3]

32 Bit: Binary in [3, 2, 1, 0]
Packed BCD in [7, 6, 5, 4]

24 Bytes
1030 Instruction Cycles (16 Bit)

```
DECBIN:   LD        CNTR,#16      ;  LOAD CNTR WITH NUMBER
                                  ;     OF BIT POSITIONS
                                  ;     IN BCD FIELD
                                  ;  #16 FOR 16 BIT (2 BYTE)
                                  ;  #'S 24/32 FOR 24/32 BIT
DB1:      LD        B,#3          ;  #'S 5/7 FOR 24/32 BIT
          RC
DB2:      LD        A,[B]         ;  PROGRAM LOOP TO
          RRC       A             ;     RIGHT SHIFT
          X         A,[B-]        ;     DECIMAL (BCD) AND
          IFBNE     #0F           ;     BINARY FIELDS.
          JP        DB2           ;     LOOP JUMP BACK
          LD        B,#3          ;  #'S 5/7 FOR 24/32 BIT
          SC                      ;  SET CARRY FOR SUBTRACT
DB3:      LD        A,[B]         ;  TEST HIGH ORDER BITS
          IFBIT     7,[B]         ;     OF BCD NIBBLES, AND
          SUBC      A,#030        ;     SUBTRACT A THREE
          IFBIT     3,[B]         ;     FROM EACH NIBBLE IF
          SUBC      A,#3          ;     HIGH ORDER BIT OF
          X         A,[B-]        ;     NIBBLE IS A ONE
          IFBNE     #1            ;  #'S 2/3 FOR 24/32 BIT
          JP        DB3           ;  LOOP BACK FOR MORE BCD BYTES
          DRSZ      CNTR          ;  DECREMENT AND TEST IF
          JP        DB1           ;  CNTR EQUAL TO ZERO
          RET                     ;  RETURN FROM SUBROUTINE
```

FDTOB—FAST DECIMAL (PACKED BCD) TO BINARY

BCD Format: Four Nibbles — W, X, Y, Z, with W = Hi Order Nibble
*** [1] = 16W + X
*** [0] = 16Y + Z

Algorithm: Binary Result is equal to 100(10W + X) + (10Y + Z)
BCD IN [1, 0]***
Temp in [2]
Binary in [4, 3]
76 Bytes
92 Instruction Cycles

```
FDTOB:  RC
        LD          B,#1
        LD          A,[B+]          ;  16W + X
        AND         A,#0F0          ;  EXTRACT 16W
        RRC         A               ;  8W
        X           A,[B]           ;  8W TO TEMP
        RRC         A               ;  4W
        RRC         A               ;  2W
        ADD         A,[B]           ;  2W + 8W = 10W
        X           A,[B-]          ;  10W TO TEMP
        LD          A,[B+]          ;  16W + X
        AND         A,#0F           ;  EXTRACT X
        ADC         A,[B]           ;  10W + X
        X           A,[B]           ;  10W + X TO TEMP
        LD          A,[B]
        ADC         A,[B]           ;  2.(10W + X)
        X           A,[B]           ;  2.(10W + X) TO TEMP
        ADC         A,[B]           ;  3.(10W + X)
        LD          B,#3            ;     = 16P + Q
        X           A,[B+]          ;  16P + Q TO [3]
        CLR         A
        IFC
        LD          A,#010          ;  16C TO A (C = CARRY)
        X           A,[B-]          ;  16C TO [4]
        LD          A,[B]           ;  16P + Q
        SWAP        A               ;  16Q + P
        X           A,[B]           ;  16Q + P TO [3]
        LD          A,[B+]          ;  16Q + P
        AND         A,#0F           ;  EXTRACT P
        ADD         A,[B]           ;  16C + P
        X           A,[B-]          ;  16C + P TO [4]**
        LD          A,[B]           ;  16Q + P
        AND         A,#0F0          ;  EXTRACT 16Q
        X           A,[B-]          ;  16Q TO [3]**
        LD          A,[B+]          ;  2.(10W + X)
        ADC         A,[B]           ;  2.(10W + X) + 16Q
```

```
X           A,[B+]          ;  2 BYTE 2.(10W + X)
CLR         A,[B-]          ;     ADD: + 48.**(10W + X)
ADC         A,[B]           ;  16C + P + NU C
X           A,[B-]          ;  50.(10W + X)
LD          A,[B]
ADC         A,[B]           ;  DOUBLE
X           A,[B+]          ;     50.(10W + X)
LD          A,[B]           ;     TO FORM
ADC         A,[B]           ;     100.(10W + X)
X           A,[B]           ;     IN [3,4]
LD          B,#0
LD          A,[B]           ;  16Y + Z
AND         A,#0F0          ;  EXTRACT 16Y
LD          B,#2
RRC         A               ;  8Y
X           A,[B]           ;  8Y TO TEMP
LD          A,[B]
RRC         A               ;  4Y
RRC         A               ;  2Y
ADC         A,[B]           ;  2Y + 8Y = 10Y
X           A,[B]           ;  10Y TO TEMP
LD          B,#0
LD          A,[B]           ;  16Y + Z
AND         A,#0F           ;  EXTRACT Z
LD          B,#2
ADD         A,[B]           ;  10Y + Z
LD          B, #3
ADC         A,[B]           ;  TWO BYTE ADD
X           A,[B+]          ;     100.(10W + X)
CLR         A               ;     + (10Y + Z)
ADC         A,[B]           ;     WITH BINARY
X           A,[B]           ;     RESULT TO [3,4]
RET
```

BINDEC—Binary to Decimal (Packed BCD)

This 25 byte subroutine represents very minimal code for translating a binary number of any length to packed BCD decimal.

ALGORITHM:

The packed BCD decimal result is resident just above the binary number. A sufficient number of bytes must be allowed for the BCD result. During each cycle of the algorithm the binary number is shifted left one bit position. The packed BCD decimal result is also shifted left one bit position, with the high order bit of the binary field being shifted up into the low order bit position of the BCD field. The shifted result in the BCD field is decimal corrected by using the DCOR instruction. Note that for addition an "ADD A, #066" instruction must be used in conjunction with the DCOR (Decimal Correct) instruction. The entire cycle is then repeated, with the total number of cycles being equal to the number of bit positions in the binary field.

16 Bit:	Binary in [1, 0]
	Packed BCD in [4, 3, 2]
24 Bit:	Binary in [2, 1, 0]
	Packed BCD in [6, 5, 4, 3]
32 Bit:	Binary in [3, 2, 1, 0]
	Packed BCD in [8, 7, 6, 5, 4]

25 Bytes
856 Instructions Cycles (16 Bit)

```
BINDEC:   LD        CNTR,#16        ;  LOAD CNTR WITH NUMBER OF BIT POSITIONS
                                    ;     IN BINARY FIELD
                                    ; #16 FOR 16 BIT (2 BYTE)
                                    ; #'S 24/32 FOR 24/32 BIT
          RC
          LD        B,#2            ; #'S 3/4 FOR 24/32 BIT
BD1:      LD        [B+],#0         ; CLEAR BCD FIELD
          IFBNE     #5              ; #'S 7/9 FOR 24/32 BIT
          JP        BD1             ; JUMP BACK FOR CLR LOOP
BD2:      LD        B,#0
BD3:      LD        A,[B]           ; PROGRAM LOOP TO
          ADC       A,[B]           ;    LEFT SHIFT
          X         A,[B+]          ;    BINARY FIELD
          IFBNE     #2              ; #'S 3/4 FOR 24/32 BIT
          JP        BD3             ; JUMP BACK FOR SHIFT LOOP1
BD4:      LD        A,[B]           ; PROGRAM LOOP TO
          ADD       A,#066          ;    LEFT SHIFT AND
          ADC       A,[B]           ;    DECIMAL CORRECT
          DCOR      A               ;    RESULT OF SHIFT
          X         A,[B+]          ;    IN BCD FIELD
          IFBNE     #5              ; #'S 7/9 FOR 24/32 BIT
          JP        BD4             ; JUMP BACK FOR SHIFT LOOP2
          DRSZ      CNTR            ; DECREMENT AND TEST IF
          JP        BD2             ;  CNTR EQUAL TO ZERO
          RET                       ; RETURN FROM SUBROUTINE
```

FBTOD—FAST BINARY TO DECIMAL (PACKED BCD)

Algorithm: This algorithm is based on the BINDEC algorithm, except that it is optimized for speed of execution.

Binary in [1, 0]
Packed BCD in [4, 3, 2]

59 Bytes
334 Instruction Cycles

```
FBTOD:  RC
        LD      B,#1
        LD      A,[B]
        SWAP    A               ;  REVERSE NIBBLES IN
        X       A,[B]           ;     UPPER BINARY BYTE
        LD      A,[B+]          ;  EXTRACT ORIGINAL UPPER
        AND     A,#0F           ;     NIBBLE OF HI BYTE
        IFGT    A,#9            ;  IF NIBBLE GREATER THAN
        ADD     A,#06           ;     NINE, THEN ADD SIX TO CORRECT BCD NIBBLE
        X       A,[B+]          ;  NIBBLE TO LOWER BCD BYTE
        LD      [B+],#0         ;  CLEAR UPPER BCD BYTES
        LD      [B],#0          ;  INITIALIZE CNTR TO COVER
        LD      CNTR,#4         ;     REMAINING HI NIBBLE (ORIGINALLY LO NIBBLE)
                                ;  IN UPPER BINARY BYTE
FBD1:   LD      B,#1            ;  PROGRAM LOOP TO
        LD      A,[B]           ;     LEFT SHIFT A BIT
        ADC     A,[B]           ;     OUT OF UPPER BINARY
        X       A,[B+]          ;     BYTE INTO LOW ORDER
        LD      A,[B]           ;     BIT POSITION OF BCD
        ADD     A,#066          ;     FIELD, AS LOWER TWO
        ADC     A,[B]           ;     BYTES OF BCD FIELD
        DCOR    A               ;     ARE LEFT SHIFTED WITH
        X       A,[B+]          ;     THE LOWER BYTE BEING
        LD      A,[B]           ;     DECIMAL CORRECTED
        ADC     A,[B]           ;  MIDDLE BYTE OF BCD FIELD
        X       A,[B]           ;     NEED NOT BE DECIMAL CORRECTED, SINCE
                                ;     MAX VALUE IS 2 (256)
        DRSZ    CNTR            ;  DECREMENT AND TEST IF
        JP      FBD1            ;     CNTR EQUAL TO ZERO
        LD      CNTR,#8         ;  INITIALIZE CNTR TO COVER
FBD2:   LD      B,#0            ;     LOWER BINARY BYTE
        LD      A,[B]           ;  PROGRAM LOOP TO
        ADC     A,[B]           ;     LEFT SHIFT A BIT
        X       A,[B]           ;     OUT OF LOWER BINARY
        LD      B,#2            ;     BYTE INTO LOW ORDER
        LD      A,[B]           ;     BIT POSITION OF BCD
        ADD     A,#066          ;     FIELD, AS BCD FIELD
        ADC     A,[B]           ;     IS LEFT SHIFTED WITH
        DCOR    A               ;     THE LOWER TWO BYTES
        X       A,[B+]          ;     OF THE FIELD BEING
        LD      A,[B]           ;     DECIMAL CORRECTED
        ADD     A,#066          ;  ADD (NOT ADC) HEX 66
        ADC     A,[B]           ;     TO SET UP "ADD" DCOR
        DCOR    A               ;  DECIMAL CORRECT MIDDLE
        X       A,[B+]          :     BYTE OF BCD FIELD
        LD      A,[B]           ;  UPPER BYTE OF BCD FIELD
        ADC     A,[B]           ;     NEED NOT BE DECIMAL
        X       A,[B]           ;     CORRECTED, SINCE MAX
                                ;     VALUE IS 6 (65535)
        DRSZ    CNTR            ;  DECREMENT AND TEST IF
        JP      FBD2            ;     CNTR EQUAL TO ZERO
        RET                     ;  RETURN FROM SUBROUTINE
```

VFBTOD—VERY FAST BINARY TO DECIMAL (PACKED BCD)

Algorithm: Decimal (Packed BCD) result is equal to summation in BCD of powers of two corresponding to 1's bits present in binary number.

Note that binary field (2 bytes) is initially one's complemented by program, in order to facilitate bypass branching when a tested bit in the binary field is found equal to zero.

Binary in [1, 0]
BCD in [4, 3, 2]

189 Bytes
144 Instruction Cycles Average
208 Instruction Cycles Maximum

```
VFBTOD:   RC
          LD        B,#0
          LD        A,[B]
          AND       A,#0F           ;  EXTRACT LO NIBBLE
          IFGT      A,#9            ;  TEST NIBBLE 9
          ADD       A,#6            ;  ADD 6 FOR CORRECTION
          LD        B,#2
          X         A,[B+]          ;  STORE IN LO BCD NIBBLE
          LD        [B+],#0         ;  CLEAR UPPER
          LD        [B],#0          ;     BCD NIBBLES
          LD        B,#1
          LD        A,[B]
          XOR       A,#0FF          ;  COMPLEMENT HI BYTE
          X         A,[B-]          ;     FOR REVERSE TESTING
          LD        A,[B]           ;     OF BINARY NUMBER
          XOR       A,#0FF          ;  COMPLEMENT LO BYTE
          X         A,[B]           ;     FOR REVERSE TESTING
          IFBIT     4,[B]           ;  TEST BINARY BIT 4
          JP        VFB1            ;     TO CONDITIONALLY
          LD        B,#2            ;     ADD BCD 16
          LD        A,#07C          ;  16 + 66
          ADC       A,[B]           ;  ADD BCD 16
          DCOR      A
          X         A,[B]
          LD        B,#0
VFB1:     IFBIT     5,[B]           ;  TEST BINARY BIT 5
          JP        VFB2            ;     TO CONDITIONALLY
          LD        B,#2            ;     ADD BCD 32
          LD        A,#098          ;  32 + 66
          ADC       A,[B]           ;  ADD BCD 32
          DCOR      A
          X         A,[B]
          LD        B,#0
VFB2:     IFBIT     6,[B]           ;  TEST BINARY BIT 6
          JP        VFB3            ;     TO CONDITIONALLY
          LD        B,#2            ;     ADD BCD 64
          LD        A,#0CA          ;  64 + 66
          ADC       A,[B]           ;  ADD BCD 64
          DCOR      A
          X         A,[B+]
          CLR       A
          ADC       A,[B]           ;  ADD CARRY
          X         A,[B]
          LD        B,#0
```

```
VFB3:   IFBIT   7,[B]           ;  TEST BINARY BIT 7
        JP      VFB4            ;     TO CONDITIONALLY
        LD      B,#2            ;     ADD BCD 128
        LD      A,#08E          ;  28 + 66
        ADC     A,[B]           ;  ADD BCD 28
        DCOR    A
        X       A,[B+]
        LD      A,#1
        ADC     A,[B]           ;  ADD BCD 1
        X       A,[B]
VFB4:   LD      B,#1            ;  HI BINARY BYTE
        IFBIT   0,[B]           ;  TEST BINARY BIT 8
        JP      VFB5            ;     TO CONDITIONALLY
        LD      B,#2            ;     ADD BCD 256
        LD      A,#0BC          ;  56 + 66
        ADC     A,[B]           ;  ADD BCD 56
        DCOR    A
        X       A,[B+]
        LD      A,#2
        ADC     A,[B]           ;  ADD BCD 2
        X       A,[B]
        LD      B,#1
VFB5:   IFBIT   1,[B]           ;  TEST BINARY BIT 9
        JP      VFB6            ;     TO CONDITIONALLY
        LD      B,#2            ;     ADD BCD 512
        LD      A,#078          ;  12 + 66
        ADC     A,[B]           ;  ADD BCD 12
        DCOR    A
        X       A,[B+]
        LD      A,#06B          ;  5 + 66
        ADC     A,[B]           ;  ADD BCD 5
        DCOR    A
        X       A,[B]
        LD      B,#1
VFB6:   IFBIT   2,[B]           ;  TEST BINARY BIT 10
        JP      VFB7            ;     TO CONDITIONALLY
        LD      B,#2            ;     ADD BCD 1024
        LD      A,#08A          ;  24 + 66
        ADC     A,[B]           ;  ADD BCD 24
        DCOR    A
        X       A,[B+]
        LD      A,#076          ;  10 + 66
        ADC     A,[B]           ;  ADD BCD 10
        DCOR    A
        X       A,[B]
        LD      B,#1
VFB7:   IFBIT   3,[B]           ;  TEST BINARY BIT 11
        JP      VFB8            ;     TO CONDITIONALLY
        LD      B,#2            ;     ADD BCD 2048
        LD      A,#0AE          ;  48 + 66
        ADC     A,[B]           ;  ADD BCD 48
        DCOR    A
        X       A,[B+]
        LD      A,#086          ;  20 + 66
        ADC     A,[B]           ;  ADD BCD 20
        DCOR    A
        X       A,[B]
        LD      B,#1
```

```
VFB8:   IFBIT   4,[B]           ;  TEST BINARY BIT 12
        JP      VFB9            ;     TO CONDITIONALLY
        LD      B,#2            ;     ADD BCD 4096
        LD      A,#0FC          ;  96 + 66
        ADC     A,[B]           ;  ADD BCD 96
        DCOR    A
        X       A,[B+]
        LD      A,#0A6          ;  40 + 66
        ADC     A,[B]           ;  ADD BCD 40
        DCOR    A
        X       A,[B]
        LD      B,#1
VFB9:   IFBIT   5,[B]           ;  TEST BINARY BIT 13
        JP      VFB10           ;     TO CONDITIONALLY
        LD      B,#2            ;     ADD BCD 8192
        LD      A,#0F8          ;  92 + 66
        ADC     A,[B]           ;  ADD BCD 92
        DCOR    A
        X       A,[B+]
        LD      A,#0E7          ;  81 + 66
        ADC     A,[B]           ;  ADD BCD 81
        DCOR    A
        X       A,[B]
        CLR     A
        ADC     A,[B]           ;  ADD CARRY
        X       A,[B]
        LD      B,#1
VFB10:  IFBIT   6,[B]           ;  TEST BINARY BIT 14
        JP      VFB11           ;     TO CONDITIONALLY
        LD      B,#2            ;     ADD BCD 16384
        LD      A,#0EA          ;  84 + 66
        ADC     A,[B]           ;  ADD BCD 84
        DCOR    A
        X       A,[B+]
        LD      A,#0C9          ;  63 + 66
        ADC     A,[B]           ;  ADD BCD 63
        DCOR    A
        X       A,[B+]
        LD      A,#1
        ADC     A,[B]           ;  ADD BCD 1
        X       A,[B]
        LD      B,#1
VFB11:  IFBIT   7,[B]           ;  TEST BINARY BIT 15
        RET                     ;     TO CONDITIONALLY
        LD      B,#2            ;     ADD BCD 32768
        LD      A,#0CE          ;  68 + 66
        ADC     A,[B]           ;  ADD BCD 68
        DCOR    A
        X       A,[B+]
        LD      A,#08D          ;  27 + 66
        ADC     A,[B]           ;  ADD BCD 27
        DCOR    A
        X       A,[B+]
        LD      A,#3
        ADC     A,[B]           ;  ADD BCD 3
        X       A,[B]
        RET
```

Pulse Width Modulation A/D Conversion Techniques with COP800 Family Microcontrollers

National Semiconductor
Application Note 607
Kevin Daugherty

1.0 BASIC TECHNIQUE

This application note describes a technique for creating an analog to digital converter using a microcontroller with other low cost components. Many applications do not require the speed associated with a dedicated hardware A/D converter and it is worth evaluating a more cost effective approach.

With a high speed CMOS microcontroller an eight bit A/D can be implemented that converts in approximately 10 ms. This method is based on the fact that if a repetitive waveform is applied to an RC network, the capacitor will charge to the average voltage, provided that the RC time constant is much larger than the pulse widths. The basic equation for computing the analog to digital result is:

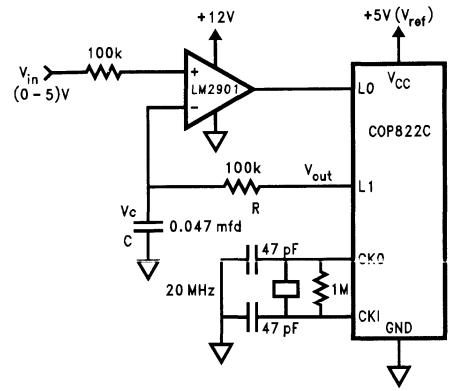
$$V_{in} = V_{ref}[T_{on}/(T_{on} + T_{off})] \quad (1)$$

With this equation it is necessary to precisely measure several time periods within both the T_{on} and T_{off} in order to achieve the desired resolution. Additionally, the waveform would have to be gradually adjusted to allow for the large RC time constant to settle out. This results in a relatively long conversion cycle. Modifying the equation and technique slightly, significantly speeds up the process. This technique works by averaging several pulses over a fixed period of time and is based on the following equation:

$$V_{in} = V_{ref}[\text{Sum of } T_{on}/(\text{Sum of } (T_{on} + T_{off}))] \quad (2)$$

2.0 IMPLEMENTATION

Figure 1 describes the basic circuit schematic that uses a National Semiconductor COP822C microcontroller, a low cost LM2901 comparator, two 100k resistors, and a 0.047 mfd film capacitor. The CMOS COP822C microcontroller provides a squarewave signal with logic levels very close to GND and V_{CC}. This generates a small ramp voltage on the capacitor for the LM2901 quad comparator input.



TL/DD/10407-1

FIGURE 1. Basic Circuit

To minimize error, a tradeoff must be made when selecting the resistor. The microcontroller output (L1) should have a large resistor to minimize the output switching offset (V_{os}), and the comparator should have a small resistor due to error caused by I_{bos} (input bias offset current).

Once the resistor is determined, the capacitor should be chosen so that the RC time constant is large enough to provide a small incremental voltage ramp. This design has a sample time of 20 μs and has a 4.7 ms time constant with a 0.047 mfd film type capacitor which has low leakage current to prevent errors. Since a 100k resistor is used in the RC network for one comparator input, another 100k resistor is required for the V_{in} input to balance the offset voltage caused by the comparator I_b (input bias current).

Figure 2 illustrates the relationship between the microcontroller squarewave output and the capacitor charge and discharge. Every 20 μs the comparator is sampled. If the capacitor voltage (V_c) is below V_{in} the RC network will receive a positive pulse. The inverse is true if V_c is above V_{in} at sample time. Note that with this approach, the PWM waveform is broken up into several small pulses over a fixed period instead of having a single pulse represent the duty cycle; thus a relatively small RC time constant can be used.

Mathematical Analysis:

let n = total number of T_{on} pulses and
m = total number of T_{off} pulses

then $V_c(t) = V_c + n[(V_{out} - V_c)(1 - e - t/RC)] - m[(V_c - V_o)(1 - e - t/RC)]$

let $V_c = V_{in}$ at start of conversion and
$K = (1 - e - t/RC)$

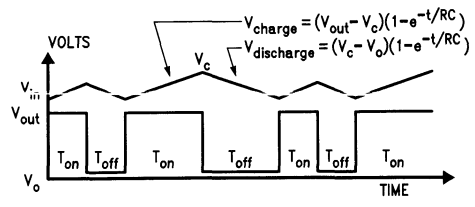
then $V_{in} = V_{in} + K_nV_{out} - K_nV_{in} - K_mV_{in} + K_mV_O$
$0 = K_nV_{out} + K_mV_o - KV_{in}(n + m)$

let $V_{out} = V_{ref} - V_{os}$

solving for V_{in}:

$$V_{in} = nV_{ref}/(n + m) - (nV_{os} - mV_o)(1/(n + m)) \quad (3)$$

Note that the RC value drops out of the equation and therefore is not an error factor.



TL/DD/10407-2

FIGURE 2. PWM Signal

3.0 SOFTWARE DESCRIPTION

Single Channel

Referring to the flow chart in *Figure 3*, and the code listed in *Figure 4*, the software counters T_{on} and TOTAL are first preloaded with the FF. The accumulator and register 0F1 are then loaded with 2 to provide for an initialization and final conversion cycle. Next, the L port is configured to complete the initialization of the microcontroller.

The comparator output is checked with the IFBIT 0,0D2 instruction. This will determine whether the RC network will receive a positive (V_{ref}) or ground pulse. You can think of the microcontroller as part of the feedback path of the comparator. The microcontroller uses the comparator output to decide what level output on L1 is required to keep the capacitor equal to the unknown input voltage. Each time the negative or GND pulse is applied, the T_{on} counter is decremented by DRSZ. Similarly, each time a sample loop is completed the TOTAL counter is decremented by DRSZ. Note that NOP instructions are used in the high and low loops. These are necessary to provide exactly the same cycles for a high or low L1 output pulse.

Once the TOTAL register is decremented to zero, the initialization loop is completed. Immediately afterwards, the L1 output is put in TRI-STATE® mode to minimize capacitor voltage variations while other instructions are completed. After the first conversion, the IFEQ A,0F1 instruction will be true and the T_{on} and TOTAL registers will be reloaded with FF. Following this, the L1 pin is restored as a high output and the 0F1 multiplier is decremented.

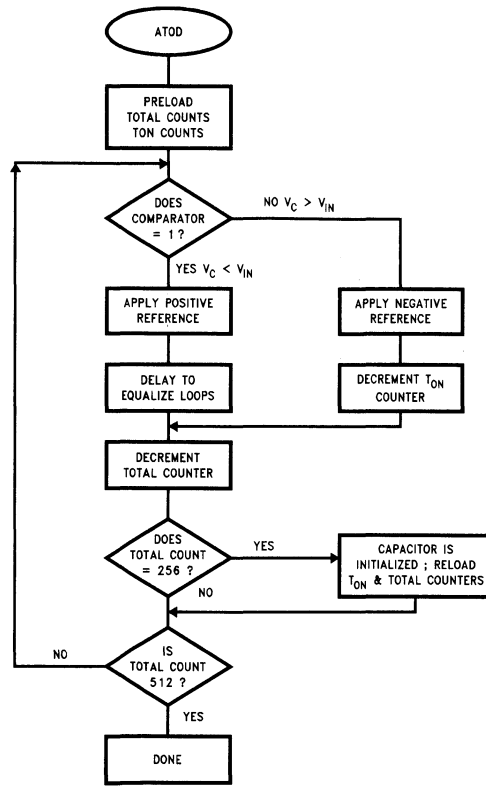
At this point the capacitor is equal to V_{in} and the actual conversion is started. When the TOTAL register is decremented to zero (255 samples later), the conversion is complete. T_{on} will not be reloaded since 0F1 was decremented and IFEQ A,0F1 will no longer be true. The accumulator is then loaded with T_{on} and stored in RAM location 00 with X A,00.

The final two instructions (RBIT 1,LCONF & RBIT 1[B]) are optional depending on the application and the amount of additional code required. This will prevent the capacitor from decaying appreciably between conversions and allow for a much quicker capacitor initialization time. Otherwise more time may be required, or a diode speed-up circuit as shown in *Figure 7d* is required to fully charge the capacitor prior to starting the actual conversion.

Eight Channel

This is bascially the same as that for the single channel. Referring to the flow chart in *Figure 5* and the code in *Figure 6*, the differences are in the front and back ends. Before the conversions are started, the X register is initialized to 00 for RAM location 00. The accumulator is then loaded with the current RAM pointer (LD A,X), OR'ed with the LDATA (OR A,LDATA), and finally the LDATA register is modified to provide for the proper output select (X A,LDTA).

Following the actual conversion cycle, the result is stored at the current RAM pointer (X A,[X+]) which also auto-increments the X register. The next conversion will use this to select the next channel and determine where to store the result. Once the eighth channel is converted, the IFEQ A,X instruction will be true and the RAM pointer will be reset (LD X,#00) before the next conversion is started.

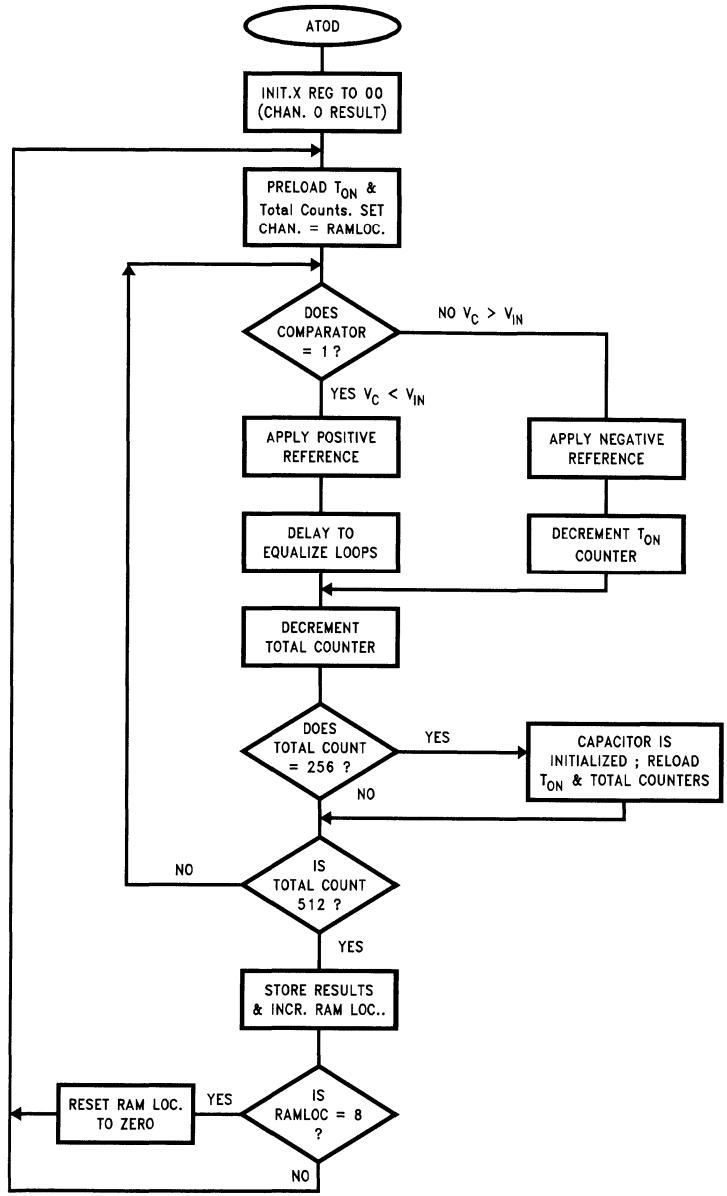


TL/DD/10407–7

FIGURE 3. PWM A/D Flow Chart

```
;The program listed below will work in any COP800 microcontroller
;(i.e. COP820, COP840, COP880, COP888). SET UP FOR .047 mfd CAP.,
;100K RES, @1 MICRO. CYCLE TIME. THE FIRST CONVERSION
;INITIALIZES, AND 2nd IS THE RESULT STORED IN RAM LOCATION 00.
.CHIP 820
LCONF=0D1
LDATA=0D0
TON=0F2
TOTAL=0F0
;
        LD A,#02          ;USED TO DETERMINE WHEN TO RELOAD
        LD TOTAL,#0FF     ;PRELOAD TOTAL COUNTS
        LD 0F1,#2         ;MULTIPLIER (255 TO INIT. PLUS 255 FOR RESULT)
        LD TON,#0FF       ;PRELOAD Ton
        LD 0FE,#0D0       ;LOAD B REG TO POINT TO LDATA REG.
        LD LDATA,#01      ;L PORT DATA REG, L0=WEAK PULL UP, L1=HIGH
        LD LCONF,#02      ;L PORT CONFIG REG, L0=INPUT, L1=OUTPUT
LOOP:   IFBIT 0,0D2       ;TEST COMPARATOR OUTPUT
        JP HIGH           ;JUMP IF L0=1
        NOP
        NOP               ;EQUALIZE TIME FOR SETTING AND RESETTING
        RBIT 1,[B]        ;DRIVE L1 LOW
        DRSZ Ton          ;DECREMENT Ton WHEN DRIVING LOW
        JMP COUNT
HIGH:   SBIT 1,[B]        ;DRIVE L1 HIGH
        NOP
        NOP
        NOP
        NOP
        NOP
        NOP               ;EQUALIZE HIGH AND LOW LOOPS
COUNT:  DRSZ TOTAL        ;DECREMENT TOTAL COUNTS
        JP LOOP
        RBIT 1,LCONF      ;TRISTATE L1 TO MINIMIZE ERRORS FROM EXTRA
        RBIT 1,[B]        ;CYCLES
        IFEQ A,0F1        ;CHECK INITIALIZATION LOOP COMPLETE
        JP RELOAD         ;JUMP IF TRUE.
        JP DEC            ;JUMP IF NOT END OF 2nd LOOP
RELOAD: LD 0F2,#0FF       ;RELOAD Ton WITH FF
        LD 0F0,#0FF       ;SYNC TOTAL AND Ton COUNTERS
DEC:    SBIT 1,[B]        ;SET L1 HIGH
        SBIT 1,LCONF      ;RESTORE L1 AS OUTPUT.
        DRSZ 0F1          ;DECREMENT MULTIPLIER UNTIL ZERO
        JMP LOOP          ;CONTINUE A/D UNTIL AFTER 2nd CONVERSION
        LD A,TON          ;LOAD A WITH Ton
        X, A,00           ;STORE RESULT IN RAM LOCATION 00
.end
```

FIGURE 4. Single Channel PWM A/D Listing



TL/DD/10407–8

FIGURE 5. 8 Channel PWM A/D Flow Chart

```
;L0,1,2 SELECTS CHANNEL OF CD4051 8:1 MUX, L3 IS THE COMP.
;OUTPUT, AND L4 DRIVES THE RC. RESULTS STORED IN RAM 00-07.
.CHIP 820
LDATA=0D0
LCONF=0D1
TON=0F2
TOTAL=0F0
        LD X,#00          ;INITIALIZE X REG FOR 1st RAM LOC.
CONVER: LD TOTAL,#0FF     ;PRELOAD TOTAL COUNTS
        LD OF1,#02        ;TOTAL LOOP COUNTER
        LD TON,#0FF       ;PRELOAD Ton
        LD OFE,#0D0       ;INIT. B REG TO POINT TO LDATA REG
        LD LDATA,#018     ;LDATA, L0-2=LOW, L3=PULLUP, L4=HIGH
        LD A,X            ;USED CURRENT RAM POINTER TO SELECT-
        OR A,LDATA        ;PROPER A/D CHANNEL.
        X A,LDATA         ;MODIFY LDATA FOR CHANNEL SELECTION.
        LD LCONF,#017     ;LCONF REG. L0-L2, L4=OUTPUT, L3=IN
LOOP:   IFBIT 3,0D2       ;TEST COMPARATOR OUTPUT AT L3 INPUT
        JMP HIGH          ;JUMP IF L3=HIGH
        NOP
        NOP               ;EQUALIZE TIME FOR SET AND RESET
        RBIT 4,[B]        ;DRIVE L4 LOW WHEN COMPARATOR IS LOW.
        DRSZ TON          ;DECREMENT Ton WHEN APPLYING NEG. REF.
        JMP COUNT         ;JUMP TO COUNT UNLESS Ton REACHES ZERO
HIGH:   SBIT 4,[B]        ;DRIVE L4 HIGH WHEN COMPARATOR IS HIGH
        NOP
        NOP
        NOP
        NOP
        NOP
        NOP               ;EQUALIZE HIGH AND LOW LOOP TIMES
COUNT:  DRSZ TOTAL        ;DEC. TOTAL COUNTS EACH LOOP
        JMP LOOP          ;JUMP UNLESS TOTAL CNTS.=0
        RBIT 4,LCONF      ;TRISTATE L4 TO MINIMIZE ERROR
        RBIT 4,[B]        ; "
        LD A,#02          ;USE TO DETERMINE WHEN TO RELOAD
        IFEQ A,OF1        ;CHECK FOR 2nd CONVERSION COMPLETE
        JP RELOAD         ;IF TRUE.
        JP DEC            ;OTHERWISE JUMP TO DEC
RELOAD: LD TON,#0FF       ;RELOAD Ton FOR START OF NEXT CONV.
        LD TOTAL,#0FF     ;SYNC Ton AND TOTAL COUNTERS
DEC:    SBIT 4,[B]        ;SET L4 HIGH
        SBIT 4,LCONF      ;RESTORE L4 AS OUTPUT.
        DRSZ OF1          ;DECREMENT TOTAL LOOP UNTIL ZERO
        JMP LOOP          ;DONE WHEN OF1 IS ZERO.
        LD A,TON          ;LOAD A WITH Ton RESULT
        X A,[X+]          ;STORE RESULT AT CURRENT RAM POINTER
                          ;AND AUTO INCREMENT POINTER
        LD A,#08          ;CHECK [X] RAM POINTER FOR
        IFEQ A,X          ;EIGHTH CHANNEL CONVERTER
        LD X,#00          ;RESET RAM POINTER IF [X]=8
        JMP CONVER
.END
```

FIGURE 6. 8-Channel PWM A/D Listing

4.0 ACCURACY AND CIRCUIT CONSIDERATIONS

The basic circuit will provide 8 bits ±1 LSB accuracy depending on the choice of comparator, and passive components. With this type of design several tradeoffs and error sources should be considered. First of all, conversion equation 2 assumes that the microcontroller output switches exactly to GND and V_{CC} (or V_{ref}). The COP822C will typically switch between 10 mV and 20 mV from GND and V_{CC} with a light load. This will cause an error equal to the offset voltage times the duty cycle (equ. 3). Fortunately, the offsets tend to cancel each other at mid range voltages. At near GND and V_{CC} input voltages the offsets are minimal due to the very small voltage drop across the resistor. If the error is undesirable, the offset voltage can be reduced by paralleling outputs with the same levels together, or by using a CMOS buffer such as a 74HC04 to drive the RC network (see *Figure 7* for suggested circuits).

Another possible source of error is with the LM2901 worst case input bias offset current of 200 nA over temperature. This will cause an error equal to R_{in} X I_{bos}, which equals 20 mV with a 100k resistor. Either the resistor or the I_{bos} can be reduced to improve the error. If the resistor is reduced then the L port offset voltages will increase so the preferred approach is to select a comparator with lower I_{bos} such as the LP339 which has an I_{bos} of only ±15 nA. The comparator V_{os} may also introduce error. The LM2901 V_{os} is ±9 mV, the LP339 V_{os} is only ±5 mV. An added benefit of using the LP339 is that since the I_{bos} is so small, the resistor for the RC network can be larger. In addition, one RC network could be used for several comparator input channels (refer to *Figure 7A*).

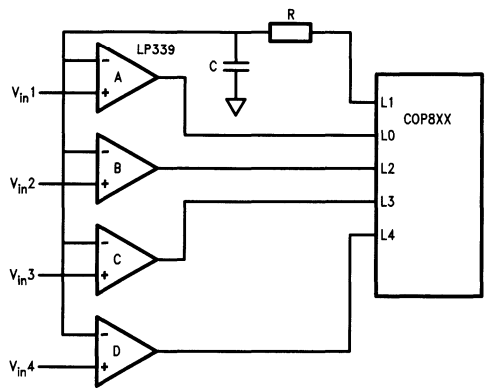
By using the LM604 *(Figure 7B)* the basic software can be easily extended for converting several channels. This will only require a control line to be selected before a conversion is started. Since the LM604 needs to be powered from a higher voltage than the input voltage range, the output voltage will also be higher than the microcontroller supply. This requires a current limiting resistor to be used in series between the LM604 output and the COP8XX. Note that two or more LM604's can be paralleled for providing several more A/D channels by utilizing the EN control input that can TRI-STATE the LM604 output when high.

When more than 4 channels of analog signals are required to be measured, the circuit in *Figure 7(d)* is recommended. This circuit utilizes an inexpensive CD4051 8:1 multiplexer with a single comparator (which could be on-board the micro). When measuring several input voltages that can vary, TRI-STATING the output driving the RC between conversions is not possible. It is necessary to provide 6x RC time constants to charge the capacitor to within 0.25%. Note that there are two 1N4148's across the comparator inputs. The diodes provide a quick capacitor charge path providing that the total input resistance is much smaller than the resistor used in the RC network (a 2k resistor will meet the requirements within 255 sample times). Once the capacitor is charged to within about 0.6V, the diodes will start turning off. At this point the microcontroller will start dominating the charge/discharge of the capacitor. After the initialization cycle is complete, the capacitor is very close to the unknown V_{in} and the diodes are effectively out of the circuit.

Depending on the speed and accuracy requirements, the total number of counts used in the conversion can be changed. Increasing the counts will give more accuracy with the practical limit of about 9–10 bits. With increased resolution, the capacitor ramp voltage per sample time should be decreased so that the capacitor can be initialized to within 1 LSB prior to conversion. This can be done by either increasing the RC time constant, or by using an initialization routine with a shorter sample time. The conversion time will depend on the total counts and the microcontroller oscillator frequency as described below:

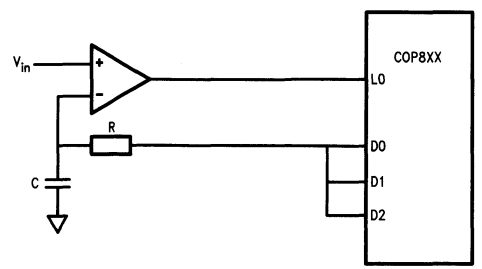
$$T_{con} = \text{Total counts} \times (20 \text{ cycles}) \times (\text{instruction cycle time})$$

Another factor to consider is when a non-ratiometric conversion is required, the reference voltage must have the tolerance to match the desired accuracy.



TL/DD/10407–4

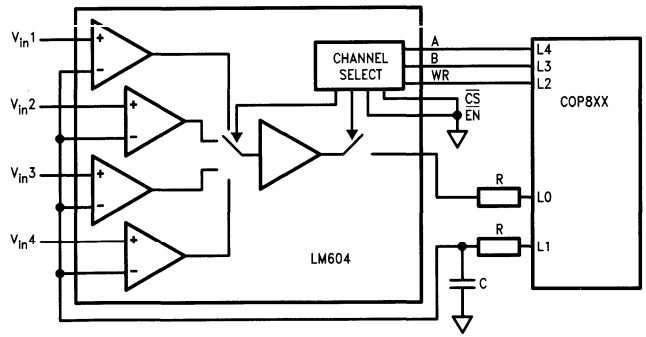
A. Multiple Channels with LP339 Low I_{bos} Comparator



TL/DD/10407–5

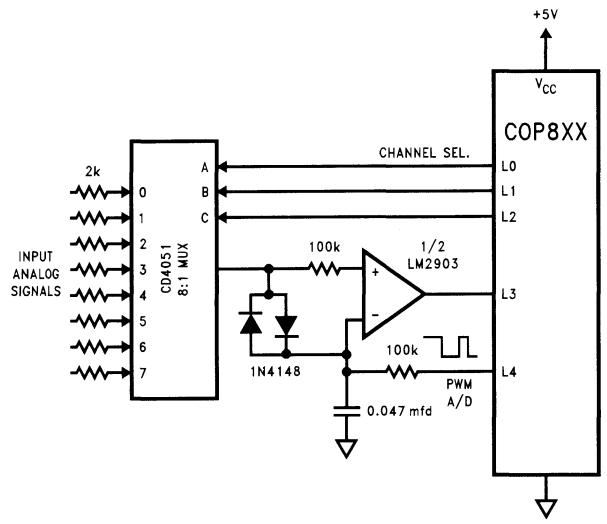
B. High Drive with Multiple Outputs

FIGURE 7. Suggested Circuits



TL/DD/10407–6

C. Four Channel A/D with LM604 MUX-Amplifier



TL/DD/10407–9

D. Eight Channel PWM A/D Circuit

FIGURE 7. Suggested Circuits (Continued)

5.0 CONCLUSION

The PWM A/D technique described in this application note provides a relatively fast discrete implementation with substantial cost savings compared to a dedicated hardware A/D. Minimal microcontroller I/O and software is required to interface with a comparator and RC network. Depending on the application requirements, the designer can tailor the basic 8-bit A/D a number of ways. By varying the total software counts, the desired speed and resolution can be adjusted. The number of A/D channels will determine the number of comparators used. In chosing the comparator, it is recommended that the designer refer to the data sheets and match the I_{bos} and V_{os} to the desired accuracy.

When other than a 1 μs instruction cycle is used, the RC time constant of 4.7 ms should be scaled to provide for a maximum peak-peak ramp voltage of <1 LSB of the desired accuracy. For example, if 8-bit accuracy is desired and the instruction cycle time is now 4 μs instead of 1 μs, multiply 4.7 ms by 4 to calculate the new RC.

Keep in mind that the comparator input voltage is limited so that you do not get erroneous/nonlinear results. Another possible problem is during development. When doing in-circuit emulation with the development equipment, note that there will be ground loops in the cable thus causing errors in your measurements. You can reduce this by connecting an extra GND and V_{CC} wire between your prototype and development system power and GND. It is still possible to see offsets in the sockets holding the COP8XX in the development board, however this should be relatively small. The best test is to take accurate measurements with an emulator in the actual prototype circuit.

COP800 Based Automated Security/Monitoring System

National Semiconductor
Application Note 662
Ramesh Sivakolundu

INTRODUCTION

National Semiconductor's COP800 family of full-feature, cost effective, fully static, single chip micro CMOS microcontrollers provide efficient system solutions with a versatile instruction set and high functionality. The heart of the ASM System prototype is a COP800 family member with at least the following features: 4k bytes of on-board program memory, 192 bytes of on-board data memory, memory mapped I/O, fourteen multi-sourced vectored interrupts and a versatile instruction set. The family member used is the COP888CG microcontroller.

This application note describes the implementation of a Security/Monitoring System using the COP888CG microcontroller. The COP888CG contains features such as:

- Low power HALT and IDLE modes
- MICROWIRE/PLUS™ serial communication
- Multiple multi-mode general purpose timers
- Multi-input wakeup/interrupt
- WATCHDOG™ and Clock monitor
- Maskable vectored interrupt scheme
- UART

In addition to these features common to the COP888 subfamily of microcontrollers, COP888CG has a full duplex, double buffered UART and two Differential Comparators.

The COP888CG based Automated Security/Monitoring (ASM) System consists of several features:

- Automatic Telephone Dialing
- Real Time Clock
- Non-Volatile storage of real time information of events
- Continuous display of events on the terminal
- Battery operated remote sensors and transmitters
- Exit and Entry delays
- Expandable to add new features

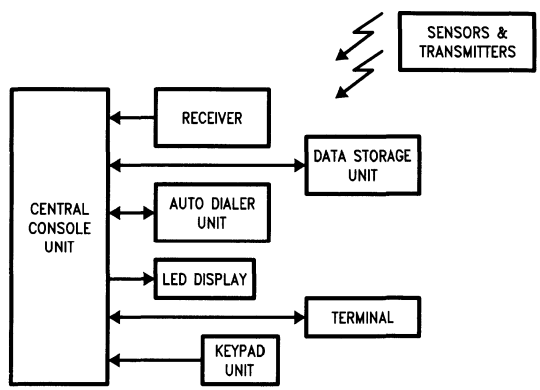
SYSTEM OVERVIEW

Figure 1 gives the block diagram of the ASM System prototype hardware. The application consists of following major blocks:

- Central Controlling Unit
- Receiver
- Sensors and Transmitters
- Keypad Unit
- Auto-Dialer Unit
- Data Storage Unit
- Display Terminal Unit
- LED Display Unit

The implementation allows easy expansion of the ASM System features by adding new blocks to the Central Controlling Unit.

COP888CG is the workhorse of the ASM System and provides the processing power to scan the keypad, service the Receiver interrupts, update the real time clock, serially communicate with the LED display unit and Data Storage Unit, activate the Auto-Dialer Unit and use the full-duplex double buffered UART to interface with the Display Terminal Unit. System capabilities may be enhanced or scaled down by simply changing the processor's algorithm. The subsequent sections describe each of the units and their interface with the COP888CG.



TL/DD/10607–1

FIGURE 1. Block Diagram of Security/Monitoring System

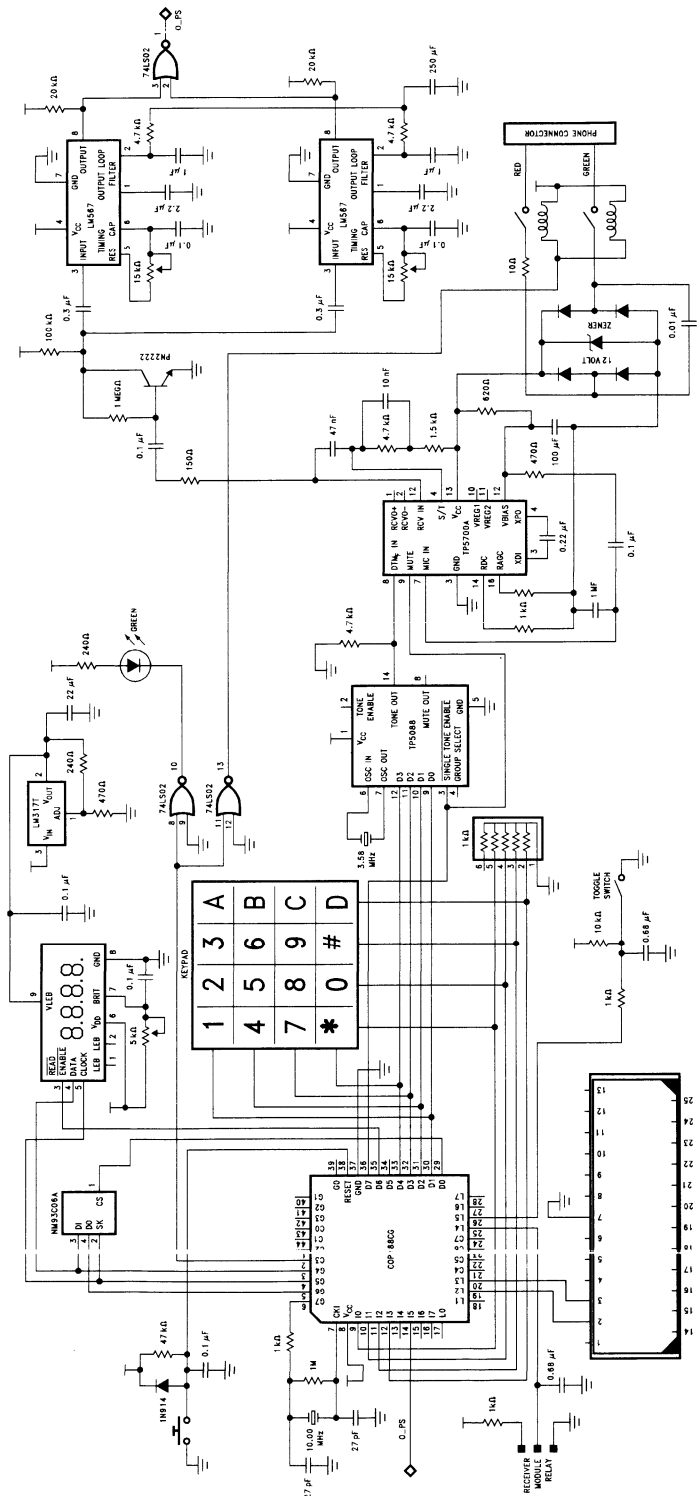


FIGURE 2. Schematic of the ASM System Prototype

HARDWARE DESCRIPTION

This section describes the various blocks in the ASM System briefly and highlights the hardware considerations in the design of the System.

Receiver Unit

The Receiver Unit operates with the Sensors and Transmitter Unit. An eight-key dip switch makes it possible to select 256 different digital codes. A detector LED indicates the level of the radio frequency (RF) energy detected by the receiver and enables the user to determine the best locations for the transmitter(s) and receiver, assuring reliable operation.

Figure 2 shows the interface between the COP888CG and the Receiver Unit on the bi-directional I/O Port L capable of functioning as Multi-Input WakeUp (MIWU). In this implementation the WR-200 series of receivers manufactured by Visonic Ltd was used. These receivers are designed to operate with Visonic standard transmitters. The receiver operates on 12 VDC. When RF signal from the transmitter(s) is detected, the receiver activates a relay which in turn interrupts the microcontroller. The output of the relay is connected to the Port L of the COP888CG whose alternate function includes, the Multi-Input WakeUp feature. The COP888CG, after a time delay of 10 seconds, activates the Auto-Dialer Unit. The microcontroller turns on a LED to indicate an alarm signal was detected and is being processed.

Sensors and Transmitters

This unit has a built-in reed switch which can be used with a magnet to activate the transmitter. An eight-key dip switch forms the code selector and each key can be set to either ON or OFF position to create a unique code. This code should match with the code selected on the receiver unit.

Model WR-100 Universal Wireless Transmitter, manufactured by Visonic Ltd. was used in the implementation of the Security/Monitoring System.

Keypad Unit

The Keypad Unit consists of 4 x 4 matrix keyboard. The *Figure 2* shows the keyboard matrix interface to COP888CG. The keyboard is scanned periodically by addressing a column in the keyboard matrix. The program senses the key closure in that column by testing the Port I lines (I0 to I3) which are connected to the rows of the keyboard matrix. Thus, each key is associated with the conjunction of one Port D output line and one Port I input line only.

The keypad unit is used to program the real time clock in order to set the time and date. The telephone number to be dialed in case of a security breach can also be programmed through the keypad as well as the terminal keyboard in the Terminal Unit.

Auto-Dialer Unit

The Auto-Dialer Unit dials the number programmed by the user upon detection of RF signal by the Receiver from the Sensors and Transmitter Unit. The unit consists of two ICs and some peripheral circuitry. National Semiconductor's TP5700A is the Telephone Speech Circuit and TP5088 is the DTMF generator. These two chips are interfaced to the COP888CG as in *Figure 2*. The COP888CG outputs the digit to be dialed to TP5088 and the output of the DTMF generator is inputted to the Speech Circuit. The Speech Circuit interfaces with the telephone lines.

TP5088 is a low cost CMOS device that provides the tone-dialing capability in microprocessor-controlled telephone applications. TP5700A is a linear bipolar device which includes the functions required to build the speech circuit of a telephone. It replaces the hybrid transformer, compensation circuit and sidetone network used traditional designs.

Data Storage Unit

The Data Storage Unit stores the real time data of events that the Receiver Unit detects and informs the Central Controlling Unit. The storage is non-volatile and can be archived for later references. The Terminal Unit can request the Central Controlling Unit to display the events and the data stored in the Storage Unit. The telephone number to be dialed by the Auto-Dialer Unit is also stored in this unit. This unit interfaces with the COP888CG using the MICROWIRE/PLUS™ serial communication protocol.

In this implementation the COP888CG microcontroller interfaces with NM93C06A Serial EEPROM Memory. The NM93C06A contains 256 bits of read/write EEPROM organized as 16 registers of 16 bits each. Written information has a retention period of at least 10 years. *Figure 2* shows the interface between COP888CG and NMC9306.

Any sequentially accessible memory device that is compatible with the MICROWIRE/PLUS™ serial communication protocol can be used as a Data Storage Unit. The Central Controlling Unit checks for the availability of memory and informs the user of the same if memory is full. Upon receipt of memory full prompt, the user can decide to overwrite or replace the memory device.

Display Terminal Unit

The Display Terminal Unit interfaces with the COP888CG through the full-duplex, double buffered UART. The COP888CG is interrupted by the terminal and the microcontroller decodes the ASCII character sent and services the corresponding request. The terminal keyboard can be used to program the telephone number to be dialed by the Auto-Dialer Unit. The real time clock is displayed on the terminal screen. The user can request the Central Controlling Unit to display the history of events monitored by the AMS System. The Central Controlling Unit retrieves the information from the Date Storage Unit and displays it on the screen.

The ASM System utilized a Visual 550 terminal. The terminal employs two independent display memories: alphanumerics and graphics. The alphanumeric functions of the V550 is ANSI X3.64 compatible and the graphics functions are fully compatible with Tectronix Plot 10® software.

With slight modification of the Central Controlling Unit's algorithm it is possible to make the ASM System interface with any other terminal unit.

LED Display Unit

The LED Display Unit is used to display the time and date information. *Figure 2* shows the interface between COP888CG and the Display Terminal Unit. The COP888CG communicates with this unit serially using the MICROWIRE/PLUS protocol.

The NSM4000A LED Display with Driver is used in the ASM System. The NSM4000A is a 4-digit 0.3" height LED display with serial data-in parallel data-out LED driver designed to operate with minimal interface to the data source. The Cen-

tral Controlling Unit does not update the display when it is servicing the Receiver Unit. The APS System has a toggle switch that enables toggling the display between Hours-Minutes to Seconds-1/80th of Seconds. The Keypad Unit is used to toggle the display between time and date.

Central Controlling Unit

This is the main unit in the application and is responsible for the efficient operation of the various units in the ASM System. The unit consists of COP888CG and the application software. The next section describes the application software in detail. The COP888CG interfaces with the various units described in the previous sections *(Figure 2)*.

The application is a real time system and is totally interrupt driven with some of the tasks being executed in the background. The various units that interface with the COP888CG can be considered as tasks and the Central Controlling Unit executes these tasks based on their priority and the sequence of occurrence. The real time clock counter is given the highest priority. The Receiver Unit uses the Multi-Input Wakeup/Interrupt feature of the COP888CG to wakeup the microcontroller and service the Alarm routine. The Display Unit has a display toggle switch which also uses the Multi-Input Wakeup/Interrupt to toggle the display between Hours-Minutes and Seconds-1/80th of Seconds.

The COP888CG communicates with the Terminal Unit through the on-board, full duplex, double buffered UART. The terminal keyboard can be used to interrupt the COP888CG to program the phone number to dial in case of an emergency. The COP888CG uses the MICROWIRE/PLUS™ serial communication protocol to display the time and date information on the LED display and also to store real time information of events in the non-volatile data storage unit. Thus the MICROWIRE/PLUS protocol is time shared between the Display Unit and Data Storage Unit.

The Keypad Unit is a 4 x 4 array of keys and the COP888CG periodically polls the keypad. The input/output ports of the COP888CG is used to read the key pressed and is decoded by the software. The Auto-Dialer Unit is driven by the input/output lines and the interface between COP888CG. This unit is activated by the COP888CG 10 seconds after the Receiver Unit interrupts the microcontroller. This delay is used to disarm the Alarm routine.

SOFTWARE DESCRIPTION

The instruction set of the COP800 family of microcontrollers provide easy optimization of program size and throughput efficiency. Most of the instructions of the COP800 family are single-byte, single-cycle instructions (approximately 60%). The COP800 family of microcontrollers has three memory mapped registers (B, X and SP). The B and X registers can be used as data store memory pointers for register indirect addressing with optional auto post incrementing or decrementing of the associated pointer. This allows greater efficiency in cycle time and program code. The COP800 family allows true bit-manipulation i.e., the ability to set, reset or test any individual bit in data memory including the memory mapped I/O ports.

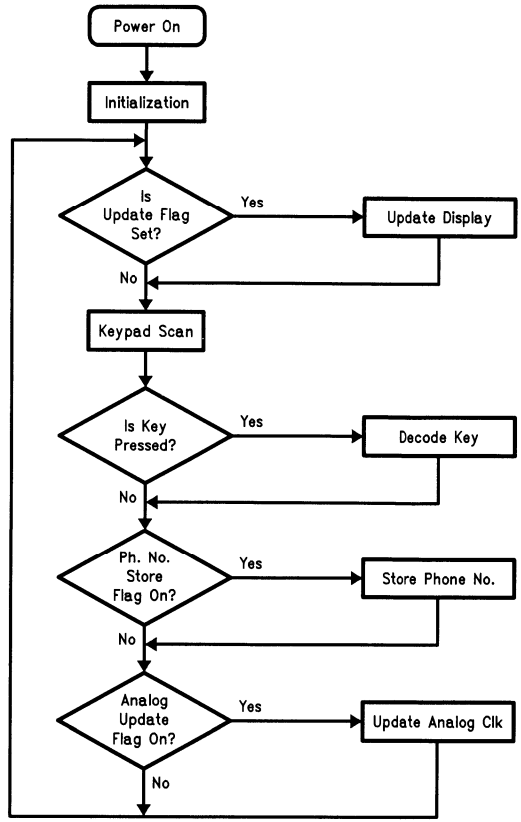
The architecture of COP800 family is based on a modified Harvard type architecture, where the Control Store Program (in ROM) is separated from the Data Store Memory (in RAM). Both types of memory have their own separate addressing space and separate address busses. This architecture allows the overlap of ROM and RAM memory accesses which is not possible with single-address bus Von Neumann-style architecture. The modified Harvard architecture allows access to ROM data tables which is not possible with the classical Harvard architecture.

The COP888 sub-family of microcontrollers support a total of sixteen vectored interrupts, of which fourteen are maskable interrupts and two high-priority, non-maskable interrupts. A 2-byte interrupt vector is reserved for each of these sixteen interrupts and they are stored in a user-defined 32-byte program memory (ROM) table. Please refer to the COP888 users manual or the Microcontrollers Databook for more detailed information on interrupts.

The MIWU feature, which utilizes the Port L, of the COP888 sub-family can be used to wakeup the microcontroller from the two power saving modes, i.e., HALT or IDLE modes. Alternately, the MIWU/Interrupt allows the user to generate eight additional edge selectable external interrupts. Three 8-bit memory mapped registers (WKEDG, WKEN and WKPND) are used to implement the MIWU/Interrupt. The three control registers each contain an associated pin for each L port pin. The WKEN register is used to select which particular Port L inputs will be used. The user can select whether the trigger condition on a selected L port pin is to be a positive edge (low to high transition) or a negative edge (high to low transition). This selection is made through the WKEDG register. The occurrence of the selected trigger condition for MIWU/Interrupt is latched into the associated bit of the Wakeup Pending Register (WKPND).

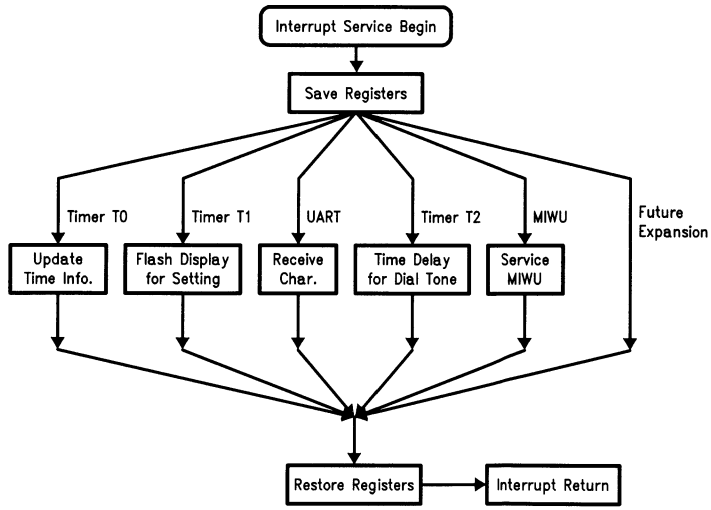
The COP800 family has the ability to detect various illegal conditions resulting from coding errors, transient noise, power supply voltage drops, runaway programs, etc. Reading an undefined ROM location gets zeroes, which results in a non-maskable software interrupt thus signalling an illegal condition has occurred. In addition to this, the COP888 sub-family supports both WATCHDOG™ and Clock Monitor. The WATCHDOG™ is used to monitor the number of instruction cycles between WATCHDOG™ services in order to avoid runaway programs or infinite loops. The Clock Monitor is used to detect the absence of a clock or a very slow clock below a specified rate. These features of the COP800 family provide easy implementation of real time applications where the proper execution of the software plays a crucial role.

The major features of the software written for the ASM System implementation are described on the flow chart *Figure 3*. The main program flow is to detect the flags set, service the flags and scan the Keypad. The rest of the software is interrupt driven. The program is real time and the interrupts are serviced as and when they occur. Some of the routines are running in the background all the time, such as, Time Keeping Routine and Keypad Scan Routine. *Figures 4* and *5* gives the flow of the various interrupt service routines. The following sub-sections briefly describe each module of software connected to the units described earlier.



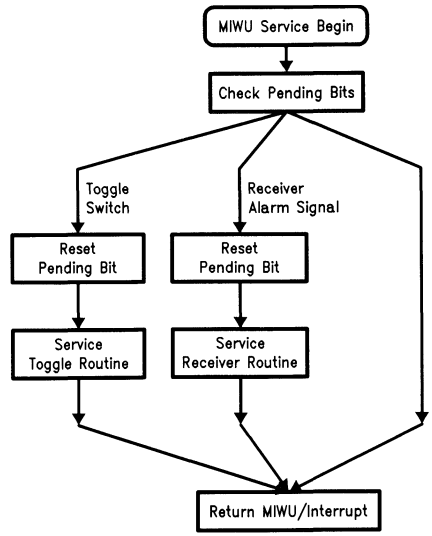
TL/DD/10607–3

FIGURE 3. ASM System Program Flow



TL/DD/10607–4

FIGURE 4. Interrupt Service Routines Flow



TL/DD/10607–5

FIGURE 5. Multi-Input Wakeup/Interrupt Service Routines

Initialization Routine

The Initialization Routine loads the Data Memory locations being used in the program with default values and initializes the various control and configuration registers. It also brings up the display on the Terminal Unit and the LED Display Unit.

Time Keeping Routine

The Time Keeping Routine is the most important routine and is executed irrespective of the other modules being executed. The program uses the IDLE Timer T0 for this purpose. The IDLE Timer is a 16-bit timer and runs continuously at a fixed rate of the instruction cycle clock. The IDLE Timer counter is not memory mapped and consequently, the user cannot read or write to it. The toggling of the twelfth bit of the IDLE counter can be programmed to generate an interrupt. This interrupt is generated every 4 ms at the maximum instruction cycle clock rate of 1 MHz. The software uses this interrupt to update counters in Data Memory for time keeping. The Time Keeping routine then sets a flag to update the display which is then used by the main program.

LED Display Routine

The COP888CG uses the MICROWIRE/PLUS to interface with NSM4000 LED Display with Driver. The time and date information is displayed on the 4-digit LED display. The user is provided with a toggle switch connected to MIWU/Interrupt feature of the COP888CG to toggle the display between Hours Minutes and Seconds 1/80th of Seconds. The toggle switch is connected to L port pin 5. Upon receipt of the MIWU/Interrupt of L port pin 5 this routine toggles the display. This routine upon receipt of the date display request through the Keypad Unit responds by switching the LED Display to show the date. The toggle switch could be used to change the display back to time. However, the display changes to time after a minute by default.

Keypad Scan Routine

This module scans the 4 x 4 matrix keyboard connected to Port D (D1–D4) as rows and to Port I (I0–I3) as columns. Thus each key in the matrix is associated with one Port D line and one Port I line. Each row in the matrix is addressed in sequence and the key closure is sensed by testing the Port I lines. The moment one key closure is detected the program jumps to load the debounce counter. The keypad scan is stopped at that particular row and the program returns to its main flow. The keypad is again scanned and when the debounce counter is decremented. When the debounce counter is zero the key pressed is accepted and decoded. The versatility of the COP888 family of instructions set allows decoding the key pressed with one instruction. The Port D (lines D1–D4) and Port I (lines I0–I3) in conjunction form an eight bit number that is unique to each key. The JID (Jump Indirect) instruction uses the contents of the accumulator to point to the indirect vector table of program address. The accumulator contents are transferred to the program counter (lower 8 bits). The data accessed from the program memory location addressed by program counter is transferred to the program counter (lower 8 bits). The JID instruction is a single-byte, three cycle instruction and provides an efficient way to decode and branch to service the appropriate routine based upon the key pressed.

The Keypad is used to set the time and data information after power up and can also be used to program the phone number to be dialed by the Auto-Dialing Unit.

Non-Volatile Data Storage Routine

The COP888CG interfaces with NM93C06A in the ASM System to store the real time data of the events monitored and also the telephone number to be dialed by the Auto-Dialer Unit. This routine is executed whenever the Receiver Unit detects a signal and the ASM System is not disarmed within 10 seconds of detection of the signal or when the

Display Terminal Unit programs the telephone number to be dialed. The Keypad can also be used to program the phone number to be dialed by the Auto-Dialer Unit. The Terminal Unit can request for the history of events, during which the COP888CG reads the NM93C06A. Please refer to the application note on MICROWIRE/PLUS for details regarding the interface between COP888CG and NMC9306.

Display Terminal Interface Routine

The Display Terminal as previously mentioned interfaces with the COP888CG through the full-duplex, double buffered UART. The terminal is used to display the history of events, real time, and sequence of operations upon detection of signal by the Receiver Unit.

The request for display of events and programming the phone number interrupts the COP888CG. However, the Time Keeping Routine updates the LED display and terminal with real time periodically, except when the COP888CG is servicing the Receiver Unit.

The operation mode of the UART may be selected in conjunction with both a prescaler and baud rate register. Character data lengths of seven, eight or nine bits are program selectable, in conjunction with a start bit, an optional parity bit, and stop bits of 7/8, 1, 1 and 7/8, or 2. The UART also contains a full set of error detection circuitry and a diagnostic test capability, as well as an ATTENTION mode to facilitate networking with other processors.

Please refer to the Users Manual or Microcontroller Databook for details.

In the ASM System the COP888CG interfaces with the V550 terminal at 2400 baud, 8 data bits, 1 Stop bit, no parity. The receiver buffer full and transmit buffer empty generates an interrupt. The Port L (pins L1, L2, L3) are used for the UART interface as CKX (clock), TDX (transmit) and RDX (receive), respectively.

The display terminal is used to display time both in analog and digital form. The V550 allows interfacing both in alphanumeric and graphic modes with separate memory for each of the modes. The COP888CG is programmed to send out the ASCII ESC sequence required to generate the graphics on the screen.

Auto-Dialing Routine

This routine is responsible for dialing the number in the event of an emergency. The COP888CG interfaces with TP5088, which in turn interfaces with TP5700A. The COP888CG activates the relay that keeps the telephone line on-hook to the off-hook position. After this it times out to get the dial tone. After a fixed amount of time, the digit to be dialed is sent out on the D port, lines D1–D4, to TP5088 along with the Chip Select. The TP5088 generates the DTMF signal for the digit. The COP888CG takes care of the timing required between two digits and also the on-time of the DTMF signal for each digit. The output of the DTMF signal goes to the TP5700A which interfaces with the Tip and Ring of the telephone lines. The TP5700A receives the signal from the telephone lines and LM567 along with the associated circuitry is used to detect whether the required frequency signal was sent by the unit responding to the telephone. The output of the LM567 is connected to Port I pin 5.

The Receiver Routine polls the Port I pin 5 periodically to check for response from the unit dialed by the Auto-Dialer Unit.

Receiver Routine

This is the main interrupt service routine of the ASM System. The Receiver Unit interfaces with the COP888CG through the L port pin 4. Upon receipt of the signal from the Sensors and Transmitter Unit the Receiver Unit activates a relay which causes a MIWU/Interrupt. The interrupt service routine then waits for 10 seconds before reacting to the signal. This time is allowed to disarm the Security/Monitoring System. The Time Keeping Routine is used to caculate the delay and if the user disarms the System by toggling a switch the signal is ignored. Otherwide the Non-Volatile Storage Routine is executed to read the telephone number and this information is passed on to the Auto-Dialer Unit. The Auto-Dialer Unit dials the number and looks for a response over the telephone line. If however, there is no response, the Receiver Routine times out after a minute and tries the same number again. The number of trials can be modified in software and the time out period can also be changed. In the ASM System the number of trials is two. With slight modification the Auto-Dialer Unit can be made to dial a different number during the second attempt. The real time and date of occurrence of the event is stored in the NMC9306 along with the outcome of the telephone call. This routine keeps track of the non-volatile memory capacity and if it overflows, it prompts the user on the terminal of the same. The user is given the choice to overwrite the non-volatile memory or replace the device.

USING THE ASM SYSTEM

The ASM System upon installation and initial power-up has some preliminary steps to be performed. The time and date should be set, the phone number to be dialed by the Auto-Dialer Unit should be programmed. The toggle switch could be used to toggle the display between Hours-Minutes and Seconds-1/80th of Seconds.

Setting Time and Date

The steps involved in setting the time and date are:

1. Press key A on the keypad. The LED display flashes.
2. Set the desired time (Hours and Minutes) using the keypad.
3. The LED display and the Terminal Screen displays the time set.
4. Press key C on the keypad. The display toggles and displays the date.
5. Press key A on the keypad. The LED display begins to flash.
6. Set the date (month and day) using the keypad.
7. The LED display now shows the date set.
8. The LED display could be toggled to show the time using the toggle switch. However, the system after one minute will default to display time.

Programming the Phone Number

The phone number to be dialed could be programmed in two ways, i.e., using the terminal or the keypad. Using the terminal, the steps to be performed are:

1. Press CNTRL B on the terminal keyboard. The COP888CG sends a carriage return to terminal.
2. Press CNTRL D on the terminal keyboard. Then type the number to be dialed. At the end press CNTRL C to end programming.

Using the keypad, perform the following steps:

1. Press "*" key on the keypad.
2. Press the digits to be dialed.

3. Press "#" key on the keypad to end programming the number.

The ASM System is now ready to start monitoring. Upon receipt of the alarm signal from the Receiving Unit the ASM System will dial the number programmed. In order to display the history of events on the terminal screen press CNTRL S from the terminal keyboard.

CONCLUSIONS

The architecture, features and flexibility of the COP800 family of microcontrollers makes it cost-effective as the workhorse of any system by eliminating external components from the circuit. This approach not only reduces the system cost and development time, but also increases the flexibility and market life of the product.

The Automated Security/Monitoring System implemented using the COP888CG illustrates a single chip system solution. The application also illustrates interfacing the COP888CG to a number of specialized peripherals using an absolute minimum number of I/O lines. The ASM System approximately uses 3k bytes of program memory (ROM) space and demonstrates an efficient method of handling multi-sourced interrupts.

Sound Effects for the COP800 Family

National Semiconductor
Application Note 663
Jerry Leventer

This application note describes the creation of sound effects using National Semiconductor's COP800 family of microcontrollers. The following applications are described in detail:

1. Whistle
2. White Noise
3. Explosion
4. Bomb
5. Laser Gun

These applications were developed on a COP820C using a 20 MHz crystal and a 1 μs instruction cycle time. By making the appropriate changes to control registers within the routines, slower clock speeds may be used. Program flow diagrams and complete source codes are included in this document.

I. WHISTLE

The whistle routine utilizes the timer underflow interrupt and employs the TIO function on pin G3. Each timer underflow causes the TIO pin to toggle. This creates a tone whose frequency remains constant as long as the timer autoreload register value remains unchanged. In order to create a desending or ascending whistle tone, the autoreload register value is increased or decreased after every thirty-two timer interrupts (FCNTR register is used to count the interrupts). When the maximum or minimum frequency has been reached, the autoreload value must be reinitialized so that the whistle frequency does not exceed the desired range.

II. WHITE NOISE

White noise is generated by using a random number generating algorithm called a RING COUNTER. One random number is extracted periodically and placed into the MICROWIRE/PLUS™ serial shift register. These bits are shifted onto the serial output (SO) pin which is wired to a transistor amplifier that drives a speaker. The serial input (SI) and serial output (SO) pins must be tied together.

The RING COUNTER is a pseudo-random number generator which operates on the principle of a linear feedback shift register (see *Figure 1*). This shift register is not to be confused with the MICROWIRE/PLUS serial shift register. Rather it is created using two bytes of data memory (RAM), and the carry flag. Each bit is called a "stage" with the carry flag being "stage 1" and bit 0 of the two byte data register being "stage 17". Using a seventeen stage shift register results in a clean tone with little distortion.

Implementation of the ring counter shift register is accomplished by a rotate right with carry instruction (RRC A). The linear feedback function is accomplished using an "exclusive or" on stages fourteen and seventeen. This particular choice of feedback stages results in a complete cycle of bit combinations, ($2^{17} - 1$), as long as the loop does not begin with zero in the RNGVAL register.

The "exclusive or" function is not explicit in that the XOR instruction is not used. Rather, stages seventeen and fourteen are tested in software using the principle that if only one of them is set then the result is a logic one, otherwise the result is logic zero. It turns out that since the rotate occurs prior to the test, the actual bits tested are the carry flag (stage 1) and bit 2 (stage 15).

A short example using four bits can be used to demonstrate how the ring counter works (see *Figure 2*). If you perform the "exclusive or" on stages three and four, then a complete cycle results. If instead, you use stages two and four, two cycles of six and one cycle of three results depending on the bit combination you begin with.

III. EXPLOSION

The explosion sound effect is generated by manipulating the white noise algorithm to begin with a high pitch and progress to a lower pitch. This is done by altering the rate (contained in the register LUPREG) at which the random numbers are extracted from the ring counter before being placed into the MICROWIRE/PLUS serial shift register (SIOR). If for example LUPREG initially contains the value 4, the white noise will be at a high pitch. By incrementing this number after every ten timer interrupts (using the register TCNTR) the white noise pitch will be reduced. Several other registers are used to provide control of strategic portions of sound within the routine. First and last tones are controlled with FIRSTR and LASTR. The value in EXITR is used to control the overall length of the explosion and the length of each tone is controlled by the register TCNTR. To vary the white noise pitch, the register LUPCNT is used. The value in LUPCNT is incremented each time the pitch of the white noise is decreased within the timer interrupt routine. Prior to entering the ring count loop, LUPCNT is loaded into LUPREG. The serial input (SI) pin must be tied to the serial output (SO) pin.

IV. BOMB

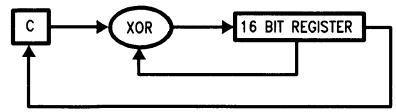
The bomb sound effect combines the descending whistle with an explosion at the end. The TIMER I/O (TIO) and serial input (SI) pins must be tied to the serial output (SO) pin. The explosion portion of this routine was altered slightly in that the first tone control register (FIRSTR) was removed. The first initialization of TCNTR, the tone control register, provides a means to control the first tone length. Subsequent tones are controlled (at label NF2 in the timer interrupt routine) where TCNTR is reinitialized. Both versions were retained for comparison and in the event that greater control of the first tone is needed.

V. LASER GUN

The laser gun sound effect combines the output from the white noise routine and the COP800 timer I/O (TIO) pin (tie TIO to SO). The SI pin is not tied to SO in this application and the ring counter uses only nine stages instead of seventeen.

The registers used for program control are EXITR, TCNTR, and the TIMER. By adjusting the value in EXITR the duration of the laser "shot" can be shortened or lengthened. (A value larger than 03F hex may create problems.) By adjusting the TIMER values (TVALO, TVALHI) and the tone counter (TCNTR) value, interesting variations in the laser sound can be attained.

NOTE: This note applies to all routines that use both the timer interrupt and the ring counter. In order to return to the main program from which the subroutine was called, the stack pointer must be manually restored during the timer interrupt before executing the return (RET) instruction. The reason for this is that the timer interrupt is two levels below the main program. A simple return statement will only serve to return to the ring counter routine from the point at which the timer interrupt occurred. By adding two to the stack pointer (SP + 2), the return statement will force the address of the instruction following the JSR in MAIN into the program counter (PC) from which point execution will continue.



TL/DD/10716–1

FIGURE 1. 17 Stage Ring Counter

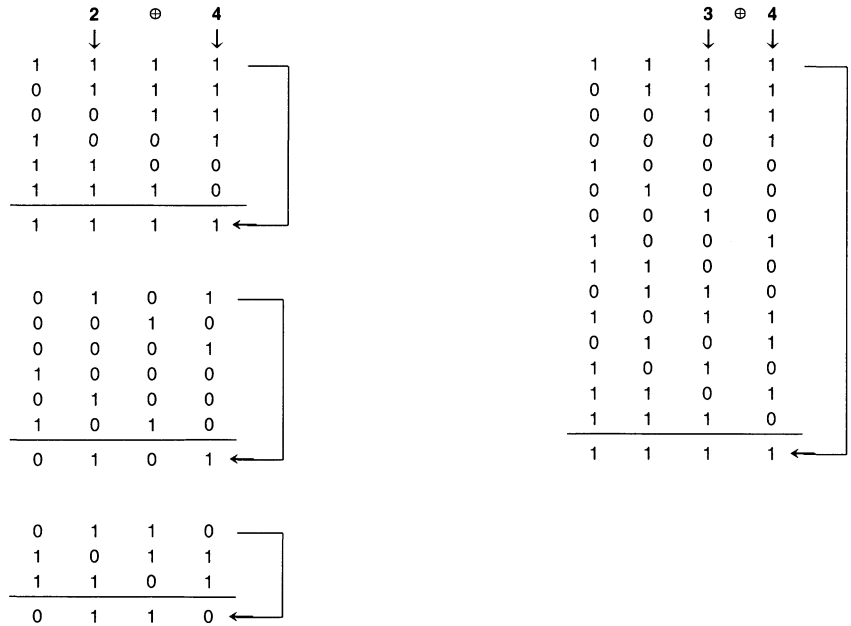
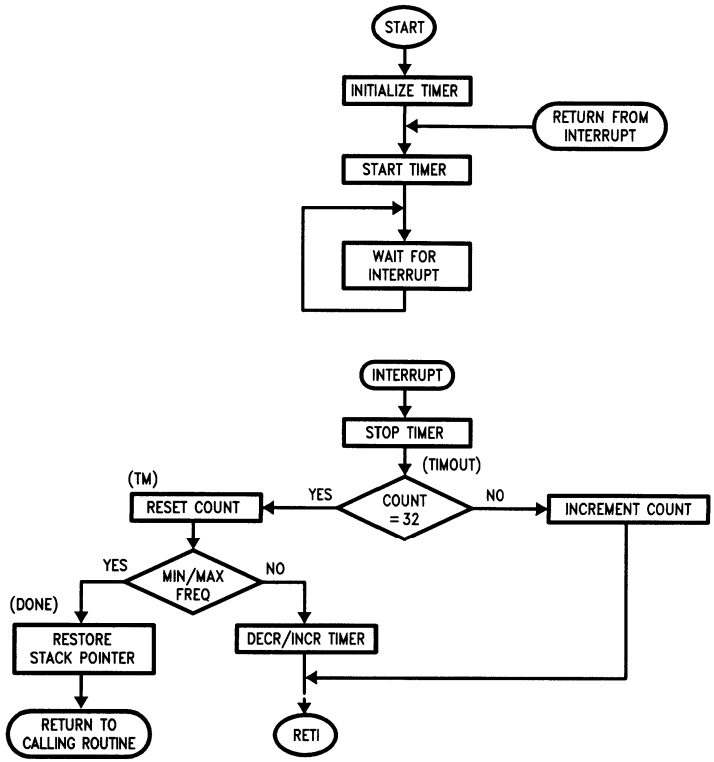


FIGURE 2. Example Showing Possible Cycles from a 4 Stage Ring Counter

Whistle Flow Diagram



TL/DD/10716–2

Descending Whistle

```
 1                      ;
 2                      ;
 3                      ; TIMER INTERRUPT IS USED.
 4                      ; OUTPUT ON TIMER I/O (TIO) PIN.
 5                      ; USE 20 MHz XTAL, 1 μs INSTR CYCLE FOR THIS DEMO.
 6                      ;
 7                      ; WRITTEN BY: JERRY LEVENTER
 8                      ; DATE:       OCTOBER 4, 1989
 9                      ;
10                             .TITLE WHISTLE1
11                             .CHIP   820
12                      ;
13       00D5                  PORTGC  =   0D5          ; PORT G CONFIGURATION
14       00E9                  SIOR    =   0E9          ; SIO SHIFT REGISTER
15       00EA                  TMRL0   =   0EA          ; TIMER LOW BYTE
16       00EB                  TMRHI   =   0EB          ; TIMER HIGH BYTE
17       00EC                  TAUL0   =   0EC          ; TIMER REGISTER LOW BYTE
18       00ED                  TAUHI   =   0ED          ; TIMER REGISTER HIGH BYTE
19       00EE                  CNTRL   =   0EE          ; CONTROL REGISTER
20       00EF                  PSW     =   0EF          ; PSW REGISTER
21       0004                  TRUN    =   4
22       0005                  TPND    =   5
23       0002                  BUSY    =   2
24       0000                  GIE     =   0
25                      ;
26                      ; ****  SPECIAL REGISTERS AND CONSTANTS ****
27                      ;
28       002F                  WSL0    =   02F          ; TIMER VALUES
29       0000                  WSLHI   =   000
30       00F0                  FCNTR   =   0F0          ; FREQUENCY COUNT REGISTER
31       0000                  FCNT    =   000
32       00FF                  MINFREQ =   0FF          ; MIN FREQUENCY CONSTANT
33                      ;
34                      ; *********************************
35                      ; **** BEGIN DEMO PROGRAM HERE ****
36                      ; *********************************
37                      ;
38 0000  DD2F           MAIN:  LD      SP,#02F          ; DEFAULT INITIALIZATION OF SP
38 0002  3005                  JSR     WHISTLE          ; ***CALLING ROUTINE FOR DEMO***
40 0004  FF                    JP      .
41 0005  BCD508         WHISTLE:LD     PORTGC,#008      ; TIO PIN (G3) AS OUTPUT
42 0008  BCEEA2                LD      CNTRL,#0A2       ; PWM WITH TIO TOGGLE, 8Tc
43 000B  BCEA2F                LD      TMRL0,#WSL0      ; WHISTLE VALUE FOR TIMER
44 000E  BCEB00                LD      TMRHI,#WSLHI
45 0011  BCEC2F                LD      TAUL0,#WSL0
46 0014  BCED00                LD      TAUHI,#WSLHI
47 0017  D000                  LD      FCNTR,#FCNT      ; INIT FREQ COUNT
48 0019  BCEF11         LUP:   LD      PSW,#011         ; ENTI, GIE = 1, TPND = 0
49 001C  BDEE7C                SBIT    TRUN,CNTRL       ; START TIMER
50 001F  FF                    JP      .                ; SELF LOOP TIL TIMER INTERRUPT
51 0020  F8                    JP      LUP              ; RUN TIL LAST HISTLE FREQ
52                      ;
53                      ; **** INTERRUPT ROUTINE ****
54                      ;
55       00FF                  .=0FF
56 00FF  BDEF75                IFBIT   TPND, PSW        ; TEST TIMER PENDING FLAG
57 0102  01                    JP      TIMOUT
58 0103  FF                    JP      .                ; ERROR
```

Descending Whistle (Continued)

```
59 0104  BDEE6C     TIMOUT: RBIT    TRUN,CNTRL        ; STOP THE TIMER
60 0107  BDF075             IFBIT   5,FCNTR           ; COUNT CYCLES
61 010A  06                 JP      TM
62 010B  9DF0               LD      A,FCNTR           ; INCREMENT COUNT
63 010D  8A                 INC     A
64 010E  9CF0               X       A,FCNTR
65 0110  8D                 RETSK
66 0111  D000       TM:     LD      FCNTR,#FCNT       ; RESET COUNT
67 0113  DEEC               LD      B,#TAUL0
68 0115  AE                 LD      A,[B]             ; CHANGE FREQUENCY
69 0116  92FF               IFEQ    A,#MINFREQ        ; TIMER = MIN FREQ?
70 0118  03                 JP      DONE              ; YES
71 0119  8A                 INC     A
72 011A  A6                 X       A,[B]             ; STORE FREQ IN AUTO RELOAD
73 011B  8D                 RETSK
74 011C  9DFD       DONE:   LD      A,SP              ; *** RESTORE STACK POINTER ***
75 011E  9402               ADD     A,#002            ; *** AND RETURN TO CALLING ***
76 0120  9CFD               X       A,SP              ; *** ROUTINE.              ***
77 0122  8E                 RET
78                          .END
```

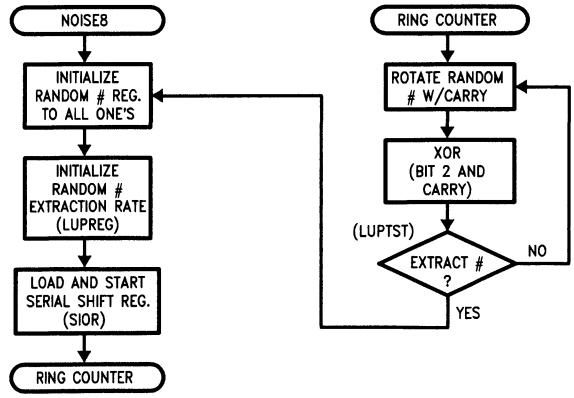
Ascending Whistle

```
 1                    ;
 2                    ;
 3                    ; OUTPUT ON TIMER I/O (TIO) PIN.
 4                    ; USES TIMER INTERRUT.
 5                    ; USE 20 MHz XTAL, 1 μs INSTR CYCLE FOR THIS DEMO.
 6                    ;
 7                    ; WRITTEN BY: JERRY LEVENTER
 8                    ; DATE:       OCTOBER 4, 1989
 9                    ;
10                    ;
11                          .TITLE WHISTLE2
12                          .CHIP   820
13                    ;
14       00D5               PORTGC  =   0D5          ; PORT G CONFIGURATION
15       00EA               TMRLO   =   0EA          ; TIMER LOW BYTE
16       00EB               TMRHI   =   0EB          ; TIMER HIGH BYTE
17       00EC               TAULO   =   0EC          ; TIMER REGISTER LOW BYTE
18       00ED               TAUHI   =   0ED          ; TIMER REGISTER HIGH BYTE
19       00EE               CNTRL   =   0EE          ; CONTROL REGISTER
20       00EF               PSW     =   0EF          ; PSW REGISTER
21       0004               TRUN    =   4
22       0005               TPND    =   5
23       0002               BUSY    =   2
24       0000               GIE     =   0
25                    ;
26                    ; **** SPECIAL REGISTERS AND CONSTANTS ****
27                    ;
28       00FF               WSLO    =   0FF          ; TIMER VALUES
29       0001               WSLHI   =   001
30       000A               MAXFREQ =   00A          ; LAST FREQUENCY CONSTANT
31       00F0               FCNTR   =   0F0          ; TIMER COUNT REGISTER
32       0010               FCNT    =   010          ; COUNTER CONSTANT
33                    ;
34                    ; *********************************
35                    ; ****    BEGIN PROGRAM HERE    ****
36                    ; ***************************** ****
37                    ;
38 0000  DD2F         MAIN:  LD      SP,#02F          ; DEFAULT INITIALIZATION OF SP
39 0002  3005                JSR     WHISTLE2         ; *** CALLING ROUTINE FOR DEMO ***
40 0004  FF                  JP      .
41                    WHISTLE2:
42 0005  BCD508              LD      PORTGC,#008      ; TIO PIN (G3) AS OUTPUT
43 0008  BCEEA0              LD      CNTRL,#0A0       ; PWM WITH TIO TOGGLE,
44 000B  BCEAFF              LD      TMRLO,#WSLO      ; WHISTLE VALUE FOR TIMER
45 000E  BCEB01              LD      TMRHI,#WSLHI
46 0011  BCECFF              LD      TAULO,#WSLO
47 0014  BCED01              LD      TAUHI,#WSLHI
48 0017  D010                LD      FCNTR,#FCNT      ; INITIALIZE COUNTER
49 0019  BCEF11       LUP:   LD      PSW,#011         ; ENTI, GIE = 1, TPND = 0
50 001C  BDEE7C              SBIT    TRUN,CNTRL       ; START TIMER
51 001F  FF                  JP      .                ; SELF LOOP UNTIL TIMER
52 0020  F8                  JP      LUP              ; INTERRUPT
```

Ascending Whistle (Continued)

```
53                      ;
54                      ; **** INTERRUPT ROUTINE ****
55                      ;
56          00FF                .=0FF
57  00FF    BDEF75              IFBIT   TPND,PSW        ; TEST TIMER PENDING FLAG
58  0102    01                  JP      TIMOUT
59  0103    FF                  JP      .
60  0104    BDEE6C      TIMOUT: RBIT    TRUN,CNTRL      ; STOP THE TIMER
61  0107    BDF075              IFBIT   5,FCNTR         ; FREQUENCY TIMED OUT?
62  010A    06                  JP      TM              ; YES, CHANGE FREQUENCY
63  010B    9DF0                LD      A,FCNTR         ; NO, KEEP GOING
64  010D    8A                  INC     A               ; INCREMENT COUNT
65  010E    9CF0                X       A,FCNTR
66  0110    8D                  RETSK                   ; RETURN
67  0111    D010        TM:     LD      FCNTR,#FCNT     ; RESET COUNTER
68  0113    9DEC                LD      A,TAUL0         ; CHANGE FREQUENCY
69  0115    920A                IFEQ    A,#MAXFREQ      ; TIMER = MAX FREQUENCY ?
70  0117    05                  JP      DONE            ; YES
71  0118    94FF                ADD     A,#0FF          ; INCREMENT FREQUENCY
72  011A    9CEC                X       A,TAUL0         ; STORE FREQ IN AUTO RELOAD
73  011C    8D                  RETSK
74  011D    9DFD        DONE:   LD      A,SP            ; *** RESTORE STACK POINTER ***
75  011F    9402                ADD     A,#002          ; *** AND RETURN TO CALLING ***
76  0121    9CFD                X       A,SP            ; *** ROUTINE.              ***
77  0123    8E                  RET
78                              .END
```

White Noise



TL/DD/10716–3

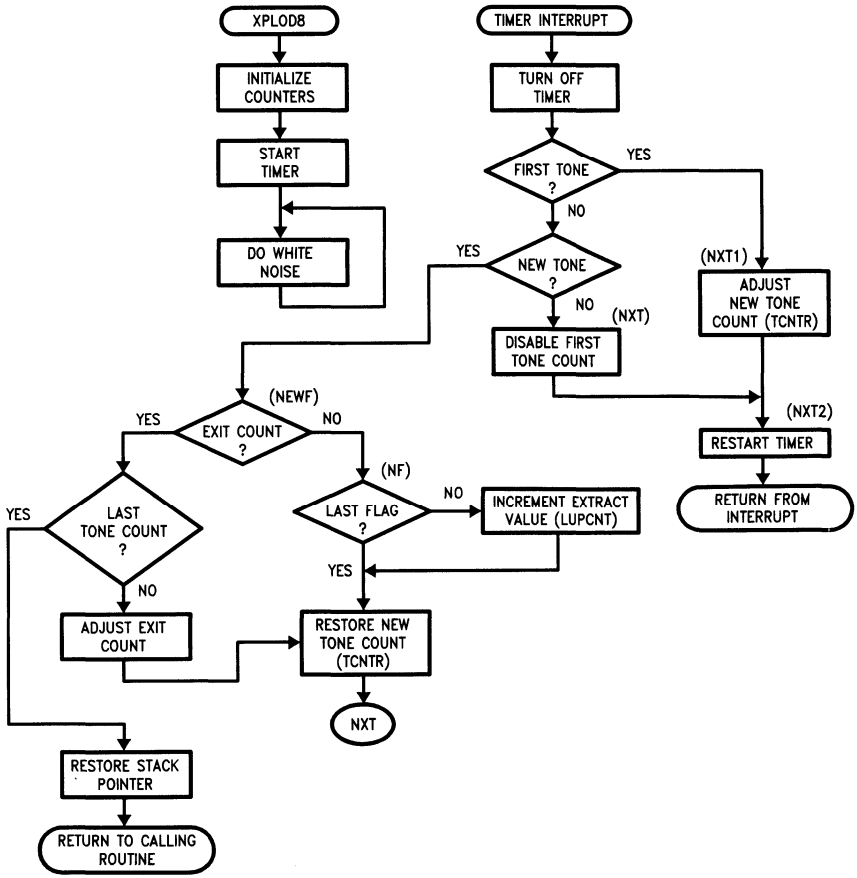
White Noise (Continued)

```
1                       ;
2                       ;
3                       ;
4                       ;
5                       ; TIE SERIAL INPUT (SI)PIN TO SERIAL OUTPUT (SO) PIN.
6                       ; OUTPUT IS ON THE SERIAL OUTPUT (SO) PIN.
7                       ; NO INTERRUPT IS USED.
8                       ; USE 20 MHz XTAL, 1 µs INSTR CYCLE FOR THIS DEMO.
9                       ;
10                      ; WRITTEN BY: JERRY LEVENTER
11                      ; DATE:       OCTOBER 4, 1989
12                      ;
13                              .TITLE NOISE8
14                              .CHIP   820
15                      ;
16      00D5                    PORTGC  =   0D5         ; PORT G CONFIGURATION
17      00E9                    SIOR    =   0E9         ; SERIAL SHIFT REGISTER
18      00EA                    TMRLO   =   0EA         ; TIMER LOW BYTE
19      00EB                    TMRHI   =   0EB         ; TIMER HIGH BYTE
20      00EC                    TAULO   =   0EC         ; TIMER REGISTER LOW BYTE
21      00ED                    TAUHI   =   0ED         ; TIMER REGISTER HIGH BYTE
22      00EE                    CNTRL   =   0EE         ; CONTROL REGISTER
23      00EF                    PSW     =   0EF         ; PSW REGISTER
24      0002                    BUSY    =   2           ; BUSY BIT
25                      ;
26                      ; ****  SPECIAL REGISTERS AND CONSTANTS ****
27                      ;
28      0002                    RNGVAL  =   002         ; RANDOM NUMBER LOCATION
29      00FF                    LUPREG  =   0FF         ; EXTRACTION RATE REGISTER
30      0000                    FLAG    =   000         ; RANDOM NUMBER BYTE FLAG
31      0004                    COUNT   =   4           ; EXTRACTION RATE CONSTANT
32                      ;
33                      ; *********************************
34                      ; ****    BEGIN PROGRAM HERE       ****
35                      ; ****************************    ****
36                      ;
37 0000 DD2F                    LD      SP,#02F         ; DEFAULT INITIALIZATION OF SP
38 0002 BCD530          NOISE:  LD      PORTGC,#030     ; SO AND SK AS OUTPUTS
39 0005 BCEE8B                  LD      CNTRL,#08B      ; SK = DIV BY 8, TIMER RELOAD
40 0008 A1                      SC                      ; INIT STAGE 1
41 0009 5D                      LD      B,#RNGVAL       ; POINT TO RANDOM # LOCATION
42 000A 9AFF                    LD      [B+],#0FF       ; INIT RING VAL TO ONE'S
43 000C 9EFF                    LD      [B],#0FF        ; B POINTS TO UPPER BYTE
44 000E 9CE9            SHIFT:  X       A,SIOR          ; PLACE # IN SIOR
45 0010 BDEF7A                  SBIT    BUSY,PSW        ; START SHIFTING
46 0013 DF04                    LD      LUPREG,#004     ; RESTORE EXTRACTION COUNT
47                      ;
```

White Noise (Continued)

```
48                  ; ********************************************
49                  ; RING COUNTER (17 STAGE)
50                  ; THIS IS A SEVENTEEN STAGE RING COUNTER (LINEAR
51                  ; FEEDBACK SHIFT REGISTER) WITH THE RRC COMMAND.
52                  ; THE COUNTER'S 14TH AND 17TH STAGES THROUGH AN
53                  ; EXCLUSIVE-OR SERVE AS THE FEEDBACK FUNCTION.
54                  ; THIS 14, 17 RING COUNTER BREAKS DOWN INTO
55                  ; 1 CYCLE OF [(2 ** 17) - 1] COUNTS. SINCE THE EXCLUSIVE OR
56                  ; OCCURS AFTER THE ROTATE, IT IS THE 15TH AND CARRY
57                  ; STAGES THAT ARE XOR'D (BIT 2 AND CARRY).
58                  ;
59                  ;                       STAGE
60                  ;                     ---------------
61                  ; BEFORE ROTATE:       14   17
62                  ; AFTER ROTATE:        15   CARRY
63                  ;
64                  ; CARRY BIT = STAGE ONE
65                  ; LOW ORDER BIT = STAGE 17
66                  ; ********************************************
67                  ;
68 0015  AE        RING:   LD      A,[B]           ; GET RANDOM #
69 0016  B0                RRC     A               ; ROTATE UPPER BYTE
70 0017  A3                X       A,[B-]
71 0018  AE                LD      A,[B]
72 0019  B0                RRC     A               ; ROTATE LOWER BYTE
73 001A  A6                X       A,[B]
74 001B  9804              LD      A,#004          ; PERFORM XOR
75 001D  85                AND     A,[B]
76 001E  9200              IFEQ    A,#000
77 0020  05                JP      LUPTST
78 0021  88                IFC
79 0022  02                JP      RC
80 0023  A1                SC
81 0024  01                JP      LUPTST
82 0025  A0        RC:     RC
83 0026  AA        LUPTST: LD      A,[B+]          ; POINT TO UPPER BYTE
84 0027  CF                DRSZ    LUPREG          ; EXTRACT THIS NUMBER ?
85 0028  EC                JP      RING            ; NO, KEEP ROTATING
86 0029  E4                JP      SHIFT           ; YES, SEND IT
87                         .END
```

Explosion



TL/DD/10716–4

Explosion (Continued)

```
 1                      ;
 2                      ;
 3                      ; TIMER INTERRUPT IS USED.
 4                      ; SI MUST BE TIED TO SO. OUTPUT ON SO.
 5                      ; USE 20 MHz XTAL, 1 µs INSTR CYCLE FOR THIS DEMO.
 6                      ;
 7                      ; WRITTEN BY: JERRY LEVENTER
 8                      ; DATE:       OCTOBER 4, 1989
 9                      ;
10                             .TITLE XPLOD8
11                             .CHIP   820
12                      ;
13          00D5               PORTGC  =   0D5          ; PORT G CONFIGURATION
14          00E9               SIOR    =   0E9          ; SIO SHIFT REGISTER
15          00EA               TMRLO   =   0EA          ; TIMER LOW BYTE
16          00EB               TMRHI   =   0EB          ; TIMER HIGH BYTE
17          00EC               TAULO   =   0EC          ; TIMER REGISTER LOW BYTE
18          00ED               TAUHI   =   0ED          ; TIMER REGISTER HIGH BYTE
19          00EE               CNTRL   =   0EE          ; CONTROL REGISTER
20          00EF               PSW     =   0EF          ; PSW REGISTER
21          0004               TRUN    =   4
22          0005               TPND    =   5
23          0002               BUSY    =   2
24                      ;
25                      ; ****  SPECIAL REGISTERS AND CONSTANTS ****
26                      ;
27                      ;       ANY REGISTER USED FOR THE DRSZ TEST MUST
28                      ;       BE INITIALIZED TO AT LEAST "1".
29                      ;
30          00F5               FIRSTR  =   0F5          ; FIRST TONE CONTROL REGISTER
31          0002                FIRST  =   002          ;  FIRST TONE CONSTANT
32          00F6               LASTR   =   0F6          ; LAST TONE CONTROL REGISTER
33          0002                LAST   =   002          ;  LAST TONE CONSTANT
34          00F7               EXITR   =   0F7          ; ROUTINE DURATION REGISTER
35          0010                EXIT   =   010          ;  EXIT CONSTANT
36          0002               RNGVAL  =   002          ; HOLDS CURRENT RANDOM #
37          00F8               TCNTR   =   0F8          ; TONE DURATION REGISTER
38          000A                TCNT   =   0A           ;  TONE CONSTANT
39          0020                TCNT1  =   020          ;   "FIRST" TONE CONSTANT
40          00F9               LUPREG  =   0F9          ; EXTRACTION RATE REGISTER
41          0004                XTRCT  =   004          ;  EXTRACT CONSTANT
42          00FA               LUPCNT  =   0FA          ; EXTRACTION VARIABLE REGISTER
43          0000               TEMP    =   000          ; LAST TONE FLAG
44          00FF               TVALO   =   0FF          ; TIMER VALUES
45          0010               TVALHI  =   010
46                      ;
47                      ; *********************************
48                      ; ****    BEGIN PROGRAM HERE    ****
49                      ; *********************************
50                      ;
51 0000  DD2F      MAIN:   LD      SP,#02F          ; DEFAULT INITIALIZATION OF SP
52 0002  3005              JSR     XPLOD            ; **** XPLOD CALLING ROUTNE ****
53 0004  FF                JP      .                ; **** SELF LOOP FOR DEMO ****
54 0005  BCD530    XPLOD:  LD      PORTGC,#030
55 0008  BCEE8A            LD      CNTRL,#08A       ; SK = DIV BY 8, PWM ON
56 000B  BCEF11            LD      PSW,#011         ; ENABLE TIMER INTERRUPT
57 000E  BCEAFF            LD      TMRLO,#TVALO     ; INITIALIZE TIMER
58 0011  BCEB10            LD      TMRHI,#TVALHI
59 0014  BCECFF            LD      TAULO,#TVALO
60 0017  BCED10            LD      TAUHI,#TVALHI
```

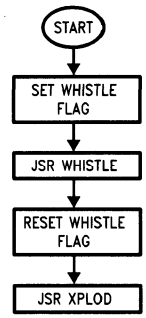
Explosion (Continued)

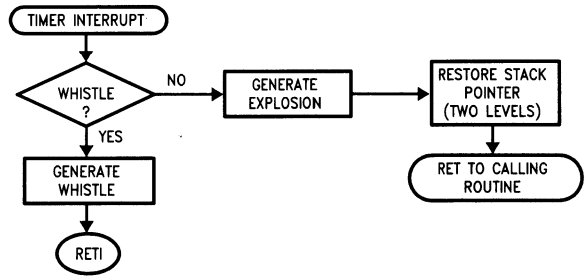
```
 61 001A  D502              LD      FIRSTR,#FIRST    ; LENGTHEN FIRST TONE
 62 001C  D602              LD      LASTR,#LAST      ; LENGTHEN LAST TONE
 63 001E  D710              LD      EXITR,#EXIT      ; INITIALIZE EXIT COUNT
 64 0020  D80A              LD      TCNTR,#TCNT      ; INITIALIZE TONE COUNT
 65 0022  DA04              LD      LUPCNT,#XTRCT    ; INITIALIZE EXTRACTION RATE
 66 0024  BD0068            RBIT    0,TEMP           ; RESET LAST TONE FLAG
 67 0027  BDEE7C            SBIT    TRUN,CNTRL       ; START TIMER
 68 002A  A1        NOISE:  SC                       ; INIT. STAGE 1
 69 002B  5D                LD      B,#RNGVAL        ; POINT TO RANDOM NUMBER
 70 002C  9AFF              LD      [B+],#OFF        ; INIT TO ALL ONE'S
 71 002E  9EFF              LD      [B],#OFF
 72 0030  9CE9      SHIFT:  X       A,SIOR           ; LOAD AND START SIOR
 73 0032  BDEF7A            SBIT    BUSY,PSW
 74 0035  9DFA              LD      A,LUPCNT         ; RESTORE EXTRACTION COUNT
 75 0037  9CF9              X       A,LUPREG
 76                 ;
 77                 ; *********************************
 78                 ; RING COUNTER (17 STAGE)
 79                 ;
 80                 ; THIS IS A SEVENTEEN STAGE RING COUNTER (LINEAR
 81                 ; FEEDBACK SHIFT REGISTER) WITH THE RRC COMMAND.
 82                 ; THE COUNTER'S 14th AND 17th STAGES THROUGH AN
 83                 ; EXCLUSIVE-OR SERVE AS THE FEEDBACK FUNCTION.
 84                 ; THIS 14, 17 RING COUNTER BREAKS DOWN INTO
 85                 ; 1 CYCLE OF [(2 ** 17) - 1] COUNTS. SINCE THE EXCLUSIVE OR
 86                 ; OCCURS AFTER THE ROTATE, IT IS THE 15th AND CARRY
 87                 ; STAGES THAT ARE XOR'D (BIT 2 AND CARRY).
 88                 ;
 89                 ;                    STAGE
 90                 ;                  ---------------
 91                 ; BEFORE ROTATE:     14   17
 92                 ; AFTER ROTATE:      15   CARRY
 93                 ;
 94                 ;  CARRY BIT = STAGE 1
 95                 ;  LOW ORDER BIT OF 16 BIT REGISTER = STAGE 17
 96                 ;  *********************************
 97                 ;
 98 0039  AE        RING:   LD      A,[B]            ; GET RANDOM #
 99 003A  B0                RRC     A                ; ROTATE UPPER BYTE
100 003B  A3                X       A,[B-]
101 003C  AE                LD      A,[B]
102 003D  B0                RRC     A                ; ROTATE LOWER BYTE
103 003E  A6                X       A,[B]
104 003F  9804              LD      A,#004           ; PERFORM XOR
105 0041  85                AND     A,[B]
106 0042  9200              IFEQ    A,#000
107 0044  05                JP      TSLUP
108 0045  88                IFC
109 0046  02                JP      RC
110 0047  A1                SC
111 0048  01                JP      TSTLUP
112 0049  A0        RC:     RC
113 004A  AA        TSTLUP: LD      A,[B+]           ; POINT TO UPPER BYTE
114 004B  C9                DRSZ    LUPREG           ; EXTRACT THIS # ?
115 004C  EC                JP      RING             ; NO, KEEP ROTATING
116 004D  AE                LD      A,[B]            ; YES
117 004E  E1                JP      SHIFT
```

Explosion (Continued)

```
118                     ;
119                     ; **** TIMER INTERRUPT ROUTINE ****
120                     ;
121          00FF       ;       .=      0FF
122 00FF  BDEF75                IFBIT   TPND,PSW        ; TEST TIMER PND FLAG
123 0102  02                    JP      TMOUT
124 0103  2005                  JMP     XPLOD
125 0105  BDEE6C        TMOUT:  RBIT    TRUN,CNTRL      ; STOP TIMER
126 0108  DEFA                  LD      B,#LUPCNT
127 010A  C5                    DRSZ    FIRSTR          ; TEST FOR FIRST TONE
128 010B  213B                  JMP     NXT1            ; AND ADJUST
129 010D  C8                    DRSZ    TCNTR           ; TEST FOR NEW TONE
130 010E  01                    JP      NXT             ; NO
131 010F  0D                    JP      NEWF
132 0110  D501          NXT:    LD      FIRSTR,#1       ; DISABLE FIRST TONE REG
133 0112  BDEF7C        NXT2:   SBIT    4,PSW           ; ENABLE TIMER INTERRUPT
134 0115  BDEF6D                RBIT    5,PSW           ; RESET TPND FLAG
135 0118  5D                    LD      B,#RNGVAL       ; POINT TO RANDOM#
136 0119  BDEE7C                SBIT    TRUN,CNTRL      ; RESTART TIMER
137 011C  8F                    RETI                    ; RETURN
138 011D  C7            NEWF:   DRSZ    EXITR           ; TEST EXIT COUNT
139 011E  10                    JP      NF              ; NO
140 011F  C6                    DRSZ    LASTR           ; ENABLE LAST TONE
141 0120  01                    JP      LST
142 0121  06                    JP      NLST
143 0122  D709          LST:    LD      EXITR,#09       ; SET LAST TONE LENGTH
144 0124  BD0078                SBIT    0,TEMP          ; SET LAST TONE FLAG
145 0127  0F                    JP      NF2
146 0128  9DFD          NLST:   LD      A,SP            ; *** RESTORE STACK POINTER  ***
147 012A  9402                  ADD     A,#002          ; *** FROM TIMER INTERRUPT   ***
148 012C  9CFD                  X       A,SP            ; *** AND RETURN TO MAIN     ***
149 012E  8E                    RET
150 012F  BD0070        NF:     IFBIT   0,TEMP          ; LAST TONE ?
151 0132  04                    JP      NF2             ; YES
152 0133  AE                    LD      A,[B]           ; NEW TONE
153 0134  9404          NF4:    ADD     A,#04           ; INCR EXTRACTION VALUE
154 0136  A6                    X       A,[B]
155 0137  D80A          NF2:    LD      TCNTR,#TCNT     ; REINITIALIZE TONE TIME
156 0139  2110                  JMP     NXT
157 013B  D820          NXT1:   LD      TCNTR,#TCNT1    ; ADJUST FIRST TONE LENGTH
158 013D  2112                  JMP     NXT2
159                             .END
```

Bomb





TL/DD/10716–5

Bomb (Continued)

```
 1                      ;
 2                      ;
 3                      ; THE SERIAL INPUT (SI) AND TIMER I/O (TIO) PINS
 4                      ; MUST BE TIED TO THE SERIAL OUTPUT (SO) PIN.
 5                      ; OUTPUT IS ON SO.
 6                      ; USE 20 MHz XTAL, 1 µs INSTR CYCLE FOR THIS DEMO.
 7                      ;
 8                      ; WRITTEN BY: JERRY LEVENTER
 9                      ; DATE:       OCTOBER 4, 1989
10                      ;
11                      ;
12                             .TITLE BOMB8
13                             .CHIP   820
14                      ;
15      00D5                   PORTGC  =   0D5          ; PORT G CONFIGURATION
16      00E9                   SIOR    =   0E9          ; SIO SHIFT REGISTER
17      00EA                   TMRLO   =   0EA          ; TIMER LOW BYTE
18      00EB                   TMRHI   =   0EB          ; TIMER HIGH BYTE
19      00EC                   TAULO   =   0EC          ; TIMER REGISTER LOW BYTE
20      00ED                   TAUHI   =   0ED          ; TIMER REGISTER HIGH BYTE
21      00EE                   CNTRL   =   0EE          ; CONTROL REGISTER
22      00EF                   PSW     =   0EF          ; PSW REGISTER
23      0004                   TRUN    =   4
24      0005                   TPND    =   5
25      0002                   BUSY    =   2
26      0000                   GIE     =   0
27                      ;
28                      ; ****  EXPLOSION REGISTERS AND CONSTANTS ****
29                      ;
30                      ; SOME OF THE FOLLOWING REGISTERS USE THE DRSZ
31                      ; TEST AND MUST THEREFORE BE INITIALIED TO AT
32                      ; LEAST "1".
33                      ;
34      00F6                   LASTR   =   0F6          ; CONTROL LAST TONE
35      0002                    LAST   =   002          ; LAST TONE CONSTANT
36      0004                    LAST2  =   004          ; EXIT CONSTANT
37      00F7                   EXITR   =   0F7          ; TOTAL TIME TILL EXIT
38      0010                    EXIT   =   010          ; EXIT CONSTANT
39      00F3                   RNGVAL  =   0F3          ; HOLDS CURRENT RING VALUE
40      00F8                   TCNTR   =   0F8          ; TIME FOR EACH TONE FREQ
41      000A                    TCNT   =   0A           ; CONSTANT VALUE
42      00F9                   LUPREG  =   0F9          ; TONE COUNT INSIDE RING
43      00FA                   LUPCNT  =   0FA          ; TONE COUNT OUTSIDE RING (VARIABLE)
44      0000                   FLAG    =   000          ; FLAG REGISTER FOR SUBROUTINES
45                      ;
46      00FF                   TVALO   =   0FF
47      001A                   TVALHI  =   01A
48                      ;
49                      ;**** WHISTLE REGISTERS AND CONSTANTS ****
50                      ;
51      002F                   WSLO    =   02F          ; TIMER VALUES
52      0000                   WSLHI   =   000
53      00FF                   MINFQ   =   0FF          ; FINAL (LOW FREQ) TIMER VALUE
54                      ;
55      00F0                   FCNTR   =   0F0          ; FREQUENCY COUNT REGISTER
56      0000                    FCNT   =   000
```

Bomb (Continued)

```
 57                     ;
 58                     ; ********************************
 59                     MAIN:
 60 0000  DD2F                 LD      SP,#02F           ; DEFAULT INITIALIZATION OF SP
 61 0002  BD0078               SBIT    0,FLAG            ; SET SUBROUTINE FLAG
 62                                                      ; 1 = WHISTLE
 63                                                      ; 0 = EXPLOSION
 64 0005  3157                 JSR     WHISTLE
 65 0007  BD0068       MAIN2:  RBIT    0,FLAG
 66 000A  300D                 JSR     BOMB
 67 000C  FF                   JP      .                 ; *** STOP HERE OR REPEAT ***
 68                     ; ********************************
 69                     ;
 70 000D  BCD530       BOMB:   LD      PORTGC,#030       ; CONFIGURE "SO" AS OUTPUT
 71 0010  BCEE8A               LD      CNTRL,#08A        ; SK = DIV BY 8, PWM ON
 72 0013  BCEF11               LD      PSW,#011          ; ENABLE TIMER INTERRUPT
 73 0016  BCEAFF               LD      TMRLO,#TVALO      ; INITIALIZE TIMER
 74 0019  BCEB1A               LD      TMRHI,#TVALHI
 75 001C  BCECFF               LD      TAULO,#TVALO
 76 001F  BCED1A               LD      TAUHI,#TVALHI
 77 0022  D602                 LD      LASTR,#LAST       ; INITIALIZE LAST TONE FLAG
 78 0024  D710                 LD      EXITR,#EXIT       ; INITIALIZE EXIT COUNT
 79 0026  D80A                 LD      TCNTR,#TCNT       ; INITIALIZE TONE COUNT
 80 0028  DA0A                 LD      LUPCNT,#10        ; INITIALIZE FIRST TONE FREQUENCY
 81 002A  BD0069               RBIT    1,FLAG            ; RESET LAST TONE FLAG BIT
 82                     ;
 83 002D  A1           NOISE:  SC                        ;
 84 002E  DEF3                 LD      B,#RNGVAL         ; POINT TO RING VALUE
 85 0030  9AFF                 LD      [B+],#0FF         ; INIT TO ALL ONE'S
 86 0032  9EFF                 LD      [B],#0FF
 87 0034  BDEE7C               SBIT    TRUN,CNTRL        ; START THE TIMER
 88 0037  BEF6A        SHIFT:  RBIT    BUSY,PSW
 89 003A  9CE9                 X       A,SIOR            ; RANDOM # TO SIO
 90 003C  BDEF7A               SBIT    BUSY,PSW
 91 003F  9DFA                 LD      A,LUPCNT          ; RESTORE EXTRACTION COUNT
 92 0041  9CF9                 X       A,LUPREG
 93                     ;
 94                     ; ********************************
 95                     ; RING COUNTER (17 STAGE)
 96                     ;
 97                     ; THIS IS A SEVENTEEN STAGE RING COUNTER (LINEAR
 98                     ; FEEDBACK SHIFT REGISTER) WITH THE RRC COMMAND.
 99                     ; THE COUNTER'S 14th AND 17th STAGES THROUGH AN
100                     ; EXCLUSIVE-OR SERVE AS THE FEEDBACK FUNCTION.
101                     ; THIS 14, 17 RING COUNTER BREAKS DOWN INTO
102                     ; 1 CYCLE OF [(2 ** 17) - 1] COUNTS. SINCE THE EXCLUSIVE OR
103                     ; OCCURS AFTER THE ROTATE, IT IS THE 15th AND CARRY
104                     ; STAGES THAT ARE XOR'D (BIT 2 AND CARRY).
105                     ;
106                     ; BEFORE ROTATE: 14 17
107                     ; AFTER ROTATE:  15 CARRY
108                     ;
109                     ;  CARRY BIT = STAGE ONE
110                     ;  LOW ORDER BIT = STAGE 17
```

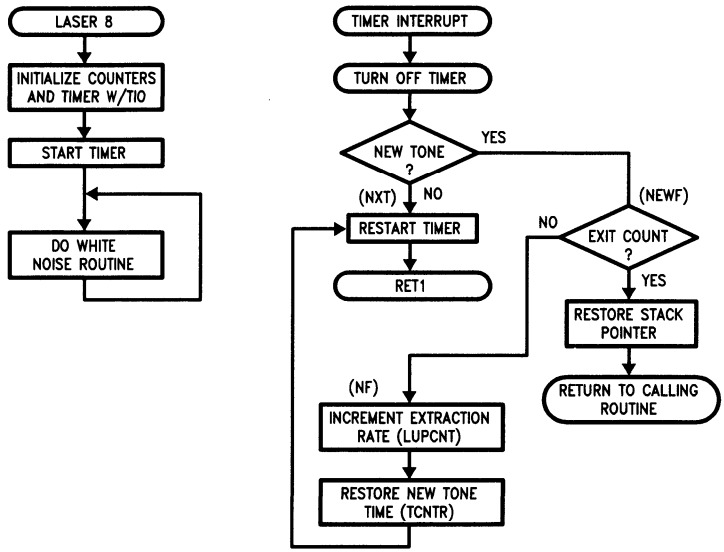
Bomb (Continued)

```
111                     ; ************************************
112 0043  AE        RING: LD      A,[B]             ;  GET RANDOM #
113 0044  B0              RRC     A                 ;  ROTATE UPPER BYTE
114 0045  A3              X       A,[B-]
115 0046  AE              LD      A,[B]
116 0047  B0              RRC     A                 ;  ROTATE LOWER BYTE
117 0048  A2              X       A,[B+]
118 0049  9804            LD      A,#004            ;  PERFORM XOR
119 004B  85              AND     A,[B]
120 004C  9200            IFEQ    A,#000
121 004E  05              JP      TSTLUP
122 004F  88              IFC
123 0050  02              JP      RC
124 0051  A1              SC
125 0052  01              JP      TSTLUP
126 0053  A0        RC:   RC
127 0054  C9        TSLUP: DRSZ    LUPREG            ;  POINT TO UPPER BYTE
128 0055  ED              JP      RING              ;  EXTRACT THIS # ?
129 0056  AE              LD      A,[B]             ;  NO, KEEP ROTATING
130 0057  2037            JMP     SHIFT             ;  YES
131                     ;
132                     ; **** INTERRUPT ROUTINE ****
133                     ;
134       00FF              .=      0FF
135 00FF  BDEF75            IFBIT   TPND,PSW          ;  TEST FOR EXIT
136 0102  01                JP      TMOUT
137 0103  FF                JP      .                 ;  ERROR
138                     ;
139 0104 BDEE6C     TMOUT   RBIT    TRUN,CNTRL        ;  STOP TIMER
140 0107 BD0070             IFBIT   0,FLAG            ;  BRANCH TO ROUTINE
141 010A  213B              JMP     WSINT             ;  SET = WHISTLE, RESET = EXPLOSION
142                     ;
143 010C DEFA               LD      B,#LUPCNT
144 010E C8                 DRSZ    TCNTR             ;  TEST FOR NEW TONE
145 010F 01                 JP      NXT               ;  NO, DON'T INCREMENT LUPCNT
146 0110 0C                 JP      NEWF              ;  YES
147 0111 BDEF7C     NXT:    SBIT    4,PSW             ;  ENABLE TIMER INTRRUPT
148 0114 BDEF6D             RBIT    5,PSW             ;  RESET TIMER PENDING FLAG
149 0117 DEF3               LD      B,#RNGVAL         ;  POINT TO RANDOM #
150 0119 BDEE7C             SBIT    TRUN,CNTRL        ;  RESTART TIMER
151 011C 8F                 RETI                      ;  RETURN TO RING COUNTER
152 011D C7         NEWF:   DRSZ    EXITR             ;  DO LAST TONE ?
153 011E 10                 JP      NF                ;  NO
154 011F  C6                DRSZ    LASTR             ;  IS LAST TONE DONE?
155 0120  01                JP      LST               ;  NO
156 0121  06                JP      NLST              ;  YES, RETURN TO MAIN
157 0122  D704       LST:   LD      EXITR,#LAST2      ;  LENGTHEN THE LAST TONE
158 0124  BD0079            SBIT    1,FLAG            ;  SET LAST TONE FLAG
159 0127  0F                JP      NF2
160 0128  9DFD       NLST:  LD      A,SP              ;  ** RESTORE STACK POINTER **
161 012A  9402              ADD     A,#002            ;  ** AND RETURN TO MAIN    **
162 012C  9CFD              X       A,SP
163 012E  8E                RET
164                     ;
165 012F  BD0071     NF:    IFBIT   1,FLAG            ;  LAST TONE ?
166 0132  04                JP      NF2               ;  YES, DON'T INCREMENT LUPCNT
167 0133  AE                LD      A,[B]             ;  NEW TONE
168 0134  9404              ADD     A,#04             ;  INCR EXTRACT COUNT (LUPCNT)
169 0136  A6                X       A,[B]
170 0137  D80A       NF2:   LD      TCNTR,#TCNT       ;  REINITIALIZE TONE TIME
171 0139  2111              JMP     NXT
```

Bomb (Continued)

```
172                     ; ************************************
173 013B  BDF075        WSINT:      IFBIT   5,FCNTR         ; READY FOR NEW FREQUENCY ?
174 013E  06                        JP      TM              ; YES
175 013F  9DF0                      LD      A,FCNTR         ; NO, INCREMENT COUNT
176 0141  8A                        INC     A
177 0142  9CF0                      X       A,FCNTR
178 0144  8D                        RETSK                   ; NO, RETURN TO WHISTLE
179 0145  D000          TM:         LD      FCNTR,#FCNT     ; RESET NEW FREQUENCY COUNT
180 0147  DEEC                      LD      B,#TAULO        ; POINT TO AUTORELOAD REG
181 0149  AE                        LD      A,[B]           ; CHANGE FREQUENCY
182 014A  92FF                      IFEQ    A,#MINFQ        ; TIMER = MIN FREQ ?
183 014C  03                        JP      DONE
184 014D  8A                        INC     A
185 014E  A6                        X       A,[B]           ; STORE FREQ IN AUTO RELOAD
186 014F  8D                        RETSK
187 0150  9DFD          DONE:       LD      A,SP            ; ** RESTORE STACK POINTER **
188 0152  9402                      ADD     A,#002          ; ** AND RETURN TO MAIN    **
189 0154  9CFD                      X       A,SP
190 0156  8E                        RET
191                     ; ********************************
192 0157  BCD508        WHISTLE:    LD      PORTGC,#008     ; TIO PIN (G3) AS OUTPUT
193 015A  BCEEA2                    LD      CNTRL,#0A2      ; PWM WITH TIO TIGGLE, 8Tc
194 015D  D000                      LD      FCNTR,#FCNT     ; INIT FREQ COUNTER
195 015F  BCEA2F                    LD      TMRLO,#WSLO     ; WHISTLE VALUE FOR TIMER
196 0162  BCEB00                    LD      TMRHI,#WSLHI
197 0165  BCEC2F                    LD      TAULO,#WSLO
198 0168  BCED00                    LD      TAUHI,#WSLHI
199                     ;
200 016B  BCEF11        BEGIN       LD      PSW,#011        ; ENTI, GIE = 1, TPND = 0
201 016E  BDEE7C                    SBIT    TRUN,CNTRL      ; START TIMER
202 0171  FF                        JP      .               ; LOOP UNTIL TIMER INTERRUPT
203 0172  F8                        JP      BEGIN           ; RETURN HERE FROM INTERRUPT
204                                 .END
```

Laser Gun



TL/DD/10716–6

Laser Gun (Continued)

```
 1                    ;
 2                    ; TIMER INTERRUPT IS USED.
 3                    ; THE SERIAL OUTPUT PIN (SO) AND THE TIO PIN MUST BE
 4                    ; TIED TOGETHER.
 5                    ; OUTPUT IS ON SO AND TIO.
 6                    ;
 7                    ; TO ALTER THE DURATION OF THE LASER SHOT CHANGE THE
 8                    ; "EXIT" VALUE, HOWEVER, DO NOT EXCEED 03F HEX.
 9                    ; THE TIMER VALUES (TVALO, TVALHI) COMBINED WITH THE
10                    ; TONE COUNT (TNCTR) CAN BE ADJUSTED TO ACHIEVE A
11                    ; VARIETY OF SOUNDS.
12                    ;
13                    ; USE 20 MHz XTAL, 1 μs INSTR CYCLE TIME FOR THIS DEMO.
14                    ;
15                    ;
16                    ; WRITTEN BY: JERRY LEVENTER
17                    ; DATE:       OCTOBER 4, 1989
18                    ;
19                    ;
20                           .TITLE LASER8
21                           .CHIP   820
22                    ;
23          00D5             PORTGC  =   0D5          ; PORT G CONFIGURATION
24          00E9             SIOR    =   0E9          ; SIO SHIFT REGISTER
25          00EA             TMRLO   =   0EA          ; TIMER LOW BYTE
26          00EB             TMRHI   =   0EB          ; TIMER HIGH BYTE
27          00EC             TAULO   =   0EC          ; TIMER REGISTER LOW BYTE
28          00ED             TAUHI   =   0ED          ; TIMER REGISTER HIGH BYTE
29          00EE             CNTRL   =   0EE          ; CONTROL REGISTER
30          00EF             PSW     =   0EF          ; PSW REGISTER
31          0004             TRUN    =   4
32          0005             TPND    =   5
33          0002             BUSY    =   2
34                    ;
35                    ; ****  SPECIAL REGISTERS AND COUNTERS ****
36                    ; ANY REGISTER THAT IS USED FOR THE DRSZ TEST,
37                    ; MUST BE INITIALIZED TO AT LEAST "1".
38                    ;
39          00F7             EXITR   =   0F7          ; ROUTINE DURATION REGISTER
40          003F              EXIT   =   03F          ;  EXIT CONSTANT
41          0002             RNGVAL  =   002          ; HOLDS CURRENT RANDOM #
42          00F8             TCNTR   =   0F8          ; TONE DURATION REGISTER
43          0020              TCNT   =   020          ;  TONE CONSTANT
44          00F9             LUPREG  =   0F9          ; EXTRACTION RATE REGISTER
45          0003              XTRCT  =   003          ;  EXTRACT CONSTANT
46          00FA             LUPCNT  =   0FA          ; EXTRACTON VARIABLE REGISTER
47          00FF             TVALO   =   0FF          ; TIMER VALUES
48          0000             TVALHI  =   000
49                    ;
50                    ;*******************************
51                    ;****   BEGIN PROGRAM HERE    ****
52                    ;*******************************
53                    ;
54 0000               MAIN:   LD      SP,#02F      DD2F; DEFAULT INITIALIZATION OF SP
55 0002               LUP:    LD      EXITR,#EXIT  D73F; INITIALIZE SHOT DURATION
56 0004                       JSR     LASER8       3018; *** LASER CALLING ROUTINE ***
57 0006                       LD      EXITR,#EXIT  D73F
58 0008                       JSR     LASER8       3018
```

Laser Gun (Continued)

```
 59 000A  D73F              LD      EXITR,#EXIT
 60 000C  3018              JSR     LASER8
 61 000E  D715              LD      EXITR,#015        ; EXIT COUNT CAN BE INITIALIZED
 62 0010  3018              JSR     LASER8            ; INSIDE PROGRAM IF SHOT RATE
 63 0012  D715              LD      EXITR,#015        ; DOES NOT CHANGE.
 64 0014  3018              JSR     LASER8
 65 0016  B8                NOP
 66 0017  EA                JP      LUP               ; **** LOOP FOR DEMO ****
 67                  ;
 68 0018  BCD530    LASER8: LD      PORTGC,#030
 69 001B  BCEEAA            LD      CNTRL,#0AA        ; SK = DIV BY 8, PWM/TIO TIMER
 70 001E  BCEF11            LD      PSW,#011          ; ENABLE TIMER INTERRUPT
 71 0021  BCEAFF            LD      TMRLO,#TVALO      ; INITIALIZE TIMER
 72 0024  BCEB00            LD      TMRHI,#TVALHI
 73 0027  BCECFF            LD      TAULO,#TVALO
 74 002A  BCED00            LD      TAUHI,#TVALHI
 75                  ;      LD      EXITR,#EXIT       ; INITIALIZE EXIT COUNT
 76 002D  D820              LD      TCNTR,#TCNT       ; INITIALIZE TONE COUNT
 77 002F  DA03              LD      LUPCNT,#XTRCT     ; INITIALIZE EXTRACTION RATE
 78 0031  BDEE7C            SBIT    TRUN,CNTRL        ; START TIMER
 79 0034  A1        NOISE:  SC                        ; INIT. STAGE 1
 80 0035  5D                LD      B,#RNGVAL         ; POINT TO RANDOM NUMBER
 81 0036  9EFF              LD      [B],#0FF          ; INIT RANDOM #
 82 0038  9CE9      SHIFT:  X       A,SIOR            ; LOAD AND START SIOR
 83 003A  BDEF7A            SBIT    BUSY,PSW
 84 003D  9DFA              LD      A,LUPCNT          ; RESTORE EXTRACTION COUNT
 85 003F  9CF9              LD      A,LUPREG
 86                  ;
 87                  ; *********************************
 88                  ; RING COUNTER
 89                  ;
 90                  ; THIS IS A NINE STAGE RING COUNTER (LINEAR
 91                  ; FEEDBACK SHIFT REGISTER) WITH THE RRC COMMAND.
 92                  ; THE COUNTER'S 8th AND 9th STAGES, THROUGH AN
 93                  ; EXCLUSIVE-OR SERVE AS THE FEEDBACK FUNCTION.
 94                  ; SINCE THE EXCLUSIVE OR OCCURS AFTER THE ROTATE,
 95                  ; IT IS THE 1st AND 9th STAGES THAT ARE XOR'D,
 96                  ; (THE CARRY FLAG AND BIT 0).
 97                  ;
 98                  ;  CARRY BIT = STAGE 1
 99                  ;  LOW ORDER BIT = STAGE 9
100                  ; *********************************
101 0041  AE        RING:   LD      A,[B]             ; GET RANDOM #
102 0042  B0                RRC     A                 ; ROTATE UPPER BYTE
103 0043  A6                X       A,[B]             ;
104 0044  9800              LD      A,#000            ; PERFORM XOR
105 0046  85                AND     A,[B]
106 0047  9200              IFEQ    A,#000
107 0049  05                JP      TSTLUP
108 004A  88                IFC
109 004B  02                JP      RC
110 004C  A1                SC
111 004D  01                JP      TSTLUP
112 004E  A0        RC:     RC
```

Laser Gun (Continued)

```
113 004F  C9          TSTLUP: DRSZ    LUPREG          ; EXTRACT THIS # ?
114 0050  F0                  JP      RING            ; NO, KEEP ROTATING
115 0051  AE                  LD      A,[B]           ; YES
116 0052  E5                  JP      SHIFT
117                   ;
118                   ; **** TIMER INTERRUPT ROUTINE ****
119                   ;
120       00FF                .=      0FF
121 00FF  BDEF75              IFBIT   TPND,PSW        ; TEST TIMER PND FLAG
122 0102  01                  JP      TMOUT
123 0103  FF                  JP      .               ; ERROR
124                   ;
125 0104  BDEE6C      TMOUT:  RBIT    TRUN,CNTRL      ; STOP TIMER
126 0107  DEFA                LD      B,#LUPCNT
127 0109  C8                  DRSZ    TCNTR           ; TEST FOR NEW TONE
128 010A  01                  JP      NXT             ; NO
129 010B  0B                  JP      NEWF
130 010C  BDEF7C      NXT:    SBIT    4,PSW           ; ENABLE TIMER INTERRUPT
131 010F  BDEF6D              RBIT    5,PSW           ; RESET TPND FLAG
132 0112  5D                  LD      B,#RNGVAL       ; POINT TO RANDOM #
133 0113  BDEE7C              SBIT    TRUN,CNTRL      ; RESTART TIMER
134 0116  8F                  RETI                    ; RETURN
135 0117  C7          NEWF:   DRSZ    EXITR           ; EXIT COUNT = 0 ?
136 0118  07                  JP      NF              ; NO
137 0119  9DFD        NLST:   LD      A,SP            ; *** RESTORE STACK POINTER ***
138 011B  9402                ADD     A,#002          ; *** FROM TIMER INTERRUPT  ***
139 011D  9CFD                X       A,SP            ; *** AND RETURN TO MAIN    ***
140 011F  8E                  RET
141 0120  AE          NF:     LD      A,[B]           ; NEW TONE
142 0121  9404                ADD     A,#04           ; INCR EXTRACTION VALUE
143 0123  A6                  X       A,[B]
144 0124  D820                LD      TCNTR,#TCNT     ; REINITIALIZE TONE TIME
145 0126  E5                  JP      NXT
146                           .END
```

DTMF Generation with a 3.58 MHz Crystal

National Semiconductor
Application Note 666
Verne H. Wilson

DTMF (Dual Tone Multiple Frequency) is associated with digital telephony, and provides two selected output frequencies (one high band, one low band) for a duration of 100 ms. DTMF generation consists of selecting and combining two audio tone frequencies associated with the rows (low band frequency) and columns (high band frequency) of a push-button touch tone telephone keypad.

This application note outlines two different methods of DTMF generation using a COP820C/840C microcontroller clocked with a 3.58 MHz crystal in the divide by 10 mode. This yields an instruction cycle time of 2.79 μs. The application note also provides a low true row/column decoder for the DTMF keyboard.

The first method of DTMF generation provides two PWM (Pulse Width Modulation) outputs on pins G3 and G2 of the G port for 100 ms. These two PWM outputs represent the selected high band and low band frequencies respectively, and must be combined externally with an LM324 op amp or equivalent feed back circuit to produce the DTMF signal.

The second method of DTMF generation uses ROM lookup tables to simulate the two selected DTMF frequencies. These table lookup values for the selected high band and low band frequencies are then combined arithmetically. The high band frequencies contain a higher bias value to compensate for the DTMF requirement that the high band frequency component be 2 dB above the low band frequency component to compensate for losses in transmission. The resultant value from the arithmetic combination of sine wave values is output on L port pins L0 to L5, and must be combined externally with a six input resistor ladder network to produce the DTMF signal. This resultant value is updated every 118 μs. The COP820C/840C timer is used to time out the 100 ms duration of the DTMF. A timer interrupt at the end of the 100 ms is used to terminate the DTMF output. The external ladder network need not contain any active components, unlike the first method of DTMF generation with the two PWM outputs into the LM324 op amp.

The associated COP820C/840C program for the DTMF generation is organized as three subroutines. The first subroutine (KBRDEC) converts the low true column/row input from the DTMF keyboard into the associated DTMF hexadecimal digit. In turn, this hex digit provides the input for the other two subroutines (DTMFGP and DTMFLP), which represent the two different methods of DTMF generation. These three subroutines contain 35, 94, and 301 bytes of COP820C/840C code respectively, including all associated ROM tables. The Program Code/ROM table breakdowns are 19/16, 78/16, and 88/213 bytes respectively.

DTMF KEYBOARD MATRIX

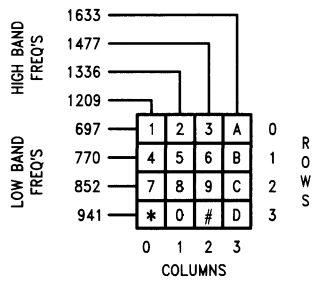
The matrix for selecting the high and low band frequencies associated with each key is shown in *Figure 1*. Each key is uniquely referenced by selecting one of the four low band frequencies associated with the matrix rows, coupled with selecting one of the four high band frequencies associated with the matrix columns. The low band frequencies are 697 Hz, 770 Hz, 852 Hz, and 941 Hz, while the high band frequencies are 1209 Hz, 1336 Hz, 1477 Hz, and 1633 Hz. The DTMF keyboard input decode subroutine assumes that the keyboard is encoded in a low true row/column format, where the keyboard is strobed sequentially with four low true column selects with each returning a low true row select. The low true column and row selects are encoded in the upper and lower nibbles respectively of the accumulator, which serves as the input to the DTMF keyboard input decode subroutine. The subroutine will then generate the DTMF hexadecimal digit associated with the DTMF keyboard input digit.

The DTMF keyboard decode subroutine (KBRDEC) utilizes a common ROM table lookup for each of the two nibbles representing the low true column and row encodings for the keyboard. The only legal low true nibbles for a single key input are E, D, B, and 7. All other low true nibble values represent multiple keys, no key, or no column strobe. Results from two legal nibble table lookups (from the same 16 byte ROM table) are combined to form a hex digit with the binary format of 0000RRCC, where RR represents the four row values and CC represents the four column values. The illegal nibbles are trapped, and the subroutine is exited with a RET (return) command to indicate multiple keys or no key. A pair of legal nibble table lookups result in the subroutine being exited with a RETSK (return and skip) command to indicate a single key input. This KBRDEC subroutine uses 35 bytes of code, consisting of 19 bytes of program code and 16 bytes of ROM table.

DTMF GENERATION USING PWM AND AN OP AMP

The first DTMF generation method (using the DTMFGP subroutine) generates the selected high band and low band frequencies as PWM (Pulse Width Modulation) outputs on pins G3 and G2 respectively of the G port. The COP820C/840C microcontrollers each contain only one timer, and three times must be generated to satisfy the DTMF application. These three times are the half periods of the two selected frequencies and the 100 ms duration period. Obviously the single timer can only generate one of the required times, while the program must generate the two remaining times. The solution lies in dividing the 100 ms duration time by the half periods for each of the eight DTMF frequencies, and then examining the respective high band and low band quotients and remainders. Naturally these divisions must be normalized to the instruction cycle time (t_C). 100 ms represents 35796 t_C's. The results of these divisions are detailed in Table I.

The four high band frequencies are produced by running the COP820C/840C timer in PWM (Pulse Width Modulation) mode, while the program produces the four low band frequencies and the 100 ms duration timeout. The programmed times are achieved by using three programmed register counters R0, R2 and R3, with a backup register R1 to reload the counter R0. These three counters represent the half period, the 100 ms quotient, and the 100 ms remainder associated with each of the four low band frequencies.



TL/DD/10740–22

FIGURE 1. DTMF Keyboard Matrix

TABLE I. Frequency Half Periods, Quotients and Remainders

	Freq. Hz	Half Period in μs	Half Period in t_C's	100 ms/0.5P in t_C's	
				Quotient	Remainder
Low Band Frequencies	697	717.36	257	139	73
	770	649.35	232	154	68
	852	586.85	210	170	96
	941	531.35	190	188	76
High Band Frequencies	1209	413.56	148	241	128
	1336	374.25	134	267	18
	1477	338.53	121	295	101
	1633	306.18	110	325	46

Note: 100 ms represents 35796 t_C's.

The DTMFGP subroutine starts by transforming the DTMF hex digit in the accumulator (with binary format 0000RRCC) into low and high frequency vectors with binary formats 0011RR11 and 0011CC00 respectively. The transformation of the hex digit 0000RRCC (where RR is the row select and CC is the column select) into the frequency vectors is shown in Table II. The conversion produces a timer vector 0011CC00 (T), and three programmed counter vectors for R1, R2, and R3. The formats for the three counter vectors are 0011RR11 (F), 0011RR10 (Q), and 0011RR01 (R). These four vectors created from the core vector are used as inputs for a 16 byte ROM table using the LAID (Load Accumulator InDirect) instruction. One of these four vectors (the T vector) is a function of the column bits (CC), while the other three vectors (F, Q, R) are a function of the row bits (RR). This correlates to only one parameter being needed for the timer (representing the selected high band frequency), while three parameters are needed for the three counters (half period, 100 ms quotient, 100 ms remainder) associated with the low band frequency and 100 ms duration. The frequency parameter ROM translation table, accessed by the T, F, Q, and R vectors, is shown in Table III.

TABLE II. DTMF Hex Digit Translation

DTMF Hex Digit— 0000RRCC

Timer Vector	Timer	T	0011CC00
Half Period Vector	R1	F	0011RR11
100 ms Quotient Vector	R2	Q	0011RR10
100 ms Remainder Vector	R3	R	0011RR01

TABLE III. Frequency Parameter ROM Translation Table

T— Timer **F— Frequency** **Q— Quotient** **R— Remainder**

Address	Data (Decimal)	Vector
0x30	147	T
0x31	10	R
0x32	140	Q
0x33	38	F
0x34	133	T
0x35	9	R
0x36	155	Q
0x37	33	F
0x38	120	T
0x39	14	R
0x3A	171	Q
0x3B	31	F
0x3C	109	T
0x3D	10	R
0x3E	189	Q
0x3F	26	F

The theory of operation in producing the selected low band frequency starts with loading the three counters with values obtained from a ROM table. The half period for the selected frequency is counted out, after which the G2 output bit is toggled. During this half period countout, the quotient counter is decremented. This procedure is repeated until the quotient counter counts out, after which the program branches to the remainder loop. During the remainder loop, the remainder counter counts out to terminate the 100 ms. Following the remainder countout, the G2 and G3 bits are both reset, after which the DTMF subroutine is exited. Great care must be taken in time balancing the half period loop for the selected low band frequency. Furthermore, the toggling of the G2 output bit (achieved with either a set or reset bit instruction) must also be exactly time balanced to maintain the half period time integrity. Local stall loops (consisting of a DRSZ instruction followed by a JP jump back to the DRSZ for a two byte, six instruction cycle loop) are embedded in both the half period and remainder loops. Consequently, the ROM table parameters for the half period and remainder counters are approximately only one-sixth of what otherwise might be expected. The program for the half period loop, along with the detailed time balancing of the loop for each of the low band frequencies, is shown in *Figure 2*.

	Program		Bytes/ Cycles	Conditional Cycles		Cycles	Total Cycles
	LD	B,#PORTGD	2/3				
	LD	X,#R1	2/3				
LUP1:	LD	A,[X−]	1/3			3	
	IFBIT	2,[B]	1/1			1	
	JP	BYP1	1/3	3	1		
	X	A,[X+]	1/3		3		
	SBIT	2,[B]	1/1		1		
	JP	BYP2	1/3		3		
BYP1:	NOP		1/1	1			
	RBIT	2,[B]	1/1	1			
	X	A,[X+]	1/3	3			
BYP2:	DRSZ	R2	1/3			3	
	JP	LUP2	1/3			3	
	JP	FINI	1/3				
LUP2:	DRSZ	R0	1/3		3	3	
	JP	LUP2	1/3		3	1	
	LD	A,[X]	1/3			3	
	IFEQ	A,#31	2/2			2	
	JP	LUP1	1/3		1	3	30
	NOP		1/1		1		
	NOP		1/1		1		
	IFEQ	A,#38	2/2		2		
	JP	LUP1	1/3	1	3		35
	LAID		1/3	3			
	NOP		1/1	1			
	JP	LUP1	1/3	3			40

Table III Frequency	Stall Loop	Total Cycles	Half Period
[(38 − 1)	× 6]	+ 35	= 257
[(33 − 1)	× 6]	+ 40	= 232
[(31 − 1)	× 6]	+ 30	= 210
[(26 − 1)	× 6]	+ 40	= 190

FIGURE 2. Time Balancing for Half Period Loop

TABLE IV. Time Balancing for Remainder Loop

Table III Remainder	Stall Loop	R Loop Overhead	Total Cycles	Table I Remainder
[(10 − 1)	× 6]	+ 20	= 74	73
[(9 − 1)	× 6]	+ 20	= 68	68
[(14 − 1)	× 6]	+ 20	= 98	96
[(10 − 1)	× 6]	+ 20	= 74	76

Note that the Q value in Table III is one greater than the quotient in Table I to compensate for the fact that the quotient count down to zero test is performed early in the half period loop. The overhead in the remainder loop is 20 instruction cycles. The detailed time balancing for the remainder loop is shown in Table IV.

The selected high band frequency is achieved by loading the half period count in t_C's minus one (from Table III) into the timer autoreload register and running the timer in PWM output mode. The minus one is necessary since the timer toggles the G3 output bit when it underflows (counts down through zero), at which time the contents of the autoreload register are transferred into the timer.

In summary, the input digit from the keyboard (encoded in low true column/row format) is translated into a digit matrix vector XXXXRRCC which is checked for 1001RRCC to indicate a single key entry. No key or multiple key entries will set a flag and terminate the DTMF subroutine. The digit matrix vector for a single key is transformed into the core vector 0000RRCC. The core vector is then translated into four other vectors (T, F, Q, R) which in turn are used to select four parameters from a 16 byte ROM table. These four parameters are used to load the timer, and the respective half period, quotient, and remainder counters. The 16 byte ROM table must be located starting at ROM location 0030 (or 0X30) in order to minimize program size, and has reference setups with the "OR A,#033" instruction for the F vector and the "OR A,#030" instruction for the T vector.

The three parameters associated with the two R bits of the core vector require a multi-level table lookup capability with the LAID instruction. This is achieved with the following section of code in the DTMF subroutine:

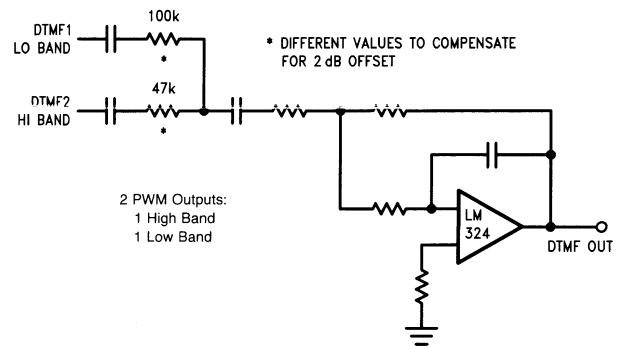
```
        LD      B,#R1
LUP:    X       A,[B]
        LD      A,[B,]
        LAID
        X       A,[B+]
        DEC     A
        IFBNE   #4
        JP      LUP
```

This program loads the F frequency vector into R1, and then decrements the vector each time around the loop. The vector is successively moved with the exchange commands from R1 to R2 to R3 as one of the same exchange commands loads the data from the ROM table into R1, R2, and R3. This successive decrementation of the F vector changes the F vector into the Q vector, and then changes the Q vector into the R vector. These vectors are used to access the ROM table with the LAID instruction. The B pointer is incremented each time around the loop after it has been used to store away the three selected ROM table parameters (one per loop). These three parameters are stored in sequential RAM locations R1, R2, and R3. The IFBNE test instruction is used to skip out of the loop once the three selected ROM table parameters have been accessed and stored away.

The timer is initialized to a count of 15 so that the first timer underflow and toggling of the G3 output bit (with timer PWM mode and G3 toggle output selected) will occur at the same time as the first toggling of the G2 output bit. The half period counts for the high band frequencies minus one are stored in the timer section of the ROM table. The selected value from this frequency ROM table is stored in the timer autoreload register. The timer is selected for PWM output mode and started with the instruction LD [B],#0B0 where the B pointer is selecting the CNTRL register at memory location 0EE.

This first DTMF generation subroutine for the COP820C/840C uses 94 bytes of code, consisting of 78 bytes of program code and 16 bytes of ROM table. A program test routine to sequentially call the DTMFGP subroutine for each of the 16 keyboard input digits is supplied with the listing for the DTMF35 program. This test routine uses a 16 byte ROM table to supply the low true encoded column/row keyboard input to the accumulator. An input from the I0 input pin of the I port is used to select which DTMF generation subroutine is to be used. The DTMFGP subroutine is selected with I0 = 0.

A TYPICAL OP AMP CONFIGURATION FOR MIXING THE TWO DTMF PWM OUTPUTS IS SHOWN IN *FIGURE 3*.



TL/DD/10740–23

FIGURE 3. Typical Op Amp Configuration for Mixing DTMF PWM Outputs

DTMF GENERATION USING A RESISTOR LADDER NETWORK

The second DTMF generation method (using the DTMFLP subroutine) generates and combines values from two table lookups simulating the two selected sine waves. The high band frequency table values have a higher base line value (16 versus 13) than the low band frequency table values. This higher bias for the high frequency values is necessary to satisfy the DTMF requirement that the high band DTMF frequencies need a value 2 dB greater than the low band DTMF frequencies to compensate for losses in transmission.

The resultant value from arithmetically combining the table lookup low band and high band frequency values is output on pins L0 to L5 of the L port in order to feed into a six input external resistor ladder network. The resultant value is updated every 117⅓ μs (one cycle of the LUP42 program loop). The LUP42 program loop contains 42 instruction cycles (t_C's) of 2.7936511 μs each for a total loop time of 117⅓ μs. The COP820C/840C timer is used to count out the 100 ms DTMF duration time.

An interrupt from the timer terminates the 100 ms DTMF output. Note that the Stack Pointer (SP) must be adjusted following the timer interrupt before returning from the DTMFLP subroutine.

The DTMFLP subroutine starts by quadrupling the value of the DTMF hex digit value in the accumulator, and then adding an offset value to reach the first value in the telephone key table. The telephone key ROM table contains four values associated with each of the 16 DTMF hex keys. These four values represent the low and high frequency table sizes and table starting addresses associated with the pair of frequencies (one low band, one high band) associated with each DTMF key. The FRLUP section of the program loads the four associated telephone key table values from the ROM table into the registers LFTBSZ (Low Freq Table Size), LFTADR (Low Freq Table Address), HFTBSZ (High Freq Table Size), and HFTADR (High Freq Table Address). The program then initializes the timer and autoreload register, starts the timer, and then jumps to LUP42. Note that the timer value in t_C's is 100 ms plus one LUP42 time, since the initial DTMF output is not until the end of the LUP42 program.

Multiples of the magic number 118 μs (approximately) are close approximations to all eight of the DTMF frequencies. The LUP42 program uses 42 instruction cycles (of 2.7936511 μs each) to yield a LUP42 time of 117⅓ μs. The purpose of the LUP42 program is to update the six L port outputs by accessing and then combining the next set of values from the selected low band and high band sine wave frequency tables in the ROM. The ROM table offset frequency pointers (LFPTR and HFPTR) must increment each time and then wrap around from top to bottom of the two selected ROM tables. The ROM table size parameters (LFTBSZ and HFTBSZ) for the selected frequencies are tested during each LUP42 to determine if the wrap around from ROM table top to bottom is necessary. The wrap around is implemented by clearing the frequency pointer in question. Note that the ROM tables are mapped from a reference of 0 to table size minus one, so that the table size is used in a direct comparison with the frequency offset pointer to test for the need for a wrap around. Also note that the offset pointer incremented value is used during the following LUP42 cycle, while the pre-incremented value of the pointer is used during the current cycle. However, it is the incremented value that is tested versus the table size for the need to wrap around.

After the low band and high band ROM table sine wave frequency values are accessed in each cycle of the LUP42 program, they are added together and then output to pins L0–L5 of the L port. As stated previously, the low band frequency values have a lower bias than the high band frequency values to compensate for the required 2 dB offset. Specifically, the base line and maximum values for the low frequency values are 13 and 26 respectively, while the base line and maximum values for the high frequency values are 16 and 32 respectively. Thus the combined base line value is 29, while the combined maximum value is 58. This gives a range of values on the L port output (L0–L5) from 0 to 58.

The minimum time necessary for the LUP42 update program loop is 36 instruction cycles including the jump back to the start of the loop. Consequently, two LAID instructions are inserted just prior to the jump back instruction at the end of LUP42 to supply the six extra NOP instruction cycles needed to increase the LUP42 instruction cycles from 36 to 42. A three cycle LAID instruction can always be used to simulate three single cycle NOP instructions if the accumulator data is not needed.

Table V shows the multiple LUP42 approximation to the eight DTMF frequencies, including the number of sine wave cycles and data points in the approximation. As an example, three cycles of a sine wave with a total of 19 data points across the three cycles is used to approximate the 1336 Hz DTMF frequency. The 19 cycles of LUP42 times the LUP42 time of 117⅓ μs is divided into the three cycles to yield a value of 1345.69 Hz. This gives an error of +0.73% when compared with the DTMF value of 1336 Hz. This is well within the 1.5% North American DTMF error range.

TABLE V. DTMF Frequency Approximation Table

DTMF Freq.	# of Sine Wave Cycles	# of Data Points	Calculation	Approx. Freq.	% Error
697	4	49	4/(49 x 117⅓)	= 695.73	−0.18
770	1	11	1/(11 x 117⅓)	= 774.79	+0.62
852	1	10	1/(10 x 117⅓)	= 852.27	+0.03
941	1	9	1/(9 x 117⅓)	= 946.97	+0.63
1209	1	7	1/(7 x 117⅓)	= 1217.53	+0.71
1336	3	19	3/(19 x 117⅓)	= 1345.69	+0.73
1477	4	23	4/(23 x 117⅓)	= 1482.21	+0.35
1633	4	21	4/(21 x 117⅓)	= 1623.38	−0.59

The frequency approximation is equal to the number of cycles of sine wave divided by the time in the total number of LUP42 cycles before the ROM table repeats.

The values in the DTMF sine wave ROM tables are calculated by computing the sine value at the appropriate points, scaling the sine value up to the base line value, and then adding the result to the base line value. The following example will help to clarify this calculation.

Consider the three cycles of sine wave across 19 data points for the 1336 Hz high band frequency. The first value in the table is the base line value of 16. With 2π radians per sine wave cycle, the succeeding values in the table represent the sine values of $1 \times (6\pi/19)$, $2 \times (6\pi/19)$, $3 \times (6\pi/19)$, . . . , up to $18 \times (6\pi/19)$. Consider the seventh and eighth values in the table, representing the sine values of $6 \times (6\pi/19)$ and $7 \times (6\pi/19)$ respectively. The respective calculatons of $16 \times \sin[6 \times (6\pi/19)]$ and $16 \times \sin[7 \times (6\pi/19)]$ yield values of -5.20 and 9.83. Rounding to the nearest integer gives values of -5 and 10. When added to the base line value of 16, these values yield the results 11 and 26 for the seventh and eighth values in the 1336 Hz DTMF ROM table. Symmetry in the loop of 19 values in the DTMF table dictates that the fourteenth and thirteenth values in the table are 21 and 6, representing values of 5 and -10 from the calculations.

The area under a half cycle of sine wave relative to the area of the surrounding rectangle is $2/\pi$, where π radians represent the sine wave half cycle. This surrounding rectangle has a length of π and a height of 1, with the height representing the maximum sine value. Consequently, the area of the surrounding rectangle is π. The integral of the area under the half sine wave from 0 to π is equal to 2. The ratio of $2/\pi$ is equal to 63.66%, so that the total of the values for each half sine wave should approximate 63.66% of the sum of the max values. The maximum values (relative to the base line) are 13 and 16 respectively for the low and high band DTMF frequencies.

For the previous 1336 Hz example, the total of the absolute values for the 19 sine values from the 1336 Hz ROM table is equal to 196. The surrounding rectangle for the three cycles of sine wave is 19 by 16 for a total area of 304. The ratio of 196/304 is 64.47% compared with the $2/\pi$ ratio of 63.66%. Thus the sine wave approximation gives an area abundance of 0.81% (equal to 64.47 − 63.66).

An application of the sine wave area criteria is shown in the generation of the DTMF 852 Hz frequency. The ten sine values calculated are 0, 7.64, 12.36, 12.36, 7.64, 0, −7.64, −12.36, −12.36, and −7.64. Rounding off to the nearest integer yields values of 0, 8, 12, 12, 8, 0, −8, −12, −12 and −8. The total of these values (absolute numbers) is 80, while the area of the surrounding rectangle is 130 (10 x 13). The ratio of 80/130 is 61.54% compared with the $2/\pi$ ratio of 63.66%. Thus the sine wave approximation gives an area deficiency of 2.12% (equal to 63.66 − 61.54), which is overly deficient. Consequently, two of the ten sine values are augmented to yield sine values of 0, 8, 12, 13*, 8, 0, −8, −12, −13*, and −8. This gives an absolute total of 82 and a ratio of 82/130, which equals 63.08% and serves as a much better approximation to the $2/\pi$ ratio of 63.66%.

The sine wave area criteria is also used to modify two values in the DTMF 941 Hz frequency. The nine sine values calculated are 0, 8.36, 12.80, 11.26, 4.45, −4.45, −11.26, −12.80, and −8.36. Rounding off to the nearest integer yields values of 0, 8, 13, 11, 4, −4, −11, −13, and −8. The total of these values (absolute numbers) is 72, while the area of the surrounding rectangle is 117 (9 x 13). The ratio of 72/117 is 61.54% compared to the $2/\pi$ ratio of 63.66%. Thus the sine wave approximation gives an area deficiency of 2.12% (equal to 63.66 − 61.54),which is overly deficient. Rounding up the two values of 4.45 and −4.45 to 5 and −5, rather than down to 4 and −4, yields values of 0, 8, 13, 11, 5, −5, −11, −13 and −8. This gives an absolute total of 74 and a ratio of 74/117, which equals 63.25% and serves as a much better approximation to the $2/\pi$ ratio of 63.66%.

With these modified values for the 852 and 941 DTMF frequencies, the area criteria ratio of $2/\pi$ = 63.66% for the sine wave compared to the surrounding rectangle has the following values:

DTMF Freq.	Sum of Values	Rectangle Area	Percentage	Diff.
697 Hz	406	49 x 13 = 637	63.74%	+0.08%
770 Hz	92	11 x 13 = 143	64.34%	+0.68%
852 Hz	82	10 x 13 = 130	63.08%	−0.58%
941 Hz	74	9 x 13 = 117	63.25%	−0.41%
1209 Hz	72	7 x 16 = 112	64.29%	+0.63%
1336 Hz	196	19 x 16 = 304	64.47%	+0.81%
1477 Hz	232	23 x 16 = 368	63.04%	−0.62%
1633 Hz	216	21 x 16 = 336	64.29%	+0.63%

The LUP42 program loop is interrupted by the COP820C/840C timer after 100 ms of DTMF output. As stated previously, the Stack Pointer (SP) must be adjusted (incremented by 2) following the timer interrupt before returning from the DTMFLP subroutine.

This second DTMF generation subroutine for the COP820C/840C uses 301 bytes of code, consisting of 88 bytes of program code and 213 bytes of ROM table. The following is a summary of the DTMFLP subroutine code allocation.

DTMFLP Code Allocation	# of Bytes
1. Subroutine Header Code	42
2. Interrupt Code	16
3. LUP42 Code	30
4. Telephone Key Table	64
5. Sine Value Tables	149
Total	301

A program test routine to sequentially call the DTMFLP subroutine for each of the 16 DTMF keyboard input digits is supplied with the listing for the DTMF35 program. This test routine uses a 16 byte ROM table to supply the low true encoded column/row keyboard input to the accumulator. An input from the I0 pin of the I port is used to select which DTMF generation subroutine is to be used. The DTMFLP subroutine is selected with I0 = 1.

A TYPICAL RESISTOR LADDER NETWORK IS SHOWN IN *FIGURE 4*.

SUMMARY

In summary, the DTMF35 program assumes a COP820C/840C clocked with a 3.58 MHz crystal in divide by 10 mode. The DTMF35 program contains three subroutines, KBRDEC, DTMFGP, and DTMFLP. The KBRDEC subroutine is a low true DTMF keyboard decoder, while the DTMFGP and DTMFLP subroutines represent the alternative methods of DTMF generation.

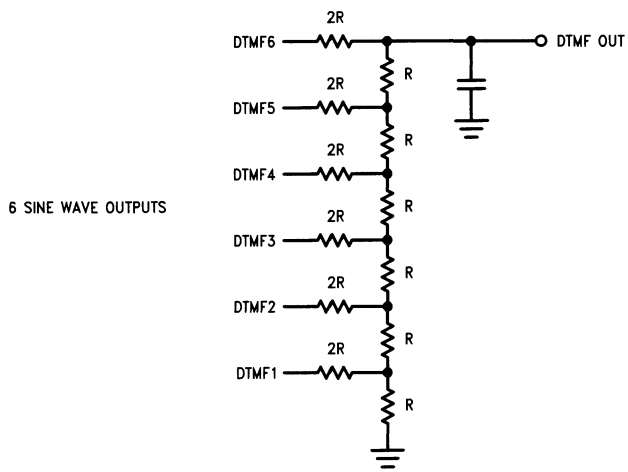
The KBRDEC subroutine provides a low true decoding of the DTMF keyboard input and assumes that the keyboard input has been encoded in a low true column/row format, with the columns of the keyboard being sequentially strobed.

The DTMFGP subroutine produces two PWM (Pulse Width Modulation) outputs (representing the selected high and low band DTMF frequencies) for combination with an external op amp network (LM324 or equivalent).

The DTMFLP subroutine produces six bits of combined high band and low band DTMF frequency output for combination in an external resistor ladder network. This output represents a combined sine wave simulation of the two selected DTMF frequencies by combining values from two selected ROM tables, and updating these values every 118 μs.

The three DTMF35 subroutines contain the following number of bytes of program and ROM table memory:

Subroutine	# of Bytes of Program	# of Bytes of ROM Table	Total # of Bytes
KBRDEC	19	16	35
DTMFGP	78	16	94
DTMFLP	88	213	301



TL/DD/10740-24

FIGURE 4. Typical Resistor Ladder Network

```
1                       ;
2                       ; DTMF GENERATION WITH A 3.58 MHZ       VERNE H. WILSON
3                       ;       CRYSTAL FOR COP820C/840C          10/28/89
4                       ;
5                       ; DTMF - DUAL TONE MULTIPLE FREQUENCY
6                       ;
7                       ; PROGRAM NAME: DTMF35.MAC
8                       ;
9                               .TITLE DTMF35
10                              .CHIP 840
11                      ;
12                      ;
13                      ; THIS DTMF PROGRAM IS BASED ON A COP820C/840C RUNNING
14                      ; WITH A CKI CLOCK OF 3.579545 MHZ (TV COLOR CRYSTAL
15                      ; FREQUENCY) IN DIVIDE BY 10 MODE, FOR AN INSTRUCTION
16                      ; CYCLE TIME OF 2.7936511 MICROSECONDS.
17                      ;
18                      ; THIS PROGRAM CONTAINS THREE SUBROUTINES, ONE FOR A
19                      ; LOW TRUE ROW/COLUMN DTMF KEYBOARD DECODING (KBRDEC),
20                      ; AND THE OTHER TWO (DTMFGP, DTMFLP) FOR ALTERNATE
21                      ; METHODS OF DTMF GENERATION.
22                      ;
23                      ; KEYBOARD INPUT DATA IS IN ACCUMULATOR WITH A
24                      ;    LOW TRUE FORMAT AS FOLLOWS:
25                      ;        BITS 7 TO 4 : LOW TRUE COLUMN VALUE (E,D,B,7)
26                      ;        BITS 3 TO 0 : LOW TRUE ROW VALUE (E,D,B,7)
27                      ;
28                      ; ASSUMPTION MADE THAT COLUMN STROBES (LOW TRUE) ARE
29                      ;    OUTPUT, WHILE ROW VALUES (LOW TRUE) ARE INPUT.
30                      ;
31                      ; THE FIRST METHOD OF DTMF GENERATION CONSISTS OF
32                      ; GENERATING TWO PWM OUTPUTS ON THE G PORT G2 AND G3
33                      ; OUTPUT PINS. THESE TWO OUTPUTS NEED TO BE MIXED
34                      ; EXTERNALLY WITH AN APPROPIATE LM324 OP AMP FEEDBACK
35                      ; CIRCUIT TO GENERATE THE DTMF.
36                      ;
37                      ; THE SECOND METHOD OF DTMF GENERATION USES ROM LOOKUP
38                      ; TABLES TO SIMULATE THE TWO DTMF SINE WAVES AND
39                      ; COMBINES THEM ARITHMETICALLY. THE RESULT IS OUTPUT ON
40                      ; THE LOWER SIX BITS OF THE L PORT (L0 - L5). THESE SIX
41                      ; OUTPUTS ARE COMBINED EXTERNALLY WITH A LADDER NETWORK
42                      ; TO GENERATE THE DTMF.
43                      ;
44                      ; THE SECOND DTMF GENERATION METHOD USES APPROXIMATELY
45                      ; THREE TIMES AS MUCH ROM CODE (INCLUDING PROGRAM CODE
46                      ; AND ROM TABLES) AS THE FIRST METHOD, BUT HAS THE
47                      ; ADVANTAGE OF ELIMINATING THE COST OF THE EXTERNAL
48                      ; ACTIVE COMPONENT (LM324 OR EQUIVALENT).
49                      ;
50                      ; BOTH OF THE DTMF SUBROUTINES GENERATE THEIR OUTPUTS
51                      ; FOR A PERIOD OF 100 MILLISECONDS.
```

TL/DD/10740–1

```
52                          ;
53                          ; DECLARATIONS:
54                          ;
55      0000                        KDATA   = 0             ; *** KEYBOARD DATA ***
56      00D0                        PORTLD  = 0D0           ; PORTL DATA REG
57      00D1                        PORTLC  = 0D1           ; PORTL CONFIG REG
58      00D4                        PORTGD  = 0D4           ; PORTG DATA REG
59      00D5                        PORTGC  = 0D5           ; PORTG CONFIG REG
60      00D7                         PORTI  = 0D7           ; PORTI INPUT PINS
61      00DC                         PORTD  = 0DC           ; PORTD REG
62      00EA                         TMRLO  = 0EA           ; TIMER LOW COUNTER
63      00EB                         TMRHI  = 0EB           ; TIMER HIGH COUNTER
64      00EC                         TAULO  = 0EC           ; TMR AUTORELOAD REG LO
65      00ED                         TAUHI  = 0ED           ; TMR AUTORELOAD REG HI
66      00EE                         CNTRL  = 0EE           ; CONTROL REG
67      00EF                           PSW  = 0EF           ; PROC STATUS WORD
68      00F0                            R0  = 0F0           ; LB FREQ LOOP COUNTER
69      00F1                            R1  = 0F1           ; LB FREQ LOOP COUNT
70      00F2                            R2  = 0F2           ; LB FREQ Q COUNT
71      00F3                            R3  = 0F3           ; LB FREQ R COUNT
72                          ;
73 0000 DD2F        START:  LD      SP,#02F     ; INITIALIZE STACK PTR
74                          ;
75                          ;                                KEYBOARD HEX DIGIT MATRIX
76                          ;                                      1  2  3  A
77 0002 DEDC                LD      B,#PORTD    ;                  4  5  6  B
78 0004 9E00                LD      [B],#0      ;                  7  8  9  C
79 0006 A0          LOOP:   RC                  ;                  *  0  #  D
80 0007 AE                  LD      A,[B]       ; DTMF TEST LOOP
81 0008 9405                ADD     A,#5        ; SEQUENCE IS 1,5,9,D,4,
82 000A A6                  X       A,[B]       ; 8,#,A,7,0,3,B,*,2,6,C
83 000B 6C                  RBIT    4,[B]       ; HEX MATRIX TO LOOKUP
84 000C 9420                ADD     A,#020      ;    TABLE FOR LOW TRUE
85 000E A4                  LAID                ;    COLUMN/ROW INPUT TO
86 000F 3210                JSR     KBRDEC      ;    KBRDEC SUBROUTINE
87 0011 A1                  SC                  ; SET C IF NOT SINGLE KEY
88 0012 DED7                LD      B,#PORTI    ; TEST BIT 0 OF PORTI TO
89 0014 70                  IFBIT   0,[B]       ;    DETERMINE WHICH
90 0015 03                  JP      BYPA        ;    DTMF SUBROUTIINE
91 0016 3040                JSR     DTMFGP      ; TWO PWM OUTPUTS ON
92 0018 02                  JP      BYPB        ;    G PORT PINS G2,G3
93 0019 308E        BYPA:   JSR     DTMFLP      ; SIX LADDER OUTPUTS ON
94                                              ;    L PORT PINS L0 - L5
95 001B DEDC        BYPB:   LD      B,#PORTD    ; D0 WILL TOGGLE FOR EACH
96 001D E8                  JP      LOOP        ;    CALL OF SUBROUTINE
97                          ;
98                          ;
99                          ;
```

TL/DD/10740–2

```
                                        .FORM
                          ;
                          ; KEYBOARD DIGIT MATRIX TABLE
                          ;
      0020                              . = 020
                          ;
                          ;                  1   5   9   D   4   8   #   A
0020 EE                             .BYTE    0EE,0DD,0BB,077,0ED,0DB,0B7,07E
0021 DD
0022 BB
0023 77
0024 ED
0025 DB
0026 B7
0027 7E
                          ;                  7   0   3   B   *   2   6   C
0028 EB                             .BYTE    0EB,0D7,0BE,07D,0E7,0DE,0BD,07B
0029 D7
002A BE
002B 7D
002C E7
002D DE
002E BD
002F 7B
                          ;
                          ;
                          ;
                          ;
                          ;
                          ; FIRST DTMF SUBROUTINE (DTMFGP) PRODUCES TWO PWM
                          ;     (PULSE WIDTH MODULATION) OUTPUTS ON PINS G3, G2
                          ;
                          ;
                          ; G PORT IS USED FOR THE TWO OUTPUTS
                          ;      -    HIGH BAND (HB) FREQUENCY OUTPUT ON G3
                          ;      -    LOW BAND (LB) FREQUENCY OUTPUT ON G2
                          ;
                          ; TIMER COUNTS OUT
                          ;      -    HB FREQUENCIES
                          ;
                          ; PROGRAM COUNTS OUT
                          ;      -    LB FREQUENCIES
                          ;      -    100 MSEC DIVIDED BY LB HALF PERIOD QUOTIENT
                          ;      -    100 MSEC DIVIDED BY LB HALF PERIOD REMAINDER
                          ;
                          ; NOTE THAT ALL COUNTS MUST BE NORMALIZED TO THE
                          ;     2.7936511 MICROSECOND INSTRUCTION CYCLE Tc
                          ;
                          ; 100 MSEC REPRESENTS 35796 Tc's
                          ;
                          ;
```

TL/DD/10740–3

```
137                     ;
138                     ;
139                     ; HALF PERIODS FOR THE 8 DTMF FREQUENCIES (697,770,852,
140                     ;     941,1209,1336,1477, AND 1633 KHZ) ARE 257,232,
141                     ;     210,190,148,134,121, AND 110 Tc's RESPECTIVELY
142                     ;
143                     ; THE 100 MSEC DIVIDED BY HALF PERIOD QUOTIENTS ARE
144                     ;     139,154,170,188,241,267,295, AND 325 RESPECTIVELY
145                     ;
146                     ; THE 100 MSEC DIVIDED BY HALF PERIOD REMAINDERS ARE
147                     ;     72,67,95,75,127,17,100, AND 45 RESPECTIVELY
148                     ;
149                     ;
150                     ;
151                     ;
152                     ; BINARY FORMAT FOR THE HEX DIGIT KEY VALUE FROM THE
153                     ;    KBRDEC SUBROUTINE IS 0000RRCC,
154                     ;       WHERE  -  RR IS ROW SELECT (LB FREQUENCIES)
155                     ;              -  CC IS COLUMN SELECT (HB FREQUENCIES)
156                     ;
157                     ; FREQUENCY VECTORS (HB & LB) FOR FREQ PARAMETER TABLE
158                     ;    MADE FROM KEY VALUE
159                     ;
160                     ; HB FREQ VECTORS (4) END WITH 00 FOR TIMER COUNTS,
161                     ;    WHERE VECTOR FORMAT IS 0011CC00
162                     ;
163                     ; LB FREQUENCY VECTORS (12) END WITH:
164                     ;    11 FOR HALF PERIOD LOOP COUNTS,
165                     ;         WHERE VECTOR FORMAT IS 0011RR11
166                     ;    10 FOR 100 MSEC DIVIDED BY HALF PERIOD QUOTIENTS,
167                     ;         WHERE VECTOR FORMAT IS 0011RR10
168                     ;    01 FOR 100 MSEC DIVIDED BY HALF PERIOD REMAINDERS,
169                     ;         WHERE VECTOR FORMAT IS 0011RR01
170                     ;
171                     ; FREQ PARAMETER TABLE AT HEX 003*  (REQUIRED LOCATION)
172                     ;
173                     ;
174                     ;
175                     ;              KEY VALUE
176                     ;               0000RRCC
177                     ;
178                     ;    TIMER     T    CC00
179                     ;    R1        F    RR11
180                     ;    R2        Q    RR10
181                     ;    R3        R    RR01
182                     ;
183                     ;
184                     ;
```

TL/DD/10740–4

```
                                        .FORM
                        ;
                        ;
                        ;FREQUENCY AND 100 MSEC PARAMETER TABLE
                        ;
0030 93                           .BYTE     147           ; T
0031 0A                           .BYTE      10           ; R
0032 8C                           .BYTE     140           ; Q
0033 26                           .BYTE      38           ; F
0034 85                           .BYTE     133           ; T
0035 09                           .BYTE       9           ; R
0036 9B                           .BYTE     155           ; Q
0037 21                           .BYTE      33           ; F
0038 78                           .BYTE     120           ; T
0039 0E                           .BYTE      14           ; R
003A AB                           .BYTE     171           ; Q
003B 1F                           .BYTE      31           ; F
003C 6D                           .BYTE     109           ; T
003D 0A                           .BYTE      10           ; R
003E BD                           .BYTE     189           ; Q
003F 1A                           .BYTE      26           ; F
                        ;
                        ;
                        ;
0040 DED5               DTMFGP:   LD        B,#PORTGC     ; CONFIGURE G PORT
0042 9B3F                         LD        [B-],#03F     ;    FOR OUTPUTS
0044 6B                           RBIT      3,[B]         ; OPTIONAL HB RESET
0045 6A                           RBIT      2,[B]         ; OPTIONAL LB RESET
0046 5F                           LD        B,#KDATA
0047 A6                           X         A,[B]         ; STORE KEY VALUE
0048 AE                           LD        A,[B]         ; KEY VALUE TO ACC
0049 9733                         OR        A,#033        ; CREATE LB FREQ VECTOR
004B DEF1                         LD        B,#R1         ;    FROM KEY VALUE
004D A6                 LUP:      X         A,[B]
004E AE                           LD        A,[B]         ; THREE PARAMETERS
004F A4                           LAID                    ;    FROM LOW BAND
0050 A2                           X         A,[B+]        ;    FREQ ROM TABLE
0051 8B                           DEC       A             ;    TO R1,R2,R3
0052 44                           IFBNE     #4
0053 F9                           JP        LUP
0054 5F                           LD        B,#KDATA
0055 AE                           LD        A,[B]         ; KEY VALUE TO ACC
0056 65                           SWAP      A             ; CREATE HB FREQ VECTOR
0057 A0                           RC                      ;    FROM KEY VALUE
0058 B0                           RRC       A
0059 B0                           RRC       A
005A 9730                         OR        A,#030
005C A4                           LAID                    ; HB FREQ TABLE
005D DEEA                         LD        B,#TMRLO      ;   (1 PARAMETER)
005F 9A0F                         LD        [B+],#15      ; INSTRUCTION CYCLE
0061 9A00                         LD        [B+],#0       ;   TIME UNTIL TOGGLE
```

TL/DD/10740–5

```
236 0063 A2                     X       A,[B+]      ; HB FREQ PARAMETER TO
237 0064 9A00                   LD      [B+],#0     ;   AUTORELOAD REGISTER
238 0066 9EB0                   LD      [B],#0B0    ; START TIMER PWM
239 0068 DED4                   LD      B,#PORTGD
240 006A DCF1                   LD      X,#R1
241 006C BB             LUP1:   LD      A,[X-]
242 006D 72                     IFBIT   2,[B]       ; TEST LB OUTPUT
243 006E 03                     JP      BYP1
244 006F B2                     X       A,[X+]
245 0070 7A                     SBIT    2,[B]       ; SET LB OUTPUT
246 0071 03                     JP      BYP2
247 0072 B8             BYP1:   NOP
248 0073 6A                     RBIT    2,[B]       ; RESET LB OUTPUT
249 0074 B2                     X       A,[X+]
250 0075 C2             BYP2:   DRSZ    R2          ; DECR. QUOT. COUNT
251 0076 01                     JP      LUP2
252 0077 0E                     JP      FINI        ; Q COUNT FINISHED
253 0078 C0             LUP2:   DRSZ    R0          ; DECR. F COUNT
254 0079 FE                     JP      LUP2        ; LB (HALF PERIOD)
255                     ;
256 007A BE                     LD      A,[X]       ; ******************
257 007B 921F                   IFEQ    A,#31       ; BALANCE         ***
258 007D EE                     JP      LUP1        ; LOW BAND        ***
259 007E B8                     NOP                 ; FREQUENCY       ***
260 007F B8                     NOP                 ; RESIDUE         ***
261 0080 9226                   IFEQ    A,#38       ; DELAY FOR       ***
262 0082 E9                     JP      LUP1        ; EACH OF THE     ***
263 0083 A4                     LAID                ; FOUR LOW BAND   ***
264 0084 B8                     NOP                 ; FREQUENCIES     ***
265 0085 E6                     JP      LUP1        ; ******************
266 0086 C3             FINI:   DRSZ    R3      ; DECR. REMAINDER COUNT
267 0087 FE                     JP      FINI    ; REM. COUNT NOT FINISHED
268 0088 BDEE6C                 RBIT    4,CNTRL ; STOP TIMER
269 008B 6B                     RBIT    3,[B]   ; OPTIONAL CLR HB OUTPUT
270 008C 6A                     RBIT    2,[B]   ; OPTIONAL CLR LB OUTPUT
271 008D 8E                     RET             ; RETURN FROM SUBROUTINE
272                     ;
273                     ;
274                     ;
```

TL/DD/10740–6

```
                        .FORM
                    ;
                    ; SECOND DTMF SUBROUTINE (DTMFLP) PRODUCES SIX
                    ;     COMBINED LOW BAND AND HIGH BAND FREQUENCY
                    ;     SINE WAVE OUTPUTS ON PINS L0 - L5
                    ;
                    ; SIX L PORT OUTPUTS (L0 - L5) FEED INTO AN EXTERNAL
                    ; RESISTOR LADDER NETWORK TO CREATE THE DTMF OUTPUT.
                    ;
                    ; FOUR VALUES FROM A KEYBOARD ROM TABLE ARE LOADED
                    ; INTO LFTBSZ (LOW FREQ TABLE SIZE), LFTADR (LOW
                    ; FREQ TABLE ADDRESS), HFTBSZ (HIGH FREQ TABLE SIZE),
                    ; AND HFTADR (HIGH FREQ TABLE ADDRESS).
                    ;
                    ; LUP42 USES THE LFPTR (LOW FREQ POINTER) AND HFPTR
                    ; (HIGH FREQ POINTER) TO ACCESS THE SINE DATA TABLES
                    ; FOR THE SELECTED FREQUENCIES ONCE PER LOOP. THESE
                    ; POINTERS ARE BOTH INCREMENTED ONCE PER LUP42.
                    ;
                    ; LUP42 PROGRAM LOOP UPDATES THE OUTPUT VALUE EVERY
                    ; 117 1/3 uSEC BY SELECTING AND THEN COMBINING NEW
                    ; VALUES FROM THE SELECTED LOW BAND AND HIGH BAND
                    ; FREQUENCY ROM TABLES WHICH SIMULATE THE SINE WAVES
                    ; FOR THE TWO FREQUENCIES.
                    ;
                    ; MULTIPLES OF THE MAGIC NUMBER OF APPROXIMATELY
                    ; 118 uSEC ARE CLOSE APPROXIMATIONS TO ALL EIGHT OF
                    ; THE DTMF FREQUENCIES.
                    ;
                    ; COP820C/840C TIMER USED TO INTERRUPT THE DTMF LUP42
                    ; PROGRAM LOOP AFTER 100 MSEC TO FINISH THE DTMF
                    ; OUTPUT AND RETURN FROM THE DTMFLP SUBROUTINE. NOTE
                    ; THAT THE STACK POINTER (SP) MUST BE ADJUSTED AFTER
                    ; THE INTERRUPT BEFORE RETURNING FROM THE SUBROUTINE.
                    ;
                    ;
                    ;
                    ;
                    ;
                    ; DECLARATIONS:
                    ;
0005                        LFPTR   =  05           ; LOW FREQ POINTER
0006                        TEMP    =  06           ; TEMPORARY
0007                        HFPTR   =  07           ; HIGH FREQ POINTER
0008                        LFTBSZ  =  08           ; LO FREQ TABLE SIZE
0009                        LFTADR  =  09           ; LO FREQ TABLE ADDR
000A                        HFTBSZ  =  0A           ; HI FREQ TABLE SIZE
000B                        HFTADR  =  0B           ; HI FREQ TABLE ADDR
                    ;
0004                        TRUN    =  04
                    ;
```

TL/DD/10740–7

```
                        ;
008E BCD1FF     DTMFLP: LD      PORTLC,#0FF     ; INITIALIZE PORT L
0091 BCD01D             LD      PORTLD,#29      ;    FOR NO TONE OUT
0094 BC0500             LD      LFPTR,#0        ; INITIALIZE OFFSET
0097 58                 LD      B,#HFPTR        ;    POINTERS FOR
0098 9A00               LD      [B+],#0         ;    DTMF SINE WAVE
009A A0                 RC                      ;    TABLE LOOKUP
009B 65                 SWAP    A               ; QUADRUPLE KEY
009C B0                 RRC     A               ;    VALUE AND ADD
009D B0                 RRC     A               ;    OFFSET FOR KEY
009E 94B8               ADD     A,#0B8          ;    TABLE LOOKUP
00A0 A6         FRLUP:  X       A,[B]           ; LOAD FOUR VALUES
00A1 AE                 LD      A,[B]           ;    FROM ROM KEY
00A2 A4                 LAID                    ;    TABLE INTO LOW
00A3 A2                 X       A,[B+]          ;    FREQ LFTBSZ,
00A4 8A                 INC     A               ;    LFTADR, AND HI
00A5 4C                 IFBNE   #0C             ;    FREQ HFTBSZ,
00A6 F9                 JP      FRLUP           ;    HFTADR
00A7 DEEA               LD      B,#TMRLO        ; INITIALIZE TIMER
00A9 9A00               LD      [B+],#0         ;    WITH A tC COUNT
00AB 9A8C               LD      [B+],#140       ;    EQUIVALENT TO
00AD 9A00               LD      [B+],#0         ;    100 MSEC PLUS
00AF 9A8C               LD      [B+],#140       ;    A LUP42 TIME
00B1 9A80               LD      [B+],#080       ; TIMER PWM, NO OUT
00B3 9B11               LD      [B-],#011       ; ENABLE TMR INTRPT
00B5 7C                 SBIT    TRUN,[B]        ; START TIMER
00B6 210F               JMP     LUP42
                        ;
                        ;
                        ;
                        ;
                        ; TELEPHONE KEY TABLE:
                        ;
                        ;    TABLE FORMAT:
                        ;         PARAMETER 1:  # OF LOW FREQ TABLE VALUES
                        ;         PARAMETER 2:  BASE ADDR. OF LOW FREQ VALUES
                        ;         PARAMETER 3:  # OF HIGH FREQ TABLE VALUES
                        ;         PARAMETER 4:  BASE ADDR. OF HIGH FREQ VALUES
                        ;
                        ; KEY 1
00B8 31                 .BYTE   49,02D,7,07C
00B9 2D
00BA 07
00BB 7C
                        ;
                        ; KEY 2
00BC 31                 .BYTE   49,02D,19,083
00BD 2D
00BE 13
00BF 83
                        ;
```

TL/DD/10740–8

```
371                             ; KEY 3
372 00C0 31                                 .BYTE       49,02D,23,096
    00C1 2D
    00C2 17
    00C3 96
373                             ;
374                             ; KEY A
375 00C4 31                                 .BYTE       49,02D,21,0AD
    00C5 2D
    00C6 15
    00C7 AD
376                             ;
377                             ; KEY 4
378 00C8 0B                                 .BYTE       11,05E,7,07C
    00C9 5E
    00CA 07
    00CB 7C
379                             ;
380                             ; KEY 5
381 00CC 0B                                 .BYTE       11,05E,19,083
    00CD 5E
    00CE 13
    00CF 83
382                             ;
383                             ; KEY 6
384 00D0 0B                                 .BYTE       11,05E,23,096
    00D1 5E
    00D2 17
    00D3 96
385                             ;
386                             ; KEY B
387 00D4 0B                                 .BYTE       11,05E,21,0AD
    00D5 5E
    00D6 15
    00D7 AD
388                             ;
389                             ; KEY 7
390 00D8 0A                                 .BYTE       10,069,7,07C
    00D9 69
    00DA 07
    00DB 7C
391                             ;
392                             ; KEY 8
393 00DC 0A                                 .BYTE       10,069,19,083
    00DD 69
    00DE 13
    00DF 83
394                             ;
395                             ; KEY 9
396 00E0 0A                                 .BYTE       10,069,23,096
    00E1 69
```

TL/DD/10740–9

```
    00E2 17
    00E3 96
397                         ;
398                         ; KEY C
399 00E4 0A                           .BYTE     10,069,21,0AD
    00E5 69
    00E6 15
    00E7 AD
400                         ;
401                         ; KEY *
402 00E8 09                           .BYTE     9,073,7,083
    00E9 73
    00EA 07
    00EB 83
403                         ;
404                         ; KEY 0
405 00EC 09                           .BYTE     9,073,19,07C
    00ED 73
    00EE 13
    00EF 7C
406                         ;
407                         ; KEY #
408 00F0 09                           .BYTE     9,073,23,096
    00F1 73
    00F2 17
    00F3 96
409                         ;
410                         ; KEY D
411 00F4 09                           .BYTE     9,073,21,0AD
    00F5 73
    00F6 15
    00F7 AD
412                         ;
413                         ;
414                         ;
415                         ;
416      00FF                         .=00FF
417                         ;
418 00FF BCD01D             INTRPT:   LD        PORTLD,#29      ; BASE LINE VALUE
419 0102 DEEF                         LD        B,#PSW          ; 100 MSEC INTERRUPT
420 0104 9B00                         LD        [B-],#0         ;    FROM TIMER
421 0106 9E00                         LD        [B],#0          ; CLR PSW AND CNTRL
422 0108 DEFD                         LD        B,#SP           ; RESTORE STACK
423 010A AE                           LD        A,[B]           ;    POINTER (SP)
424 010B 8A                           INC       A               ;    TO ITS VALUE
425 010C 8A                           INC       A               ;    BEFORE THE
426 010D A6                           X         A,[B]           ;    INTERRUPT
427 010E 8E                           RET                       ; RETURN FROM
428                                                             ;    SUBROUTINE
429                         ;
430                         ;
```

TL/DD/10740–10

```
                                .FORM
                    ;
                    ; LUP42 CONSISTS OF 42 COP840C INSTRUCTION CYCLE TIMES
                    ; LUP42 TIMING LOOP IS 42 / 0.3579545 = 117 1/3 uSEC
                    ;
                    ;
010F 5A             LUP42:    LD        B,#LFPTR
0110 AE                       LD        A,[B]          ; INCREMENT LOW FREQ
0111 8A                       INC       A              ;    OFFSET POINTER
0112 57                       LD        B,#LFTBSZ      ; TEST IF LFPTR
0113 82                       IFEQ      A,[B]          ;    BEYOND LIMIT
0114 64                       CLR       A              ; REINITIALIZE LFPTR
0115 5A                       LD        B,#LFPTR       ;    FOR NEXT TIME
0116 A6                       X         A,[B]
0117 56                       LD        B,#LFTADR      ; ADD PTR TO LO FREQ
0118 84                       ADD       A,[B]          ;    TABLE ADDRESS
0119 A4                       LAID                     ; LOW FREQ COMPONENT
011A 59                       LD        B,#TEMP        ; RESULT TO TEMP
011B A2                       X         A,[B+]
011C AE                       LD        A,[B]          ; INCREMENT HI FREQ
011D 8A                       INC       A              ;    OFFSET POINTER
011E 55                       LD        B,#HFTBSZ      ; TEST IF HFPTR
011F 82                       IFEQ      A,[B]          ;    BEYOND LIMIT
0120 64                       CLR       A              ; REINITIALIZE HFPTR
0121 58                       LD        B,#HFPTR       ;    FOR NEXT TIME
0122 A6                       X         A,[B]
0123 54                       LD        B,#HFTADR      ; ADD PTR TO HI FREQ
0124 84                       ADD       A,[B]          ;    TABLE ADDRESS
0125 A4                       LAID                     ; HI FREQ COMPONENT
0126 59                       LD        B,#TEMP        ; ADD LOW FREQ VALUE
0127 84                       ADD       A,[B]          ;   TO HI FREQ VALUE
0128 9CD0                     X         A,PORTLD       ; RESULT TO PORT L
012A A4                       LAID                     ; EQUIVALENT OF
012B A4                       LAID                     ;    SIX NOP'S
012C E2                       JP        LUP42          ; TIMING LOOP OF
                                                       ;    117 1/3 uSEC
                    ;
                    ;
                    ;
                    ;
```

TL/DD/10740–11

```
                        .FORM
                ;
                ; THE FREQUENCY APPROXIMATION IS EQUAL TO THE NUMBER OF
                ; CYCLES OF SINE WAVE DIVIDED BY THE TIME IN THE TOTAL
                ; NUMBER OF LUP42 CYCLES BEFORE THE REPETITION OF THE
                ; ROM TABLE. AS AN EXAMPLE, CONSIDER THE THREE CYCLES
                ; OF SINE WAVE AND 19 VALUES IN THE ASSOCIATED 1336 HZ
                ; ROM TABLE. THE 19 CYCLES OF LUP42 TIMES THE LUP42
                ; TIME OF 117 1/3 uSEC IS DIVIDED INTO THE THREE CYCLES
                ; OF SINE WAVE TO YIELD A VALUE OF 1345.69 HZ AS THE
                ; 1336 HZ APPROXIMATION.
                ;
                ; THE VALUES IN THE ROM TABLES FOR THE DTMF SINE WAVES
                ; SHOULD WRAP AROUND END TO END IN EITHER DIRECTION TO
                ; FORM A SYMETRICAL LOOP. THE FIRST VALUE IN THE ROM
                ; TABLE REPRESENTS THE BASE LINE FOR THAT FREQUENCY.
                ;
                ; THE HIGH BAND DTMF FREQUENCIES HAVE A BASE LINE VALUE
                ; OF 16 AND A MAXIMUM VALUE OF 32. THE LOW BAND DTMF
                ; FREQUENCIES HAVE A BASE LINE VALUE OF 13 AND A
                ; MAXIMUM VALUE OF 26. THIS DIFFERENCE IN BASE LINE
                ; VALUES IS NECESSARY TO SATISFY THE REQUIREMENT OF THE
                ; HIGH BAND FREQUENCIES NEEDING A LEVEL 2 dB ABOVE THE
                ; LEVEL OF THE LOW BAND FREQUENCIES TO COMPENSATE FOR
                ; LOSSES IN TRANSMISSION. THE SUM OF THE TWO BASE LINE
                ; VALUES YIELDS A BASE LINE VALUE OF 29, WHILE THE SUM
                ; OF THE TWO MAXIMUM VALUES YIELDS A MAXIMUM VALUE OF
                ; 58. THUS THE SIX BIT DTMF OUTPUT FROM THE L PORT TO
                ; THE LADDER NETWORK RANGES FROM 0 TO 58, WITH A BASE
                ; LINE VALUE OF 29.
                ;
                ; THE VALUES IN THE DTMF SINE WAVE TABLES ARE
                ; CALCULATED BY COMPUTING THE SINE VALUE AT THE
                ; APPROPIATE POINTS, SCALING THE SINE VALUE UP TO THE
                ; BASE LINE VALUE, AND THEN ADDING THE RESULT TO THE
                ; BASE LINE VALUE. THE FOLLOWING EXAMPLE WILL HELP TO
                ; CLARIFY THIS CALCULATION.
                ;
                ; CONSIDER THE THREE CYCLES OF SINE WAVE ACROSS 19
                ; DATA POINTS FOR THE 1336 HZ DTMF HIGH BAND FREQUENCY.
                ; THE FIRST VALUE IN THE TABLE IS THE BASE LINE VALUE
                ; OF 16. WITH 2 PI RADIANS PER SINE WAVE CYCLE,
                ; THE SUCCEEDING VALUES IN THE TABLE REPRESENT THE
                ; SINE VALUES OF 1 X (6 PI / 19), 2 X (6 PI / 19),
                ; 3 X (6 PI / 19), . . . . , UP TO 18 X (6 PI / 19).
                ; LET US NOW CONSIDER THE SEVENTH AND EIGHTH VALUES
                ; IN THE TABLE, REPRESENTING THE SINE VALUES OF
                ; 6 X (6 PI / 19) AND 7 X (6 PI / 19) RESPECTIVELY.
                ; THE CALCULATIONS OF 16 X SIN [6 X (6 PI / 19)] AND
                ; 16 X SIN [7 X (6 PI / 19)] YIELD VALUES OF - 5.20 AND
                ; 9.83 RESPECTIVELY. ROUNDED TO THE NEAREST INTEGER
```

TL/DD/10740–12

```
                        ; GIVES VALUES OF - 5 AND 10. WHEN ADDED TO THE BASE
                        ; LINE VALUE OF 16, THESE VALUES YIELD THE RESULTS
                        ; 11 AND 26 FOR THE SEVENTH AND EIGHTH VALUES IN THE
                        ; 1336 HZ DTMF TABLE. SYMMETRY IN THE LOOP OF 19 VALUES
                        ; IN THE DTMF TABLE DICTATES THAT THE FOURTEENTH AND
                        ; THIRTEENTH VALUES IN THE TABLE ARE 21 AND 6,
                        ; REPRESENTING VALUES OF 5 AND - 10 FROM THE
                        ; CALCULATIONS.
                        ;
                        ; THE AREA UNDER A HALF CYCLE OF SINE WAVE RELATIVE TO
                        ; THE AREA OF THE SURROUNDING RECTANGLE IS 2/PI, WHERE
                        ; PI RADIANS REPRESENT THE SINE WAVE HALF CYCLE. THIS
                        ; SURROUNDING RECTANGLE HAS A LENGTH OF PI AND A HEIGHT
                        ; OF 1, WITH THE HEIGHT REPRESENTING THE MAXIMUM SINE
                        ; VALUE. CONSEQUENTLY, THE AREA OF THIS SURROUNDING
                        ; RECTANGLE IS PI. THE INTEGRAL OF THE AREA UNDER THE
                        ; HALF SINE WAVE FROM 0 TO PI IS EQUAL TO 2. THE RATIO
                        ; OF 2/PI IS EQUAL TO 63.66 % , SO THAT THE TOTAL OF
                        ; THE VALUES FOR EACH HALF SINE WAVE SHOULD APPROXIMATE
                        ; 63.66 % OF THE SUM OF THE MAX VALUES. THE MAXIMUM
                        ; VALUES (RELATIVE TO THE BASE LINE) ARE 13 AND 16
                        ; RESPECTIVELY, FOR THE LOW AND HIGH BAND FREQUENCIES.
                        ;
                        ;
                        ;
                        ;
                        ;
                        ; LF697:   4 CYCLES OF SINE WAVE SPREAD
                        ;                 ACROSS 49 TIMING LOOP (LUP42) CYCLES
                        ;
                        ;         FREQ. = 4 / (49 X 117 1/3) = 695.73 HZ
                        ;         ERROR = (697 - 695.73) / 697  = - 0.18 %
                        ;
012D 0D                         .BYTE     13,19,24,26,25,20,14,7,2,0
012E 13
012F 18
0130 1A
0131 19
0132 14
0133 0E
0134 07
0135 02
0136 00
0137 01                         .BYTE     1,5,11,18,23,26,25,21,15,9
0138 05
0139 0B
013A 12
013B 17
013C 1A
013D 19
013E 15
```

TL/DD/10740–13

```
    013F 0F
    0140 09
557 0141 03                         .BYTE     3,0,1,4,10,16,22,25,26,23
    0142 00
    0143 01
    0144 04
    0145 0A
    0146 10
    0147 16
    0148 19
    0149 1A
    014A 17
558 014B 11                         .BYTE     17,11,5,1,0,3,8,15,21,25
    014C 0B
    014D 05
    014E 01
    014F 00
    0150 03
    0151 08
    0152 0F
    0153 15
    0154 19
559 0155 1A                         .BYTE     26,24,19,12,6,1,0,2,7
    0156 18
    0157 13
    0158 0C
    0159 06
    015A 01
    015B 00
    015C 02
    015D 07
560                         ;
561                         ;
562                         ; LF770:    1 CYCLE OF SINE WAVE SPREAD
563                         ;                   ACROSS 11 TIMING LOOP (LUP42) CYCLES
564                         ;
565                         ;           FREQ. = 1 / (11 X 117 1/3) = 774.79 HZ
566                         ;           ERROR = (774.79 - 770) / 770  = + 0.62 %
567                         ;
568 015E 0D                             .BYTE     13,20,25,26,23,17,9,3,0,1
    015F 14
    0160 19
    0161 1A
    0162 17
    0163 11
    0164 09
    0165 03
    0166 00
    0167 01
569 0168 06                             .BYTE     6
570                         ;
```

TL/DD/10740–14

```
571                             ;
572                             ; LF852:    1 CYCLE OF SINE WAVE SPREAD
573                             ;                   ACROSS 10 TIMING LOOP (LUP42) CYCLES
574                             ;
575                             ;           FREQ. = 1 / (10 X 117 1/3) = 852.27 HZ
576                             ;           ERROR = (852.27 - 852) / 852  = + 0.03 %
577                             ;
578 0169 0D                             .BYTE     13,21,25,26,21,13,5,1,0,5
    016A 15
    016B 19
    016C 1A
    016D 15
    016E 0D
    016F 05
    0170 01
    0171 00
    0172 05
579                             ;
580                             ;
581                             ; LF941:    1 CYCLE OF SINE WAVE SPREAD
582                             ;                   ACROSS 9 TIMING LOOP (LUP42) CYCLES
583                             ;
584                             ;           FREQ. = 1 / (9 X 117 1/3) = 946.97 HZ
585                             ;           ERROR = (946.97 - 941) / 941  = + 0.63 %
586                             ;
587 0173 0D                             .BYTE     13,21,26,24,18,8,2,0,5
    0174 15
    0175 1A
    0176 18
    0177 12
    0178 08
    0179 02
    017A 00
    017B 05
588                             ;
589                             ;
590                             ;
591                             ; HF1209:   1 CYCLE OF SINE WAVE SPREAD
592                             ;                   ACROSS 7 TIMING LOOP (LUP42) CYCLES
593                             ;
594                             ;           FREQ. = 1 / (7 X 117 1/3) = 1217.53 HZ
595                             ;           ERROR = (1217.53 - 1209) / 1209  = + 0.71 %
596                             ;
597 017C 10                             .BYTE     16,29,32,23,9,0,3
    017D 1D
    017E 20
    017F 17
    0180 09
    0181 00
    0182 03
598                             ;
```

TL/DD/10740-15

```
599                             ;
600                             ; HF1336:  3 CYCLES OF SINE WAVE SPREAD
601                             ;                  ACROSS 19 TIMING LOOP (LUP42) CYCLES
602                             ;
603                             ;          FREQ. = 3 / (19 X 117 1/3) = 1345.69 HZ
604                             ;          ERROR = (1345.69 - 1336) / 1336  = + 0.73 %
605                             ;
606 0183 10                             .BYTE     16,29,31,19,4,0,11,26,32,24
    0184 1D
    0185 1F
    0186 13
    0187 04
    0188 00
    0189 0B
    018A 1A
    018B 20
    018C 18
607 018D 08                             .BYTE     8,0,6,21,32,28,13,1,3
    018E 00
    018F 06
    0190 15
    0191 20
    0192 1C
    0193 0D
    0194 01
    0195 03
608                             ;
609                             ;
610                             ; HF1477:  4 CYCLES OF SINE WAVE SPREAD
611                             ;                  ACROSS 23 TIMING LOOP (LUP42) CYCLES
612                             ;
613                             ;          FREQ. = 4 / (23 X 117 1/3) = 1482.21 HZ
614                             ;          ERROR = (1482.21 - 1477) / 1477  = + 0.35 %
615                             ;
616 0196 10                             .BYTE     16,30,29,14,1,4,20,32,26,10
    0197 1E
    0198 1D
    0199 0E
    019A 01
    019B 04
    019C 14
    019D 20
    019E 1A
    019F 0A
617 01A0 00                             .BYTE     0,8,24,32,22,6,0,12,28,31
    01A1 08
    01A2 18
    01A3 20
    01A4 16
    01A5 06
    01A6 00
```

TL/DD/10740–16

```
    01A7 0C
    01A8 1C
    01A9 1F
618 01AA 12                         .BYTE     18,3,2
    01AB 03
    01AC 02
619                         ;
620                         ;
621                         ; HF1633:  4 CYCLES OF SINE WAVE SPREAD
622                         ;                  ACROSS 21 TIMING LOOP (LUP42) CYCLES
623                         ;
624                         ;          FREQ. = 4 / (21 X 117 1/3) = 1623.38 HZ
625                         ;          ERROR = (1633 - 1623.38) / 1633  = - 0.59 %
626                         ;
627 01AD 10                         .BYTE     16,31,27,9,0,11,29,30,14,0
    01AE 1F
    01AF 1B
    01B0 09
    01B1 00
    01B2 0B
    01B3 1D
    01B4 1E
    01B5 0E
    01B6 00
628 01B7 07                         .BYTE     7,25,32,18,2,3,21,32,23,5
    01B8 19
    01B9 20
    01BA 12
    01BB 02
    01BC 03
    01BD 15
    01BE 20
    01BF 17
    01C0 05
629 01C1 01                         .BYTE     1
630                         ;
631                         ;
632                         ;
```

TL/DD/10740–17

```
633                                     .FORM
634                             ;
635                             ; DTMF KEYBOARD DECODE SUBROUTINE (KBRDEC)
636                             ;
637                             ; KEYBOARD INPUT DATA IS IN ACCUMULATOR WITH A
638                             ;    LOW TRUE FORMAT AS FOLLOWS:
639                             ;        BITS 7 TO 4 : LOW TRUE COLUMN VALUE (E,D,B,7)
640                             ;        BITS 3 TO 0 : LOW TRUE ROW VALUE (E,D,B,7)
641                             ;
642                             ; ASSUMPTION MADE THAT COLUMN STROBES (LOW TRUE) ARE
643                             ;    OUTPUT, WHILE ROW VALUES (LOW TRUE) ARE INPUT.
644                             ;
645                             ; LOW TRUE COLUMN/ROW INPUT DIGIT IN ACCUMULATOR IS
646                             ;     TRANSFORMED INTO A DTMF HEX DIGIT KEY VALUE
647                             ;
648                             ; TABLE LOOKUP TRANSFORMATION CHECKS FOR MULTIPLE KEYS,
649                             ;     NO KEY, OR NO COLUMN SELECT, AND THEN PRODUCES
650                             ;     A DTMF HEX DIGIT KEY VALUE WITH A BINARY FORMAT
651                             ;     OF 0000RRCC FOR A SINGLE KEY INPUT,
652                             ;     WHERE    -  RR IS LOW BAND (LB) FREQUENCY SELECT
653                             ;              -  CC IS HIGH BAND (HB) FREQUENCY SELECT
654                             ;
655                             ; KBRDEC SUBROUTINE IS EXITED WITH A RETURN (RET)
656                             ;     COMMAND TO INDICATE MULTIPLE KEYS, NO KEY,
657                             ;     OR NO COLUMN SELECT
658                             ;
659                             ; KBRDEC SUBROUTINE IS EXITED WITH A RETURN AND SKIP
660                             ;     (RETSK) COMMAND TO INDICATE A SINGLE KEY ENTRY
661                             ;
662                             ;
663     0200                            .=0200
664                             ;
665                             ; LOW TRUE TRANSLATION TABLE  - ONLY E,D,B,7 ACCEPTABLE
666                             ;
667 0200 C0                             .BYTE  0C0,0C0,0C0,0C0,0C0,0C0,0C0,0C
    0201 C0
    0202 C0
    0203 C0
    0204 C0
    0205 C0
    0206 C0
    0207 0C
668 0208 C0                             .BYTE  0C0,0C0,0C0,8,0C0,4,0,0C0
    0209 C0
    020A C0
    020B 08
    020C C0
    020D 04
    020E 00
    020F C0
669                             ;
```

TL/DD/10740–18

```
670                             ;
671 0210 5F                     KBRDEC:   LD        B,#KDATA
672 0211 A6                               X         A,[B]     ; STORE LOW TRUE
673 0212 AE                               LD        A,[B]     ;   COLUMN/ROW VALUE
674 0213 95F0                             AND       A,#0F0    ; EXTRACT LOW TRUE COLUMN
675 0215 65                               SWAP      A         ;   & PUT IN LOWER NIBBLE
676 0216 A4                               LAID                ; 0000CC00 FROM TABLE
677 0217 A0                               RC                  ; SHIFT TABLE VALUE DOWN
678 0218 B0                               RRC       A         ;   TWO BITS TO PRODUCE
679 0219 B0                               RRC       A         ;   000000CC
680 021A A6                               X         A,[B]     ; STORE RESULT
681 021B 950F                             AND       A,#0F     ; EXTRACT LOW TRUE ROW
682 021D A4                               LAID                ; 0000RR00 FROM TABLE
683 021E 84                               ADD       A,[B]     ; ADD TO PRODUCE 0000RRCC
684 021F 930F                             IFGT      A,#0F     ; RETURN IF MULTIPLE KEYS,
685 0221 8E                               RET                 ;    NO KEYS, OR NO COLUMN
686 0222 8D                               RETSK               ; RETURN AND SKIP
687                                                           ;    IF SINGLE KEY
688                             ;
689                             ;
690                             ;
691                                       .END
```

TL/DD/10740–19

```
B       00FE          BYP1    0072          BYP2    0075          BYPA    0019
BYPB    001B          CNTRL   00EE          DTMFGP  0040          DTMFLP  008E
FINI    0086          FRLUP   00A0          HFPTR   0007          HFTADR  000B
HFTBSZ  000A          INTRPT  00FF  *       KBRDEC  0210          KDATA   0000
LFPTR   0005          LFTADR  0009          LFTBSZ  0008          LOOP    0006
LUP     004D          LUP1    006C          LUP2    0078          LUP42   010F
PORTD   00DC          PORTGC  00D5          PORTGD  00D4          PORTI   00D7
PORTLC  00D1          PORTLD  00D0          PSW     00EF          R0      00F0
R1      00F1          R2      00F2          R3      00F3          SP      00FD
START   0000  *       TAUHI   00ED  *       TAULO   00EC  *       TEMP    0006
TMRHI   00EB  *       TMRLO   00EA          TRUN    0004          X       00FC
```

TL/DD/10740–20

2-Way Multiplexed LCD Drive and Low Cost A/D Converter Using V/F Techniques with COP8 Microcontrollers

National Semiconductor
Application Note 673
Volker Soffel

ABSTRACT

This application note is intended to show a general solution for implementing a low cost A/D and a 2-way multiplexed LCD drive using National Semiconductor's COP840C 8-bit microcontroller. The implementation is demonstrated by means of a digital personal scale. Details and function of the weight sensor itself are not covered in this note. Also the algorithms used to calculate the weight from the measured frequency are not included, as they are too specific and depend on the kind of sensor used.

Typical Applications

- Weighing scales
- Sensors with voltage output
- Capacitive or resistive sensors
- All kinds of measuring equipment
- Automotive test and control systems

Features

- 2-way multiplexed LCD drive capability up to 30 segments (4 digit and 2 dot points)
- Precision frequency measurement
- Low current consumption
- Current saving HALT mode
- Additional computing power for application specific tasks

INTRODUCTION

Today's most popular digital scales all have the following characteristics:

They are battery powered and use a LCD to display the weight. Instead of using a discrete A/D-converter, in many cases a V/F converter is used, which converts an output voltage change of the weight sensor to a frequency change. This frequency is measured by a microcontroller and is used to calculate the weight. The advantages of a V/F over an A/D converter are multifold. Only one line from the V/F to the microcontroller is needed, whereas a parallel A/D needs at least 8 lines or even more (National also offers A/Ds with serial output). A V/F can be constructed very simply using National Semiconductor's low cost, precision voltage to frequency converters LM331 or LM331A. Other possibilities are using Op-amps or a 555-timer in astable mode.

V/F-CONVERSION

Hardware

The basic configuration of the scale described in this application note is shown in *Figure 1*.

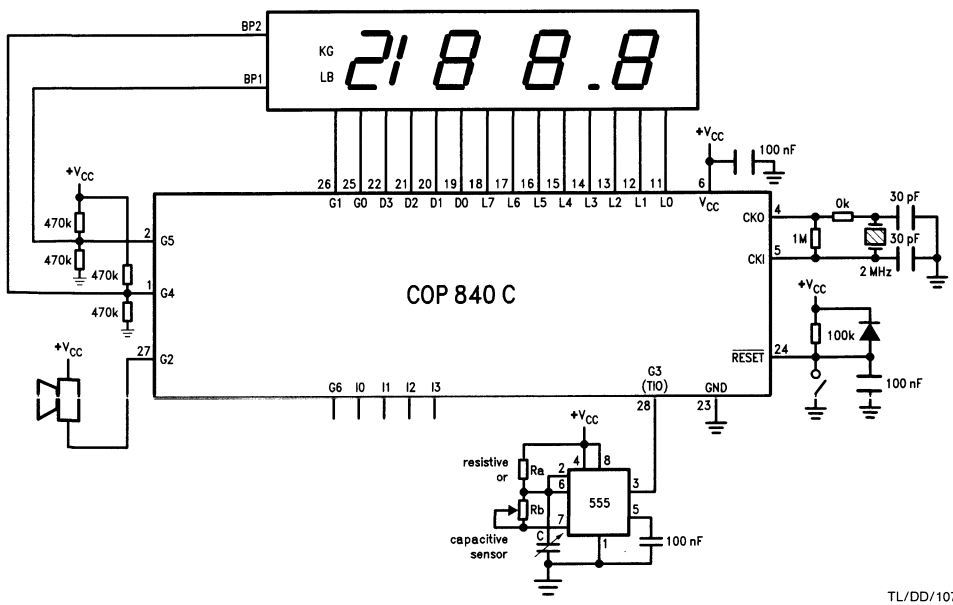


FIGURE 1. System Diagram

A capacitive or resistive sensor's weight related capacitance or resistance change is transformed by a 555 timer (in astable mode) to a change of frequency. The output frequency f is determined by the formula:

$$f = 1.44/((Ra + 2Rb) *C)$$

The output high time is given by:

$$t1 = 0.693* (Ra + Rb) *C$$

The output low time is given by:

$$t2 = 0.693* Rb* C$$

This frequency is measured using the COP800 16-bit timer in the "input capture" mode. After calculation, the weight is displayed on a 2-way multiplexed LCD. Using this configuration a complete scale can be built using only two ICs and a few external passive components.

For more information on V/F converters generally used with voltage output sensors, refer to the literature listed in the reference section.

Frequency Measurement

The COP 16-bit timer is ideally suited for precise frequency measurements with minimum software overhead. This timer has three programmable operating modes, of which the "input capture" mode is used for the frequency measurement. Allocated with the timer is a 16-bit "autoload/capture register". The G3-I/O-pin serves as the timer capture input (TIO). In the "input capture" mode the timer is decremented with the instruction cycle frequency (tc). Each positive going edge at TIO (also neg. edge programmable) causes the timer value to be copied automatically to the autoload/capture register without stopping the timer or destroying its contents. The "timer pending" flag (TPND) in the PSW-register is set to indicate a capture has occurred, and if the timer-interrupt is enabled, an interrupt is generated. The frequency measurement routine listed below executes the following operations (refer to the RAM/register definition file listed at the beginning for symbolic names used in the routines):

The timer is preset with FFFF Hex and is started by setting the TRUN bit, after which the software checks the TPND-flag in a loop (timer interrupt is disabled). When the TPND flag is set the first time, the contents of the capture register is saved in RAM locations STALO and STAHI (start value). The TPND pending flag now must be reset by the software. Then, another 255 positive going edges are counted (equal to 255 pulses) before the capture register is saved in RAM locations ENDLO, ENDHI (end value). The shortest time period that can be measured depends on the number of instruction cycles needed to save the capture register, because with the next positive going edge on TIO the contents of the capture register is overwritten (worst case is 18 instruction cycles, which equals a max. frequency of 55.5 kHz at tc = 1 μs).

The end-value is subtracted from the start-value and the result is restored in RAM locations STALO, STAHI. This value can then be used to calculate the time period of the frequency applied to TIO (G3) by multiplying it with the tc-time and dividing the result by the number of pulses measured (N = 255).

$$T = (\text{startvalue} - \text{endvalue}) *tc/N$$

```
;THE FOLLOWING "INCLUDE FILE" IS USED
;AS PART OF THE DEFINITION- AND INITIALIZATION PHASE
;IN COP800 PROGRAMS.
;REGISTER NAMES,CONTROL BITS ETC ARE NAMED IN THE
;SAME WAY IN THE COP800 DATA-SHEETS.


;        --- COP800  MEMORY MAPPED ---


; ******************************************************
; * PORT  -, CONFIGURATION  -  AND CONTROL REGISTERS *
; ******************************************************


     PORTLD     =     0D0          ; L-PORT DATA REGISTER
     PORTLC     =     0D1          ; L-PORT CONFIGURATION
```

TL/DD/10788-2

7

```
PORTLP    =    0D2          ; L-PORT INPUT REGISTER

PORTGD    =    0D4          ; G-PORT DATA REGISTER
PORTGC    =    0D5          ; G-PORT CONFIGURATION
PORTGP    =    0D6          ; G-PORT INPUT REGISTER

PORTD     =    0DC          ; D-PORT (OUTPUT)
PORTI     =    0D7          ; I-PORT (INPUT)


SIOR      =    0E9          ; MWIRE SHIFT REGISTER
TMRLO     =    0EA          ; TIMER LOW-BYTE
TMRHI     =    0EB          ; TIMER HIGH-BYTE
TAULO     =    0EC          ; T.-AUTO REG.LOW BYTE
TAUHI     =    0ED          ; T.-AUTO REG.HIGH BYTE

CNTRL     =    0EE          ; CONTROL REGISTER
PSW       =    0EF          ; PSW-REGISTER
               .FORM
;            **********************
;            *  CONSTANT DECLARE  *
;            **********************


;        ---  CONTROL REGISTER BITS ---


S0        =     00      ; MICROWIRE CLOCK DIVIDE BY
                        ;       --- BIT 0 ---
S1        =     01      ; MICROWIRE CLOCK DIVIDE BY
                        ;       --- BIT 1 ---
IEDG      =     02      ; EXTERNAL INTERRUPT EDGE
                        ; POLARITY SELECT (0=RISING
                        ; EDGE,1=FALLING EDGE)
MSEL      =     03      ; ENABLE MICROWIRE FUNCTION
                        ;     --- SO AND SK ---
TRUN      =     04      ; START/STOP THE TIM/COUNT.
                        ;     (1=RUN;0=STOP)
TEDG      =     05      ; TIMER INPUT EDGE POL.SEL.
                        ; (0=RIS. EDGE;1=FAL. EDGE)
CSEL      =     06      ; SELECTS THE CAPTURE MODE
                        ;
TSEL      =     07      ; SELECTS THE TIMER MODE


;         ---  P S W    REGISTER ---


GIE       =     00      ; GLOBAL INTERRUPT ENABLE
```

TL/DD/10788–3

```
ENI         =       01      ; EXTERNAL INTERRUPT ENABLE
BUSY        =       02      ; MICROWIRE BUSY SHIFTING
IPND        =       03      ; EXTERNAL INTERR. PENDING
ENTI        =       04      ; TIMER INTERRUPT ENABLE
TPND        =       05      ; TIMER INTERRUPT PENDING
C           =       06      ; CARRY FLAG
HC          =       07      ; HALF CARRY FLAG

;****           RAM-DEFINITIONS      *****


        BCDLO     = 000   ;CALCULATED WEIGHT IN BCD
                          ;LOW BYTE
        BCDHI     = 001   ;CALCULATED WEIGHT IN BCD
                          ;HIGH BYTE
        MWBUF0    = 003   ;7SEGMENT DATA FOR LCD DISPL
                          ;L-PORT
        MWBUF1    = 004   ;D-PORT
        MWBUF2    = 005   ;G-PORT
        OFF1      = 006   ;OFFSET REGISTERS FOR
        OFF2      = 007   ;7 SEGMENT CODE TABLE
        OFF3      = 008   ;

        STALO     = 009   ;START VALUE,LOW BYTE
        STAHI     = 00A   ;START VALUE,HIGH BYTE
        ENDLO     = 00B   ;END VALUE LOW BYTE
        ENDHI     = 00C   ;END VALUE HIGH BYTE

        DIV0      = 00D   ;DIVISOR FOR DINBI248 ROUTINE

       ;022.. 02F RESERVED FOR STACK WITH COP820
       ;062..06F  RESERVED FOR STACK WITH COP840


;****           REGISTER DEFINITIONS      ****

        COUNT  = 0F0
        COUNT2 = 0F1
        COUNT3 = 0F2
        FLAG   = 0FF      ;FLAG REGISTER


;****           BIT DEFINITIONS FLAG REGISTER  ****

        POUND = 04      ;POUND=1:DISPLAY POUND SEGMENT
                        ;POUND=0:DISPLAY kg SEGMENT

;*****       G-PORT BIT DEFINITIONS         *****

        BP1   = 05      ;BACKPLANE 1
```

TL/DD/10788–4

```
        BP2  = 04        ;BACKPLANE 2

;TIME OF 255 PULSES, USING TIMER INPUT CAPTURE MODE

FMEAS:
                         ;PERIOD TIME=
                         ;(START-ENDVALUE)*tc/255
                         ;DIFFERENCE START-ENDVALUE
                         ;IS STORED IN ENDLO,ENDHI
        LD       COUNT,#000  ;LOAD PULSE COUNTER (255 PULSES)
        LD       X,#TAULO    ;POINT TO AUTO REG. LOW B.
        LD       B,#TMRLO    ;PRESET TIMER
        LD       [B+],#0FF   ;REG. WITH FFFFh
        LD       [B],#0FF
        LD       B,#CNTRL
        LD       [B+],#0D0   ;CNTRL-REG.: TIMER CAPTURE
                             ;MODE,TIO POS. TRIGGERED,
                             ;START TIMER

        RBIT     #TPND,[B]   ;RESET TIMER PENDING FLAG
L1:     IFBIT    #TPND,[B]
        JP       SSTORE
        JP       L1
SSTORE:                      ;STORE START VALUE
        RBIT     #TPND,[B]
        LD       A,[X+]      ;LOAD TIMER CAPTURE REG.
                             ;LOW BYTE
        X        A,STALO     ;STORE IN RAM
        LD       A,[X-]      ;LOAD HIGH BYTE CAPTURE,
                             ;POINT TO LOW BYTE CAPTURE
        X        A,STAHI     ;STORE IN RAM
        LD       B,#PSW
L256:
        IFBIT    #TPND,[B]
        JP       DCOU
        JP       L256
DCOU:   RBIT     #TPND,[B]   ;RESET TIMER PENDING FLAG
        DRSZ     COUNT       ;DECREMENT PULSE COUNTER
                             ;COUNTER = 0 ?
        JP       L256        ;NO,LOOP 'TIL 255 PULSES
                             ;HAVE BEEN MEASURED

ESTORE:                      ;STORE END VALUE

        LD       CNTRL,#00   ;STOP TIMER
        LD       B,#STALO    ;POINT TO START VALUE LOW BYTE
        LD       A,[X+]      ;LOAD END VALUE LOW BYTE
        X        A,[B]       ;LOAD ACCU WITH STARTVALUE LOW BYTE
                             ;& STALO WITH END VALUE LOW BYTE
```

TL/DD/10788-5

```
        SC
        SUBC     A,[B]       ;SUBTRACT ENDVALUE LOW BYTE
                             ;FROM STARTVALUE LOW BYTE
        X        A,[B+]      ;STORE RESULT IN STALO,
                             ;POINT TO STAHI
        LD       A,[X]       ;LOAD ACCU WITH ENDVALUE HIGH BYTE
        X        A,[B]       ;LOAD ACCU WITH STARTVALUE HIGH BYTE
                             ;& STAHI WITH ENDVALUE HIGH BYTE
        SUBC     A,[B]       ;SUBTRACT ENDVALUE HIGH BYTE FROM
                             ;STARTVALUE HIGH BYTE
        X        A,[B]       ;STORE RESULT IN STAHI
        RET
        .END
```

TL/DD/10788-6

2-WAY MULTIPLEXED LCD DRIVE

Today a wide variety of LCDs, ranging from static to multiplex rates of 1:64 are available on the market. The multiplex rate of a LCD can be determined by the number of its backplanes (segment-common plate). The higher the multiplex rate the more individual segments can be controlled using only one line. e.g. a static LCD only has one backplane; only one segment can be controlled with one line. A two-way multiplexed LCD has two backplanes and two segments can be controlled with one line, etc.

Common to all LCDs is the fact that the drive voltage applied to the backplane(s) and segments has to be alternating. DC-components higher than 100 mV can cause electrochemical reactions (refer to manufacturer's spec), which reduce reliability and lifetime of the display.

If the multiplex ratio of the LCD is N and the amount of available outputs is M, the number of segments that can be driven is:

$$S = (M - N) * N$$

So the maximum number of a 2-way mux LCD's segments that can be driven with a COP800 in 28-pin package (if all outputs can be used to drive the LCD) is:

$$S = (18 - 2) * 2 = 32$$

During one LCD refresh cycle tx (typical values for $1/tx = fx$ are in the range 30 Hz . . . 60 Hz), three different voltages levels: Vop, 0.5*Vop and 0V have to be generated. The "off" voltage across a segment is not 0V as with static LCDs and also the "on" voltage is not Vop, but only a fraction of it. The ratio of "on" to "off" r.m.s.-voltage (discrimination) is determined by the multiplex ratio and the number of voltage levels involved. The most desirable discrimination ratio is one that maximizes the ratio of V_{ON} to V_{OFF}, allowing the maximum voltage difference between activated and non-activated states. In general the maximum achievable ratio for any particular value of N is given by:

$$(V_{ON}/V_{OFF})\ max = SQR((SQR(N) + 1)/(SQR(N) - 1))$$

$$SQR = \text{square root}$$

Using this formula the maximum achievable discrimination ratio for a 2-way multiplex LCD is 2.41, however, it is also possible to order a customized display with a smaller ratio. For ease of operation, most LCD drivers use equal voltage steps (0V, 0.5 *Vop, Vop). Thus a discrimination ratio of 2.24 is achieved. When using the COP800 to drive a 2-way multiplexed LCD the only external hardware required to achieve the three voltage steps are 4 equal resistors that form two voltage dividers—one for each backplane *(Figure 1)*. The procedure is to set G4 and G5 to "0" for 0V, to HI-Z (TRI-STATE®) for 0.5*Vop and to "1" in order to establish Vop at the backplane electrodes.

With the COP800 each I/O pin can be set individually to TRI-STATE, "1" or "0", so this procedure can be implemented very easily.

The current consumption of typical LCDs is in the range of 3 μA to 4 μA (at Vop = 4.5V, refresh rate 60 Hz) per square centimeter of activated area. Thus the backplane and segment terminals can be treated as Hi-Z loads. At high refresh rates the LCD's current consumption increases dramatically, which is the reason why many LCD manufacturers recommend not using a refresh frequency higher than 60 Hz.

Timing Considerations

As shown in *Figures 2* and *3*, one LCD refresh cycle tx is subdivided into four equally distant time sections ta, tb, tc and td during which the backplane and segment terminals have to be updated in order to switch a specific segment on or off. Considering a refresh frequency of 50 Hz (tx = 20 ms) ta, tb, tc and td are equal to 5 ms; a COP800 running from an external clock of 2 MHz has an internal instruction cycle time of 5 μs and a typical current consumption of less than 350 μA (at V_{CC} = 3V and room temperature), thus meeting both the requirements of low current consumption and additional computing power between LCD refreshes.

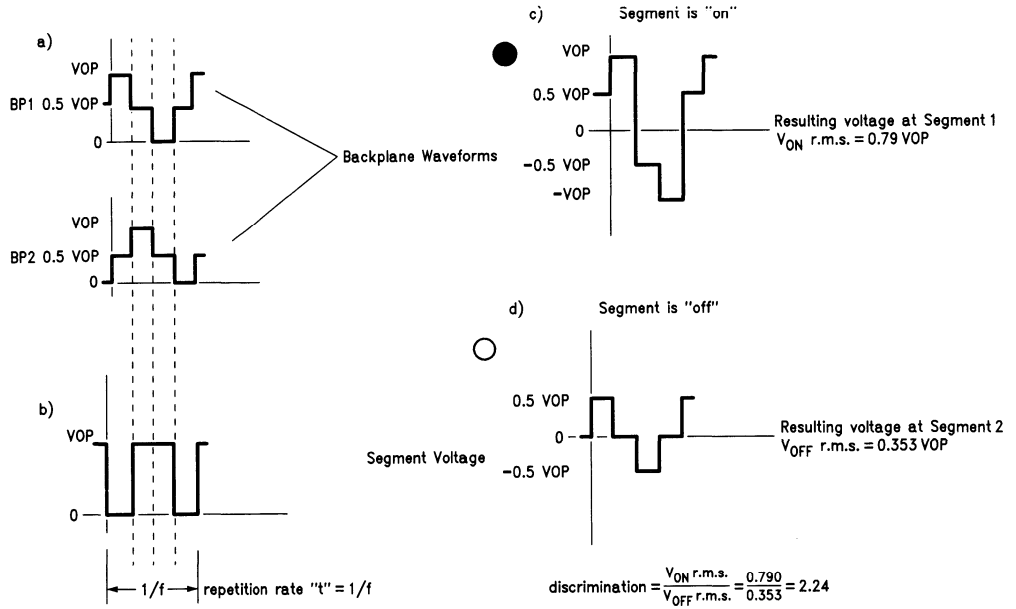
The timing is done using the COP800's 16-bit timer in the PWM autoload mode. The timer and the assigned 16-bit autoload register are preset with proper values. By setting the TRUN-flag in the CNTRL-register the timer is decremented each instruction cycle. A flag (TPND) is set at underflow and the timer is automatically reloaded with the value stored in the autoload-register. Timer underflow can also be programmed to generate an interrupt.

Segment Control

Figure 2 shows the voltage-waveforms applied to the two backplane-electrodes (a) and the waveform at a segement-electrode (b), which is needed to switch segment A on and segment B off. The resulting voltage over the segments (c and d) is achieved by subtracting waveform (b) from BP1 (segment A) and waveform (b) from BP2 (segment B).

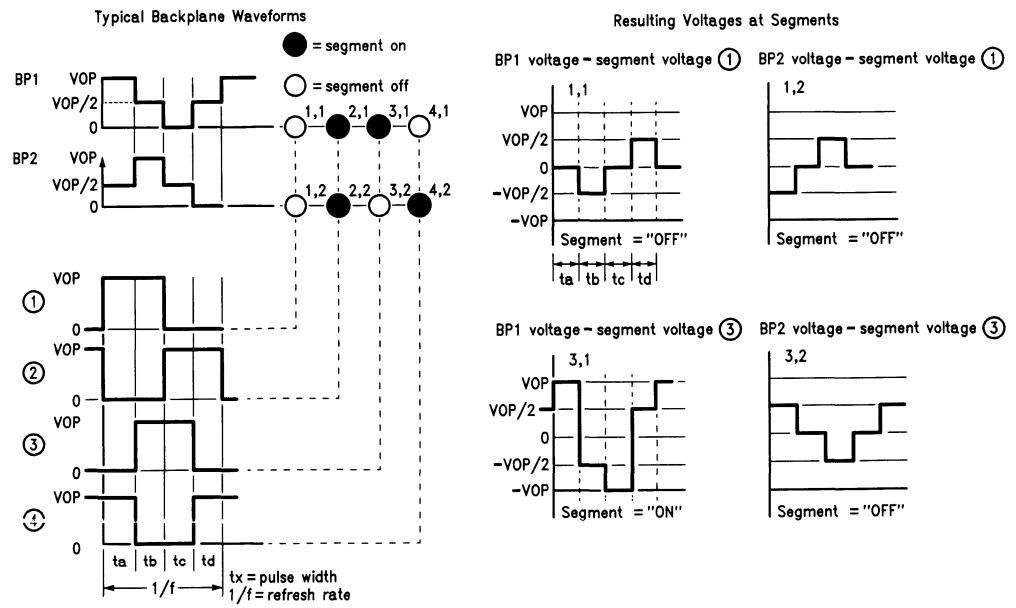
Figure 3 shows the four different waveforms which must be generated to meet all possible combinations of two segments connected to the same driving terminal (off-off, on-off, off-on, on-on).

Figure 4 shows the internal segment and backplane connections for a typical 2-way mux LCD.



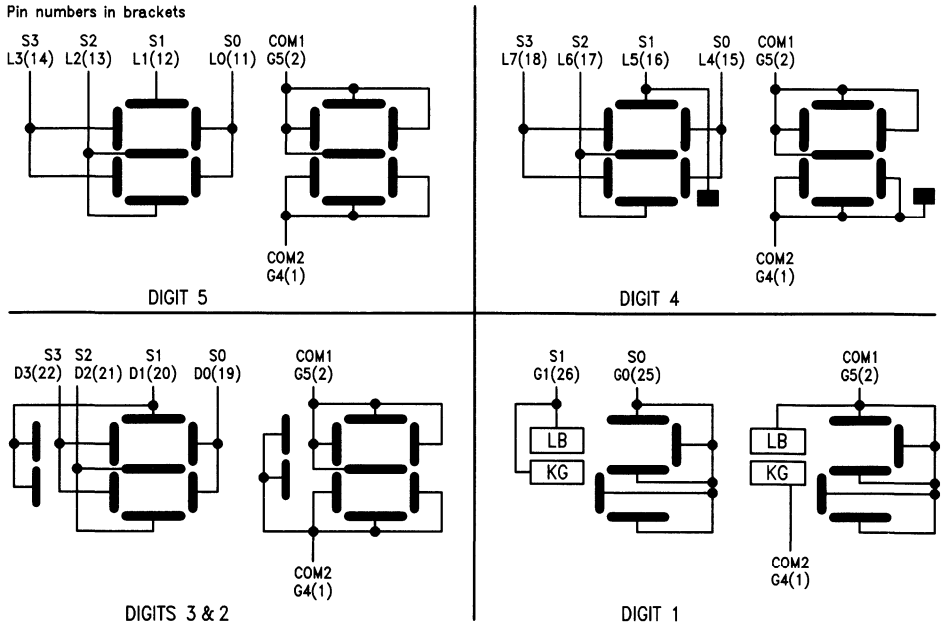
TL/DD/10788–7

FIGURE 2. LCD Waveforms



TL/DD/10788–8

FIGURE 3. Backplane and Segment Voltage Scheme for 1:2 Mux LCD-Drive



TL/DD/10788–9

FIGURE 4. Customized LCD Display (Backplane and Segment Organization)

LCD Drive Subroutine

The LCD drive subroutine DISPL converts a 16-bit binary value to a 24-bit BCD-value for easier display data fetch. The drive subroutine itself is built up of a main routine doing the backplane refresh and 7 subroutines (SEG0, SEG1, SEG2, SEG3, SEGOUT, TTPND, DISPD). The subroutines SEG0 to SEG4 are used to get the LCD segment data from a look-up table in ROM for time phases ta, tb, tc and td respectively. Subroutine SEGOUT writes the segment data for each time phase to the corresponding output ports. One time phase takes 5 ms, giving a total refresh cycle time of 20 ms (50 Hz). The exact timing is done by using the COP800 16-bit timer in the PWM autoload mode. In that mode the timer is reloaded with the value stored in the autoload register on every timer underflow. At the same time the timer pending flag is set. The subroutine TTPND checks this flag in a loop. If the timer pending flag is set, this subroutine resets it and returns to the calling program. Thus a 5 ms time delay is created before the segment and backplane data for the next time phase is written to the output ports. Finally the subroutine DISPD switches off the LCD by setting the backplane and segment connections to "0". In this digital scale application a frequency measurement is made while the LCD is off. Then the weight is calculated from this frequency and is displayed for 10s. After this 10s the LCD is switched off again and the COP800 is programmed to enter the current saving HALT mode ($I_{DD} < 10\ \mu A$). A new weight cycle on the digital scale is initiated by pressing a push button, which causes a reset of the microcontroller.

CONCLUSIONS

National Semiconductor's COP800 Microcontroller family is ideally suited for use with V/F converters and 2-way multiplexed LCDs, as they offer features, which are essential for these types of applications. The high resolution, 3-mode programmable 16-bit timer allows precise frequency measurement in the input capture mode with minimum software overhead. The timer's PWM autoreload mode offers an easy way to implement a precise timebase for the LCD refresh. The COP800's programmable I/O ports provide flexibility in driving 2-way multiplexed LCDs directly. The COP800 family, fabricated using M2CMOS technology, offers both low voltage (min V_{CC} of 2.5V) and low current drain.

REFERENCES

1. National Semiconductor, "Linear Databook 2, Rev. 1" LM331, LM331A datasheets pages 3–285 ff.
2. National Semiconductor, "Linear Applications Databook, 1986", "Versatile monolithic V/Fs can compute as well as convert with high accuracy", pages 1213 ff.
3. National Semiconductor, "Microcontrollers Databook, Rev. 1", COP820C/COP840C datasheets pages 2–7 ff.
4. U. Tietze, Ch. Schenk, "Halbleiter-Schaltungstechnik" 8.Auflage 1986, Springer Verlag, ISBN 0-387-16720-X, "Funktionsgeneratoren mit steuerbarer Frequenz", pages 465 ff, "Multivibratoren", pages 183 ff.
5. Lucid Displays, "LCD design guide", English Electric Valve Company Ltd., Chelmsford, Essex, Great Britain.

APPENDIX—Software Routines

```
;LOOKUP TABLE FOR  CUSTOMIZED 2-WAY MULIPLEX LCD

. = X'200       ;START LOOK-UP TABLE AT ROM ADRESS 200
;TIMEPHASE Ta 7 SEGMENT DATA
.BYTE   004     ;"0" AND ".0"
.BYTE   00E     ;"1" AND ".1"
.BYTE   008     ;"2" AND ".2"
.BYTE   008     ;"3" AND ".3"
.BYTE   002     ;"4" AND ".4"
.BYTE   001     ;"5" AND ".5"
.BYTE   001     ;"6" AND ".6"
.BYTE   00C     ;"7" AND ".7"
.BYTE   000     ;"8" AND ".8"
.BYTE   000     ;"9" AND ".9"
.BYTE   00F     ;" " AND ". "

;SPECIAL SEGMENTS TIMPHASE Ta
.BYTE   001     ;"LB"
.BYTE   000     ;"LB 2"
.BYTE   003     ;"KG"
.BYTE   002     ;"KG 2"

. = .+ 1
;TIMEPHASE Tb 7 SEGMENT DATA
.BYTE   002     ;"0"
.BYTE   00E     ;"1"
.BYTE   003     ;"2"
.BYTE   00A     ;"3"
.BYTE   00E     ;"4"
.BYTE   00A     ;"5"
.BYTE   002     ;"6"
.BYTE   00E     ;"7"
```

TL/DD/10788–10

```
        .BYTE   002     ;"8"
        .BYTE   00A     ;"9"
        .BYTE   00F     ;" "
        .BYTE   000     ;".0"
        .BYTE   00C     ;".1"
        .BYTE   001     ;".2"
        .BYTE   008     ;".3"
        .BYTE   00C     ;".4"
        .BYTE   008     ;".5"
        .BYTE   000     ;".6"
        .BYTE   00C     ;".7"
        .BYTE   000     ;".8"
        .BYTE   008     ;".9"
        .BYTE   00D     ;". "

        .LOCAL
TTPND:
        LD      B,#PSW
$LOOP:
        IFBIT   #TPND,[B]
        JP      $END
        JP      $LOOP
$END:
        RBIT    #TPND,[B]
        LD      B,#PORTGD
        RET
        .LOCAL

        . = .+1
        ;TIMEPHASE Tc 7 SEGMENT DATA
        .BYTE   00B     ;"0" AND ".0"
        .BYTE   001     ;"1" AND ".1"
        .BYTE   007     ;"2" AND ".2"
        .BYTE   007     ;"3" AND ".3"
        .BYTE   00D     ;"4" AND ".4"
        .BYTE   00E     ;"5" AND ".5"
        .BYTE   00E     ;"6" AND ".6"
        .BYTE   003     ;"7" AND ".7"
        .BYTE   00F     ;"8" AND ".8"
        .BYTE   00F     ;"9" AND ".9"
        .BYTE   000     ;" " AND ". "

COPY:                       ;COPY 2BYTES POINTED TO
                            ;BY B AND B+1 TO RAM
                            ;POINTED TO BY X AND X+1
        LD      A,[B+]
        X       A,[X+]
        LD      A,[B+]
        X       A,[X+]
        RET
        .LOCAL
```

TL/DD/10788–11

```
        ;TIMEPHASE Td 7 SEGMENT DATA
        .BYTE   00D     ;"0"
        .BYTE   001     ;"1"
        .BYTE   00C     ;"2"
        .BYTE   005     ;"3"
        .BYTE   001     ;"4"
        .BYTE   005     ;"5"
        .BYTE   00D     ;"6"
        .BYTE   001     ;"7"
        .BYTE   00D     ;"8"
        .BYTE   005     ;"9"
        .BYTE   000     ;" "
        .BYTE   00F     ;".0"
        .BYTE   003     ;".1"
        .BYTE   00E     ;".2"
        .BYTE   007     ;".3"
        .BYTE   003     ;".4"
        .BYTE   007     ;".5"
        .BYTE   00F     ;".6"
        .BYTE   003     ;".7"
        .BYTE   00F     ;".8"
        .BYTE   007     ;".9"
        .BYTE   002     ;". "

        ;SPECIAL SEGMENTS TIMEPHASE Tb
        .BYTE   003     ;"LB"
        .BYTE   003     ;"LB 2 "
        .BYTE   001     ;"KG"
        .BYTE   001     ;"KG 2"

        ;SPECIAL SEGMENTS TIMPHASE Tc
        .BYTE   002     ;"LB"
        .BYTE   003     ;"LB 2"
        .BYTE   000     ;"KG"
        .BYTE   001     ;"KG 2"

        ;SPECIAL SEGMENTS TIMEPHASE Td
        .BYTE   000     ;"LB"
        .BYTE   000     ;"LB 2"
        .BYTE   002     ;"KG"
        .BYTE   002     ;"KG 2"

        .END
;DISPL:
;INPUT PARAMETER: COUNT2 =RAM REGISTER, WHICH CONTAINS
;THE DISPLAY TIME IN SEC.
;EXAMPLE COUNT2= 1-> DISPLAY TIME IS 1SEC.

;LCD DRIVE ROUTINE FOR CUSTOMIZED 2 WAY MULTIPLEX
;LCD
```

TL/DD/10788-12

```
;ROUTINE CONVERTS BCD DATA STORED IN RAM LOCATIONS
;BCDLO, BCDHI INTO LCD OUTPUT DATA STORED AT
;MWBUF0 = LPORT DATA
;MWBUF1 = DPORT DATA
;MWBUF2 = G-PORT DATA (G0,G1 ONLY, OTHER BITS
;                      STAY UNCHANGED)
;SUBROUTINES INCLUDED:
;SEG0: GETS LCD SEGMENT DATA FOR TIMEPHASE TA
;SEG1: GETS LCD SEGMENT DATA FOR TIMEPHASE TB
;SEG2: GETS LCD SEGMENT DATA FOR TIMEPHASE TC
;SEG3: GETS LCD SEGMENT DATA FOR TIMEPHASE TD
;DISPD:  SWITCHES THE DISPLAY OFF AND
;        CONFIGURES G-,L- AND D-PORTS
;TTPND:  CHECKS TIMER PENDING FLAG (REFRESH
;        RATE GENERATION)
;SEGOUT: OUTPUTS LCD SEGMENT AND BACKPLANE DATA
;SUBROUTINES  SEG0...  SEG1  MUST FOLLOW DIRECTLY  AFTER  LOOK-UP
;TABLE, BECAUSE OF THE USE OF THE LAID-INSTRUCTION

        .LOCAL
SEG0:
        LD      B,#OFF1  ;POINT TO OFFSET 1 REG.
        LD      [B+],#000
        LD      [B+],#000
        LD      A,#00B
$TWO:
        IFBIT   #05,BCDHI ;WEIGHT >= 200 POUNDS?
        INCA              ;YES DISPLAY DIGIT5 ("2")

$POUND:
        IFBIT   #POUND,FLAG
        JP      $LPORT
        ADD     A,#002
$LPORT:
        X       A,[B]
        LD      X,#BCDLO
        LD      B,#MWBUF0
        LD      A,[X]
        AND     A,#00F  ;ELIMINATE  DIGIT1 BITS
        ADD     A,OFF2
        LAID            ;GET DIGIT1 DATA
        X       A,[B]   ;SAVE DIGIT1 DATA
                        ;IN MWBUF0
        LD      A,[X+]
        AND     A,#0F0  ;ELIMINATE DIGIT1 BITS
        SWAP    A
        ADD     A,OFF1  ;ALWAYS DISPLAY DECIMAL POINT
        LAID            ;GET DIGIT1 DATA
        SWAP    A
        OR      A,[B]   ;STORE DIGIT1 AND
        X       A,[B+]  ;DIGIT2 DATA IN MWBUF0
```

TL/DD/10788–13

7

```
$DPORT:
        LD      A,[X]
        IFBIT   #04,BCDHI
        JP      $ADD1
        AND     A,#00F
        ADD     A,OFF2  ;DISPLAY NO LEADING ZERO
        JP      $GET
$ADD1:
        AND     A,#00F
        ADD     A,OFF1  ;DISPLAY "1" (DIGIT4)
$GET:
        LAID            ;GET DIGIT3 DATA
        X       A,[B+]  ;STORE DIGIT3 DATA IN
                        ;MWBUF1
$GPORT:
        LD      A,OFF3
        LAID            ;GET DIGIT5 ("2") AND SPECIAL
                        ;SEGMENT DATA
        OR      A,#0FC  ;SET BITS 2...7 TO 1
        X       A,[B]   ;SAVE  DATA IN MWBUF2
        RET
SEG1:
        LD      B,#OFF1
        LD      [B+],#01B
        LD      [B+],#010
        LD      A,#056
        JP      $TWO

SEG2:
        LD      B,#OFF1
        LD      [B+],#030
        LD      [B+],#030
        LD      A,#05A
        JP      $TWO

SEG3:
        LD      B,#OFF1
        LD      [B+],#04B
        LD      [B+],#040
        LD      A,#05E
        JP      $TWO
        .LOCAL


DISPL:
        IFBIT   #POUND,FLAG
        JP      MULT2
        JP      LDT
MULT2:                  ;CALCULATE WEIGHT IN POUNDS
        LD      B,#BUF12LO  ;(Multiplication of kg *2.2)
        LD      [B+],#22
```

TL/DD/10788-14

```
        LD      X,#STALO
        JSR     MULBI168
        LD      B,#BUF12LO
        JSR     COPY
        LD      STAHI+1,#00
        LD      DIV0,#10
        JSR     DIVBI248
LDT:
        JSR     BINBCD16    ;CONVERT BINARY TO BCD WEIGHT
        LD      COUNT,#50   ;REPEAT DISPLAY LOOP 50 TIMES
                            ;(=1 SEC DISPLAY TIME)
        LD      B,#TMRLO
        LD      [B+],#0E8 ;LOAD TIMER WITH 1000(03E8h)
        LD      [B+],#003 ;(=50 Hz LCD REFRESH AT tc=5us)
        LD      [B+],#0E8 ;LOAD AUTOREG. WITH 1000
        LD      [B+],#003
        LD      [B+],#090 ;CNTRL-REG.:"TIMER WITH AUTO-
                          ;LOAD"- MODE,START TIMER
        LD      [B+],#010 ;PSW-REG.:RESET TPND FLAG
DISP1:
        JSR     SEG0        ;GET 7-SEGM. DATA FOR REFRESH
                            ;TIMEPHASE Ta
        JSR     TTPND       ;TEST TIMER PENDING FLAG
TP0:                        ;BACKPLANE REFRESH Ta
        SBIT    #BP1,[B]
        LD      A,[B+]      ;POINT TO G-CONFIG.-REG.
        RBIT    #BP2,[B]
        SBIT    #BP1,[B]
        LD      A,[B-]      ;POINT TO G-DATA REG.
        RBIT    #BP2,[B]
        JSR     SEGOUT      ;SEGMENT DATA OUT
        JSR     SEG1        ;GET 7-SEG. DATA FOR Tb
        JSR     TTPND
TP1:
        SBIT    #BP2,[B]
        LD      A,[B+]      ;POINT TO G-CONF.-REG.
        RBIT    #BP1,[B]
        SBIT    #BP2,[B]
        LD      A,[B-]      ;POINT TO G-DATA REG.
        RBIT    #BP1,[B]
        JSR     SEGOUT
        JSR     SEG2        ;GET 7-SEGM. DATA FOR Tc
        JSR     TTPND
TP2:
        RBIT    #BP1,[B]
        LD      A,[B+]      ;POINT TO G-CONFIG.-REG.
        RBIT    #BP2,[B]
        SBIT    #BP1,[B]
        LD      A,[B-]      ;POINT TO G-DATA-REG.
        RBIT    #BP2,[B]
        JSR     SEGOUT
```

TL/DD/10788-15

7

```
        JSR     SEG3
        JSR     TTPND
TP3:
        RBIT    #BP1,[B]
        RBIT    #BP2,[B]
        LD      A,[B+]
        RBIT    #BP1,[B]
        SBIT    #BP2,[B]
        JSR     SEGOUT
        DRSZ    COUNT
        JP      DISP1
        LD      COUNT,#50
        DRSZ    COUNT2      ;10SEC OVER?
        JP      DISP1       ;NO, DISPLAY WEIGHT
        JSR     DISPD
        RET                 ;YES ROUTINE FINISHED


DISPD:                      ;SWITCH DISPLAY OFF
        LD      B,#PORTLD
        LD      [B+],#000   ;OUTPUT 0 TO L PORT
        LD      [B+],#0FF   ;L-PORT = OUTPUT PORT
        LD      B,#PORTGD
        LD      [B+],#000   ;OUTPUT 0 TO G OUTPUTS
        LD      [B+],#037   ;G0..G2,G4,G5=OUTPUTS
        LD      PORTD,#000 ;OUTPUT 0 TO D-PORT
        RET

SEGOUT:
        LD      B,#MWBUF0
        LD      A,[B+]     ;POINT TO MWBUF1
        X       A,PORTLD   ;OUTPUT 7 SEG. DATA IN
                           ;MWBUF0 TO L-PORT
        LD      A,[B+]     ;POINT TO MWBUF2
        X       A,PORTD    ;OUTPUT MWBUF1 TO D-PORT
        LD      X,#PORTGD
        LD      A,[X]
        AND     A,[B]      ;AND MWBUF2 WITH PORTGD
                           ;LEAVE BITS 2...7 UNCHANGED
        X       A,[B]      ;STORE RESULT (A')IN
                           ;MWBUF2,LOAD A WITH
                           ;ORIGINAL MWBUF2 VALUE
        AND     A,#003     ;AND 007 WITH ORIGINAL
                           ;MWBUF2 (A''),SET BITS 0,1 TO
                           ;CORRECT VALUE
        OR      A,[B]      ;OR  A' WITH A'',RESTORE ORIGINAL
                           ;G2...G7 BITS
        X       A,[X]      ;OUTPUT RESULT TO G-PORT
        RET
```

TL/DD/10788–16

```
;16 BIT BINARY TO BCD CONVERSION
;THE MEMORY ASSIGNMEMTS ARE AS FOLLOWS:

;BINLO: RAM ADRESS BINARY LOW BYTE
;BCDLO: RAM ADRESS BCD LOW BYTE
;COUNT: RAM ADRESS SHIFT COUNTER (0F0...0FB,0FF)

;BCD NUMBER IN BCDLO,BCDLO+1,BCDLO+2
;
;MEMORY ADRESS        M(BINLO+1)  M(BINLO)
;DATA                 BINARY HB   BINARY LOW BYTE
;
;MEMORY ADRESS        M(BCDLO+2)  M(BCDLO+1)  M(BCDLO)
;DATA                 BCD HB      BCD         BCD LOW BYTE
;


        BINLO =  STALO
        .LOCAL
        $BCDT = (BCDLO + 3) & 0F
        $BINT = (BINLO + 2) & 0F
BINBCD:
        LD      COUNT,#16 ;LOAD CONTROL REGISTER WITH
                          ;NUMBER OF LEFTSHIFTS TO
                          ;EXECUTE
        LD      B,#BCDLO  ;LOAD BCD-NUMBER LOWEST BYTE
                          ;ADRESS

$CBCD:                    ;CLEAR BCD RAM-REGISTERS
        LD      [B+],#00
        IFBNE   #$BCDT
        JP      $CBCD
$LSH:                     ;LEFTSHIFT BINARY NUMBER
        LD      B,#BINLO
        RC
$LSHFT:
        LD      A,[B]
        ADC     A,[B]     ;IF MSB IS SET, SET CARRY
        X       A,[B+]
        IFBNE   #$BINT
        JP      $LSHFT
        LD      B,#BCDLO
$BCDADD:
        LD      A,[B]
        ADD     A,#066    ;ADD CORRECTION FACTOR
        ADC     A,[B]     ;LEFTSHIFT BCD NUMBER
                          ;(BCD=2**WEIGHT OF
                          ;BINARY BIT(=CARRY BIT))
        DCOR    A         ;DECIMAL CORRECT ADDITION
        X       A,[B+]
        IFBNE   #$BCDT
```

TL/DD/10788-17

```
        JP      $BCDADD
        DRSZ    COUNT     ;DECREMENT SHIFT COUNTER
        JP      $LSH
        RET
        .LOCAL
```

TL/DD/10788-18

```
;BINARY DIVIDE 24BIT BY  8BIT (Q=Y/Z)
;YL: LOW BYTE RAM ADRESS DIVIDEND
;ZL: LOW BYTE RAM ADRESS DIVISOR
;CNTR: RAM ADRESS SHIFT COUNTER (0F0...0FB,0FF)

;QUOTIENT AT RAM LOCATIONS YL..YL+2
;REMAINDER AT YL+3
;QUOTIENT IS ALL '1's IF DIVIDE BY ZERO, REMAINDER
;THEN CONTAINS YL

;THE MEMORY ASSIGNMENTS ARE AS FOLLOWS:
;
;       M(YH+1)  M(YH)          M(YL+1)  M(YL)
;       0        Y(HIGH BYTE)   Y        Y(LOW BYTE)
;      -------------------------------------------------
;       M(ZL)
;       Z
;
;ROUTINE NEEDS 1.21ms FOR EXECUTION AT tc=1us

        ZL      =  DIV0
        YL      =  STALO
        CNTR    =  COUNT
        .LOCAL
        $YH      = YL+2
        $BTY     = ($YH&00F)+2 ;PARAMETER FOR "IFBNE"-INSTR.

DIVBI248:
        LD      CNTR,#018  ;INITIALIZE SHIFT COUNTER
        LD      B,#$YH+1   ;FOR 24 COUNTS
        LD      [B],#000   ;PUT 0 IN M(YH+1)
        LD      X,#$YH+1
$LSHFT:
        LD      B,#YL   ;LEFT SHIFT DIVIDEND
        RC
$LUP:
        LD      A,[B]
        ADC     A,[B]
        X       A,[B+]
        IFBNE   #$BTY
        JP      $LUP
        LD      B,#ZL
        IFC
        JP      $SUBT
$TSUBT:
                        ;SUBTRACT AND TEST
        SC              ;SUBTRACT Z FROM M(YH+1,YH+2)
        LD      A,[X]
        SUBC    A,[B]
        IFNC
        JP      $TEST
```

TL/DD/10788-19

```
$SUBT:
                        ;SUBTRACT Z FROM M(YH+1,YH+2)
        LD      A,[X]
        SUBC    A,[B]
        X       A,[X]
        LD      B,#YL
        SBIT    #0,[B]
$TEST:
        DRSZ    CNTR    ;24 SHIFTS EXECUTED?
        JP      $LSHFT  ;NO, LEFT SHIFT DIVIDEND
        RET
        .LOCAL

;BINARY MULTIPLIES A 16BIT VALUE (X1)
;WITH A 8BIT VALUE (X2): M = X1 * X2

;X1L: RAM ADRESS X1 LOW BYTE
;X2L: RAM ADRESS X2
;COUNT RAM ADRESS SHIFT COUNTER

;M IS STORED AT RAM ADRESSES X2L...X2L+2

;THE MEMORY ASSIGNMEMTS ARE AS FOLLOWS:
;MEMORY       M(X2L+2)  M(X2L+1)       M(X2L)
;DATA         0         0              X2
;-----------------------------------------------------
;MEMORY        M(X1L+1)   M(X1L)
;DATA          X1(H.B.)   X1(LOW BYTE)

;THE EXECUTION TIME FOR THE ROUTINE AT tc=1us IS 240us
;


        .LOCAL

MULBI168:

        LD      COUNT,#9 ;PRESET SHIFT COUNTER
        LD      [B+],#00 ;PRESET X2L+1,X2L+2 WITH '0'
        LD      [B],#00
        RC
$LOOP:
        LD      A,[B]    ;RIGHT SHIFT
        RRCA
        X       A,[B-]
        LD      A,[B]
        RRCA
        X       A,[B-]
        LD      A,[B]
        RRCA
        X       A,[B+]
```

TL/DD/10788-20

7

```
        LD      A,[B+] ;INCREMENT B POINTER
        IFNC           ;MOST SIGN. BIT OF X2 SET?
        JP      $TEST  ;NO,TEST SHIFT COUNTER
        RC             ;YES,RESET CARRY
        LD      A,[B-] ;POINT TO 2nd HIGHEST BYTE
                       ;OF RESULT
        LD      A,[X+] ;DO WEIGHTED ADD
        ADC     A,[B]
        X       A,[B+]
        LD      A,[X-]
        ADC     A,[B]
        X       A,[B]
$TEST:
        DRSZ    COUNT  ;8 RIGHT SHIFTS EXECUTED?
        JP      $LOOP  ;NO,SHIFT
        RET            ;YES,MULIPLICATION FINISHED
        .LOCAL
        .END
```

TL/DD/10788-21

PC® MOUSE Implementation Using COP800

National Semiconductor
Application Note 681
Alvin Chan

ABSTRACT

The mouse is a very convenient and popular device used in data entry in desktop computers and workstations. For desktop publishing, CAD, paint or drawing programs, using the mouse is inevitable. This application note will describe how to use the COP822C microcontroller to implement a mouse controller.

INTRODUCTION

Mouse Systems was the first company to introduce a mouse for PCs. Together with Microsoft and Logitech, they are the most popular vendors in the PC mouse market. Most mainstream PC programs that use pointing devices are able to support the communication protocols laid down by Mouse Systems and Microsoft.

A typical mouse consists of a microcontroller and its associated circuitry, which are a few capacitors, resistors and transistors. Accompanying the electronics are the mechanical parts, consisting of buttons, roller ball and two disks with slots. Together they perform several major functions: motion detection, host communication, power supply, and button status detection.

MOTION DETECTION

Motion detection with a mouse consists of four commonly known mechanisms. They are the mechanical mouse, the opto-mechanical mouse, the optical mouse and the wheel mouse.

The optical mouse differs from the rest as it requires no mechanical parts. It uses a special pad with a reflective surface and grid lines. Light emitted from the LEDs at the bottom of the mouse is reflected by the surface and movement is detected with photo-transistors.

The mechanical and the opto-mechanical mouse use a roller ball. The ball presses against two rollers which are connected to two disks for the encoding of horizontal and vertical motion. The mechanical mouse has contact points on the disks. As the disks move they touch the contact bars, which in turn generates signals to the microcontroller. The opto-mechanical mouse uses disks that contain evenly spaced slots. Each disk has a pair of LEDs on one side and a pair of photo-transistors on the other side.

The wheel mouse has the same operation as the mechanical mouse except that the ball is eliminated and the rollers are rotated against the outside surface on which the mouse is placed.

HOST COMMUNICATION

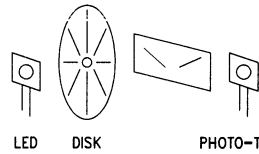
Besides having different operating mechanisms, the mouse also has different modes of communication with the host. It can be done through the system bus, the serial port or a special connector. The bus mouse takes up an expansion slot in the PC. The serial mouse uses one of the COM ports.

Although the rest of this report will be based on the opto-mechanical mouse using the serial port connection, the same principle applies to the mechanical and the wheel mouse.

MOTION DETECTION FOR THE OPTO-MECHANICAL MOUSE

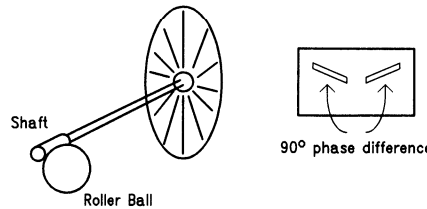
The mechanical parts of the opto-mechanical mouse actually consist of one roller ball, two rollers connected to the disks and two pieces of plastic with two slots on each one for LED light to pass through. The two slots are cut so that they form a 90 degree phase difference. The LEDs and the photo-transistors are separated by the disks and the plastic. As the disks move, light pulses are received by the photo-transistors. The microcontroller can then use these quadrature signals to decode the movement of the mouse.

Figure 1a shows the arrangement of the LEDs, disks, plastic and photo-transistors. The shaft connecting the disk and the ball is shown separately on *Figure 1b*. *Figure 2* shows the signals obtained from the photo-transistors when the mouse moves. The signals will not be exactly square waves because of unstable hand movements.



TL/DD/10799-1

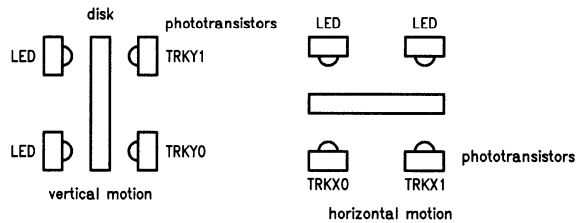
a



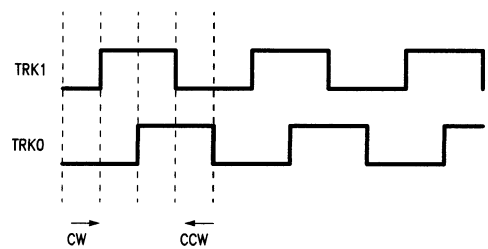
TL/DD/10799-2

b

FIGURE 1



TL/DD/10799-3



TL/DD/10799-4

Signals at phototransistors are similar for vertical and horizontal motion.
Track 1 leads track 0 by 90 degrees

FIGURE 2

RESOLUTION, TRACKING SPEED AND BAUD RATE

The resolution of the mouse is defined as the number of movement counts the mouse can provide for each fixed distance travelled. It is dependent on the physical dimension of the ball and the rollers. It can be calculated by measuring the sizes of the mechanical parts.

An example for the calculation can be shown by making the following assumptions:

- The disks have 40 slots and 40 spokes
- Each spoke has two data counts
  (This will be explained in the section "An Algorithm for Detecting Movements")
- Each slot also has two data counts
- The roller has a diameter of 5mm

For each revolution of the roller, there will be $40 \times 2 \times 2 = 160$ counts of data movement. At the same time, the mouse would have travelled a distance of $\pi \times 5 = 15.7$mm. Therefore the resolution of the mouse is $15.7/160 = 0.098$mm per count. This is equivalent to 259 counts or dots per inch (dpi).

The tracking speed is defined as the fastest speed that the mouse can move without the microcontroller losing track of the movement. This depends on how fast the microcontroller can sample the pulses from the photo-transistors. The effect of a slow tracking speed will contribute to jerking movements of the cursor on the screen.

The baud rate is fixed by the software and the protocol of the mouse type that is being emulated. For mouse systems and microsoft mouse, they are both 1200. Baud rate will affect both the resolution and the tracking speed. The internal movement counter may overflow while the mouse is still sending the last report with a slow baud rate. With a fast baud rate, more reports can be sent for a certain distance moved and the cursor should appear to be smoother.

POWER SUPPLY FOR THE SERIAL MOUSE

Since the serial port of the PC has no power supply lines, the RTS, CTS, DTR and DSR RS232 interface lines are utilized. Therefore the microcontroller and the mouse hardware should have very little power consumption. National Semiconductor's COP822C fits into this category perfectly. The voltage level in the RS232 lines can be either positive or negative. When they are positive, the power supply can be obtained by clamping down with diodes. When they are negative, a 555 timer is used as an oscillator to transform the voltage level to positive. The 1988 National Semiconductor Linear 3 Databook has an example of how to generate a variable duty cycle oscillator using the LMC555 in page 5-282.

While the RTS and DTR lines are used to provide the voltage for the mouse hardware, the TXD line of the host is utilized as the source for the communication signals. When idle, the TXD line is in the mark state, which is the most negative voltage. A pnp transistor can be used to drive the voltage of the RXD pin to a voltage level that is compatible with the RS232 interface standard.

AN ALGORITHM FOR DETECTING MOVEMENTS

The input signal of the photo-transistors is similar to that shown in *Figure 2*. Track 1 leads track 0 by 90 degrees. Movement is recorded as either of the tracks changes state. State tables can be generated for clockwise and counter-clockwise motions.

With the two tracks being 90 degrees out of phase, there could be a total of four possible track states. It can be observed that the binary values formed by combining the present and previous states are unique for clockwise and counter-clockwise motion. A sixteen entry jump table can be formed to increment or decrement the position of the cursor. If the value obtained does not correspond to either the clockwise or counter-clockwise movement, it could be treated as noise. In that case either there is noise on the microcontroller input pins or the microcontroller is tracking motions faster than the movement of the mouse. A possible algorithm can be generated as follows. The number of instruction cycles for some instructions are shown on the left.

$(TRK1, TRK0)_t$		$(TRK1, TRK0)_{t-1}$		Binary Value
		CCW		
0	1	0	0	4
1	1	0	1	D
1	0	1	1	B
0	0	1	0	2

$(TRK1, TRK0)_t$		$(TRK1, TRK0)_{t-1}$		Binary Value
		CW		
1	0	0	0	8
0	0	0	1	1
0	1	1	1	7
1	1	1	0	E

```
CYCLES    ;*************************************************
          ;       SAMPLE SENSOR INPUT
          ;       INC OR DEC THE POSITION
          ;*************************************************
          ;
          SENSOR:
1                 LD        B,#GTEMP
3                 LD        A,PORTGP
1                 RRC       A
2                 AND       A,#03C          ; G6,G5,G4,G3
1                 X         A, [B]          ; (GTEMP)
          ;
2                 LD        A, [B+]         ; (GTEMP) X IN 3,2
1                 RRC       A
1                 RRC       A
2                 AND       A, #03
1                 OR        A, [B]          ; (TRACKS)
2                 OR        A, #0B0         ; X MOVEMENT TABLE
3                 JID
          ;
          NOISEX: JP        YDIR
          ;
3         INCX:   LD        A,XINC
1                 INC       A
3                 JP        COMX
          ;
          DECX:   LD        A,XINC
                  DEC       A
          COMX:
2                 IFEQ      A, #080
1                 JP        YDIR
3                 X         A, XINC
1                 LD        B, #CHANGE
1                 SBIT      RPT, [B]
1                 LD        B, #TRACKS
          ;
          YDIR:
2                 LD        A, [B-]         ; (TRACKS) Y IN 5, 4
1                 SWAP      A
1                 RRC       A
1                 RRC       A
1                 RRC       A
2                 AND       A, #0C0
1                 OR        A, [B]          ; (GTEMP)
```

```
1                   SWAP    A
2                   OR      A, #0C0         ; Y MOVEMENT TABLE
3                   JID
            ;
            NOISEY: JP      ESENS
            ;
3           INCY:   LD      A, YINC
1                   INC     A
3                   JP      COMY
            DECY:
                    LD      A, YINC
                    DEC     A
            COMY:
2                   IFEQ    A, #080
1                   JP      ESENS
3                   X       A, YINC
1                   LD      B, #CHANGE
1                   SBIT    RPT, [B]
1                   LD      B, #GTEMP
            ESENS:
2                   LD      A, [B+]         ; (GTEMP) IN5, 4, 1, 0
1                   X       A, [B]          ; (TRACKS) NEW TRACK STATUS
5                   RET
            ;
                    .=0B0
            MOVEMX:
                    .ADDR   NOISEX          ; 0
                    .ADDR   INCX            ; 1
                    .ADDR   DECX            ; 2
                    .ADDR   NOISEX          ; 3
                    .ADDR   DECX            ; 4
                    .ADDR   NOISEX          ; 5
                    .ADDR   NOISEX          ; 6
                    .ADDR   INCX            ; 7
                    .ADDR   INCX            ; 8
                    .ADDR   NOISEX          ; 9
                    .ADDR   NOISEX          ; A
                    .ADDR   DECX            ; B
                    .ADDR   NOISEX          ; C
                    .ADDR   DECX            ; D
                    .ADDR   INCX            ; E
                    .ADDR   NOISEX          ; F
            ;
                    .=0C0
            MOVEMY:
                    .ADDR   NOISEY          ; 0
                    .ADDR   INCY            ; 1
                    .ADDR   DECY            ; 2
                    .ADDR   NOISEY          ; 3
                    .ADDR   DECY            ; 4
                    .ADDR   NOISEY          ; 5
                    .ADDR   NOISEY          ; 6
                    .ADDR   INCY            ; 7
                    .ADDR   INCY            ; 8
                    .ADDR   NOISEY          ; 9
                    .ADDR   NOISEY          ; A
                    .ADDR   DECY            ; B
                    .ADDR   NOISEY          ; C
                    .ADDR   DECY            ; D
                    .ADDR   INCY            ; E
                    .ADDR   NOISEY          ; F
```

Going through the longest route in the sensor routine takes 75 instruction cycles. So at 5 MHz the microcontroller can track movement changes within 150 μs by using this algorithm.

MOUSE PROTOCOLS

Since most programs in the PC support the mouse systems and microsoft mouse, these two protocols will be discussed here. The protocols are byte-oriented and each byte is framed by one start-bit and two stop-bits. The most commonly used reporting mode is that a report will be sent if there is any change in the status of the position or of the buttons.

MICROSOFT COMPATIBLE DATA FORMAT

6	5	4	3	2	1	0	Bit Number
1	L	R	Y7	Y6	X7	X6	Byte 1
0	X5	X4	X3	X2	X1	X0	Byte 2
0	Y5	Y4	Y3	Y2	Y1	Y0	Byte 3

L, R = Key data (Left, Right key) 1 = key depressed

X0–X7 = X distance 8-bit two's complement value −128 to +127

Y0–Y7 = Y distance 8-bit two's complement value −128 to +127

Positive = South

In the Microsoft Compatible Format, data is transferred in the form of seven-bit bytes. Y movement is positive to the south and negative to the north.

FIVE BYTE PACKED BINARY FORMAT (MOUSE SYSTEMS CORP)

7	6	5	4	3	2	1	0	Bit Number
1	0	0	0	0	L*	M*	R*	Byte 1
X7	X6	X5	X4	X3	X2	X1	X0	Byte 2
Y7	Y6	Y5	Y4	Y3	Y2	Y1	Y0	Byte 3
X7	X6	X5	X4	X3	X2	X1	X0	Byte 4
Y7	Y6	Y5	Y4	Y3	Y2	Y1	Y0	Byte 5

L*, M*, R* = Key data (Left, Middle, Right key), 0 = key depressed

X0–X7 = X distance 8-bit two's complement value −127 to +127

Y0–Y7 = Y distance 8-bit two's complement value −127 to +127

In the Five Byte Packed Binary Format data is transferred in the form of eight-bit bytes (eight data bits without parity). Bytes 4 and 5 are the movement of the mouse during the transmission of the first report.

THE COP822C MICROCONTROLLER

The COP822C is an 8-bit microcontroller with 20 pins, of which 16 are I/O pins. The I/O pins are separated into two ports, port L and port G. Port G has built-in Schmitt-triggered inputs. There is 1k of ROM and 64 bytes of RAM. In the mouse application, the COP822C's features used can be summarized below. Port G is used for the photo-transistor's input. Pin G0 is used as the external interrupt input to monitor the RTS signal for the microsoft compatible protocol. The internal timer can be used for baud rate timing and interrupt generation. The COP822C draws only 4 mA at a crystal frequency of 5 MHz. The instruction cycle time when operating at this frequency is 2 μs.

A MOUSE EXAMPLE

The I/O pins for the COP822C are assigned as follows:

Pin	Function
G0	Interrupt Input (Monitoring RTS Toggle)
G1	Reserved for Input Data (TXD of Host)
G2	Output Data (RXD of Host)
G3–G6	LED Sensor Input
L0–L2	Button Input
L3	Jumper Input (for Default Mouse Mode)

The timer is assigned for baud rate generation. It is configured in the PWM auto-reload mode (with no G3 toggle output) with a value of 1A0 hex in both the timer and the auto-reload register. When operating at 5 MHz, it is equivalent to 833 μs or 1200 baud. When the timer counts down, an interrupt is generated and the service routine will indicate in a timer status byte that it is time for the next bit. The subroutine that handles the transmission will look at this status byte to send the data.

The other interrupt comes from the G0 pin. This is implemented to satisfy the microsoft mouse requirement. As the RTS line toggles, it causes the microcontroller to be interrupted. The response to the toggling is the transmission of the character "M" to indicate the presence of the mouse.

The main program starts by doing some initializations. Then it loops through four subroutines that send the report, sense the movement, sense the buttons, and set up the report format.

Subroutine "SDATA" uses a state table to determine what is to be transmitted. There are 11 or 12 states because microsoft has only 7 data bits and mouse systems has 8. The state table is shown below:

SENDST	State
0	IDLE
1	START BIT
2–8	DATA (FOR MICROSOFT)
2–9	DATA (FOR MOUSE SYSTEMS)
9–10	STOP BIT (FOR MICROSOFT)
10–11	STOP BIT (FOR MOUSE SYSTEMS)
11	NEXT WORD (FOR MICROSOFT)
12	NEXT WORD (FOR MOUSE SYSTEMS)

The G2 pin is set to the level according to the state and the data bit that is transmitted.

Subroutine "SENSOR" checks the input pins connected to the LEDs. The horizontal direction is checked first followed by the vertical direction. Two jump tables are needed to decode the binary value formed by combining the present and previous status of the wheels. The movements are recorded in two counters.

Subroutines "BUTUS" and "BUTMS" are used for polling the button input. They compare the button input with the value polled last time and set up a flag if the value changes. Two subroutines are used for the ease of setting up reports for different mice. The same applies for subroutines "SRPTMS" and "SRPTUS" which set up the report format for transmission. The status change flag is checked and the report is formatted according to the mouse protocol. The

movement counters are then cleared. Since the sign of the vertical movement of mouse systems and microsoft is reversed, the counter value in subroutine "SRPTMS" is complemented to form the right value.

There is an extra subroutine "SY2RPT" which sets up the last two bytes in the mouse systems' report. It is called after the first three bytes of the report are sent.

The efficiency of the mouse depends solely on the effectiveness of the software to loop through sensing and transmission subroutines. For the COP822C, one of the most effective addressing modes is the B register indirect mode. It uses only one byte and one instruction cycle. With autoincrement or autodecrement, it uses one byte and two instruction cycles. In order to utilize this addressing mode more often, the organization of the RAM data has to be carefully thought out. In the mouse example, it can be seen that by placing related variables next to each other, the saving of code and execution time is significant. Also, if the RAM data can fit in the first 16 bytes, the load B immediate instruction is also more efficient. The subroutine "SRPTMS" is shown below and it can be seen that more than half the instructions are B register indirect which are efficient and compact.

```
;
;     VARIABLES
;
      WORDPT   =       000       ;WORD POINTER
      WORD1    =       001       ;BUFFER TO STORE REPORTS
      WORD2    =       002
      WORD3    =       003
      CHANGE   =       004       ;MOVEMENT CHANGE OR BUTTON PRESSED
      XINC     =       005       ;X DIRECTION COUNTER
      YINC     =       006       ;Y DIRECTION COUNTER
      NUMWORD  =       007       ;NUMER OF BYTES TO SEND
      SENDST   =       008       ;SERIAL PROTOCOL STATE
;
;*****************************************************
;     SUBROUTINE SET UP REPORT 'SRPT' FOR MOUSE SYSTEMS
;     CHANGE OF STATUS DETECTED
;     SET UP THE FIRST 3 WORDS FOR REPORTING
;     IF IN IDLE STATE
;*****************************************************
;
SRPTMS:
      LD       A,CHANGE
      IFEQ     A, #0             ; EXIT IF NO CHANGE
      RET
;
      RBIT     GIE, PSW          ; DISABLE INTERRUPT
      LD       B, #WORDPT
      LD       [B+], #01         ; (WORDPT) SET WORD POINTER
      LD       A, BUTSTAT
      X        A, [B+]           ; (WORD1)
;
      LD       A, XINC
      X        A, [B+]           ; (WORD2)
;
      SC
      CLR      A
      SUBC     A, YINC           ; FOR MOUSE SYSTEM NEG Y
      X        A, [B+]           ; (WORD3)
;
      RBIT     RPT, [B]          ; (CHANGE) RESET CHANGE OF STATUS
      SBIT     SYRPT, [B]                  ; (CHANGE)
      LD       A, [B+]           ; INC B
      LD       [B+], #0          ; (XINC)
      LD       [B+], #0          ; (YINC)
;
      LD       [B+], #03         ; (NUMWORD) SEND 3 BYTES
      LD       [B], #01          ; (SENDST) SET TO START BIT STATE
;     SBIT     GIE, PSW          ; ENABLE INTERRUPT
      RET
;
```

CONCLUSION

The COP822C has been used as a mouse controller. The code presented is a minimum requirement for implementing a mouse systems and microsoft compatible mouse. About 550 bytes of ROM code has been used. The remaining ROM area can be used for internal diagnostics and for communicating with the host's mouse driver program. The unused I/O pins can be used to turn the LED's on only when necessary to save extra power. This report demonstrated the use of the efficient instruction set of the COP800 family. It can be seen that the architecture of the COP822C is most suitable for implementing a mouse controller. The table below summarizes the advantages of the COP822C.

Feature	Advantage
Port G	Schmitt Triggered Input for Photo-Transistors
G0	External Interrupt for RTS Toggling
Timer	For Baud Rate Generation
Low Power	4 mA at 5 MHz
Small Size	20-Pin DIP

REFERENCE

The mouse still reigns over data entry—Electronic Engineering Times, October 1988.

MICE for mainstream applications—PC Magazine, August 1987.

Logimouse C7 Technical Reference Manual—Logitech, January 1986.

APPENDIX A—MEMORY UTILIZATION

RAM Variables

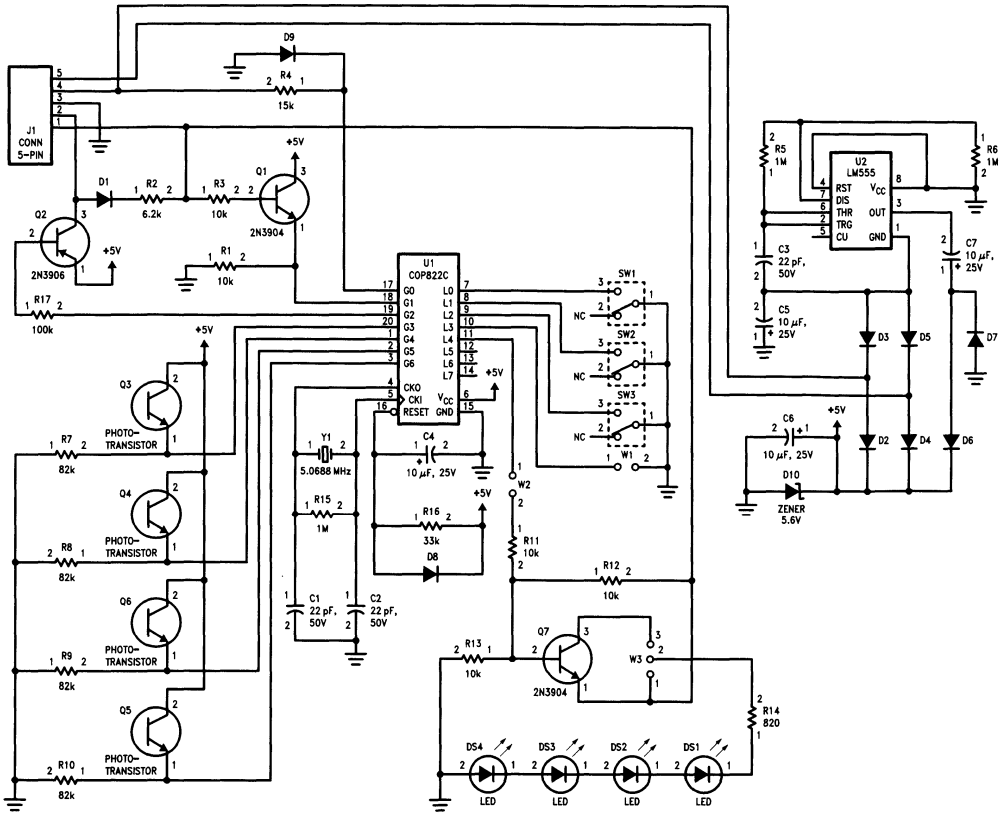
TEMP	=	0F1	Work Space
ASAVE	=	0F4	Save A Register
PSSAVE	=	0F6	Save PSW Register
WORDPT	=	000	Word Pointer
WORD1	=	001	Buffer to Store Report
WORD2	=	002	Buffer
WORD3	=	003	Buffer
CHANGE	=	004	Movement or Button Change
XINC	=	005	X Direction Counter
YINC	=	006	Y Direction Counter
NUMWORD	=	007	Number of Bytes to Send
SENDST	=	008	Serial Protocol State
TSTATUS	=	00A	Counter Status
MTYPE	=	00B	Mouse Type
GTEMP	=	00C	Track Input from G Port
TRACKS	=	00D	Previous Track Status
BTEMP	=	00E	Button Input from L Port
BUTSTAT	=	00F	Previous Button Status

APPENDIX B—SUBROUTINE SUMMARY

Subroutine	Location	Function
MLOOP	03D	Main Program Loop
SENSOR	077	Sample Photo-Transistor Input
INTRP	0FF	Interrupt Service Routines
SRPTUS	136	Set Up Report for Microsoft
SRPTMS	16C	Set Up 1st 3 Bytes Report for Mouse Systems
SDATA	191	Drive Data Transmission Pin According to Bit Value of Report
SY2RPT	1D1	Set Up Last 2 Bytes Report for Mouse Systems
BUTUS	200	Sample Button Input for Microsoft
BUTMS	210	Sample Button Input for Mouse Systems

APPENDIX C—SYSTEM SCHEMATIC, SYSTEM

Flowchart, complete program listing.



TL/DD/10799–5

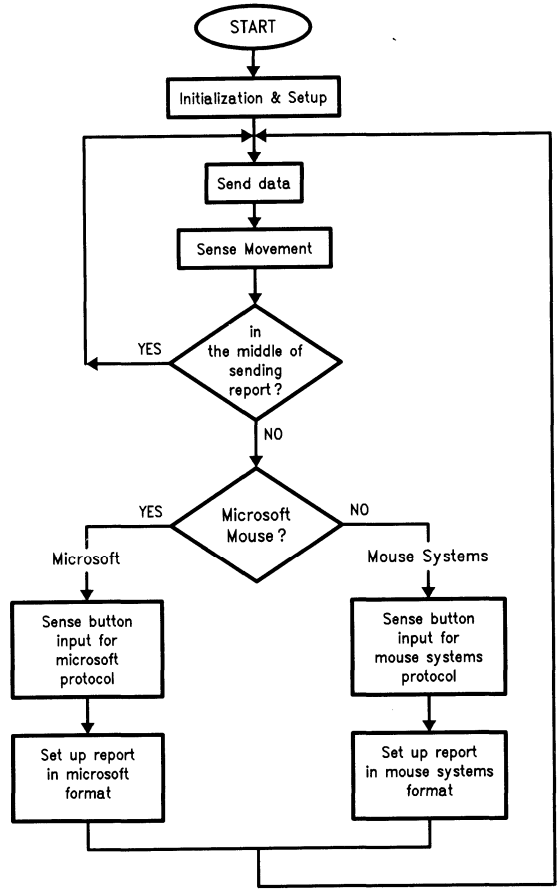
Note 1: All diodes are 1N4148.

Note 2: All resistor values are in ohms, 5%, 1/8W.

Note: Unless otherwised specified

FIGURE 3. System Schematic

Flowchart for Mouse Systems and Microsoft Mouse



TL/DD/10799–6

```
NATIONAL SEMICONDUCTOR CORPORATION
COP800 CROSS ASSEMBLER,REV:D1,12 OCT 88
AMOUSE


  1                     ;
  2                     ;       MICROSOFT AND MOUSE SYSTEM COMPATIBLE MOUSE
  3                     ;       02/14/89
  4                     ;       NAME : AMOUSE.MAC
  5                     ;
  6                             .TITLE AMOUSE
  7                             .CHIP 820
  8                     ;
  9                     ;
 10     00D0            PORTLD  =       0D0     ; PORT L DATA
 11     00D1            PORTLC  =       0D1     ; PORT L CONFIG
 12     00D2            PORTLP  =       0D2     ; PORT L PIN
 13                     ;
 14     00D4            PORTGD  =       0D4     ; PORT G DATA
 15     00D5            PORTGC  =       0D5     ; PORT G CONFIG
 16     00D6            PORTGP  =       0D6     ; PORT G PIN
 17                     ;
 18     00EA            TMRLO   =       0EA     ; TIMER LOW BYTE
 19     00EB            TMRHI   =       0EB     ; TIMER HIGH BYTE
 20     00EC            TAULO   =       0EC     ; TIMER REGISTER LOW BYTE
 21     00ED            TAUHI   =       0ED     ; TIMER REGISTER HIGH BYTE
 22                     ;
 23     00EE            CNTRL   =       0EE     ; CONTROL REGISTER
 24     00EF            PSW     =       0EF     ; PSW REGISTER
 25                     ;
 26                     ;       CONSTANT DECLARE
 27                     ;
 28     0000            INTR    =       0
 29     0003            TIO     =       3
 30     0004            SO      =       4
 31     0005            SK      =       5
 32     0006            SI      =       6
 33     0007            CKO     =       7
 34                     ;
 35     0007            TSEL    =       7
 36     0006            CSEL    =       6
 37     0005            TEDG    =       5
 38     0004            TRUN    =       4
 39     0003            MSEL    =       3
 40     0002            IEDG    =       2
 41     0001            S1      =       1
 42     0000            S0      =       0
 43                     ;
 44     0007            HCARRY  =       7
 45     0006            CARRY   =       6
 46     0005            TPND    =       5
 47     0004            ENTI    =       4
 48     0003            IPND    =       3
 49     0002            BUSY    =       2
 50     0001            ENI     =       1
 51     0000            GIE     =       0
```

TL/DD/10799–7

```
 52                     ;
 53                     ;       TSTATUS BITS
 54                     ;
 55       0002          TBAUB   =       2       ;BAUD RATE TIMER BIT
 56                     ;
 57       0000          RPT     =       0       ;REPORT BIT OF CHANGE (CHANGE)
 58       0001          SYRPT   =       1       ;SET UP MOUSE SYSTEM LAST 2 WORDS (CHANGE)
 59       0007          USOFT   =       7       ;MICROSOFT (MTYPE)
 60       0002          XMT     =       2       ;G2 AS XMT BIT (PORTGD)
 61                     ;
 62       0003          SW      =       3       ;SLIDE SWITCH (PORTLP,MTYPE)
 63                     ;
 64                     ;       REGISTER ASSIGNMENTS
 65                     ;
 66       00F0                  RSVD    =       0F0
 67       00F1                  TEMP    =       0F1
 68       00F3                  TBAU    =       0F3     ;BAUD RATE TIMER
 69       00F4                  ASAVE   =       0F4     ;SAVE A
 70       00F5                  BSAVE   =       0F5     ;SAVE B
 71       00F6                  PSSAVE  =       0F6     ;SAVE PSW
 72                     ;
 73                     ;       VARIABLES
 74                     ;
 75       0000                  WORDPT  =       000     ;WORD POINTER
 76       0001                  WORD1   =       001     ;BUFFER TO STORE REPORTS
 77       0002                  WORD2   =       002
 78       0003                  WORD3   =       003
 79                     ;
 80       0004                  CHANGE  =       004     ;MOVEMENT CHANGE OR BUTTON PRESSED
 81       0005                  XINC    =       005     ;X DIRECTION COUNTER
 82       0006                  YINC    =       006     ;Y DIRECTION COUNTER
 83       0007                  NUMWORD =       007     ;NUMBER OF BYTES TO SEND
 84       0008                  SENDST  =       008     ;SERIAL PROTOCOL STATE
 85                     ;
 86       0009                  TBAUR   =       009     ;BAUD RATE TIMER RELOAD
 87       000A                  TSTATUS =       00A     ;COUNTER STATUS
 88       000B                  MTYPE   =       00B     ;MOUSE TYPE
 89                     ;
 90       000C                  GTEMP   =       00C     ;TRACK INPUT FROM G
 91       000D                  TRACKS  =       00D     ;PREVIOUS TRACK STATUS
 92                     ;
 93       000E                  BTEMP   =       00E     ;BUTTON INPUT
 94       000F                  BUTSTAT =       00F     ;PREVIOUS BUTTON STATUS
 95                     ;
 96                     ;
 97                     ; MOST POSITIVE = SPACE = HI = ON = 0 = START BIT = RBIT
 98                     ; MOST NEGATIVE = MARK = LO = OFF = 1 = STOP BIT = SBIT
 99                     ;
100                     ;       MICROSOFT FORMAT
101                     ;
102                     ;       1  L  R  Y7  Y6  X7  X6
```

TL/DD/10799–8

```
103                     ;       0  X5 .............  X0
104                     ;       0  Y5 .............  Y0
105                     ;
106                     ;       1200 BAUD 7 BIT NO PARITY 2 STOP BITS
107                     ;
108                     ;       MOUSE SYSTEMS FORMAT (FIVE BYTE PACKED BINARY)
109                     ;
110                     ;       1  0   0   0   0   L*  M*  R*
111                     ;       X7 ......................  X0
112                     ;       Y7 ......................  Y0
113                     ;       X7 ......................  X0
114                     ;       Y7 ......................  Y0
115                     ;
116                     ;       1200 BAUD 7 BIT NO PARITY 2 STOP BITS
117                     ;
118                     ;       G6,G5,G4,G3 ARE SENSOR INPUTS
119                     ;
120                     ;       L0, L1 AND L2 ARE BUTTON INPUTS
121                     ;
122                     ;       G0 IS INTERRUPT INPUT FOR DETECTING RTS TOGGLE
123                     ;
124                     ;       USE G2 AS TRANSMIT
125                     ;
126                     ;       G1 USED FOR RECEIVING COMMANDS FROM HOST (RESERVED)
127                     ;
128                     START:
129 0000 DD2F                   LD      SP,#02F
130 0002 BCEF00                 LD      PSW,#0          ;DISABLE INTR
131 0005 BCEE80                 LD      CNTRL,#080      ;10000000 - AUTORELOAD
132                                                     ;RISING EDGE EXT INT
133 0008 BCD504                 LD      PORTGC,#004     ;G2 AS OUTPUT, OTHERS AS HI-Z
134 000B BCD404                 LD      PORTGD,#004     ;G2 DATA 1 "MARK"
135                     ;
136 000E BCD130                 LD      PORTLC,#030     ;HI-Z INPUTS FOR L6-7,OUTPUT L4,5
137 0011 BCD00F                 LD      PORTLD,#0F      ;WEAK PULL UP FOR L0-3
138                     ;
139                     ;       INIT RAM
140                     ;
141 0014 5B                     LD      B,#CHANGE
142 0015 9A00                   LD      [B+],#0         ;(CHANGE)
143 0017 9A00                   LD      [B+],#0         ;(XINC)
144 0019 9A00                   LD      [B+],#0         ;(YINC)
145 001B BC0A00                 LD      TSTATUS,#0
146                     ;
147 001E 9DD6                   LD      A,PORTGP
148 0020 B0                     RRC     A
149 0021 953C                   AND     A,#03C          ;NOW IN 6,5,4,3
150 0023 9C0D                   X       A,TRACKS        ;GET INITIAL VALUE OF SENSORS
151                     ;
152 0025 3067                   JSR     SELECT          ;SELECT MOUSE TYPE
153                     ;
```

TL/DD/10799–9

```
154                     ;*************************************************
155                     ;
156                     ;       CRYSTAL FREQ = 4.96 MHZ  2.016 US INST CYCLE
157                     ;       FOR 1200 BAUD - TIMER = 413 COUNT
158                     ;
159                     ;*************************************************
160                     ;
161                     LTIMER:
162 0027 DEEA                   LD      B,#TMRLO
163 0029 9A9D                   LD      [B+],#09D       ;FOR 2.016 US CYCLE
164 002B 9A01                   LD      [B+],#01
165 002D 9A9D                   LD      [B+],#09D
166 002F 9E01                   LD      [B],#01
167                     ;
168 0031 BC0800                 LD      SENDST,#0       ;SET TO IDLE STATE
169 0034 9DEF                   LD      A,PSW
170 0036 9713                   OR      A,#013          ;ENABLE INTRS SET GIE
171 0038 9CEF                   X       A,PSW
172 003A BDEE7C                 SBIT    TRUN,CNTRL      ;START TIMER
173                     ;
174                     MLOOP:
175 003D BCD03F                 LD      PORTLD,#03F     ;TURN ON LED (NOT USED)
176 0040 3191                   JSR     SDATA
177 0042 3077                   JSR     SENSOR
178 0044 9D08                   LD      A,SENDST        ;IF SENDING REPORT
179 0046 9300                   IFGT    A,#0            ;JUST DO SENSOR
180 0048 F4                     JP      MLOOP
181                     ;
182 0049 9DD2                   LD      A,PORTLP        ;GET INPUT FROM BUTTONS (L0,L1,L2)
183 004B B0                     RRC     A               ;PUT IN CARRY FOR CHECKING
184 004C 51                     LD      B,#BTEMP        ;PREPARATION TO SEE WHAT BUTTON IS PRESSED
185                     ;
186 004D BD0B77                 IFBIT   USOFT,MTYPE
187 0050 0B                     JP      LPUS
188                     ;
189 0051 3210                   JSR     BUTMS           ;MOUSE SYSTEMS
190 0053 316C                   JSR     SRPTMS
191                     ;
192 0055 BDD273                 IFBIT   SW,PORTLP
193 0058 E4                     JP      MLOOP           ;CONTINUE IF NO CHANGE IN SWITCH
194 0059 306B                   JSR     USM             ;ELSE NEW SET UP
195 005B E1                     JP      MLOOP
196                     LPUS:
197 005C 3200                   JSR     BUTUS           ;MICROSOFT
198 005E 3136                   JSR     SRPTUS
199                     ;
200 0060 BDD273                 IFBIT   SW,PORTLP
201 0063 3071                   JSR     SYM             ;IF CHANGED IN SWITCH, NEW SET UP
202 0065 203D           +       JP      MLOOP
203                     ;
204                     ;*************************************************
```

TL/DD/10799–10

```
                        ;       SELECT MOUSE TYPE
                        ;***********************************************
                        ;
                        SELECT:
                                IFBIT   SW,PORTLP       ;CHECK JUMPER
0067 BDD273                     JP      SYM
006A 06
                        ;
                        USM:
006B 54                         LD      B,#MTYPE
006C 7F                         SBIT    USOFT,[B]       ;(MTYPE) IS MICROSOFT MOUSE
006D BC0F87                     LD      BUTSTAT,#087    ;NO KEY PRESSED
0070 8E                         RET
                        ;
                        SYM:
0071 54                         LD      B,#MTYPE
0072 6F                         RBIT    USOFT,[B]       ;(MTYPE) IS MOUSE SYSTEMS
0073 BC0F00                     LD      BUTSTAT,#0      ;NO KEY PRESSED
0076 8E                         RET
                        ;
                        ;***********************************************
                        ;       SAMPLE SENSOR INPUT
                        ;       INC OR DEC THE POSITION
                        ;       -127 IS USED INSTEAD OF -128 IN CHECKING
                        ;       NEGATIVE GOING POSITION SO THAT BOTH
                        ;       MICROSOFT AND MOUSE SYSTEMS FIT IN
                        ;***********************************************
                        ;
                        SENSOR:
0077 53                         LD      B,#GTEMP
0078 9DD6                       LD      A,PORTGP
007A BCD00F                     LD      PORTLD,#0F      ;(NOT USED) TURN OFF LED
007D B0                         RRC     A
007E 953C                       AND     A,#03C          ;G5,G4,G3,G2
0080 A6                         X       A,[B]           ;(GTEMP)
                        ;
                        ;
                        ;               (TRK1,TRK0)t-1  (TRK1,TRK0)t
                        ;       CCW       0    1          0    0       4
                        ;                 1    1          0    1       D
                        ;                 1    0          1    1       B
                        ;                 0    0          1    0       2
                        ;
                        ;       CW        1    0          0    0       8
                        ;                 0    0          0    1       1
                        ;                 0    1          1    1       7
                        ;                 1    1          1    0       E
                        ;
0081 AA                         LD      A,[B+]          ;(GTEMP) X IN 3,2
0082 B0                         RRC     A
0083 B0                         RRC     A
0084 9503                       AND     A,#03           ;GET X TRACKS
```

TL/DD/10799–11

```
256 0086 87                     OR      A,[B]           ;OVERLAY WITH PREVIOUS (TRACKS)
257 0087 97B0                   OR      A,#0B0          ;X MOVEMENT TABLE
258 0089 A5                     JID
259                     ;
260 008A 0F             NOISEX: JP      YDIR
261                     ;
262                     INCX:
263 008B 9D05                   LD      A,XINC
264 008D 8A                     INC     A
265 008E 03                     JP      COMX            ;CHECK IF LIMIT IS REACHED
266                     DECX:
267 008F 9D05                   LD      A,XINC
268 0091 8B                     DEC     A
269                     COMX:                           ;CHECK FOR LIMIT
270 0092 9250                   IFEQ    A,#80
271 0094 05                     JP      YDIR            ;YES DO NOTHING
272 0095 9C05                   X       A,XINC          ;ELSE NEW POSITION
273 0097 5B                     LD      B,#CHANGE
274 0098 78                     SBIT    RPT,[B]         ;(CHANGE)
275 0099 52                     LD      B,#TRACKS
276                     ;
277                     YDIR:
278 009A 52                     LD      B,#TRACKS
279 009B AB                     LD      A,[B-]          ;(TRACKS) Y IN 5,4
280 009C 65                     SWAP    A
281 009D B0                     RRC     A
282 009E B0                     RRC     A
283 009F B0                     RRC     A
284 00A0 95C0                   AND     A,#0C0
285 00A2 87                     OR      A,[B]           ;(GTEMP)
286 00A3 65                     SWAP    A
287 00A4 97C0                   OR      A,#0C0          ;Y MOVEMENT TABLE
288 00A6 A5                     JID
289                     ;
290     00B0                    .=0B0
291                     MOVEMX:
292 00B0 8A                     .ADDR   NOISEX          ;0
293 00B1 8F                     .ADDR   DECX            ;1
294 00B2 8B                     .ADDR   INCX            ;2
295 00B3 8A                     .ADDR   NOISEX          ;3
296 00B4 8B                     .ADDR   INCX            ;4
297 00B5 8A                     .ADDR   NOISEX          ;5
298 00B6 8A                     .ADDR   NOISEX          ;6
299 00B7 8F                     .ADDR   DECX            ;7
300 00B8 8F                     .ADDR   DECX            ;8
301 00B9 8A                     .ADDR   NOISEX          ;9
302 00BA 8A                     .ADDR   NOISEX          ;A
303 00BB 8B                     .ADDR   INCX            ;B
304 00BC 8A                     .ADDR   NOISEX          ;C
305 00BD 8B                     .ADDR   INCX            ;D
306 00BE 8F                     .ADDR   DECX            ;E
```

TL/DD/10799–12

```
00BF 8A                 .ADDR   NOISEX          ;F
                ;
      00C0              .=0C0
                MOVEMY:
00C0 D0                 .ADDR   NOISEY          ;0
00C1 D1                 .ADDR   INCY            ;1
00C2 D5                 .ADDR   DECY            ;2
00C3 D0                 .ADDR   NOISEY          ;3
00C4 D5                 .ADDR   DECY            ;4
00C5 D0                 .ADDR   NOISEY          ;5
00C6 D0                 .ADDR   NOISEY          ;6
00C7 D1                 .ADDR   INCY            ;7
00C8 D1                 .ADDR   INCY            ;8
00C9 D0                 .ADDR   NOISEY          ;9
00CA D0                 .ADDR   NOISEY          ;A
00CB D5                 .ADDR   DECY            ;B
00CC D0                 .ADDR   NOISEY          ;C
00CD D5                 .ADDR   DECY            ;D
00CE D1                 .ADDR   INCY            ;E
00CF D0                 .ADDR   NOISEY          ;F
                ;
00D0 0F         NOISEY: JP      ESENS
                ;
00D1 9D06       INCY:   LD      A,YINC
00D3 8A                 INC     A
00D4 03                 JP      COMY
                DECY:
00D5 9D06               LD      A,YINC
00D7 8B                 DEC     A
                COMY:
00D8 9280               IFEQ    A,#080
00DA 05                 JP      ESENS
00DB 9C06               X       A,YINC
00DD 5B                 LD      B,#CHANGE
00DE 78                 SBIT    RPT,[B]         ;(CHANGE)
00DF 53                 LD      B,#GTEMP
                ;
                ESENS:
00E0 53                 LD      B,#GTEMP
00E1 AA                 LD      A,[B+]          ;(GTEMP) IN 5,4,1,0
00E2 A6                 X       A,[B]           ;(TRACKS)NEW TRACK STATUS
00E3 8E                 RET
                ;
                ;
      00FF              .=0FF
                ;
                ;*************************************************
                ;       INTERRUPT ROUTINES
                ;*************************************************
                ;
00FF 9CF4       INTRP:  X       A,ASAVE
```

TL/DD/10799–13

```
358                     ;
359 0101 BDEF75         IFBIT   IPND,PSW
360 0104 07             JP      TINTR
361 0105 BDEF73         IFBIT   IPND,PSW
362 0108 0A             JP      XINTR
363                     ;
364                     INTRET:                         ;INTERRUPT RETURN
365 0109 9DF4           LD      A,ASAVE
366 010B 8F             RETI
367                     ;
368                     ;************************************************
369                     ;       TIMER INTERRUPT
370                     ;       UPDATE ALL THE COUNTERS
371                     ;************************************************
372                     ;
373                     TINTR:
374 010C BDEF6D         RBIT    TPND,PSW
375 010F BD0A7A         SBIT    TBAUB,TSTATUS   ;SET BIT IN TSTATUS
376 0112 F6             JP      INTRET
377                     ;
378                     ;******************************************************
379                     ;       EXTERNAL INTERRUPT
380                     ;       RESPONSE TO RTS TOGGLING
381                     ;       BY SENDING AN 'M' 4DH
382                     ;******************************************************
383                     ;
384 0113 BDEF6B         XINTR:  RBIT    IPND,PSW
385 0116 BD0B77                 IFBIT   USOFT,MTYPE     ;ONLY IF MICROSOFT PROTOCOL
386 0119 01                     JP      XINTR1          ;CONTINUE
387 011A EE                     JP      INTRET          ;ELSE DO NOTHING
388                     XINTR1:
389 011B BC01FF                 LD      WORD1,#0FF      ;ALL MARK
390 011E BC024D                 LD      WORD2,#'M'
391 0121 BC0702                 LD      NUMWORD,#02
392                     ;
393 0124 9D08                   LD      A,SENDST
394 0126 9200                   IFEQ    A,#0            ;IF IDLE, SEND 'M'
395 0128 05                     JP      RTSR2
396                     ;
397 0129 BC0001                 LD      WORDPT,#WORD1   ;FAKE CONTINUE LAST CHAR
398 012C 2109           +       JP      INTRET
399                     ;
400                     RTSR2:
401 012E BC0002                 LD      WORDPT,#WORD2   ;'M' ONLY
402 0131 BC0801                 LD      SENDST,#01
403 0134 2109           +       JP      INTRET
404                     ;
405                     ;******************************************************
406                     ;       SUBROUTINE SET UP REPORT 'SRPT' FOR MICROSOFT
407                     ;                                       -------------
408                     ;       CHANGE OF STATUS DETECTED
```

TL/DD/10799–14

```
409                     ;       SET UP THE 3 WORDS FOR REPORTING IF IN IDLE STATE
410                     ;*************************************************************
411                     ;
412                     SRPTUS:
413 0136 5B                     LD      B,#CHANGE
414 0137 70                     IFBIT   RPT,[B]
415 0138 01                     JP      SRUS1
416 0139 8E                     RET                     ;EXIT IF NOT CHANGE
417                     ;
418                     SRUS1:
419 013A BDEF68                 RBIT    GIE,PSW         ;DISABLE INTERRUPT
420 013D 5F                     LD      B,#WORDPT
421 013E 9A01                   LD      [B+],#WORD1     ;(WORDPT)SET WORD POINTER
422 0140 9D05                   LD      A,XINC
423 0142 65                     SWAP    A
424 0143 B0                     RRC     A
425 0144 B0                     RRC     A
426 0145 9503                   AND     A,#03           ;X7,X6
427 0147 A6                     X       A,[B]           ;(WORD1)
428                     ;
429 0148 9D06                   LD      A,YINC
430 014A 65                     SWAP    A
431 014B 950C                   AND     A,#0C           ;Y7,Y6
432 014D 87                     OR      A,[B]           ;(WORD1)
433 014E 9740                   OR      A,#040          ;SET BIT 6
434 0150 BD0F87                 OR      A,BUTSTAT       ;GET BUTTON STATUS
435 0153 A2                     X       A,[B+]          ;(WORD1)
436                     ;
437 0154 9D05                   LD      A,XINC
438 0156 953F                   AND     A,#03F          ;X0-X5
439 0158 A2                     X       A,[B+]          ;(WORD2)
440                     ;
441 0159 9D06                   LD      A,YINC
442 015B 953F                   AND     A,#03F          ;Y0-Y5
443 015D A2                     X       A,[B+]          ;(WORD3)
444 015E 68                     RBIT    RPT,[B]         ;(CHANGE)RESET CHANGE OF STATUS
445 015F AA                     LD      A,[B+]          ;INC B
446 0160 9A00                   LD      [B+],#0         ;(XINC)
447 0162 9A00                   LD      [B+],#0         ;(YINC)
448                     ;
449 0164 9A03                   LD      [B+],#03        ;(NUMWORD)SEND 3 BYTES
450 0166 9E01                   LD      [B],#01         ;(SENDST)SET TO START BIT STATE
451                     ;
452 0168 BDEF78                 SBIT    GIE,PSW         ;ENABLE INTERRUPT
453 016B 8E                     RET
454                     ;
455                     ;*************************************************************
456                     ;       SUBROUTINE SET UP REPORT 'SRPT' FOR MOUSE SYSTEMS
457                     ;                               ----------------
458                     ;       CHANGE OF STATUS DETECTED
459                     ;       SET UP THE FIRST 3 WORDS FOR REPORTING
```

TL/DD/10799–15

```
460                     ;       IF IN IDLE STATE
461                     ;*******************************************************
462                     ;
463                     SRPTMS:
464 016C 5B                     LD      B,#CHANGE
465 016D 70                     IFBIT   RPT,[B]
466 016E 01                     JP      SRMS1
467 016F 8E                     RET                     ;EXIT IF NO CHANGE
468                     ;
469                     SRMS1:
470 0170 BDEF68                 RBIT    GIE,PSW         ;DISABLE INTERRUPT
471 0173 5F                     LD      B,#WORDPT
472 0174 9A01                   LD      [B+],#WORD1     ;(WORDPT)SET WORD POINTER
473 0176 9D0F                   LD      A,BUTSTAT
474 0178 A2                     X       A,[B+]          ;(WORD1)
475                     ;
476 0179 9D05                   LD      A,XINC
477 017B A2                     X       A,[B+]          ;(WORD2)
478                     ;
479 017C A1                     SC
480 017D 64                     CLR     A
481 017E BD0681                 SUBC    A,YINC          ;FOR MOUSE SYSTEM NEG Y
482 0181 A2                     X       A,[B+]          ;(WORD3)
483                     ;
484 0182 68                     RBIT    RPT,[B]         ;(CHANGE)RESET CHANGE OF STATUS
485 0183 79                     SBIT    SYRPT,[B]       ;(CHANGE)
486 0184 AA                     LD      A,[B+]          ;INC B
487 0185 9A00                   LD      [B+],#0         ;(XINC)
488 0187 9A00                   LD      [B+],#0         ;(YINC)
489                     ;
490 0189 9A03                   LD      [B+],#03        ;(NUMWORD)SEND 3 BYTES
491 018B 9E01                   LD      [B],#01         ;(SENDST)SET TO START BIT STATE
492                     ;
493 018D BDEF78                 SBIT    GIE,PSW         ;ENABLE INTERRUPT
494 0190 8E                     RET
495                     ;
496                     ;
497                     ;*******************************************************
498                     ;       SUBROUTINE TO SEND DATA 'SDATA'
499                     ;       CHECK THE BIT TO SEND AND DRIVE THE OUTPUT TO THE
500                     ;       DESIRED VALUE
501                     ;
502                     ;       SENDST          STATE
503                     ;         0             IDLE
504                     ;         1             START BIT
505                     ;         2-8           DATA
506                     ;         2-9           DATA    (FOR MOUSE SYSTEMS)
507                     ;         9-10          STOP BIT
508                     ;         10-11         STOP BIT   (FOR MOUSE SYSTEMS)
509                     ;         11            NEXT WORD
510                     ;         12            NEXT WORD  (FOR MOUSE SYSTEMS)
```

TL/DD/10799-16

```
511                     ;
512                     ;*******************************************************
513                     ;
514 0191 55             SDATA:  LD      B,#TSTATUS
515 0192 72                     IFBIT   TBAUB,[B]       ;(TSTATUS)CHECK IF BAUD RATE TIMER ENDS
516 0193 01                     JP      SDATA1
517 0194 8E                     RET
518                     ;
519                     SDATA1:
520 0195 6A                     RBIT    TBAUB,[B]       ;(TSTATUS)
521 0196 AA                     LD      A,[B+]          ;INC B TO (MTYPE)
522 0197 9D08                   LD      A,SENDST
523 0199 97F0                   OR      A,#0F0
524 019B A5                     JID
525                     ;
526 019C 8E             IDLE:   RET                     ;EXIT IF IDLE
527                     ;
528 019D 77             STAT9:  IFBIT   USOFT,[B]       ;(MTYPE)
529 019E 16                     JP      STOPB
530                     DATAB:
531 019F 9D00                   LD      A,WORDPT
532 01A1 9CFE                   X       A,B             ;B POINTS TO THE WORD
533                     ;
534 01A3 A0                     RC
535 01A4 AE                     LD      A,[B]
536 01A5 B0                     RRC     A               ;XMIT LEAST SIG BIT
537 01A6 A6                     X       A,[B]
538 01A7 DED4                   LD      B,#PORTGD
539 01A9 88                     IFC
540 01AA 7A                     SBIT    XMT,[B]
541 01AB 89                     IFNC
542 01AC 6A                     RBIT    XMT,[B]
543                     ;
544 01AD 9D08           NEXT:   LD      A,SENDST
545 01AF 8A                     INC     A
546 01B0 9C08                   X       A,SENDST
547 01B2 8E                     RET                     ;EXIT
548                     ;
549 01B3 77             STAT11: IFBIT   USOFT,[B]       ;(MTYPE)
550 01B4 04                     JP      NXWORD
551                     ;
552 01B5 BDD47A         STOPB:  SBIT    XMT,PORTGD
553 01B8 F4                     JP      NEXT
554                     ;
555 01B9 9D00           NXWORD: LD      A,WORDPT
556 01BB 8A                     INC     A
557 01BC BD0783                 IFGT    A,NUMWORD       ;NUMBER OF WORDS TO SEND
558 01BF 09                     JP      ENDRPT          ;END OF REPORT
559 01C0 9C00                   X       A,WORDPT
560 01C2 BC0801                 LD      SENDST,#01      ;SEND START BIT
561                     ;
```

TL/DD/10799–17

```
562 01C5 BDD46A         STARTB: RBIT    XMT,PORTGD      ;SEND START BIT
563 01C8 E4                     JP      NEXT
564                     ;
565 01C9 BD0471         ENDRPT: IFBIT   SYRPT,CHANGE
566 01CC 04                     JP      SY2RPT
567                     ;
568 01CD BC0800                 LD      SENDST,#0
569 01D0 8E                     RET
570                     ;
571                     ;***********************************************
572                     ;       SET UP LAST 2 WORDS IN MOUSE SYSTEM FORMAT
573                     ;***********************************************
574                     ;
575                     SY2RPT:
576 01D1 BDEF68                 RBIT    GIE,PSW         ;DISABLE INTERRUPT
577                     ;
578 01D4 5F                     LD      B,#WORDPT
579 01D5 9A01                   LD      [B+],#WORD1     ;(WORDPT)SET WORD POINTER
580 01D7 9D05                   LD      A,XINC
581 01D9 A2                     X       A,[B+]          ;(WORD1)
582                     ;
583 01DA A1                     SC
584 01DB 64                     CLR     A
585 01DC BD0681                 SUBC    A,YINC          ;FOR MOUSE SYSTEM NEG Y
586 01DF A2                     X       A,[B+]          ;(WORD2)
587                     ;
588 01E0 AA                     LD      A,[B+]          ;INC B
589 01E1 69                     RBIT    SYRPT,[B]       ;(CHANGE)RESET CHANGE OF STATUS
590 01E2 AA                     LD      A,[B+]          ;INC B
591 01E3 9A00                   LD      [B+],#0         ;XINC
592 01E5 9A00                   LD      [B+],#0         ;YINC
593                     ;
594 01E7 9A02                   LD      [B+],#02        ;(NUMWORD)SEND 2 BYTES
595 01E9 9E01                   LD      [B],#01         ;(SENDST)SET TO START BIT STATE
596                     ;
597 01EB BDEF78                 SBIT    GIE,PSW         ;ENABLE INTERRUPT
598 01EE 21C5           +       JP      STARTB
599                     ;
600     01F0                    .=01F0
601                     ;
602 01F0 9C                     .ADDR   IDLE            ;0
603 01F1 C5                     .ADDR   STARTB          ;1
604 01F2 9F                     .ADDR   DATAB           ;2
605 01F3 9F                     .ADDR   DATAB           ;3
606 01F4 9F                     .ADDR   DATAB           ;4
607 01F5 9F                     .ADDR   DATAB           ;5
608 01F6 9F                     .ADDR   DATAB           ;6
609 01F7 9F                     .ADDR   DATAB           ;7
610 01F8 9F                     .ADDR   DATAB           ;8
611 01F9 9D                     .ADDR   STAT9           ;9
612 01FA B5                     .ADDR   STOPB           ;10
```

TL/DD/10799–18

```
613 01FB B3                 .ADDR   STAT11          ;11
614 01FC B9                 .ADDR   NXWORD          ;12
615 01FD 9C                 .ADDR   IDLE            ;13
616 01FE 9C                 .ADDR   IDLE            ;14
617 01FF 9C                 .ADDR   IDLE            ;15
618                 ;
619                 ;
620                 ;***********************************************
621                 ;       SAMPLE BUTTON INPUT     FOR MICROSOFT
622                 ;                               ------------
623                 ;       INDICATE BUTTON STATUS
624                 ;***********************************************
625                 ;
626                 BUTUS:
627 0200 9E00               LD      [B],#0          ;(BTEMP), (A=PORTLP, CARRY ROTATED)
628 0202 89                 IFNC                    ;MICROSOFT: 1=KEY DEPRESSED
629 0203 7D                 SBIT    5,[B]           ;(BTEMP)
630                 ;
631 0204 B0                 RRC     A
632 0205 B0                 RRC     A
633 0206 89                 IFNC
634 0207 7C                 SBIT    4,[B]           ;(BTEMP)
635                 ;
636 0208 AA                 LD      A,[B+]          ;(BTEMP)
637 0209 82                 IFEQ    A,[B]           ;(BUTSTAT)
638 020A 8E                 RET                     ;NO CHANGE
639                 ;
640 020B A6                 X       A,[B]           ;(BUTSTAT)
641 020C BD0478             SBIT    RPT,CHANGE      ;INDICATE TO SEND DATA
642 020F 8E                 RET
643                 ;
644                 ;
645                 ;***********************************************
646                 ;       SAMPLE BUTTON INPUT     FOR MOUSE SYSTEMS
647                 ;                               ----------------
648                 ;       INDICATE BUTTON STATUS
649                 ;***********************************************
650                 ;
651                 BUTMS:
652 0210 9E87                 LD      [B],#087        ;(BTEMP)
653                 ;
654 0212 89                 IFNC                    ;MOUSE SYSTEM: 0=KEY DEPRESSED
655 0213 6A                 RBIT    2,[B]           ;(BTEMP)
656                 ;
657 0214 B0                 RRC     A
658 0215 89                 IFNC
659 0216 69                 RBIT    1,[B]           ;(BTEMP)
660                 ;
661 0217 B0                 RRC     A
662 0218 89                 IFNC
663 0219 68                 RBIT    0,[B]           ;(BTEMP)
```

TL/DD/10799–19

```
664                     ;
665 021A AA                    LD      A,[B+]          ;(BTEMP)
666 021B 82                    IFEQ    A,[B]           ;(BUTSTAT)
667 021C 8E                    RET                     ;NO CHANGE
668                     ;
669 021D A6                    X       A,[B]           ;(BUTSTAT)
670 021E BD0478                SBIT    RPT,CHANGE      ;INDICATE TO SEND DATA
671 0221 8E                    RET
672                     ;
673                     ;*********************************************************
674                     ;
675     03D0                    .=03D0
676 03D0 28                    .BYTE   '(C) 1990 NATIONAL SEMICONDUCTOR AMOUSE VER 1.0'
    03D1 43
    03D2 29
    03D3 20
    03D4 31
    03D5 39
    03D6 39
    03D7 30
    03D8 20
    03D9 4E
    03DA 41
    03DB 54
    03DC 49
    03DD 4F
    03DE 4E
    03DF 41
    03E0 4C
    03E1 20
    03E2 53
    03E3 45
    03E4 4D
    03E5 49
    03E6 43
    03E7 4F
    03E8 4E
    03E9 44
    03EA 55
    03EB 43
    03EC 54
    03ED 4F
    03EE 52
    03EF 20
    03F0 41
    03F1 4D
    03F2 4F
    03F3 55
    03F4 53
    03F5 45
    03F6 20
    03F7 56
    03F8 45
    03F9 52
    03FA 20
    03FB 31
    03FC 2E
    03FD 30
677                     ;
678                     .END
```

TL/DD/10799-20

TL/DD/10799-21

```
NATIONAL SEMICONDUCTOR CORPORATION
COP800 CROSS ASSEMBLER,REV:D1,12 OCT 88
AMOUSE
SYMBOL TABLE


ASAVE  00F4        B      00FE        BSAVE  00F5  *     BTEMP  000E
BUSY   0002  *     BUTMS  0210        BUTSTA 000F        BUTUS  0200
CARRY  0006  *     CHANGE 0004        CKO    0007  *     CNTRL  00EE
COMX   0092        COMY   00D8        CSEL   0006  *     DATAB  019F
DECX   008F        DECY   00D5        ENDRPT 01C9        ENI    0001  *
ENTI   0004  *     ESENS  00E0        GIE    0000        GTEMP  000C
HCARRY 0007  *     IDLE   019C        IEDG   0002  *     INCX   008B
INCY   00D1        INTR   0000  *     INTRET 0109        INTRP  00FF  *
IPND   0003        LPUS   005C        LTIMER 0027  *     MLOOP  003D
MOVEMX 00B0  *     MOVEMY 00C0  *     MSEL   0003  *     MTYPE  000B
NEXT   01AD        NOISEX 008A        NOISEY 00D0        NUMWOR 0007
NXWORD 01B9        PORTGC 00D5        PORTGD 00D4        PORTGP 00D6
PORTLC 00D1        PORTLD 00D0        PORTLP 00D2        PSSAVE 00F6  *
PSW    00EF        RPT    0000        RSVD   00F0  *     RTSR2  012E
S0     0000  *     S1     0001  *     SDATA  0191        SDATA1 0195
SELECT 0067        SENDST 0008        SENSOR 0077        SI     0006  *
SK     0005  *     SO     0004  *     SP     00FD        SRMS1  0170
SRPTMS 016C        SRPTUS 0136        SRUS1  013A        START  0000  *
STARTB 01C5        STAT11 01B3        STAT9  019D        STOPB  01B5
SW     0003        SY2RPT 01D1        SYM    0071        SYRPT  0001
TAUHI  00ED  *     TAULO  00EC  *     TBAU   00F3  *     TBAUB  0002
TBAUR  0009  *     TEDG   0005  *     TEMP   00F1  *     TINTR  010C
TIO    0003  *     TMRHI  00EB  *     TMRLO  00EA        TPND   0005
TRACKS 00DD        TRUN   0004        TSEL   0007  *     TSTATU 000A
USM    006B        USOFT  0007        WORD1  0001        WORD2  0002
WORD3  0003  *     WORDPT 0000        X      00FC        XINC   0005
XINTR  0113        XINTR1 011B        XMT    0002        YDIR   009A
YINC   0006
```

TL/DD/10799–22

```
NATIONAL SEMICONDUCTOR CORPORATION
COP800 CROSS ASSEMBLER,REV:D1,12 OCT 88
AMOUSE
MACRO TABLE


   NO WARNING LINES

   NO ERROR LINES

  556  ROM BYTES USED

SOURCE CHECKSUM = 987A
OBJECT CHECKSUM = 0A39

INPUT   FILE D:BMOUSE.MAC
LISTING FILE D:BMOUSE.PRN
OBJECT  FILE D:BMOUSE.LM
```

TL/DD/10799–23

Using COP800 Devices to Control DC Stepper Motors

National Semiconductor
Application Note 714
Michelle Giles

INTRODUCTION

COP800 devices can be used to control DC stepper motors with limited effort. This application note describes the use of a COP820 to control the speed, direction and rotation angle of a stepper motor. In addition to the COP820, this application requires a quad high current peripheral driver (DS3658) to meet the high current needs of the stepper motor.

DC STEPPER MOTOR

A DC stepper motor translates current pulses into rotor movement. A typical motor contains four winding coils labeled red, yellow/white, red/white, and yellow. Applying current to these windings forces the motor to step. For normal operation, two windings are activated (pulsed) concurrently. The motor moves clockwise one step per change in windings activated with the following activation sequence: red and yellow, yellow and red/white, red/white and yellow/white, yellow/white and red, repeat. Half-steps may be generated by altering the sequence to: red and yellow, yellow, yellow and red/white, red/white, red/white and yellow/white, yellow/white, yellow/white and red, red, repeat. The motor runs in a counterclockwise direction if either sequence is applied in reverse order. The speed of rotation (number of steps/second) is controlled by the frequency of the pulses.

COP820 CONTROL OF STEPPER MOTOR

The COP820 controls the stepper motor by sending pulse sequences to the motor windings in response to control commands. Commands executed by the code in this application include: single step the motor in a clockwise or counterclockwise direction (i.e. rotate the rotor through a certain number of degrees), run the motor continuously at one of four speeds in a clockwise or counterclockwise direction, and stop the motor.

Note: Half-stepping is not implemented in this example.

During continuous mode operation, the 16-bit timer of the COP820 is used to control the speed of the stepper motor. The timer is set up with a value that causes an underflow once every x seconds or at a frequency of 1/x. Each underflow of the timer interrupts the microcontroller. In response to the timer interrupt, the microcontroller generates a new pulse and causes a single step of the motor. Thus the motor steps at the frequency of the timer underflows. This application sets up the timer to generate interrupts at four different frequencies. These frequencies produce the following motor speeds: 25 steps/second, 100 steps/second, 200 steps/second, and 400 steps/second.

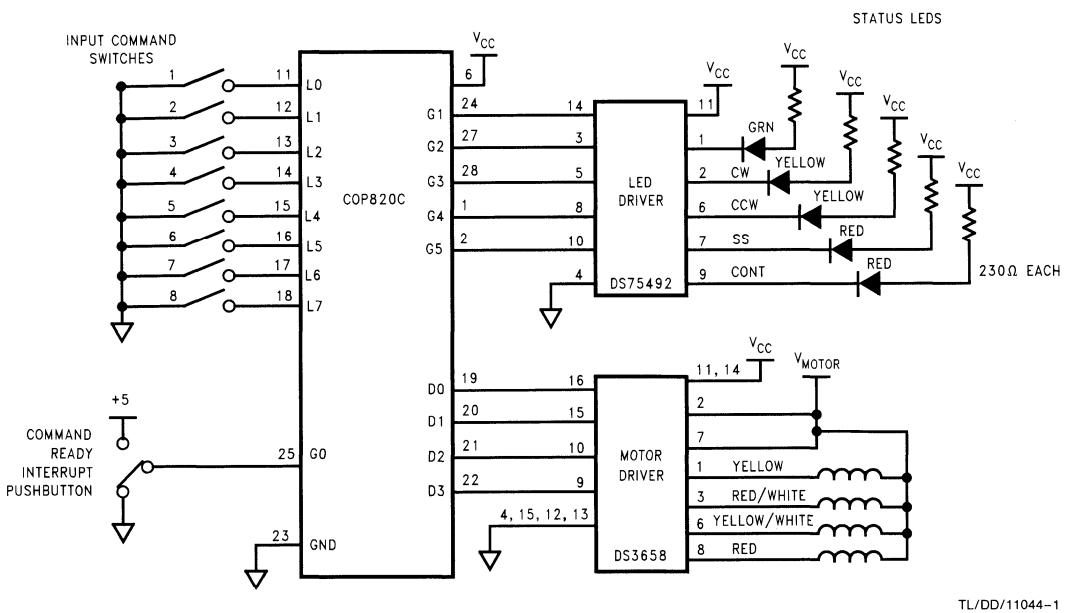
The determination of which windings to activate and deactivate to step the motor is performed by a single subroutine in this example. A block of memory is allocated to store a step pointer and the four possible stepper drive values are shown in Table I (9,C,6,3). Consecutive memory locations are used to store the stepper drive values so that applying the value from location X and then location X+1 (or X−1) causes the motor to step once. The motor drive subroutine increments or decrements the pointer to the current drive value based on the selection of a clockwise or counterclockwise direction. Writing the value from the newly selected location to the motor causes a single step of the motor in the appropriate direction.

During single step operation, the microcontroller steps the motor the exact number of times requested in the control command. Each step corresponds to 1.8 degrees of rotor movement. Therefore, a request to perform 200 steps will rotate the rotor through one complete revolution (360 degrees) at a fixed speed.

A block diagram of the application is shown in *Figure 1*. A flowchart of the code used to control the motor is given in *Figure 2*. The complete code is given at the end.

TABLE I. Stepper Motor Drive Sequence

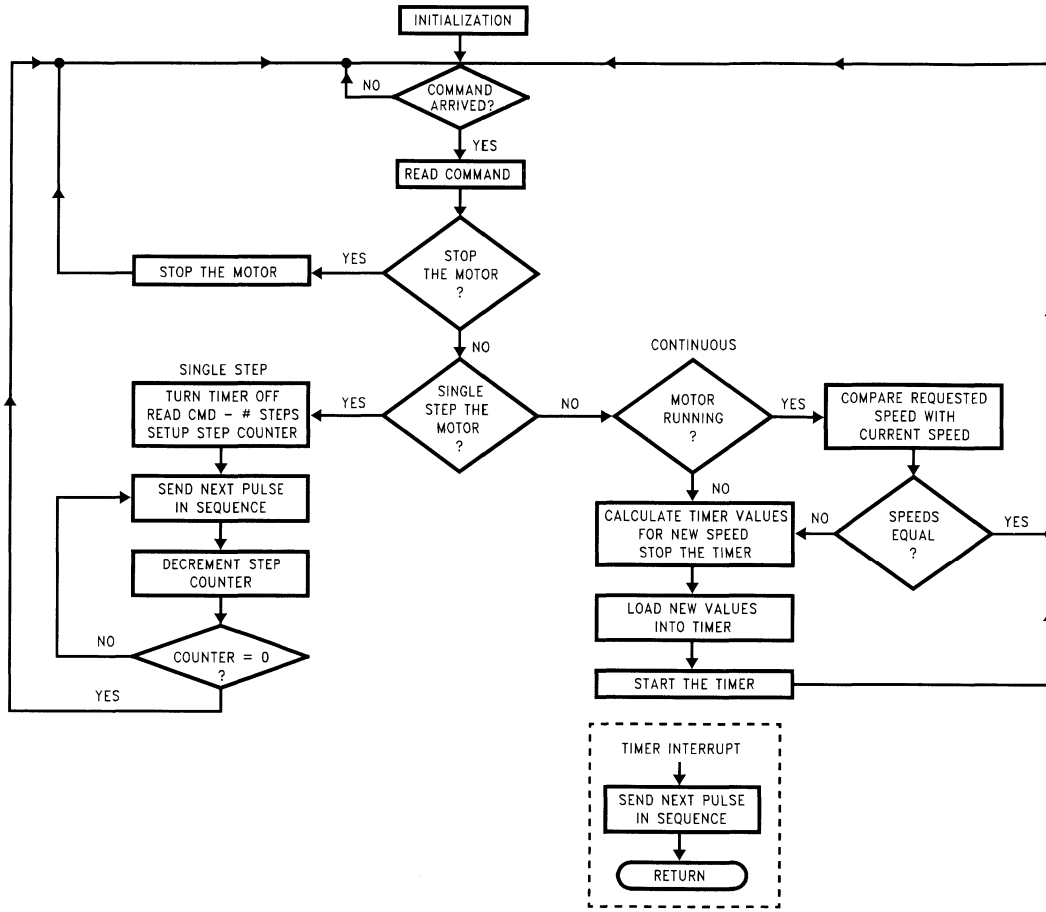
Step	Yellow	Red/White	Yellow/White	Red	Hex Value
0	ON	OFF	OFF	ON	9
1	ON	ON	OFF	OFF	C
2	OFF	ON	ON	OFF	6
3	OFF	OFF	ON	ON	3
4	ON	OFF	OFF	ON	9



TL/DD/11044–1

FIGURE 1. Schematic Diagram

Program Code Flow Chart



TL/DD/11044-2

FIGURE 2. Program Flowchart

NATIONAL SEMICONDUCTOR CORPORATION
COP800 CROSS ASSEMBLER, REV: D1, 12 OCT 88

```
                        ;STEPPER MOTOR CONTROL PROGRAM
                        ;MAY 1990
                        ;
                        ;This program controls the speed, direction, and degree of rotation of
                        ;a DC stepper motor.
                        ;
                        ;                                Memory Map
                        ; RAM    CONTENTS
                        ;  00    (MS0) step motor drive value 09H     (two windings active per pulse)
                        ;  01    (MS1) step motor drive value 0CH
                        ;  02    (MS2) step motor drive value 06H
                        ;  03    (MS3) step motor drive value 03H
                        ;  04    (CMD) control command
                        ;        bit7 - bit4 = motor speed or upper nibble of # single steps
                        ;        bit 3       = unused
                        ;        bit 2       = (MODE) single step or continuous mode select (1 = ss)
                        ;        bit 1       = (DIR) cw or ccw direction select (1 = cw)
                        ;        bit 0       = (GO)  motor go or motor stop select (1 = stop)
                        ;  05    (STEPS) lower byte of number of single steps
                        ;  07    (FLGREG) flag register
                        ;        bit 0       = (INT) ready to read in cmd (ext int occured)
                        ;        bit1 - bit7 = unused
                        ;  14    (TVAL0) value to load in lower byte of timer for speed X
                        ;  15    (TVAL1) value to load in upper byte of timer for speed X
                        ;  D2    (PORTLP) port L input pins used for incomming commands
                        ;  D4    (PORTGD) port G data pins used to drive status LEDs
                        ;  DC    (PORTD)  port D data pins used to ouput pulses to the stepper motor
                        ;  F0    (CREG0) step counter register zero
                        ;  F1    (CREG1) step counter register one
                        ;  F2    (STPPTR) pointer to current step motor drive value (RAM 00 - 03)
                        ;
                        ;REGISTER AND CONSTANT DEFINITIONS

                        ;COMMAND BITS
0000                            GO      = 0             ;GO COMMAND BIT
                                                        ;  1 = STOP          0 = GO
0001                            DIR     = 1             ;DIRECTION COMMAND BIT
                                                        ;  1 = CW            0 = CCW
0002                            MODE    = 2             ;MODE COMMAND BIT
                                                        ;  1 = SINGLE STEP  0 = CONTINUOUS

                        ;PORTG BITS
0000                            INT     = 0             ;FLAG BIT (SET IF EXTINT OCCURS)
0001                            READY   = 1             ;READY LED
0002                            CW      = 2             ;CLOCKWISE LED
0003                            CCW     = 3             ;COUNTER CLOCKWISE LED
0004                            SS      = 4             ;SINGLE STEP LED
0005                            NS      = 5             ;CONTINUOUS (NON-STOP) LED

                        ;REGISTERS
0004                            CMD     = 04            ;INPUT COMMAND STORAGE REGISTER
```

TL/DD/11044-3

NATIONAL SEMICONDUCTOR CORPORATION
COP800 CROSS ASSEMBLER, REV: D1, 12 OCT 88

```
 52      0005                    STEPS   = 05            ;INPUT #STEPS/SPEED REGISTER
 53      00F0                    CREG0   = 0F0           ;COUNTER REGISTER
 54      00F1                    CREG1   = 0F1           ;COUNTER REGISTER
 55      0007                    FLGREG  = 07            ;FLAG REGISTER (FLAG BITS)
 56      00F2                    STPPTR  = 0F2           ;CURRENT MOTOR STEP POINTER
 57      0014                    TVAL0   = 014           ;MOTOR SPEED LOAD VALUES
 58      0015                    TVAL1   = 015
 59      0000                    MS0     = 00            ;STEPPER MOTOR DRIVE VALUES
 60      0001                    MS1     = 01
 61      0002                    MS2     = 02
 62      0003                    MS3     = 03
 63
 64                      ;ASSIGNMENTS FOR COP820
 65
 66      00D5                    PORTGC  = 0D5
 67      00D4                    PORTGD  = 0D4
 68      00D6                    PORTGP  = 0D6
 69      00D1                    PORTLC  = 0D1
 70      00D0                    PORTLD  = 0D0
 71      00D2                    PORTLP  = 0D2
 72      00DC                    PORTD   = 0DC
 73      00D7                    PORTI   = 0D7
 74      00E9                    SIOR    = 0E9
 75      00EA                    TMRLO   = 0EA
 76      00EB                    TMRHI   = 0EB
 77      00EC                    TAULO   = 0EC
 78      00ED                    TAUHI   = 0ED
 79      00EE                    CNTRL   = 0EE
 80      00EF                    PSW     = 0EF
 81
 82
 83      0000                    GIE     = 0
 84      0001                    ENI     = 1
 85      0002                    BUSY    = 2
 86      0003                    IPND    = 3
 87      0004                    ENTI    = 4
 88      0005                    TPND    = 5
 89
 90      0002                    IEDG    = 2
 91      0003                    MSEL    = 3
 92      0004                    TRUN    = 4
 93      0005                    TC3     = 5
 94      0006                    TC2     = 6
 95      0007                    TC1     = 7
 96
 97
 98                              .CHIP   820
 99
100                      ;INITIALIZATION OF REGISTERS    ;*******************************
101 0000 DD2F                    LD      SP,#02F
102 0002 BCEE80                  LD      CNTRL,#080
```

TL/DD/11044-4

NATIONAL SEMICONDUCTOR CORPORATION
COP800 CROSS ASSEMBLER, REV: D1, 12 OCT 88

```
103 0005 BCEF03                 LD      PSW,#003                ;GLOBAL INT ENABLE/EXTINT ENABLE
104 0008 BCD401                 LD      PORTGD,#01
105 000B BCD53E                 LD      PORTGC,#03E             ;CONFIG PORTG FOR OUTPUTS
106 000E BCDC09                 LD      PORTD,#09               ;START MOTOR DRIVE VALUE
107 0011 BCD100                 LD      PORTLC,#00              ;CONFIG PORTL FOR INPUTS
108 0014 BCD0FF                 LD      PORTLD,#0FF             ;CONFIG PORTL FOR WEAK PULL-UPS
109 0017 5F                     LD      B,#MS0                  ;SETUP MOTOR DRIVE VALUES
110 0018 9A09                   LD      [B+],#09
111 001A 9A0C                   LD      [B+],#0C
112 001C 9A06                   LD      [B+],#06
113 001E 9E03                   LD      [B],#03
114 0020 D200                   LD      STPPTR,#00              ;INIT STEP POINTER
115 0022 BC0700                 LD      FLGREG,#00              ;INIT FLAG REGISTER
116
117
118                     ;READ, DECODE, AND EXECUTE COMMAND
119                                                             ;*****************************
120 0025 BDD479         TOP:    SBIT    READY,PORTGD            ;TURN ON READY FOR NEXT CMD LED
121 0028 3081                   JSR     WAIT                    ;WAIT FOR CMD AND READ CMD
122 002A BDD469                 RBIT    READY,PORTGD            ;TURN OFF READY FOR NEXT CMD LED
123 002D 9C04                   X       A,CMD                   ;STORE IN CMD REGISTER
124 002F BD0470                 IFBIT   GO,CMD                  ;IF STOP BIT SET
125 0032 08                     JP      STOP                    ;THEN STOP MOTOR
126 0033 BD0472                 IFBIT   MODE,CMD                ;ELSE CHEK MODE
127 0036 3041                   JSR     SSTEP                   ;IF MODE SET THEN GO SINGLE STEP
128 0038 305F                   JSR     CONT                    ;ELSE GO CONTINUOUS
129 003A EA                     JP      TOP                     ;GO WAIT FOR NEXT COMMAND
130                     STOP:                                   ;STOP THE MOTOR
131 003B 308E                   JSR     TMRSET                  ;STOP THE TIMER
132 003D BCD401                 LD      PORTGD,#01              ;TURN OFF ALL LEDS
133 0040 E4                     JP      TOP                     ;GO WAIT FOR NEXT CMD
134
135
136                     ;SINGLE STEP THE MOTOR  (SS)
137
138                     SSTEP:                                  ;*****************************
139 0041 308E                   JSR     TMRSET                  ;STOP TIMER
140 0043 BCD410                 LD      PORTGD,#010             ;TURN ON SS LED (RST ALL OTHER LEDS)
141 0046 3081                   JSR     WAIT                    ;WAIT FOR CMD BYTE 2 (# STEPS)
142 0048 8A                     INC     A                       ;ADD 1 TO CORRECT FOR LOOP
143 0049 9CF0                   X       A,CREG0                 ;STORE #STEPS IN LOBYTE COUNT REG
144 004B 9D04                   LD      A,CMD                   ;LOAD HIBYTE # STEPS
145 004D 65                     SWAP    A                       ;MOVE TO LOWER NIBBLE
146 004E 950F                   AND     A,#0F                   ;GET RID OF UPPER BITS
147 0050 8A                     INC     A                       ;ADD 1 TO CORRECT FOR LOOP
148 0051 9CF1                   X       A,CREG1                 ;MOVE TO HIBYTE OF COUNT REG
149 0053 C0             TP2:    DRSZ    CREG0                   ;DECR LOBYTE AND IF NOT ZERO
150 0054 05                     JP      DO                      ;THEN GO DO A STEP
151 0055 C1             MID:    DRSZ    CREG1                   ;ELSE DECR HIBYTE AND IF NOT ZERO
152 0056 01                     JP      DO2                     ;THEN GO DO A STEP AND RST LO COUNT
153 0057 8D                     RETSK                           ;ELSE END OF LOOP    RETURN
```

TL/DD/11044–5

```
154 0058 D0FF          DO2:    LD      CREG0,#0FF              ;RESET LOBYTE OF COUNTER
155 005A 3098          DO:     JSR     NXTVAL                  ;STEP THE MOTOR
156 005C 3158                  JSR     DELAY                   ;SLOW THE STEPPING
157 005E F4                    JP      TP2                     ;GO TO TOP OF LOOP
158
159
160                    ;RUN THE MOTOR CONTINUOUSLY  (NS = NON-STOP = CONTINUOUSLY)
161
162                    CONT:                                   ;*******************************
163 005F BDEE74                IFBIT   TRUN,CNTRL              ;IF MOTOR ALREADY RUNNING NS
164 0062 01                    JP      CHKSPD                  ;THEN CHECK THE CURRENT SPEED
165 0063 03                    JP      SETGO                   ;ELSE GO START THE MOTOR
166 0064 3148          CHKSPD: JSR     SPEED                   ;COMPARE INPUT WITH ACTUAL SPD
167 0066 8E                    RET                             ;IF EQUAL RET ELSE RESTART MOTOR
168 0067 308E          SETGO:  JSR     TMRSET                  ;STOP THE TIMER
169 0069 BCD420                LD      PORTGD,#020             ;TURN ON CONTINUOUS LED
170 006C 3126                  JSR     TIMVAL                  ;CALCULATE TIMER (SPEED) VALUE
171 006E AE                    LD      A,[B]                   ;LOAD A WITH TVAL1
172 006F 9CEB                  X       A,TMRHI                 ;MOVE SPEED VAL INTO TIMER
173 0071 AB                    LD      A,[B-]                  ;LOAD A WITH TVAL1 POINT TO TVAL0
174 0072 9CED                  X       A,TAUHI                 ;MOVE SPEED VAL INTO AUTORELOAD REG
175 0074 AE                    LD      A,[B]                   ;LOAD A WITH TVAL0
176 0075 9CEA                  X       A,TMRLO                 ;MOVE SPEED VAL INTO TIMER
177 0077 AE                    LD      A,[B]                   ;LOAD A WITH TVAL0
178 0078 9CEC                  X       A,TAULO
179 007A BDEF7C                SBIT    ENTI,PSW                ;ENABLE TIMER INTERRUPT
180 007D BDEE7C                SBIT    TRUN,CNTRL              ;START THE TIMER
181 0080 8E                    RET                             ;RET TO MAIN AND WAIT FOR TMRINT
182
183                    ;SUPPORT ROUTINES ***********************************************************
184
185                    WAIT:                                   ;*******************************
186                            ;WAIT FOR AN EXTERNAL INTERRUPT TO SIGNAL AN INCOMMING COMMAND
187                            ;READ THE INCOMMING COMMAND FROM PORT L
188 0081 BD0770                IFBIT   INT,FLGREG              ;IF EXTERNAL INTERRUPT OCCURED
189 0084 01                    JP      OUT                     ;THEN JUMP OUT OF LOOP
190 0085 FB                    JP      WAIT                    ;ELSE CONTINUE TO WAIT
191 0086 BD0768        OUT:    RBIT    INT,FLGREG              ;RESET EXTERNAL INTERRUPT FLAG
192 0089 9DD2                  LD      A,PORTLP                ;READ INCOMMING COMMAND
193 008B 96FF                  XOR     A,#0FF                  ;COMPLEMENT INCOMMING COMMAND
194 008D 8E                    RET                             ;RETURN COMMAND IN ACC
195
196
197                    TMRSET:                                 ;*******************************
198                            ;RESET THE TIMER
199 008E BDEE6C                RBIT    TRUN,CNTRL              ;STOP THE TIMER
200 0091 BDEF6D                RBIT    TPND,PSW                ;RESET THE TIMER PENDING BIT
201 0094 BDEF6C                RBIT    ENTI,PSW                ;DISABLE TIMER INTERRUPT
202 0097 8E                    RET
203
204
```

TL/DD/11044–6

NATIONAL SEMICONDUCTOR CORPORATION
COP800 CROSS ASSEMBLER, REV: D1, 12 OCT 88

```
205                     NXTVAL:                                 ;*******************************
206                             ;SEND THE NEXT DRIVE VALUE TO STEP THE MOTOR ONE STEP IN THE
207                             ;APPROPRIATE DIRECTION (CW OR CCW)
208 0098 9DF2                   LD      A,STPPTR                ;LOAD STEP VALUE POINTER
209 009A DED4                   LD      B,#PORTGD               ;POINT TO PORT G
210 009C BD0471                 IFBIT   DIR,CMD                 ;IF CLOCKWISE
211 009F 11                     JP      IPTR                    ;THEN GO INCREMENT POINTER
212 00A0 6A             DPTR:   RBIT    CW,[B]                  ;ELSE RST CW LED
213 00A1 7B                     SBIT    CCW,[B]                 ;TURN ON CCW LED
214 00A2 8B                     DEC     A                       ;AND DECREMENT POINTER
215 00A3 92FF                   IFEQ    A,#0FF                  ;IF OFF BOTTOM OF STEPS
216 00A5 9803                   LD      A,#03                   ;THEN LOOP TO TOP OF STEPS
217 00A7 9CF2           WRVAL:  X       A,STPPTR                ;A -> STPPTR (SAVE NEW STPPTR)
218 00A9 9DF2                   LD      A,STPPTR                ;[STPPTR] -> PORTD (LOOKUP VAL)
219 00AB 9CFE                   X       A,B
220 00AD AE                     LD      A,[B]
221 00AE 9CDC                   X       A,PORTD                 ;WRITE STEP VALUE TO MOTOR
222 00B0 8E                     RET
223 00B1 6B             IPTR:   RBIT    CCW,[B]                 ;TURN OFF CCW LED
224 00B2 7A                     SBIT    CW,[B]                  ;TURN ON CW LED
225 00B3 8A                     INC     A                       ;INCREMENT THE STEP POINTER
226 00B4 9204                   IFEQ    A,#04                   ;IF OFF TOP OF STEPS
227 00B6 64                     CLR     A                       ;THEN LOOP TO BOTTOM OF STEPS
228 00B7 EF                     JP      WRVAL                   ;GO WRITE VALUE TO MOTOR
229
230
231                     ;INTERRUPT HANDLERS
232         00FF                . = 0FF                         ;*******************************
233                             ;BRANCH TO THE APPROPRIATE INTERRUPT HANDLER
234 00FF BDEF75                 IFBIT   TPND,PSW                ;TIMER UNDERFLOW
235 0102 08                     JP      TMRINT
236 0103 BDEF73                 IFBIT   IPND,PSW                ;EXTERNAL INTERRUPT
237 0106 16                     JP      EXTINT
238 0107 BDEF78                 SBIT    GIE,PSW                 ;SOFTWARE TRAP
239 010A 8D                     RETSK
240
241                     TMRINT:                                 ;*******************************
242                             ;RESET THE TIMER INTERRUPT PENDING BIT AND STEP THE MOTOR
243 010B 9CF9                   X       A,0F9                   ;CONTEXT SAVE ROUTINE
244 010D 9DFE                   LD      A,B
245 010F 9CFA                   X       A,0FA
246 0111 BDEF6D                 RBIT    TPND,PSW                ;RESET PENDING BIT
247 0114 3098                   JSR     NXTVAL                  ;STEP THE MOTOR
248 0116 9DFA                   LD      A,0FA                   ;CONTEXT RESTORE ROUTINE
249 0118 9CFE                   X       A,B
250 011A 9DF9                   LD      A,0F9
251 011C 8F                     RETI
252
253                     EXTINT:                                 ;*******************************
254 011D BD0778                 SBIT    INT,FLGREG              ;SET INTERRUPT OCCURED FLAG
255 0120 3158                   JSR     DELAY                   ;WAIT
```

TL/DD/11044–7

```
256 0122 BDEF6B                 RBIT    IPND,PSW                ;RESET PENDING BIT
257 0125 8F                     RETI
258
259                     ;SUPPORT ROUTINES CONTINUED
260
261                     TIMVAL:                                 ;*******************************
262                             ;During continuous operation, the motor is stepped once every
263                             ;timer underflow. Therefore, a timer value is calculated that will
264                             ;produce timer underflows every X microseconds causing the motor
265                             ;to step Xsteps/second.
266                             ;For example:  To step 100 times per second.
267                             ;       microseconds/step = 1000000uS/sec x 1sec/100steps = 10000
268                             ;       10000uS/step = 02718Hex uS/step
269                             ;       1uS = one count down of the timer
270                             ; Therefore, load the timer with 02718H for 100 steps/sec.
271
272 0126 DE14                      LD      B,#TVAL0                ;POINT TO STORAGE FOR TIMVAL
273 0128 BD0474                 FASTER: IFBIT   4,CMD                   ;IF LOWEST SPEED BIT SET
274 012B 17                     JP      SLOWER                  ;THEN USE SLOWEST SPEED
275 012C BD0475                 IFBIT   5,CMD                   ;IF SECOND LOWEST SPD BIT SET
276 012F 0E                     JP      SLOW                    ;THEN USE SLOW SPEED
277 0130 BD0476                 IFBIT   6,CMD                   ;IF SECOND HIGHEST SPD BIT SET
278 0133 05                     JP      FAST                    ;THEN USE FAST SPEED
279 0134 9A02                   LD      [B+],#02                ;ELSE USE FASTEST SPEED
280 0136 9E08                   LD      [B],#08                 ;400steps/sec = 2rev/sec
281 0138 8E                     RET
282 0139 9A88                   FAST:   LD      [B+],#088               ;200steps/sec = 1rev/sec
283 013B 9E13                   LD      [B],#013
284 013D 8E                     RET
285 013E 9A18                   SLOW:   LD      [B+],#018               ;100steps/sec = .5rev/sec
286 0140 9E27                   LD      [B],#027
287 0142 8E                     RET
288 0143 9A54                   SLOWER: LD      [B+],#054               ;25steps/sec = .125rev/sec
289 0145 9E9C                   LD      [B],#09C
290 0147 8E                     RET
291
292
293                     SPEED:                                  ;*******************************
294                             ;COMPARE CURRENT MOTOR SPEED WITH DESIRED MOTOR SPEED
295 0148 3126                   JSR     TIMVAL                  ;CALCULATED DESIRED SPEED VAL
296 014A 9D14                   LD      A,TVAL0
297 014C BDEC82                 IFEQ    A,TAULO                 ;IF DESIRED LBYTE EQUALS CURRENT LBYTE
298 014F 01                     JP      TSTHI                   ;THEN GO TEST HI-BYTE
299 0150 8D                     RETSK                           ;ELSE NOT EQUAL RETURN AND SKIP
300 0151 9D15                   TSTHI:  LD      A,TVAL1
301 0153 BDED82                 IFEQ    A,TAUHI                 ;IF HI-BYTE EQUALS CURRENT HI-BYTE
302 0156 8E                     RET                             ;THEN DESIRED = CURRENT RETURN
303 0157 8D                     RETSK                           ;ELSE DESIRED != CURRENT RET & SKIP
304
305                     DELAY:                                  ;*******************************
306                             ;INSERT A DELAY
```

TL/DD/11044–8

NATIONAL SEMICONDUCTOR CORPORATION
COP800 CROSS ASSEMBLER, REV: D1, 12 OCT 88

```
307 0158 D301                    LD      0F3,#01              ;FOR SINGLE STEP & EXTINT DEBOUNCE
308 015A D4FF            DLY1:   LD      0F4,#0FF             ;APPROX .256mS X 6
309 015C C4              DLY2:   DRSZ    0F4
310 015D FE                      JP      DLY2
311 015E C3                      DRSZ    0F3
312 015F FA                      JP      DLY1
313 0160 8E                      RET
314
315                              .END
```

TL/DD/11044–9

```
B       00FE          BUSY    0002  *       CCW     0003          CHKSPD  0064
CMD     0004          CNTRL   00EE          CONT    005F          CREG0   00F0
CREG1   00F1          CW      0002          DELAY   0158          DIR     0001
DLY1    015A          DLY2    015C          DO      005A          DO2     0058
DPTR    00A0  *       ENI     0001  *       ENTI    0004          EXTINT  011D
FAST    0139          FASTER  0134  *       FLGREG  0007          GIE     0000
GO      0000          IEDG    0002  *       INT     0000          IPND    0003
IPTR    00B1          MID     0055  *       MODE    0002          MS0     0000
MS1     0001  *       MS2     0002  *       MS3     0003  *       MSEL    0003  *
NS      0005  *       NXTVAL  0098          OUT     0086          PORTD   00DC
PORTGC  00D5          PORTGD  00D4          PORTGP  00D6  *       PORTI   00D7  *
PORTLC  00D1          PORTLD  00D0          PORTLP  00D2          PSW     00EF
READY   0001          SETGO   0067          SIOR    00E9  *       SLOW    013E
SLOWER  0143          SP      00FD          SPEED   0148          SS      0004  *
SSTEP   0041          STEPS   0005  *       STOP    003B          STPPTR  00F2
TAUHI   00ED          TAULO   00EC          TC1     0007  *       TC2     0006  *
TC3     0005  *       TIMVAL  0126          TMRHI   00EB          TMRINT  010B
TMRLO   00EA          TMRSET  008E          TOP     0025          TP2     0053
TPND    0005          TRUN    0004          TSTHI   0151          TVAL0   0014
TVAL1   0015          WAIT    0081          WRVAL   00A7          X       00FC
```

TL/DD/11044–10

MACRO TABLE

```
   NO WARNING LINES

   NO ERROR LINES

  282  ROM BYTES USED

SOURCE CHECKSUM = 80C0
OBJECT CHECKSUM = 0520

INPUT   FILE C:MOTOR.MAC
LISTING FILE C:MOTOR.PRN
OBJECT  FILE C:MOTOR.LM
```

TL/DD/11044–11

MF2 Compatible Keyboard with COP8 Microcontrollers

National Semiconductor
Application Note 734
Volker Soffel

ABSTRACT

This application note describes the implementation of an IBM MF2 compatible keyboard with National Semiconductor's COP888CL or COP943C/COP880CL microcontrollers. Two different solutions have been developed. One solution, suitable for laptop/notebook keyboards is based on the COP888CL with special power saving techniques. The other for most price competitive standard desktop keyboards is based on the COP943C/COP880C microcontrollers. The same principles can be applied to all types of keyboards or data input devices.

FEATURES

- Single chip solution
- Low cost R/C or ceramic oscillator optional
- LED direct drive capability
- I/Os with software programmable on chip pull-ups
- Current saving M2CMOS technology
- Multi-input wakeup and HALT mode for further power consumption reduction (COP888CL only)
- Software key rollover
- Schmitt triggers on keyboard data and clock lines

INTRODUCTION

The expression MF2 keyboard stands for multi-functional keyboard version 2. This type of keyboard was first developed and defined by IBM for use with all types of PC (XT, AT, PS/2). In the meantime it has become an industry standard and today nearly all PCs have an MF2 compatible keyboard. As the name suggests, this keyboard features all operation modes which are necessary to stay compatible with the older XT and AT type keyboards. In the following chapters the features and functions of an MF2 keyboard as well as their implementation with a COP8 microcontroller are described.

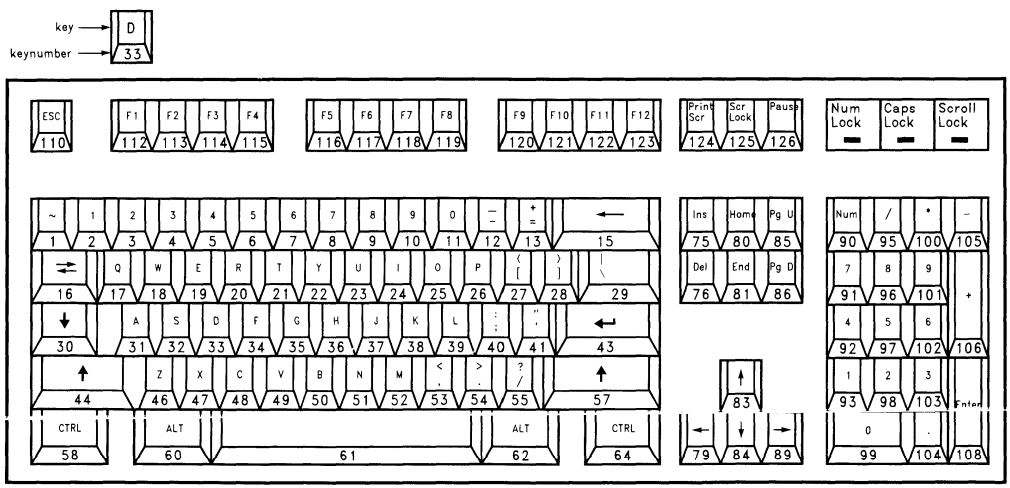
MF2 KEYBOARD KEY-LAYOUT

Figure 1 shows the key layout of the U.S. version of an MF2 keyboard. Its outer appearance is characterized by 101 keys (102 for some countries), a separate cursor and numeric key pad, and 12 function keys in the upper row. The keyboard sends a "make" code if a key is depressed and a "break" code if the key is released. These make and break codes are independent of any country-specific keyboard layouts, which means they are independent of the symbols printed on the keys. These codes are solely determined by the physical position of a key on the keyboard. The physical position of a key on an MF2 keyboard is defined by its assigned key number, which is shown in *Figure 1.*

HARDWARE

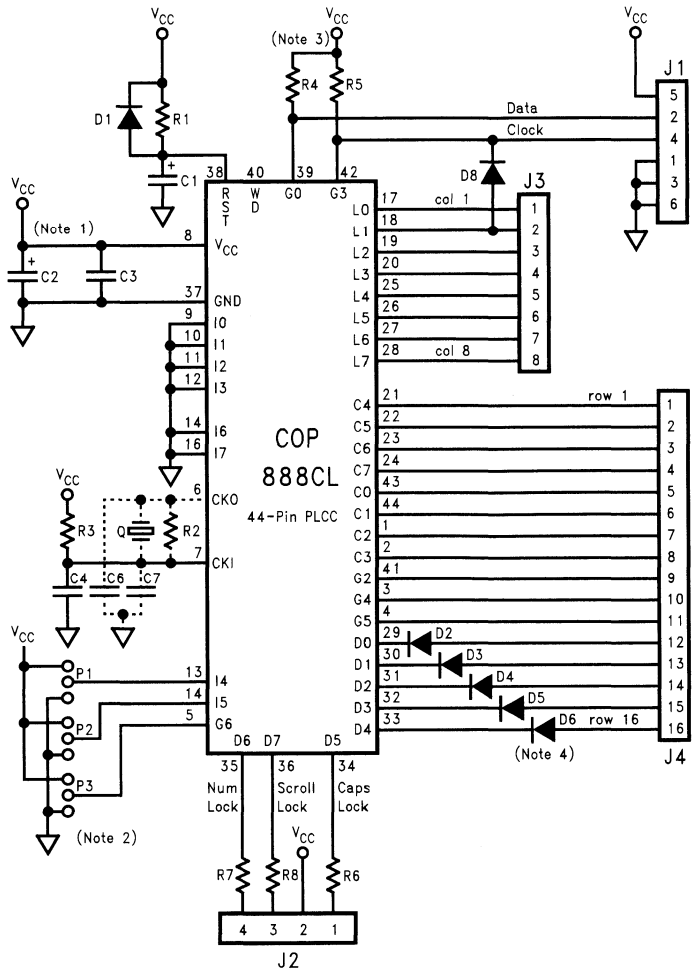
Laptop/Notebook Keyboard With COP888CL

Figure 2 shows the schematics of an MF2 keyboard with a COP888CL microcontroller. The G, C and L ports of the COP888CL are software programmable I/Os and can be programmed either as TRI-STATE® inputs, inputs with weak pull-up, push-pull output low, or push-pull output high.



TL/DD/11091–10

FIGURE 1. MF2 Keyboard U.S. Layout



TL/DD/11091–11

Note 1: C2 (47 μF level off capacitor) can be removed when the power supply ripple < ±10%, 0.5 V/ms.

Note 2: Jumper P1: Mode select: 0 = XT-mode, 1 = AT-mode. Jumper P2, P3: not used.

Note 3: Care must be taken if there are pullups in the computer system that clock/data line current < 3 mA.

Note 4: Diodes D2–D6 should be removed if keyboard has hardware keyrollover (diodes in matrix).

FIGURE 2. MF-2 Keyboard Schematics with a 44-Pin COP888CL

The keyboard is organized as an 8 input by 16 output matrix. The COP888CL's L port is configured as a weak pull-up input port, thus allowing the use of the multi-input wakeup feature. Most of the time the chip is in the current saving HALT mode (Idd $\leq$ 10 μA). Any keystroke or a data transmission from the computer will create a high to low transition on one of the L lines, which wakes up the μC from HALT mode. After returning from the HALT mode, the keyboard is scanned in order to detect which key is pressed and the appropriate key code is sent to the computer. This event-driven keyboard scanning results in lowest possible current consumption as HALT mode is even entered between successive single keystrokes. The diodes in the D-lines of the key matrix prevent a high current from being drawn. When two keys in the same column are pressed, two outputs could be potentially connected together: one of the D output lines, which is high and the polled line, which is pulled low. In this case, excessive current would be drawn without the protection diodes. These diodes can be omitted if the keyboard already has decoupling diodes in its matrix (hardware key rollover). All other matrix lines source current in the μA range and there is no need for current limiting diodes.

The G0 and G3 pins are used for the keyboard data and clock lines. The pull-ups on these lines ensure a defined logic "1" level. The keyboard interface on the computer side uses open collector drivers and the G0, G3 pins of the COP888CL are configured as TRI-STATE (Hi–Z) inputs when a "1" is written to the data or clock line. To output a logic "0" the μC pulls the data or clock line low (push-pull low output). A maximum current of 3 mA can be sunk into the data and clock pins. Schmitt triggers on the data and clock line inputs reduce the risk of errors in the data received by the keyboard.

The microcontroller provides the option of using a low cost R/C oscillator with frequency variation tight enough to fulfill the requirements for a keyboard, in addition to the option of using a crystal or a ceramic clock.

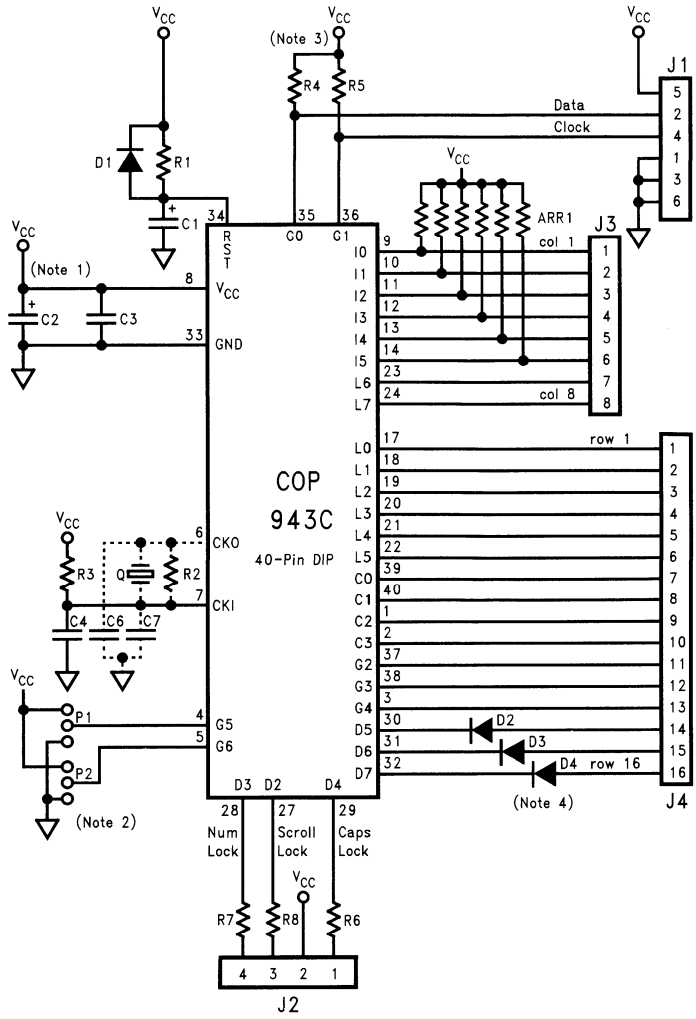
The XT or AT/PS-2 operation mode can be selected via a hardware switch. Additional inputs for customer specific settings are available.

The three LEDs of an MF2 keyboard are driven directly by three of the COP888CL's high sink D-lines (max. 15 mA for each pin), thus eliminating the need for additional LED drivers or transistors.

The keyboard logic generates a Power-On Reset (POR) signal when the power is first applied to the keyboard. After POR the keyboard performs the Basic Assurance Test (BAT). The BAT consists of a keyboard controller self-test. During the BAT, any activity on the data and clock lines is ignored. The 3 keyboard LEDs are turned on at the beginning and turned off at the end of the BAT. Upon satisfactory completion of the BAT, the keyboard sends the BAT completion code (hex AA) to the computer and keyboard scanning begins. Any code other than hex AA is interpreted by the computer as a BAT error.

Desktop Keyboard with COP943C or COP880C

Figure 3 shows the schematic for an MF2 keyboard with the COP943C/COP880C. The only difference compared to COP888CL solution is that the COP943C/COP880C microcontrollers do not have the multi-input wakeup feature, which allows an event driven keyboard scanning. The key matrix is therefore continuously scanned in a loop. With the COP943C/COP880C solution a part of the I port is used as the key matrix input. The I port is a TRI-STATE (Hi–Z) input port (requires external pull-ups).



TL/DD/11091–12

Note 1: C2 (47 μF level off capacitor) can be removed when the power supply ripple < ±10%, 0.5 V/ms.

Note 2: Jumper P1: Mode select: 0 = XT-mode, 1 = AT-mode. Jumper P2: P3: not used.

Note 3: Care must be taken if there are pullups in the computer system that clock/data line current < 3 mA.

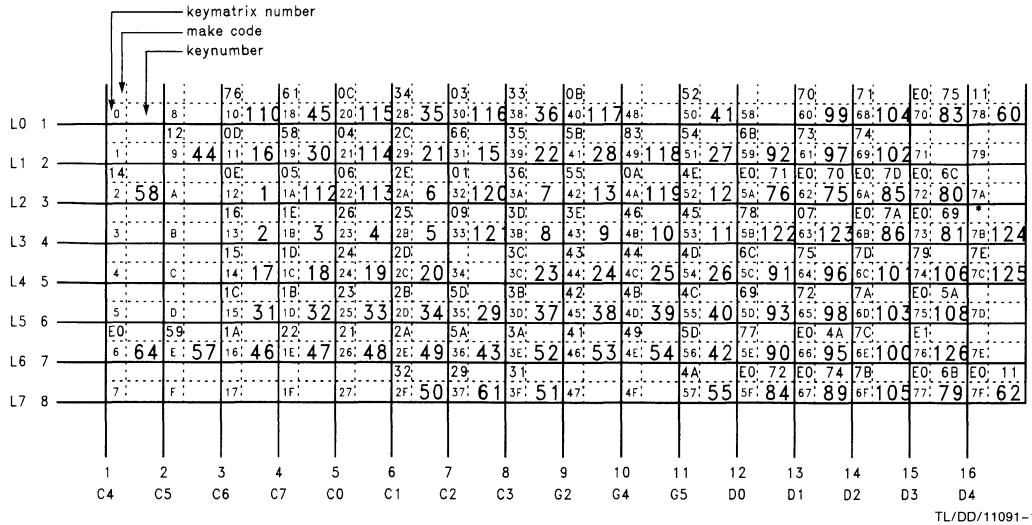
Note 4: Diodes D2–D4 should be removed if keyboard has hardware keyrollover (diodes in matrix).

FIGURE 3. MF-2 Keyboard Schematic with a 40-Pin COP943C/COP880C

Key Matrix Organization

Figure 4 shows an example of what an MF2 keyswitch matrix could look like. Each key position in the matrix is marked with its key number.

For example: Key number "58" is located at the key matrix position number "2" and has the AT-set make code "14 Hex". Looking at *Figure 1,* one can see that key number "58" belongs to the left "CNTRL" key. Note that the "SHIFT", "CNTRL" and "ALT" keys are located in their own matrix lines, separate from all other keys. The reasons for that will be explained in the chapter "Software Key Rollover".



TL/DD/11091–13

FIGURE 4. Keyboard Matrix COP888CL AT Code Set

Code Sets

The MF2 keyboard supports 3 different sets of make and break codes. Code set 1 is used for XT/PC and PS/2–30 compatible computers. Code set 2 is used for AT and all other PS/2 models compatible computers and code set 3 is used for workstations and terminal emulations on the PC. The country specific keyboard driver on the PC side converts the "key position" codes from the keyboard into the ASCII codes that correspond to the characters printed on the keycaps (as long as the right driver is installed on the PC). Appendix 1 gives a complete overview of the key numbers and their make and break codes for all 3 code sets. The symbols of the U.S. keyboard layout are only listed for reference and are different for other country layouts. The break code for code set 1 is equal to the make code with the most significant bit set. The make codes preceded with a "F0 Hex" code give the break codes of code sets 2 and 3.

KEYBOARD SOFTWARE

The software of the keyboard microcontroller can be subdivided into the following five main tasks:

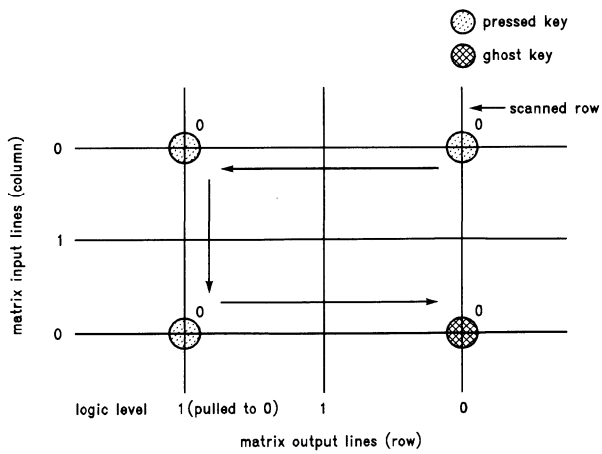
- key detection
- software key rollover
- key decoding and encoding
- keycode transmission
- keyboard command set

Key Detection

Key detection is done by scanning the keyboard matrix in the following way. Sequentially each of the 16 matrix output lines are pulled low, while all the others are high. The 8 matrix input lines are read and the 8-bit input value is compared with the result of the previous scanning of the same matrix output line (a history of the previous scan is kept in the μC's RAM). Thus the keyboard microcontroller's key detection routine detects any key change in that matrix output line (key pressed or released) since the previous scan. It is important to recognize released keys, as the MF2 keyboard not only sends a key's "make" code when the key is pressed, but also a key's "break" code when the key is released. Key debouncing is performed by software by making sure that the time between two scans is bigger than the key bounce time (typically 8 ms).

Software Key Rollover

Software key rollover means that no decoupoing diodes are used in the key switch matrix. However, the keyboard action is still N key rollover in nature. That is, if N keys are depressd in some sequence and held down, the make code of these keys is transmitted in that sequence. However, if three keys from three corners of a rectangle in the key switching matrix are depressed, a "ghost" key (a key which is not really pressed) would be created (see *Figure 5*). To prevent this, a special algorithm, which checks for such special key combinations, has been implemented into the keyboard software. If a "ghost" key has been detected the keyboard outputs the "key detection error code" and the N key rollover reverts to a 2 key rollover. To ensure that all 3-key combinations used on a PC (e.g., CNTRL + ALT + DEL) are still possible, keyboard manufacturers using this method organize the key switch matrix accordingly (an example is given in *Figure 4*).



TL/DD/11091–14

FIGURE 5. Software Key Rollover

```
;       SOFTWARE KEY ROLLOVER
;
;LENGTHC: COUNTER  FOR NO.  OF BYTES (15 FOR A 16 BY  8  MATRIX)
;         WHICH  HAVE  TO BE COMPARED WITH THE  ACTUAL SCANNED
;         BYTE.
;LASTSCN: RAM LOCATION WHICH CONTAINS THE RESULT OF THE ACTUAL
;         SCANNED LINE
;
;PNTSCAN: RAM  LOCATION  WHICH CONTAINS A POINTER  TO  THE  RAM
;         CELL IN THE  SCAN HISTORY TABLE THAT STORES THE RESULT
;         OF  THE PREVIOUS SCAN FOR THE ACTUAL  SCANNED  MATRIX
;         LINE
;SCNLOT:  START ADRESS OF THE RAM SCAN HISTORY TABLE (16 BYTES)
;MATLEN:  MATRIX LENGTH (IN THIS CASE MATLEN=16dec)
;BITC  :  SHIFT COUNTER FOR BYTE SHIFT
;TYPSAV:  RAM ADRESS OF TYPEMATIC RATE SAVE REGISTER
;TYPST :  RAM ADRESS FOR TYPEMATIC RATE VALUE
;STATUS:  RAM ADRESS OF GENERAL STATUS FLAG REGISTER
;STAT2 :  RAM ADRESS OF GENERAL STATUS FLAG REGISTER 2
;TYPCO1:  RAM ADRESS OF REGISTER THAT CONTAINS TYPEMATIC KEY
;         MAKE CODE
;SCNCNT:  SCAN COUNTER FOR 16 MATRIX LINES
;
;
;
        .LOCAL
KEYROL:

        LD      LENGTHC,#00F    ;LOAD TABLE LENGTH COUNTER
        LD      X,#LASTSCN      ;POINT TO RAM LOCATION WHERE
                                ;RESULT OF PREVIOUS SCAN IS
                                ;STORED
        LD      A,PNTSCAN       ;LOAD POINTER TO ACTUAL SCAN
                                ;LINE
        INC     A
        X       A,B             ;POINT TO THE NEXT SCAN LINE
$NEXTB:
        IFBNE   #((SCNLOT+MATLEN)&00F) ;IF END OF HISTORY SCANTABLE
                                       ;IN RAM NOT REACHED
        JP      $1                     ;THEN OK
        LD      B,#SCNLOT       ;ELSE POINT TO BEGINNING OF TABLE
$1:
        LD      A,[X]           ;COMPARE NEW SCANNED MATRIX LINE
        OR      A,[B]           ;WITH ALL OTHER PREVIOUS SCANNED
                                ;BYTES IN TABLE
        IFEQ    A,#0FF          ;IF NO KEYS PRESSED IN
                                ;SAME INPUT LINE
        JP      $INCB           ;THEN COMPARE WITH NEXT BYTE
                                ;IN SCAN TABLE
                                ;ELSE LOOK IF MORE THAN
                                ;TWO KEYS ARE PRESSED
                                ;IN ONE OF THE TWO
                                ;COMPARED BYTES
        LD      A,[X]           ;LOAD 1ST OF COMP.BYTES
```

TL/DD/11091–1

```
        LD      BITC,#08        ;LOAD BIT COUNTER
$ZERO1:
        RRC     A
        IFNC                    ;IF 1 KEY PRESSED
        JP      $ZERO3          ;THEN TEST IF 2ND
                                ;KEY IS PRESSED
        DRSZ    BITC            ;IF NOT ALL BITS CHECKED
        JP      $ZERO1          ;THEN CONTINUE CHECK
        JP      $INCB

$ZERO2:
        RRC     A
        IFNC                    ;IF 2ND KEY PRESSED
        JP      $ENDLP          ;THEN ERROR: "GHOST KEY"
$ZERO3: DRSZ    BITC            ;IF NOT ALL BITS CHECKED
        JP      $ZERO2          ;THEN CONTINUE CHECK
$INCB:
        LD      A,[B+]          ;INC B
        DRSZ    LENGTHC         ;IF NEW SCANNED MATRIX LINE
                                ;NOT COMPARED WITH ALL OTHER
                                ;BYTES IN TABLE
        JP      $NEXTB          ;THEN COMP. WITH NEXT
                                ;BYTE IN TABLE
        SC                      ;IF ALL COMPARED, SET NO ERROR
                                ;FLAG
$ENDLP:
        LD      B,#STAT2        ;POINT TO STATUS FLAG REGISTER
        IFNC                    ;ERROR DURING THIS SCAN?
        JP      $ERROR          ;YES, DO ERROR PROCEDURE
        IFBIT   ERR2,[B]        ;ERROR DURING PREVIOUS SCANS,
                                ;BUT NO ERROR DURING THIS
                                ;SCAN?
        JP      $RESTORE        ;YES, RESTORE TYPEMATIC RATE
        RET

$RESTORE:
        RBIT    ERR2,[B]
        JSR     TSTOP           ;STOP TYPEMATIC TIMER
        LD      A,TYPSAV        ;LOAD SAVED TYPEMATIC VALUE
        X       A,TYPST         ;RESTORE OLD TYPEMATIC VALUE
        RET                     ;NO ERROR DURING THIS SCAN:
                                ;RETURN

$ERROR:
        IFBIT   ERR2,[B]        ;IF ERROR OCURRED ALREADY
                                ;DURING PREVIOUS SCAN
        JP      $ERREND         ;THEN DO NOTHING
        SBIT    ERR2,[B]        ;ELSE SET PREVIOUS ERROR FLAG
        LD      B,#TYPST        ;POINT TO TYPEMATIC VALUE
                                ;REGISTER
        LD      A,[B]
        X       A,TYPSAV        ;SAVE TYPEMATIC RATE/DELAY
        LD      [B],#07F        ;SET TYPEMATIC TO 1s DELAY,
                                ;2 CHARACTERS/s FOR ERROR CODE
```

TL/DD/11091–2

```
                                    ;REPETITION
        LD      A,#000              ;IF SET2,3 ERROR CODE 00
        LD      B,#STATUS           ;POINT TO STATUS FLAG REGISTER
        IFBIT   SET1,[B]
        LD      A,#0FF              ;ELSE ERROR CODE FF
        X       A,TYPCO1            ;PUT IN TYPEMATIC BUFFER
        JSR     TSTART              ;INIT & START TYPEMATIC TIMER
$ERREND:
        LD      A,SCNCNT            ;INCREMENT SCAN COUNTER
        INCA
        X       A,SCNCNT
        RETSK                       ;RET AND SKIP FOR ROLLOVER ERROR
        .LOCAL
        .END
```

TL/DD/11091–3

Key Decoding and Encoding

After detection of a key change (pressing or releasing a key), the software first has to determine the physical location of the key in the key matrix. This decoding process is done by calculating an internal key number out of the key matrix column and row position of the changed key. At the same time, it is determined if the key has been pressed or released. A pressed or released key is then signaled by setting or resetting a "key down" flag in RAM. The internal key number and the "key down" status flag are the input parameters to the key encoding procedure. The internal key number is used to get the "make" code for the key out of a ROM look-up table, which has been matched to the physical matrix organization of the keyboard. If the "key down" flag is reset (key is released) the software calculates the key "break" code out of the previously fetched key "make" code. In this way, each pressed or released key is encoded with its appropriate "make" or "break" code, which is then written to the keyboard controllers 16 byte output buffer (FIFO) until the computer interface is ready to receive it. Before writing to the FIFO the software checks whether there is still enough capacity to store the key code.

Key Repetition

All keys are typematic (repetitive) by default. That means when a key is pressed and held down, the μC continues to send the "make" code for that key until it is released. When two or more keys are held down, only the code for the last key pressed is repeated. Typematic operation will stop when this key is released, even if other keys are still held down.

The default values for typematic operation are:

delay time = 500 ms
repetition rate = 10 characters/second,

where the delay time is the time which is inserted before a character is repeated for the first time.

Operating Protocol

There are two different transmission protocols for an MF2 keyboard: the AT transmission protocol and the XT transmission protocol. Data transmission to and from the keyboard is synchronous serial, the data format for the XT mode is:

9 bits in length
1 start bit (high)
8 data bits (LSB first)

The data format for AT and PS/2 modes is:

11 bits in length
1 start bit (low)
8 data bits (LSB first)
1 parity bit (odd)
1 stop bit (high)

If no data is transmitted, both data and clock lines are in the high state. The clock signal is always provided by the keyboard. *Figure 6* shows the XT and the AT protocol timings.

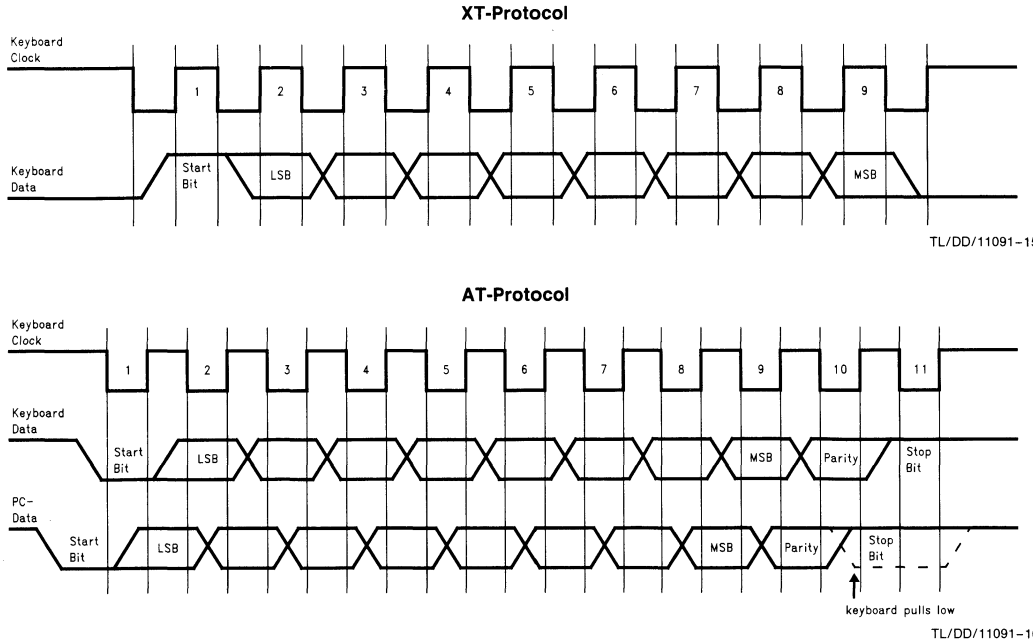


FIGURE 6. XT and AT Protocol Timings

Keyboard Data Transmission in XT Format

At the falling edge of the clock, the start bit (high) is shifted out, followed by the 8 data bits (least significant bit first). Data is valid on the rising edge of the clock and changes after the falling edge of the clock.

Keyboard Data Transmission in AT Format

Before sending data, the keyboard monitors the clock and data lines. If the clock line is low, then the keyboard is disabled by the computer and no data is transmitted. The microcontroller continues to scan the keyboard and stores key data in its output buffer. If the data line is low, while the clock line is high, the computer requests to send and the keyboard goes into receive mode. The keyboard is only allowed to transmit data when both data and clock lines are high.

The keyboard pulls the data line low (start bit) and starts the clock. The 8 data bits (least significant bit first) are shifted out, followed by the parity (odd) and stop bit (high). Data is valid after the falling edge of the clock and changes after the rising edge of the clock. If no data is transmitted both data and clock lines are high. If the computer pulls the clock line low for at least 60 μs before the 10th bit is transmitted, the keyboard stops transmission and stores the aborted data in its output buffer.

```
;  SENDBY:   SEND BYTE TO COMPUTER
;INPUT PARAMETER:
;BYTSEN: RAM LOCATION CONTAINING THE
;        BYTE TO BE TRANSMITTED
;OUTPUT:
;        DATSEN FLAG IN STATUS REGISTER
;        1=BYTE SENT,0=BYTE NOT SENT
;PARCNT: PARITY COUNTER REGISTER
;BITC  : DATA LENGTH COUNTER FOR TRANSMISSION LOOP
;
;CLOCK HIGH TIME (=CLOCK LOW TIME) = 40us
;AT 3.58MHz CLOCK (INSTR. CYCLE = 2.79us)
;
;DATA REGISTER OF PORT G DATA AND CLOCK LINES IS
;PRESET WITH "0"

        .LOCAL

SendBy:
        LD      B,#STATUS       ;POINT TO  STATUS FLAG  REGISTER
        RBIT    DatSen,[B]      ;RESET "BYTE SEND" FLAG
        LD      A,BytSen        ;LOAD BYTE TO SEND
        LD      BITC,#009       ;DATA LENGTH
        IFBIT   PCXT,[B]        ;IF XT MODE
        JMP     PCMode          ;THEN JUMP TO XT
                                ;SEND ROUTINE
                                ;ELSE SEND AT PROTOCOL
$ATSEND:
        LD      PARCNT,#10      ;LOAD PARITY COUNTER
        LD      B,#PORTGP       ;POINT TO GPORT INPUT
                                ;REGISTER
WAITS:
```

TL/DD/11091–4

```
        IFBIT   ClockL,[B]      ;IF CLOCKLINE HIGH
        JP      $ClocOK         ;THEN OK
        JP      WAITS           ;ELSE KEYBOARD DISABLED:
                                ;WAIT

$ClocOK:
        IFBIT   DataLn,[B]      ;IF DATALINE IS HIGH
        JP      $DataOK         ;THEN OK
        RET                     ;ELSE PC SENDS DATA:
                                ;RETURN (GOTO RECEIVE)


$DataOK:
        LD      B,#PORTGC       ;POINT TO PORT G CONFIGURATION
                                ;REGISTER
        RC                      ;STARTBIT = 0
$SendBt:
        RBIT    ClockL,[B]      ;SET CLOCKLINE HIGH (TRI-STATE)
        IFBIT   ClockL,PORTGP   ;IF PC DOES NOT PULL CLOCKL LOW
        JP      $ClockH         ;THEN OK
        RBIT    DataLn,[B]      ;ELSE SET DATA LINE BACK TO HIGH
        RET                     ;STOP TO SEND

$ClockH:
        IFC                     ;IF BIT TO TRANSMIT = "1"
        JP      $DATHI          ;THEN DATALINE HIGH
        SBIT    DataLn,[B]      ;ELSE DATALINE LOW
        JP      $CLKLOW         ;SET CLOCKLINE LOW
$DATHI:
        RBIT    DataLn,[B]      ;SET DATALINE HIGH (TRI-STATE)

$CLKLOW:
        SBIT    ClockL,[B]      ;SET CLOCKLINE LOW
        IFC                     ;IF BIT=1
        DRSZ    PARCNT          ;THEN DECR. PARITY COUNTER
        RRC     A               ;SHIFT NEXT BIT INTO CARRY
        NOP
        DRSZ    BITC            ;IF NOT ALL BITS SENT
        JP      $SendBt         ;THEN TRANSMIT NEXT BIT
$PARITY:                        ;SEND PARITY BIT
        NOP                     ;DELAY
        NOP
        NOP
        RBIT    ClockL,[B]      ;SET CLOCKLINE HIGH
        IFBIT   00,PARCNT       ;IF NUMBER OF "1" = ODD
        JP      $DLOW           ;THEN PARITY = 0
        RBIT    DataLn,[B]      ;ELSE PARITY = 1
        JP      $CLKL2
$DLOW:
        SBIT    DataLn,[B]      ;SET DATALINE LOW
        NOP
$CLKL2:
        NOP                     ;DELAY
        NOP
```

TL/DD/11091–5

```
        NOP
        SBIT    ClockL,[B]      ;SET CLOCKLINE LOW
        JSR     DEL12           ;INSERT DELAY  12 INSTR. CYCLES

        RBIT    ClockL,[B]      ;SET CLOCKLINE HIGH

                                ;TRANSMIT STOP BIT
        RBIT    DataLn,[B]      ;SET DATA LINE HIGH (STOP BIT)
        JSR     DEL11           ;INSERT DELAY 11 INSTR. CYCLES
        SBIT    Clockl,[B]      ;SET CLOCKLINE LOW
        JSR     DEL12           ;INSERT DELAY 12 INSTR. CYCLES
$ENDSB:
        RBIT    ClockL,[B]      ;SET CLOCKLINE HIGH
        RBIT    DATALN,[B]      ;DATA HIGH (XT MODE)
        LD      B,#STATUS       ;POINT TO STATUS FLAG REG.
        SBIT    DatSen,[B]      ;SET DATA SENT FLAG

        LD      A,BYTSEN
        IFEQ    A,#0FE          ;IF SENT BYTE = RESEND
                                ;COMMAND
        RET                     ;THEN DON'T SAVE
        X       A,SENBYT        ;ELSE SAVE LAST SENT BYTE
                                ;IN SENBYT IN CASE PC ASKS
                                ;KEYBOARD TO RESEND
        RET

;XT TRANSMISSION PROTOCOL
PCMode:
        IFBIT   CLOCKL,PORTGP   ;CLOCKLINE HIGH?
        JP      $PCSND          ;YES,START TO SEND
        JMP     POWRUP          ;ELSE RESET
$PCSND:
        LD      B,#PORTGC
        SBIT    DATALN,[B]      ;DATA LINE LOW BEFORE
                                ;START TO SEND
        SC                      ;START BIT = 1
$PCSEND:
        SBIT    ClockL,[B]      ;CLOCKLINE LOW
        IFC                     ;IF BIT TO SEND=1
        JP      $DATH2          ;THEN SET DATALINE HIGH
        SBIT    DataLn,[B]      ;ELSE SET DATALINE LOW
        NOP                     ;DELAY
        NOP
        NOP
        NOP
        NOP
        NOP
        JP      $CLKHI
$DATH2:
        RBIT    DataLn,[B]      ;SET DATALINE HIGH
        IFBIT   DATALN,PORTGP   ;IF DATALINE HIGH
        JP      $CLKHI          ;THEN OK
                                ;ELSE KEYBOARD DISABLED
        RBIT    CLOCKL,[B]      ;CLOCKLINE HIGH
```

TL/DD/11091-6

```
        RET                         ;STOP TO SEND
$CLKHI:
        RBIT    ClockL,[B]          ;SET CLOCKLINE HIGH
        RRC     A                   ;SHIFT NEXT BIT TO TRANSMIT
                                    ;INTO CARRY
        NOP                         ;DELAY
        NOP
        NOP
        NOP
$PCOK:
        DRSZ    BITC                ;IF NOT ALL BITS SENDED
        JP      $PCSEND             ;THEN CONTINUE
        SBIT    CLOCKL,[B]          ;ELSE CLOCKLINE LOW
        SBIT    DATALN,[B]          ;DATA LOW
        JSR     DELAYD              ;10 INSTR. CYCLES DELAY
        JP      $ENDSB

DEL12:  NOP
DEL11:  NOP
DELAYD: RET
        .LOCAL
        .END
```

TL/DD/11091–7

Keyboard Receives Data

The keyboard can only receive data from the computer in AT-PS/2 mode. The computer pulls the data line low (start bit) after which the keyboard starts to shift out 11 clock pulses within 15 ms. Transmission has to be completed within 2 ms. Data from the computer changes after the falling edge of the clock and is valid before the rising edge of the clock. After the start bit, 8 data bits (least significant bit first), followed by the parity bit (odd) and the stop bit (high) are shifted out by the computer with the clock signal provided by the keyboard. The keyboard pulls the stop bit low in order to acknowledge the receipt of the data. If a transmission error occurred (parity error or similar) the keyboard issues the "RESEND" command to the PC.

```
;
;  RECDAT:   RECEIVE DATA COMMING FROM PC
;
;RETURN, IF PARITY ERROR
;
;RETURN SKIP , IF BYTE WAS RECEIVED
;WITHOUT ERROR
;
;BTRECV: RAM LOCATION CONTAINING THE
;        RECEIVED BYTE
;
;
;BITC  : RECEIVE LOOP COUNTER REGISTER
;PARCNT: PARITY COUNTER REGISTER
;

RecDat:

        CLRA
        LD      B,#PORTGC       ;B POINT TO PORT G
                                ;CONFIGURATION
        LD      X,#BTRECV       ;X POINT TO RECEIVED BYTE
                                ;RAM CELL
        LD      PARCNT,#10      ;LOAD PARITY COUNTER
        LD      BITC,#009       ;LOAD RECEIVE LOOP COUNTER
                                ;(8 DATABITS + 1 PARITY BIT)
        RC                      ;START BIT= "0"
RdByte:

        SBIT    ClockL,[B]      ;SET CLOCKLINE LOW
                                ;(CLOCK IN START BIT)
        IFC                     ;IF "1"-BIT RECEIVED
        DRSZ    PARCNT          ;THEN DECR. PARITY COUNTER
        RRC     A               ;SHIFT CARRY TO BIT 7 OF ACCU
        X       A,[X]           ;STORE RECEIVED BYTE
        LD      A,[X]           ;RESTORE AS  LONG AS NOT
                                ;FULL BYTE RECEIVED


        RBIT    ClockL,[B]      ;SET CLOCKLINE HIGH
;READ IN RECEIVED BIT
        RC                      ;RECEIVED BIT= "0"
        IFBIT   DataLn,PORTGP   ;IF DATALINE = "1"
        SC                      ;THEN RECEIVED BIT= "1"
        DRSZ    BITC            ;9 BITS RECEIVED?
        JP      RdByte          ;NO,LOOP

;CLOCK LOW PULSE AFTER PARITY HAS BEEN RECEIVED
        SBIT    ClockL,[B]      ;SET CLOCKLINE LOW
        JSR     DELAYD          ;INSERT 10 INSTR. CYCLES DELAY
        RBIT    ClockL,[B]      ;SET CLOCKLINE HIGH
;PC SENDS STOP BIT
        SBIT    DataLn,[B]      ;PULL STOP BIT LOW
```

TL/DD/11091-8

```
                                    ;TO ACKNOWLEDGE RECEIPT OF BYTE
        JSR     DELAYD              ;INSERT DELAY
;CLOCK LOW PULSE (CLOCK ACKNOWLEDGE FOR PC)
        SBIT    ClockL,[B]          ;SET CLOCKLINE LOW
        JSR     DELAYD              ;INSERT DELAY
        RBIT    ClockL,[B]          ;SET CLOCKLINE HIGH

        RBIT    DataLn,[B]          ;RETURN DATA TO HIGH

;PARITY CHECK
        IFBIT   00,PARCNT           ;IF NO. OF RECEIVED DATA "1"=ODD
        JP      PAR0                ;THEN PARITY BIT MUST BE "0"
ParOne:                             ;ELSE PARITY BIT MUST BE "1"
        IFC                         ;IF RECEIVED PARITY BIT=1
        RETSK                       ;THEN OK,RETURN SKIP
        JP      PARERR              ;ELSE PARITY ERROR
PAR0:
        IFNC                        ;IF RECEIVED PARITY BIT =0
        RETSK                       ;THEN OK,RETURN SKIP
                                    ;ELSE PARITY ERROR

ParErr: LD      BytSen,#0FE         ;LOAD "RESEND" CODE
        JSR     SByWPo              ;SEND RESEND CODE TO PC
        RET                         ;ERROR,RETURN
        .END
```

TL/DD/11091–9

Commands from the Computer

The following table shows the commands and their hexadecimal values the computer may send to the keyboard. Only AT-PS/2 compatible computers can send commands to the keyboard and the keyboard can only receive the commands when operated in the AT-mode.

The commands can be sent to the keyboard at any time. The keyboard responds within 20 ms to any valid transmission with ACK (FA Hex), except for the ECHO command where the keyboard responds with EE Hex, the RESEND command and the reserved commands.

Command	Hex Value
Set/Reset Mode Indicators	ED
Echo	EE
Reserved	EF
Select Alternate Code Set	F0
Reserved	F1
Read Keyboard ID	F2
Set Typematic Rate/Delay	F3
Enable	F4
Default Disable	F5
Set Default	F6
Set All Keys	
Typematic/No Break	F7
Make/Break/No Typematic	F8
Make/No Typematic	F9
Typem./Make/Br.	FA
Set Key Type	
Typematic/No Break	FB
Make/Break/No Typematic	FC
Make/No Typematic	FD
Resend	FE
Reset	FF

In the XT mode the keyboard only accepts the RESET command, which is assumed when the computer pulls the clock line low for at least 10 ms.

Commands to the Computer

The following table shows the commands and their hexadecimal values the keyboard may send to the system.

Command	Hex Value
Key Detection Error/ Buffer Overrun	00 (Code Sets 2 and 3)
Keyboard ID	83AB
BAT Completion Code	AA
BAT Failure Code	FC
Echo	EE
Acknowledge	FA
Resend	FE
Key Detection Error/ Buffer Overrun	FF (Code Set 1)

SUMMARY

When using National Semiconductor's microcontroller to implement the functions of an MF2 keyboard, very few external components are necessary. *Figure 2* shows the complete schematic of an MF2 keyboard based on the COP888CL. The implementation of software key rollover eliminates the need for decoupling diodes in the 16 by 8 key matrix. LED direct drive capability of the COP8 and a RC oscillator with tolerances tight enough to meet the requirements for a keyboard further reduce component count and price. Schmitt triggers on the ports used for the keyboards data and clock lines add additional security against transmission errors. Where low power consumption is the most important design factor (e.g., laptop or notebook computers) the COP8's M2CMOS technology and the multi-input wakeup feature offer a remarkable improvement over the NMOS controllers used in most of today's existing solutions.

National Semiconductor offers three chips tailored for the needs of a keyboard designer. Starting with the most price competitive 2.5k ROM device COP943C, an upgrade path is provided with the COP880C to 4k ROM. Both devices are intended for the use in standard MF2 desktop keyboards. The COP888CL is ideally suited for notebook or laptop keyboards, as it has special power saving features. The complete software for an MF2 keyboard as well as complete demo keyboards and keyboard evaluation boards for the COP888CL and COP943C/COP880C microcontrollers are available. Contact National Semiconductor's μC marketing or applications for further information.

APPENDIX I. KEY NUMBERS AND THEIR CORRESPONDING MAKE/BREAK CODES FOR ALL THREE CODE SETS

Key Position and Symbol		Table I (XT and PS/2 30)		Table II (AT and PS/2 50, 60, 80)		Table III (Terminal MODE)	
		Make	Break	Make	Break	Code	Type
01	~	29	A9	0E	F0-0E	0E	Typematic
02	! 1	02	82	16	F0-16	16	Typematic
03	@ 2	03	83	1E	F0-1E	1E	Typematic
04	# 3	04	84	26	F0-26	26	Typematic
05	$ 4	05	85	25	F0-25	25	Typematic
06	% 5	06	86	2E	F0-2E	2E	Typematic
07	^ 6	07	87	36	F0-36	36	Typematic
08	& 7	08	88	3D	F0-3D	3D	Typematic
09	* 8	09	89	3E	F0-3E	3E	Typematic
10	(9	0A	8A	46	F0-46	46	Typematic
11	) 0	0B	8B	45	F0-45	45	Typematic
12	_ -	0C	8C	4E	F0-4E	4E	Typematic
13	+ =	0D	8D	55	F0-55	55	Typematic
15	B.S. ←	0E	8E	66	F0-66	66	Typematic
16	TAB	0F	8F	0D	F0-0D	0D	Typematic
17	Q	10	90	15	F0-15	15	Typematic
18	W	11	91	1D	F0-1D	1D	Typematic
19	E	12	92	24	F0-24	24	Typematic
20	R	13	93	2D	F0-2D	2D	Typematic
21	T	14	94	2C	F0-2C	2C	Typematic
22	Y	15	95	35	F0-35	35	Typematic
23	U	16	96	3C	F0-3C	3C	Typematic
24	I	17	97	43	F0-43	43	Typematic
25	O	18	98	44	F0-44	44	Typematic
26	P	19	99	4D	F0-4D	4D	Typematic
27	{ [	1A	9A	54	F0-54	54	Typematic
28	}]	1B	9B	5B	F0-5B	5B	Typematic
29*	| \	2B	AB	5D	F0-5D	5C	Typematic
30	Caps Lk	3A	BA	58	F0-58	14	Make/Break
31	A	1E	9E	1C	F0-1C	1C	Typematic
32	S	1F	9F	1B	F0-1B	1B	Typematic
33	D	20	A0	23	F0-23	23	Typematic
34	F	21	A1	2B	F0-2B	2B	Typematic
35	G	22	A2	34	F0-34	34	Typematic

Key Position and Symbol		Table I (XT and PS/2 30)		Table II (AT and PS/2 50, 60, 80)		Table III (Terminal MODE)	
		Make	Break	Make	Break	Code	Type
36	H	23	A3	33	F0-33	33	Typematic
37	J	24	A4	3B	F0-3B	3B	Typematic
38	K	25	A5	42	F0-42	42	Typematic
39	L	26	A6	4B	F0-4B	4B	Typematic
40	: ;	27	A7	4C	F0-4C	4C	Typematic
41	" '	28	A8	52	F0-52	52	Typematic
42**	| \	2B	AB	5D	F0-5D	53	Typematic
43	Enter (L)	1C	9C	5A	F0-5A	5A	Typematic
44	Shift (L)	2A	AA	12	F0-12	12	Typematic
45**	Macro	56	D6	61	F0-61	13	Typematic
46	Z	2C	AC	1A	F0-1A	1A	Typematic
47	X	2D	AD	22	F0-22	22	Typematic
48	C	2E	AE	21	F0-21	21	Typematic
49	V	2F	AF	2A	F0-2A	2A	Typematic
50	B	30	B0	32	F0-32	32	Typematic
51	N	31	B1	31	F0-31	31	Typematic
52	M	32	B2	3A	F0-3A	3A	Typematic
53	< ,	33	B3	41	F0-41	41	Typematic
54	> .	34	B4	49	F0-49	49	Typematic
55	? /	35	B5	4A	F0-4A	4A	Typematic
57	Shift (R)	36	B6	59	F0-59	59	Make/Break
58	Ctrl (L)	1D	9D	14	F0-14	11	Make/Break
60	Alt (L)	38	B8	11	F0-11	19	Make/Break
61	Space	39	B9	29	F0-29	29	Typematic
62	Alt (R)	E0-38	E0-B8	E0-11	E0-F0-11	39	Make
64	Ctrl (R)	E0-1D	E0-9D	E0-14	E0-F0-14	58	Make
90	Num Lk	45	C5	77	F0-77	76	Make
91	7 Home	47	C7	6C	F0-6C	6C	Make
92	4 ←	4B	CB	6B	F0-6B	6B	Make
93	1 End	4F	CF	69	F0-69	69	Make
96	8 ↑	48	C8	75	F0-75	75	Make
97	5	4C	CC	73	F0-73	73	Make
98	2 ↓	50	D0	72	F0-72	72	Make
99	0 Ins	52	D2	70	F0-70	70	Make
100	*	37	B7	7C	F0-7C	7E	Make

*101-Keyboard only

**102-Keyboard only

7

Key Position and Symbol		Table I (XT and PS/2 30)		Table II (AT and PS/2 50, 60, 80)		Table III (Terminal MODE)	
		Make	Break	Make	Break	Code	Type
101	9 Pg UP	49	C9	7D	F0-7D	7D	Make
102	6 →	4D	CD	74	F0-74	74	Make
103	3 Pg DN	51	D1	7A	F0-7A	7A	Make
104	Del	53	D3	71	F0-71	71	Make
105	-	4A	CA	7B	F0-7B	84	Make
106	+	4E	CE	79	F0-79	7C	Make
108	Enter	E0-1C	E0-9C	E0-5A	E0-F0-5A	79	Typematic
110	Esc	01	81	76	F0-76	08	Make
112	F1	3B	BB	05	F0-05	07	Make
113	F2	3C	BC	06	F0-06	0F	Make
114	F3	3D	BD	04	F0-04	17	Make
115	F4	3E	BE	0C	F0-0C	1F	Make
116	F5	3F	BF	03	F0-03	27	Make
117	F6	40	C0	0B	F0-0B	2F	Make
118	F7	41	C1	83	F0-83	37	Make
119	F8	42	C2	0A	F0-0A	3F	Make
120	F9	43	C3	01	F0-01	47	Make
121	F10	44	C4	09	F0-09	4F	Make
122	F11	57	D7	78	F0-78	56	Make
123	F12	58	D8	07	F0-07	5E	Make
125	Scr Lk	46	C6	7E	F0-7E	5F	Make

Key Position and Symbol		Cursor Pad <NUM Lock Off/Shift Off> or <NUM Lock On/Shift On>				Table III (Terminal Mode)	
		Table I (XT and PS/2 30)		Table II (AT and PS/2 50, 60, 80)			
		Make	Break	Make	Break	Code	Type
75	Insert	E0-52	E0-D2	E0-70	E0-F0-70	67	Make
76	Delete	E0-53	E0-D3	E0-71	E0-F0-71	64	Typematic
79	←	E0-4B	E0-CB	E0-6B	E0-F0-6B	61	Typematic
80	Home	E0-47	E0-C7	E0-6C	E0-F0-6C	6E	Make
81	End	E0-4F	E0-CF	E0-69	E0-F0-69	65	Make
83	↑	E0-48	E0-C8	E0-75	E0-F0-75	63	Typematic
84	↓	E0-50	E0-D0	E0-72	E0-F0-72	60	Typematic
85	PG UP	E0-49	E0-C9	E0-7D	E0-F0-7D	6F	Make
86	PG DN	E0-51	E0-D1	E0-7A	E0-F0-7A	6D	Make
89	→	E0-4D	E0-CD	E0-74	E0-F0-74	6A	Typematic

***. Cursor Pad Key—<NUM Lock On/Shift Off>**

Table I: Make Code = = E0-2A-Make Code
Break Code = = Break Code-E0-AA

Table II: Make Code = = E0-12-Make Code
Break Code = = Break Code E0-F0-12

***. Cursor Pad Key—<NUM Lock Off/Shift On>**

Table I: Make Code = E0-AA-Make Code
Break Code = Break Code-E0-2A

Table II: Make Code = E0-F0-12-Make Code
Break Code = Break Code E0-12

Key Code of "Pause", "PRTSC" and "/" Keys

TABLE I. XT and PS/2 30

Key Position and Symbols		Make	Break
126	Pause	E1-1D-45-E1-9D-C5	No Break Code (Make Only)
	Ctrl-"Pause"	E0-46-E0-C6	No Break Code (Make Only)
124	Print Screen	E0-2A-E0-37	E0-B7-E0-AA
	Shift-"PRTSC"	E0-37	E0-B7
	Ctrl-"PRTSC"	E0-37	E0-B7
	Alt-"PRTSC"	54	D4
95	/	E0-35	E0-B5
	Shift-"/"	E0-AA-E0-35	E0-B5-E0-2A

TABLE II. AT and PS/2 50, 60, 80

Key Position and Symbols		Make	Break
126	Pause	E1-14-77-E1-F0-14-F0-77	No Break Code (Make Only)
	Ctrl-"Pause"	E0-7E-E0-F0-7E	No Break Code (Make Only)
124	Print Screen	E0-12-E0-7C	E0-F0-7C-E0-F0-12
	Shift-"PRTSC"	E0-7C	E0-F0-7C
	Ctrl-"PRTSC"	E0-7C	E0-F0-7C
	Alt-"PRTSC"	84	F0-84
95	/	E0-4A	E0-F0-4A
	Shift-"/"	E0-F0-12-E0-4A	E0-F0-4A-E0-12

TABLE III. Terminal Mode

Key Position and Symbols		Code	Type
126	Pause	62	Make
124	Print Screen	57	Make
95	/	77	Make

APPENDIX II. REFERENCES

1. IBM Technical Reference Manuals XT, AT and PS/2
2. Chicony, Chicony Keyboards General Specification, 1988
3. C' T Magazin fuer Computertechnik, No. 6, 1988, pages 148ff. No. 7, 1988, pages 178ff. Martin Gerdes, "Knoepfchen, Knoepfchen"

RS-232C Interface with COP800

National Semiconductor
Application Note 739
Michelle Giles

INTRODUCTION

This application note describes an implementation of the RS-232C interface with a COP888CG. The COP888CG 8-bit microcontroller features three 16-bit timer/counters, MICROWIRE/PLUS™ Serial I/O, multi-source vectored interrupt capability, two comparators, a full duplex UART, and two power saving modes (HALT and IDLE). The COP888CG feature set allows for efficient handling of RS-232C hardware handshaking and serial data transmission/reception.

SYSTEM OVERVIEW

In this application, a COP888CG is connected to a terminal using the standard RS-232C interface. The serial port of the terminal is attached to the COP888CG interface hardware using a standard ribbon cable with DB-25 connectors on either end. The terminal keyboard transmits ASCII characters via the cable to the COP888CG interface. All characters received by the COP888CG are echoed back to the terminal screen. If the COP888CG detects a parity or framing error, it transmits an error message back to the terminal screen.

HARDWARE DESCRIPTION

The COP888CG features used in this application include the user programmable UART, the 8-bit configurable L PORT, and vectored interrupts. In addition to the COP888CG, the RS-232C interface requires a DS14C88 driver and a DS14C89A receiver. The DS14C88 converts TTL/CMOS level signals to RS-232C defined levels and the DS14C89A does the opposite. *Figure 1* contains a diagram of the COP888CG interface hardware.

The COP888CG is configured as data communications equipment (DCE) and the terminal is assumed to be data terminal equipment (DTE). The following RS-232C signals are used to communicate between the COP888CG (DCE) and the terminal (DTE):

RS-232C Signal Name	Signal Origin
TxD (Transmit Data)	DTE
RxD (Receive Data)	DCE
CTS (Clear To Send)	DCE
RTS (Request To Send)	DTE
DSR (Data Set Ready)	DCE
DTR (Data Terminal Ready)	DTE
DCD (Data Carrier Detect)	DCE

Five general purpose I/O pins on the COP888CG L PORT are used for the control signals CTS, DSR, DCD, RTS and DTR. Two additional L PORT pins are used for TxD and RxD. These two general purpose pins are configured for their alternate functions, UART transmit (TDX) and UART receive (RDX). According to the RS-232C interface standard, DCE transmits data to DTE on RxD and receives data from DTE on TxD. Therefore, the UART transmit data pin (TDX) is used for the RS-232C receive data signal (RxD) and the UART receive data pin (RDX) is used for the RS-232C transmit data signal (TxD). In this example, all handshaking between DCE and DTE is performed in hardware.

The terminal is setup to interface with the COP888CG by selecting the 9600 baud, 7 bits/character, odd parity and one stop bit options. The local echo back of characters is disabled to allow the COP888CG to perform the echo back function. The terminal is also configured to use the hardware control signals (CTS, DSR, RTS, DTR) for handshaking.

SOFTWARE DESCRIPTION

The software for this application consists of an initialization routine, several interrupt routines, and a disable routine. These routines handle RS-232C handshaking, transmitting and receiving of characters, error checking, and echoing back of received characters. *Figures 2* thru *5* contain flowcharts of the routines. The complete code is given at the end of this application note.

The initialization routine configures the UART, initializes the transmit/receive data buffer, and enables the 8-bit L PORT handling of RS-232C control signals. In this particular example, the UART is configured to operate at 9600 BAUD in full duplex, asynchronous mode. The framing format is chosen to be: 7 bits/character, odd parity, and one stop bit. Different baud rates, modes of operation, and framing formats may be selected by setting the ENUCMD, ENUICMD, BAUDVAL and PSRVAL constants located at the beginning of the code to alternative values. (Refer to the COP888CG data sheet or COP888 Family User's Manual for details on configuring the UART.) Each RS-232C control signal is assigned to an L PORT pin. Pins L0, L2, L5 and L6 are configured as outputs for the DCD, TxD, CTS and DSR signals, respectively. Pins L3, L4 and L7 are configured as inputs for TxD, RTS and DTR, respectively. The transmit/receive data buffer is a circular buffer whose location and size is selected by setting the START and END constants located at the beginning of the program. The initialization routine sets up the buffer based on these constants.

The interrupt routines respond to transmit buffer empty, receive buffer full, and L PORT interrupts. A generic context switching routine is used for entering and exiting all interrupts. This routine saves the contents of the accumulator, the PSW register and the B pointer before vectoring to the appropriate interrupt routine. It also restores the contents of saved registers before a return from interrupt is executed.

The UART transmitter interrupt is called when the transmit buffer empty flag (TBMT) is set. This routine checks for active RTS and DTR control signals. If both signals are active and there is data to be transmitted, a byte of data is loaded into the UART transmit buffer. Otherwise, the UART transmitter is disabled.

The L PORT interrupts are used to indicate an active-low transition of RTS and/or DTR. When both signals are active (the remote receiver is ready to accept data), this routine enables the UART transmitter.

The UART receiver interrupt routine is called when the receive buffer full flag (RBFL) is set. This routine reads the

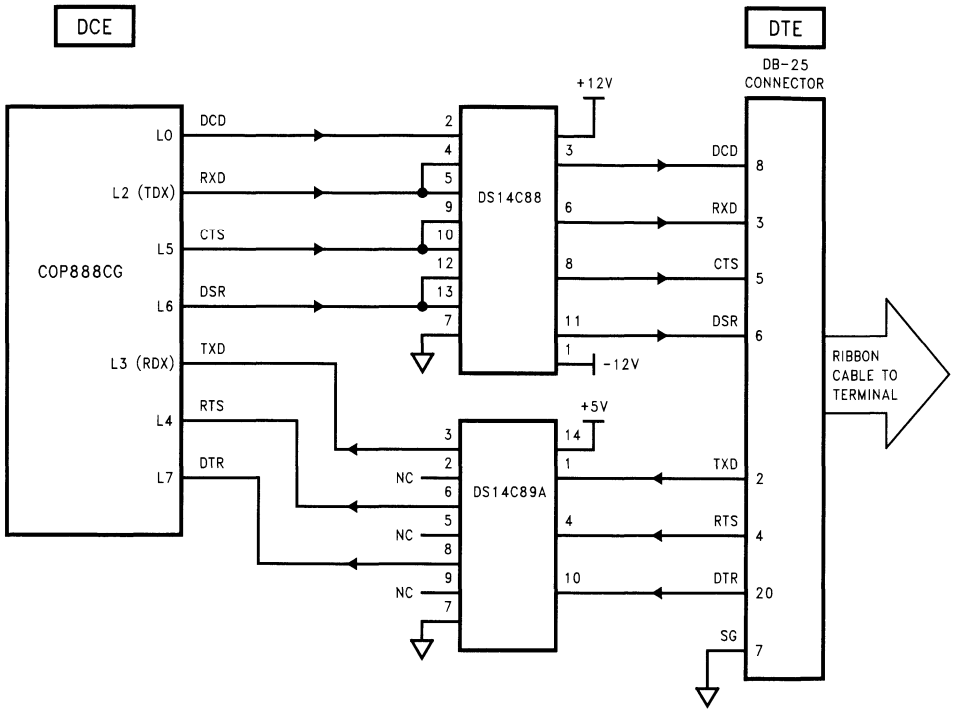
UART receive buffer and checks for errors. If no errors are detected, the incoming data is placed in the data buffer for echoing. If errors are detected, an error message is queued for transmission.

The receiver interrupt disables the remote transmitter by deactivating CTS whenever the transmit/receive data buffer is almost full. This action prevents the data buffer from overflowing. Note that CTS is turned off before the buffer is completely full to insure buffer space will exist for storing characters which are in the process of being sent when CTS is deactivated.

The disable routine clears the UART control registers, disables the L PORT interrupts, and de-activates the RS-232C control signals.

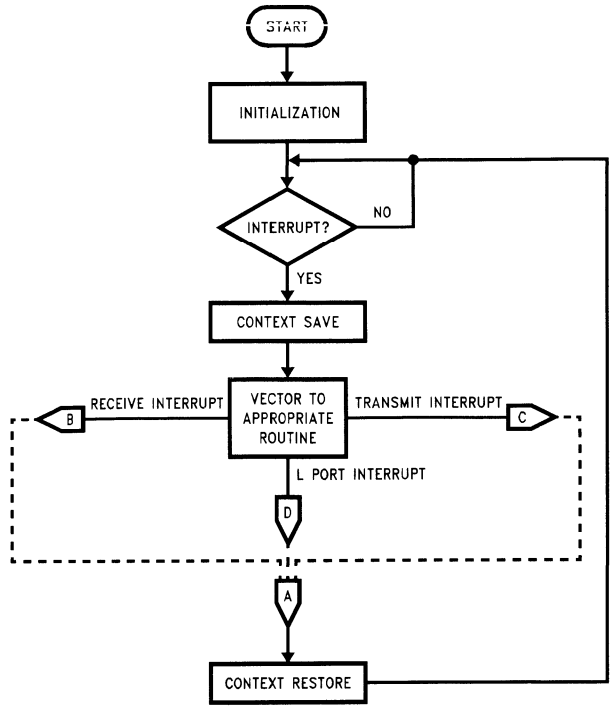
CONCLUSION

The user configurable UART, multiple external interrupt capabilities, and vectored interrupt scheme of the COP888CG microcontroller allow for an efficient implementation of the RS-232C interface standard. This application note shows how the COP888CG may be configured for connection to a terminal using these features. However, the code for this application can be easily adapted to other applications requiring different baud rates or framing formats, connection to a modem (DCE), separate transmit and receive buffers, incoming command decoding and/or handling of character strings. The versatility of the RS-232C standard and the COP888CG provides a means to develop practical solutions for many applications.



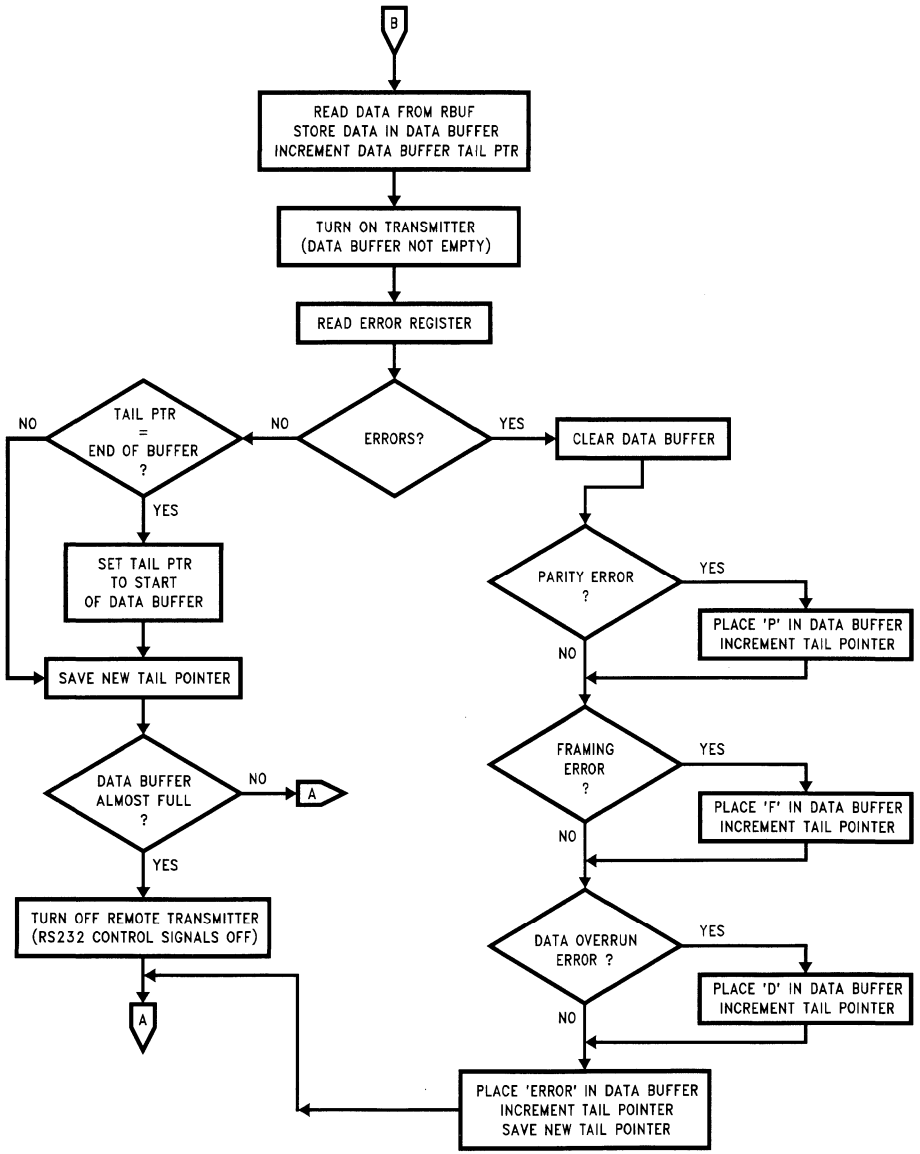
TL/DD/11110–1

FIGURE 1. Interface Diagram



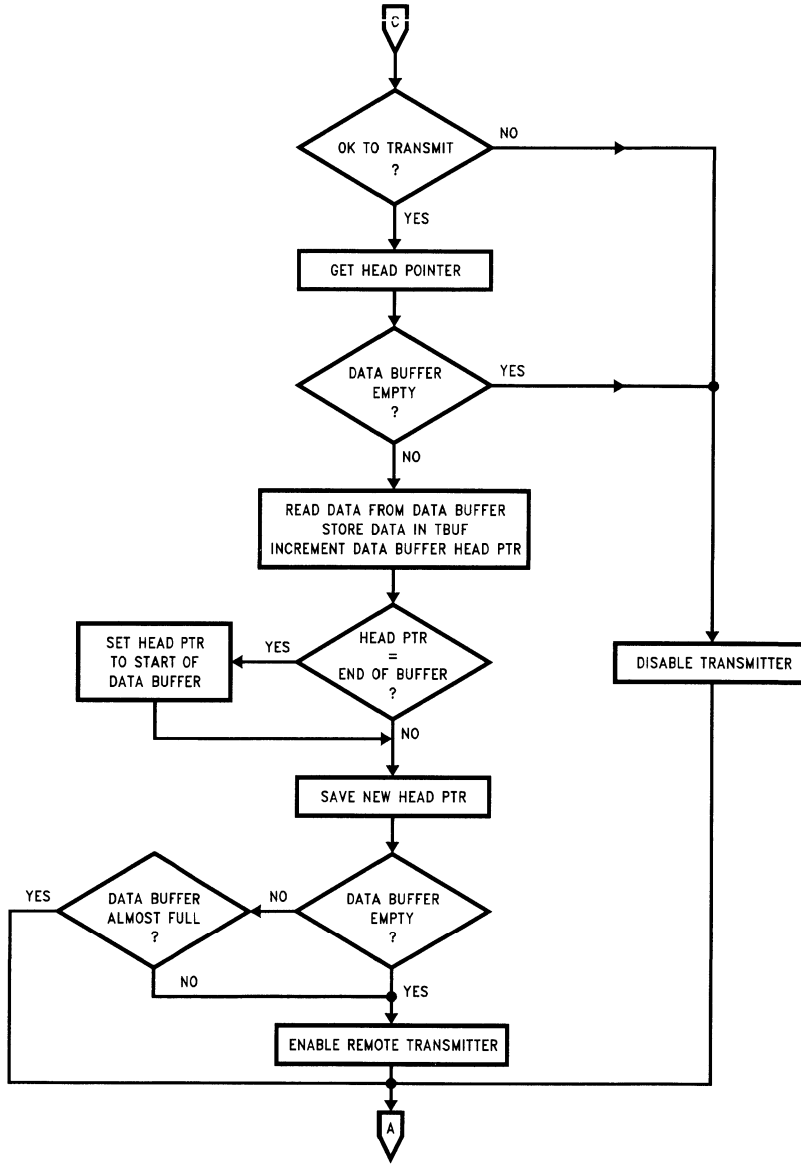
TL/DD/11110–2

FIGURE 2. Main Program Flow



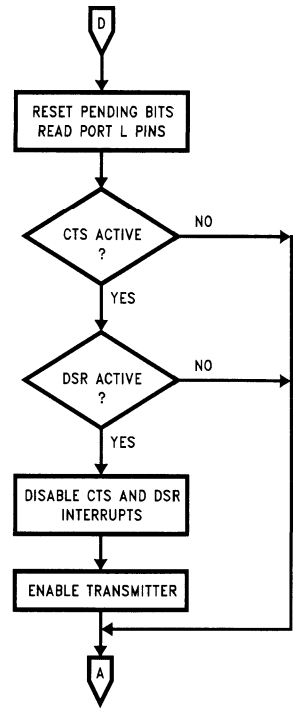
TL/DD/11110–3

FIGURE 3. Receiver Interrupt Routine



TL/DD/11110–4

FIGURE 4. Transmitter Interrupt Routine



TL/DD/11110–5

FIGURE 5. L Port Interrupt Routine

```
NATIONAL SEMICONDUCTOR CORPORATION
COP800 CROSS ASSEMBLER, REV:D1,12 OCT 88


                    ;The following set of routines uses the COP888CG UART and several I/O pins
                    ;to simulate an RS232 port interface. The code handles hardware control
                    ;signals, echo back of received characters, and error checking.  A single
                    ;routine called INIT initializes the UART and hardware control signals.
                    ;The transmitting and receiving of characters is handled in several
                    ;interrupt routines. The UART is disabled by calling the DISABLE routine.
                    ;The user must select values for several constants before compiling
                    ;this code.
                    ;
                    ;NOTES:
                    ;       * The COP tranmitter is enabled only when the transmit/receive
                    ;         buffer is not empty and the appropriate RS232 control signals
                    ;         from the remote receiver are present.
                    ;       * The COP receiver is always enabled.  the remote transmitter
                    ;       * The remote transmitter is disabled whenever the transmit/
                    ;         receive data buffer is full.

                    ;Definition of Constants
0089                        ENUCMD  =  089      ;Value to put in the ENU register
                                                ;Selects bits per char and parity option
                                                ;DEFAULT = 081  (7 bits/char and odd parity)

0020                        ENUICMD =  020      ;Value to put in the ENUI register
                                                ;Selects number of stop bits, uart clock option,
                                                ;sync/async option, xmit/rcv interrupt enable,
                                                ;and TDX pin enable
                                                ;DEFAULT = 023 ( 1 stop bit, internal BRG,
                                                ;async operation, no interrupt, and TDX enabled)

0004                        BAUDVAL = 04        ;Baud rate divisor equals N - 1
00C8                        PSRVAL  = 0C8       ;Baud rate prescalar
                                                ; BR = FC/(16 * N*P) where
                                                ;       FC = CKI frequency
                                                ;       N  = Baud Divisor
                                                ;       P  = Prescalar
                                                ;GIVEN:         CALCULATE:   BAUDVAL:  PSRVAL:
                                                ;CKI = 10MHz     N = 5
                                                ;BR  = 9600      P = 13        04       0C8
                                                ;
                                                ;CKI = 10MHz     N = 10
                                                ;BR  = 4800      P = 13        09       0C8
                                                ;
                                                ;See tables in users manual for translation
                                                ;of N and P to BAUDVAL and PSRVAL

0010                        START   =  010      ;Beginning address of the xmit/rcv buffer
001D                        END     =  01D      ;End address of the xmit/rcv buffer
001E                        HEAD    =  01E      ;RAM address where current head of buffer stored
001F                        TAIL    =  01F      ;RAM address where current tail of buffer stored
000E                        SIZE    =  0E       ;Size of transmit/receive data buffer
```

TL/DD/11110–6

```
NATIONAL SEMICONDUCTOR CORPORATION
COP800 CROSS ASSEMBLER, REV:D1, 12 OCT 88


  52      0000                  DCD     =  00       ;Bit position of DCD signal on L port pins
  53      0005                  CTS     =  05       ;Bit position of CTS signal on L port pins
  54      0007                  DTR     =  07       ;Bit position of DTR signal on L port pins
  55      0004                  RTS     =  04       ;Bit position of RTS signal on L port pins
  56      0006                  DSR     =  06       ;Bit position of DSR signal on L port pins
  57      0005                  ETDX    =  05       ;Bit position of TDX enable pin in ENUI
  58      0000                  TIE     =  00       ;Bit position of TX interrupt enable bit
  59      0001                  RIE     =  01       ;Bit position of RX interrupt enable bit
  60      0005                  PE      =  05       ;Bit position of parity error in ENUR
  61      0006                  FE      =  06       ;Bit position of framing error in ENUR
  62      0007                  DOE     =  07       ;Bit position of data overrun error in ENUR
  63
  64
  65
  66                            .INCLD  COP888.INC
  67
  68 0002 3008          MAIN:   JSR     INIT                ;INITIALIZE UART
  69 0004 FF                    JP      .                   ;DO OTHER TASKS
  70 0005 3044                  JSR     DISABLE             ;DISABLE UART
  71 0007 FF                    JP      .                   ;DO OTHER TASKS
  72
  73                    INIT:
  74 0008 9FEF                  LD      B,#PSW
  75 000A 68                    RBIT    GIE,[B]             ;DISABLE ALL INTERRUPTS
  76 000B BCBE00                LD      PSR,#00             ;UART OFF (POWERDOWN)
  77 000E BCD165                LD      PORTLC,#065         ;SET I/O
  78 0011 9FD0                  LD      B,#PORTLD           ;NOT READY TO RECEIVE
  79 0013 7E                    SBIT    DSR,[B]             ;   TURN OFF DATA SET READY
  80 0014 7D                    SBIT    CTS,[B]             ;   TURN OFF CLEAR TO SEND
  81 0015 68                    RBIT    DCD,[B]             ;   TURN ON DATA CARRIER DETECT
  82 0016 BC1E10                LD      HEAD,#START         ;INIT HEAD POINTER
  83 0019 BC1F10                LD      TAIL,#START         ;INIT TAIL POINTER
  84 001C 9FE8                  LD      B,#ICNTRL           ;CONFIGURE PORTL INTERRUPTS
  85 001E 6E                    RBIT    LPEN,[B]            ;  DISABLE PORTL INTERRUPTS
  86 001F BCC890                LD      WKEDG,#090          ;  SELECT FALLING EDGE FOR RTS AND DTR
  87 0022 BCC990                LD      WKEN,#090           ;  ENABLE RTS AND DTR INTERRUPT
  88 0025 BCCA00                LD      WKPND,#00           ;  CLEAR PORTL INTERRUPT PENDING FLAGS
  89 0028 7E                    SBIT    LPEN,[B]            ;  ENABLE PORT L INTERRUPTS
  90 0029 BCBA89                LD      ENU,#ENUCMD         ;SELECT BITS/CHAR AND PARITY OPTION
  91 002C BCBB00                LD      ENUR,#00            ;CLEAR ERROR BITS
  92 002F BCBC20                LD      ENUI,#ENUICMD       ;SELECT CLOCK, INTERRUPTS, STOPBITS
  93 0032 BCBD04                LD      BAUD,#BAUDVAL       ;SETUP BRG
  94 0035 9FBC                  LD      B,#ENUI
  95 0037 78                    SBIT    TIE,[B]             ;ENABLE TRANSMITTER INTERRUPT
  96 0038 79                    SBIT    RIE,[B]             ;ENABLE RECEIVER INTERRUPT
  97 0039 BCBEC8                LD      PSR,#PSRVAL         ;UART ON
  98 003C 9FD0                  LD      B,#PORTLD           ;READY TO RECEIVE
  99 003E 6E                    RBIT    DSR,[B]             ;  TURN ON DATA SET READY
 100 003F 6D                    RBIT    CTS,[B]             ;  TURN ON CLEAR TO SEND
 101 0040 9FEF                  LD      B,#PSW
 102 0042 78                    SBIT    GIE,[B]             ;ENABLE ALL INTERRUPTS
```

TL/DD/11110–7

```
NATIONAL SEMICONDUCTOR CORPORATION
COP800 CROSS ASSEMBLER, REV:D1, 12 OCT 88


 103 0043 8E                    RET
 104
 105                    DISABLE:
 106 0044 BDEF68                RBIT    GIE, PSW            ;DISABLE INTERRUPTS
 107 0047 BCD061                LD      PORTLD, #061        ;TURN OFF HANDSHAKING SIGNALS
 108 004A BCBE00                LD      PSR, #00            ;UART POWERDOWN
 109 004D BCBA00                LD      ENU, #00            ;CLEAR UART CONTROL REGISTERS
 110 0050 BCBC00                LD      ENUI, #00
 111 0053 BCBB00                LD      ENUR, #00
 112 0056 9FC9                  LD      B, #WKEN            ;DISABLE RTS AND DTR INTERRUPTS
 113 0058 6C                    RBIT    RTS, [B]
 114 0059 6F                    RBIT    DTR, [B]
 115 005A BDEF78                SBIT    GIE, PSW            ;ENABLE INTERRUPTS
 116 005D 8E                    RET
 117
 118
 119                    ;INTERRUPT ROUTINES
 120
 121      00FF                  . = 0FF                     ;INTERRUPT START ADDRESS
 122 00FF 67                    PUSH    A                   ;CONTEXT SAVE
 123 0100 9DFE                  LD      A, B
 124 0102 67                    PUSH    A
 125 0103 9DEF                  LD      A, PSW
 126 0105 67                    PUSH    A
 127 0106 B4                    VIS
 128 0107 8C            REST:   POP     A                   ;CONTEXT RESTORE
 129 0108 9CEF                  X       A, PSW
 130 010A 8C                    POP     A
 131 010B 9CFE                  X       A, B
 132 010D 8C                    POP     A
 133 010E 8F                    RETI
 134
 135
 136                    ;PORT L INTERRUPTS
 137                    ;  The port L interrupts are used to indicate a return to active
 138                    ; state of the DTR and RTS signals from the remote receiver.
 139                    ; If both DTR and RTS are active, the remote receiver is ready
 140                    ; to accept data and the COP transmitter is enabled.
 141
 142                    LINT:                               ;PORT L INTERRUPT
 143 010F BCCA00                LD      WKPND, #00          ;RESET PENDING BITS
 144 0112 9DD2                  LD      A, PORTLP           ;READ PORT L PINS
 145 0114 6010                  IFBIT   #RTS, A             ;IF RTS (ACTIVE LOW) NOT PRESENT
 146 0116 06                    JP      NOTRDY              ;THEN REMOTE NOT READY TO RECEIVE
 147 0117 6080                  IFBIT   #DTR, A             ;IF DTR (ACTIVE LOW) NOT PRESENT
 148 0119 03                    JP      NOTRDY              ;THEN REMOTE NOT READY TO RECEIVE
 149 011A 9FBC          READY:  LD      B, #ENUI
 150 011C 78                    SBIT    TIE, [B]            ;RE-ENABLE TRANSMITTER INTERRUPT
 151 011D E9            NOTRDY: JP      REST                ;EXIT INTERRUPT
 152
 153
```

TL/DD/11110–8

```
NATIONAL SEMICONDUCTOR CORPORATION
COP800 CROSS ASSEMBLER, REV:D1, 12 OCT 88


 154                    ;UART RECEIVE INTERRUPT
 155                    ;  The UART receive interrupt does the following:
 156                    ;       1. Reads the received data
 157                    ;       2. Checks for receiver errors
 158                    ;       3. If no errors detected, places the received data in
 159                    ;          the transmit/receive buffer and enables the transmitter.
 160                    ;       4. If errors detected, the transmit/receive buffer is cleared
 161                    ;          of ALL data and an error message is placed in the data buffer.
 162                    RCVINT:                                 ;RECEIVER INTERRUPT
 163 011E 9D1F                 LD      A,TAIL
 164 0120 9CFE                 X       A,B                     ;GET TAIL POINTER
 165 0122 9DB9                 LD      A,RBUF                  ;READ RECEIVED DATA
 166 0124 A2                   X       A,[B+]                  ;STORE RECEIVED DATA
 167 0125 9DBB                 LD      A,ENUR                  ;READ ERROR REGISTER
 168 0127 BDBC78               SBIT    TIE,ENUI                ;ENABLE TRANSMITTER INTERRUPT
 169 012A 60E0                 ANDSZ   A,#0E0                  ;CHECK FOR PE,DOE,FE
 170 012C 1A                   JP      ERROR                   ;THROW DATA AWAY IN BUFFER
 171 012D 9DFE                 LD      A,B                     ;LOAD ACC WITH NEW TAIL PTR
 172 012F 921E                 IFEQ    A,#END+1                ;IF END OF DATA BUFFER
 173 0131 9810                 LD      A,#START                ; SET TAIL PTR TO START OF BUFFER
 174 0133 9C1F                 X       A,TAIL                  ;SAVE TAIL PTR
 175 0135 9D1E                 LD      A,HEAD                  ;IS DATA BUFFER FULL?
 176 0137 A1                   SC
 177 0138 BD1F81               SUBC    A,TAIL                  ;A = HEAD - TAIL
 178 013B 89                   IFNC                            ;IF BORROWED (TAIL > HEAD)
 179 013C 940E                 ADD     A,#SIZE                 ;THEN ADD BUFFER SIZE TO RESULT
 180 013E 9303                 IFGT    A,#03                   ;IF DATA BUFFER NOT FULL
 181 0140 2107                 JMP     REST                    ;  THEN EXIT INTERRUPT
 182 0142 BDD07D        RXOFF:  SBIT    CTS,PORTLD              ;  ELSE TURN OFF REMOTE TRANSMITTER
 183 0145 2107                 JMP     REST                    ;EXIT INTERRUPT
 184                    ERROR:
 185 0147 BC1E10               LD      HEAD,#START             ;CLEAR BUFFER
 186 014A 9F10                 LD      B,#START                ;POINT TO START OF BUFFER
 187 014C 6020                 IFBIT   PE,A
 188 014E 9A50                 LD      [B+],#'P'               ;P = PARITY
 189 0150 6040                 IFBIT   FE,A
 190 0152 9A46                 LD      [B+],#'F'               ;F = FRAMING
 191 0154 6080                 IFBIT   DOE,A
 192 0156 9A44                 LD      [B+],#'D'               ;D = DATA OVERRUN
 193 0158 9A20                 LD      [B+],#020               ;BLANK SPACE
 194 015A 9A45                 LD      [B+],#'E'
 195 015C 9A52                 LD      [B+],#'R'
 196 015E 9A52                 LD      [B+],#'R'
 197 0160 9A4F                 LD      [B+],#'O'
 198 0162 9A52                 LD      [B+],#'R'
 199 0164 9A0A                 LD      [B+],#0A                ;LINE FEED
 200 0166 9A0D                 LD      [B+],#0D                ;CARRIAGE RETURN
 201 0168 9DFE          OUTERR: LD      A,B                     ;SAVE NEW TAIL PTR
 202 016A 9C1F                 X       A,TAIL
 203 016C 2107                 JMP     REST
 204
```

TL/DD/11110–9

```
NATIONAL SEMICONDUCTOR CORPORATION
COP800 CROSS ASSEMBLER, REV:D1,12 OCT 88


 205
 206                    ;UART TRANSMIT INTERRUPT
 207                    ;  The UART transmit interrupt does the following:
 208                    ;       1. Checks for RTS and DTR signals (OK to transmit?)
 209                    ;       3. If OK to transmit and buffer not empty, transmits data.
 210                    ;       4. If not OK to transmit or buffer empty, disables transmitter.
 211
 212                    XMITINT:                        ;TRANSMITTER INTERRUPT
 213 016E 9DD2                  LD      A,PORTLP
 214 0170 6090                  ANDSZ   A,#090          ;IS IT OK TO TRANSMITT?
 215 0172 219C                  JMP     IDLE            ;NO:  GO TURN OFF TRANSMITTER
 216 0174 9D1E                  LD      A,HEAD          ;YES: GET PTR TO DATA
 217 0176 BD1F82                IFEQ    A,TAIL          ;IF DATA BUFFER EMPTY
 218 0179 219C                  JMP     IDLE            ;THEN TURN OFF TRANSMITTER
 219 017B 9CFE                  X       A,B             ;ELSE
 220 017D AA                    LD      A,[B+]          ;GET TRANSMIT DATA
 221 017E 9CB8                  X       A,TBUF          ;SEND TRANSMIT DATA
 222 0180 9DFE                  LD      A,B             ;LOAD ACC WITH NEW HEAD PTR
 223 0182 921E                  IFEQ    A,#END+1        ;IF END OF DATA BUFFER
 224 0184 9810                  LD      A,#START        ;  SET HEAD PTR TO START OF BUFFER
 225 0186 9C1E                  X       A,HEAD          ;SAVE HEAD PTR
 226 0188 9D1E                  LD      A,HEAD          ;IS DATA BUFFER FULL?
 227 018A BD1F82                IFEQ    A,TAIL          ;IF BUFFER EMPTY
 228 018D 09                    JP      NFULL           ;   THEN NOT FULL
 229 018E A1                    SC                      ;   ELSE CHECK HOW FULL
 230 018F BD1F81                SUBC    A,TAIL          ;A = HEAD - TAIL
 231 0192 89                    IFNC                    ;IF BORROWED (TAIL ) HEAD)
 232 0193 940E                  ADD     A,#SIZE         ;THEN ADD BUFFER SIZE TO RESULT
 233 0195 9303                  IFGT    A,#03           ;IF DATA BUFFER NOT FULL
 234 0197 BDD06D        NFULL:  RBIT    CTS,PORTLD      ;   THEN TURN ON REMOTE TRANSMITTER
 235 019A 2107                  JMP     REST            ;   ELSE EXIT INTERRUPT
 236 019C 9FBC          IDLE:   LD      B,#ENUI
 237 019E 68                    RBIT    TIE,[B]         ;DISABLE TRANSMITTER INTERRUPT
 238 019F 2107                  JMP     REST            ;EXIT INTERRUPT
 239
 240                    ;Software Trap
 241                    ;
 242 01A1 B5            SFTINT: RPND
 243 01A2 2000                  JMP     00              ;RESTART
 244
 245                    ;VECTOR INTERRUPT TABLE
 246
 247      01E2                  .=01E2
 248 01E2 010F                  .ADDRW  LINT            ;L PORT INTERRUPT
 249      01EC                  .=01EC
 250 01EC 016E                  .ADDRW  XMITINT         ;TRANSMITTER INTERRUPT
 251 01EE 011E                  .ADDRW  RCVINT          ;RECEIVER INTERRUPT
 252      01FE                  .=01FE
 253 01FE 01A1                  .ADDRW  SFTINT          ;SOFTWARE INTERRUPT/TRAP
 254                            .END
```

TL/DD/11110-10

```
NATIONAL SEMICONDUCTOR CORPORATION
COP800 CROSS ASSEMBLER, REV:D1, 12 OCT 88

SYMBOL TABLE

B       00FE        BAUD    00BD        BAUDVA 0004        CNTRL   00EE  *
CTS     0005        DCD     0000        DISABL 0044        DOE     0007
DSR     0006        DTR     0007        END    001D        ENU     00BA
ENUCMD  0089        ENUI    00BC        ENUICM 0020        ENUR    00BB
ERROR   0147        ETDX    0005  *     FE     0006        GIE     0000
HEAD    001E        ICNTRL  00E8        IDLE   019C        INIT    0008
LINT    010F        LPEN    0006        MAIN   0002  *     NFULL   0197
NOTRDY  011D        OUTERR  0168  *     PE     0005        PORTLC  00D1
PORTLD  00D0        PORTLP  00D2        PSR    00BE        PSRVAL  00C8
PSW     00EF        RBUF    00B9        RCVINT 011E        READY   011A  *
REST    0107        RIE     0001        RTS    0004        RXOFF   0142  *
SFTINT  01A1        SIZE    000E        SP     00FD        START   0010
TAIL    001F        TBUF    00B8        TIE    0000        WKEDG   00C8
WKEN    00C9        WKPND   00CA        X      00FC        XMITIN  016E
```

TL/DD/11110–11

```
NATIONAL SEMICONDUCTOR CORPORATION
COP800 CROSS ASSEMBLER, REV:D1, 12 OCT 88

MACRO TABLE


   NO WARNING LINES

   NO ERROR LINES

  267  ROM BYTES USED

SOURCE CHECKSUM = 6884
OBJECT CHECKSUM = 096B

INPUT   FILE C:UART.MAC
LISTING FILE C:UART.PRN
OBJECT  FILE C:UART.LM
```

TL/DD/11110–12

NM95C12 Flexibility in Industrial Control Applications

National Semiconductor
Application Note 755
Sean Long

INTRODUCTION

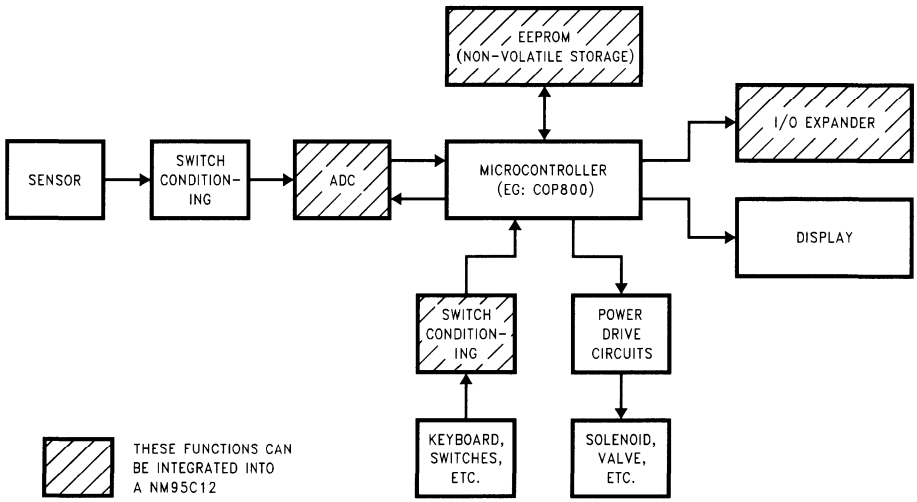
This application note describes a general purpose industrial controller and details how a NM95C12 can be used to integrate a number of different functions typically found in such a design.

General purpose application examples of the use of the NM95C12 are presented rather than a specific design. Each design idea and software can be incorporated into the designer's required application.

The basic building blocks of an industrial controller (for example, heating, process control, etc.) are a microcontroller, an Analogue to Digital Converter (ADC), an EEPROM, a display (LCD, LED, etc.), I/O interfaces, and power driver/control circuits. The NM95C12 forms the basis of this design performing the non-volatile parameter storage, a low cost ADC, an I/O expander, and providing a simple control interface.

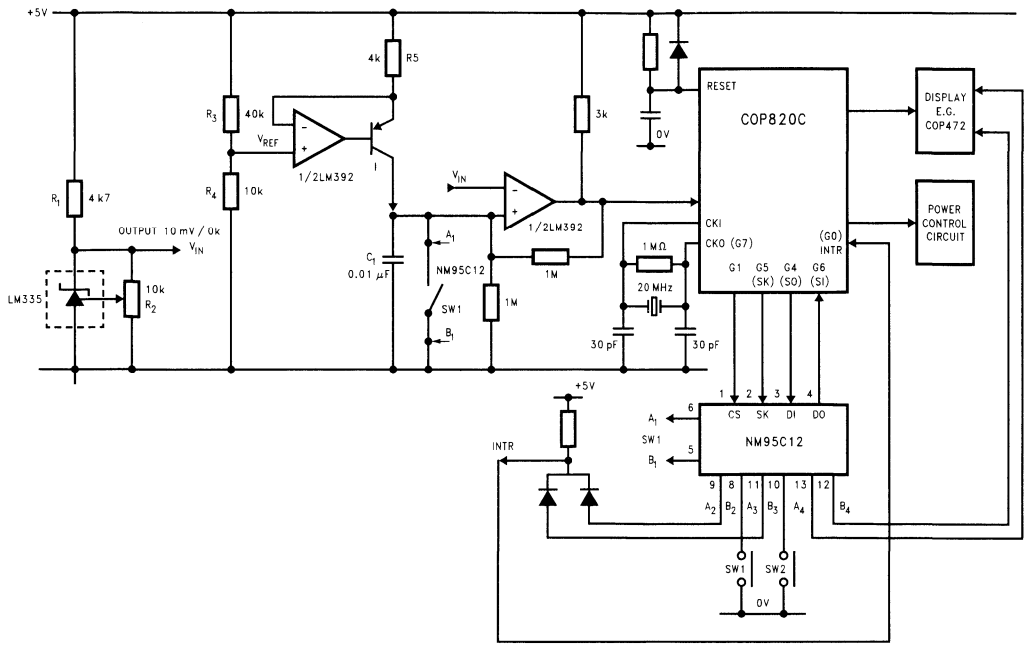
Figure 1 shows the basic block diagram with the shaded parts representing the functions performed by the NM95C12.

This application note will describe the theory of operation behind the design and give detailed software examples to show how to interface a popular microcontroller to the NM95C12.



TL/D/11160–1

FIGURE 1. Typical Industrial Controller Block Diagram



TL/D/11160–2

COP820/NM95C12 Industrial Controller

THE NM95C12 1024-BIT CMOS EEPROM WITH DIP SWITCHES

The NM95C12 features 1K-bit EEPROM memory with 8 switch logic terminals. These switch logic terminals are individually programmable outputs which may be used as DIP switch positions or as SPST switch positions.

The NM5C12 uses the MICROWIRE™ serial I/O interface which is fully compatible with COPS™ microcontrollers via 4 simple control lines:

SK — Serial Clock
CS — Chip Select
DI — Data In
DO — Data Out

The EPROM array (addresses 0 to 60) is addressed via five instructions:

READ — Read Data from register
WEN — Write enable
WRITE — Writes data to register
WRALL — Writes to all registers
WDS — Disables all programming instructions

This area of memory is used for the normal EEPROM applications such as the storage of user changeable, non-volatile parameters such as time on/off, temperature on/off limits, etc.

CONTROLLING THE SWITCH LOGIC

Address locations 61 to 63 control the switch operation.

Address	Name	Description
61	ISS	Provides the initial switch configuration automatically on power-up. Controlled via a WRITE operation.
62	SCR	The SCR is not an E2 location and hence is volatile. The SCR is loaded automatically from address 61 on power-up. The SCR controls the switch terminals A1–A4 and B1–B4.
63	SRR	The SRR allows the current logic levels of the switch terminals to be read back via the MICROWIRE bus.

THEORY OF OPERATION

The relationship for charge of a capacitor is as follows:

Charge (Q) = Voltage (V) × Capacitance (C)
= Current (I) × Time (T)

Therefore the voltage across the capacitor, V_{CAP}

$V_{CAP} = (I \times T)/C$

Assuming that the current I is a constant source, and the capacitance value C does not vary gives:

V_{CAP} is proportional to T.

Mode of Operation

— initially switch S1 is closed to short out V_{CAP} to measure input voltage V_{IN}

To Measure V_{IN}:

— microcontroller opens S1 and starts internal timer at T1
— V_{CAP} is proportional to time T
— when $V_{CAP} > V_{IN}$ then comparator output V_{COMP} goes high
— microcontroller stops internal timer at T2
— V_{IN} is proportional to time T = T2 − T1
— microcontroller closes S1 ready for next measurement

CURRENT SOURCE/VOLTAGE COMPARATOR FOR ADC

This is based on a LM932 which has an Operational Amplifier and a Voltage Comparator in the same 8-pin package. This device operates from a single +5V supply.

Refer to the National Semiconductor General Purpose Linear Databook for further details of the LM392.

INPUT SENSOR

For this example assume temperature needs to be controlled.

LM335: This is a precision, low-cost, easily calibrated two terminal temperature sensor that behaves like a zener diode with a voltage of +10 mV/degree Kelvin. The initial accuracy is ±1° and can be externally trimmed with a potentiometer connected to the ADJ pin.

Refer to the National Semiconductor Linear Databook 2 for further details of the LM335 Temperature Sensors.

NM95C12 SWITCH LOGIC APPLICATIONS/ CONFIGURATIONS

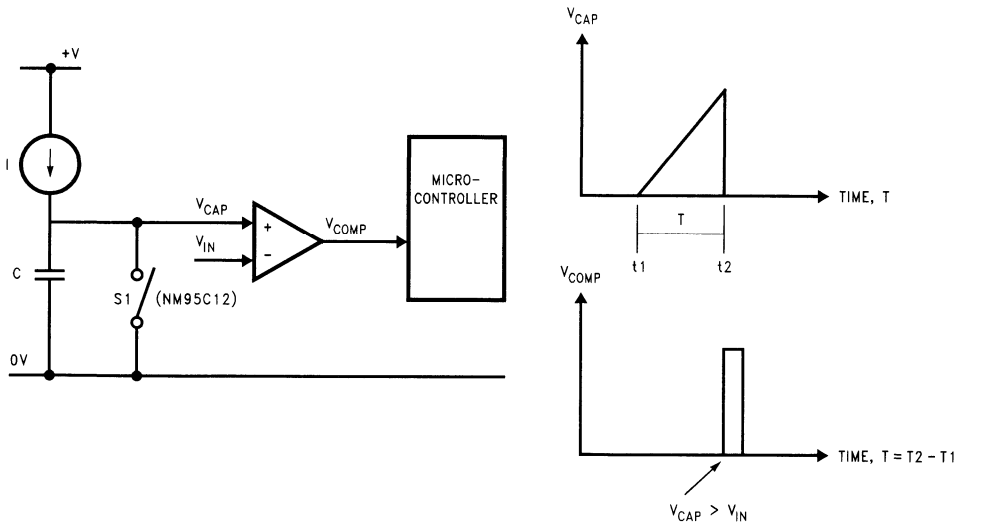
A_1, B_1—Control the Charge/Discharge of Capacitor for ADC

Switch Configuration:

Analog Switch Open: Mode 12, ZYXW = 110?
(? = don't care)

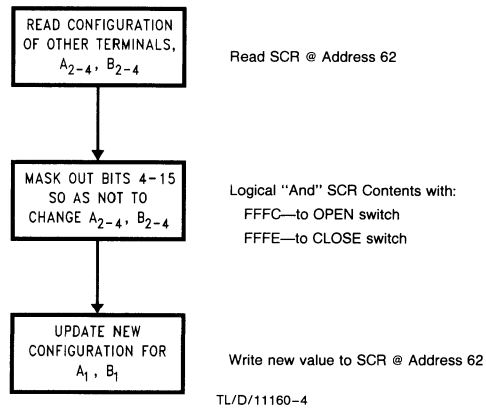
Analog Switch Closed: Mode 13, ZYXW = 111?

To change the state of the switch terminals A_1, B_1, follow the flowchart in *Figure 3*.



TL/D/11160–3

FIGURE 2. Single Slope Analogue to Digital Converter



TL/D/11160–4

FIGURE 3. Controlling Switch Terminals A1, B1

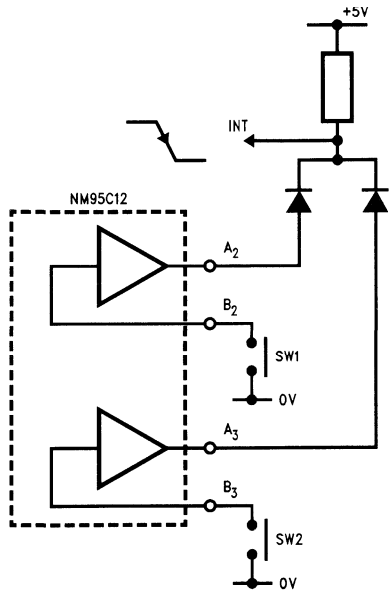
A_2, B_2 and A_3, B_3 — Switch Debouncing

The switch logic configuration is shown in *Figure 4*. When either of the mechanical switches SW1 or SW2 are pressed, this causes the interrupt line (INT) to be pulled low signalling to the microcontroller that a switch has been pressed. As part of the interrupt service routine the microcontroller can generate a delay to allow time for mechanical switch debouncing, before reading the NM95C12 SRR to determine which mechanical switch was pressed.

The advantage of this design is that it saves input pins on the microcontroller and means that the software does not have to perform periodic polling of the inputs to determine the mechanical switch status since the circuit is interrupt driven.

Switch Configuration: both A_2,B_2 and A_3,B_3 will be configured in mode 5; ZYXW = 0101.

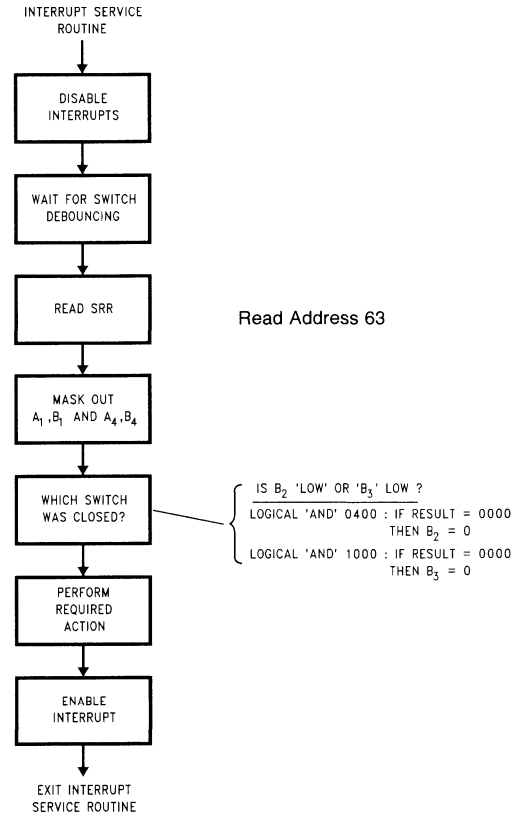
To change the state of the switch terminals A_2,B_2 and A_3,B_3 follow the flowchart in *Figure 5*.



TL/D/11160–5

A_2,B_2, A_3,B_3 Configured in mode 5, ZYXW = 0101

FIGURE 4. Switching Conditioning



TL/D/11160–6

FIGURE 5. Controlling Switch Terminals A_2,B_2, A_3,B_3

A_4,B_4 Programmable I/O

These two terminals use mode 1 to 4 according to the logic level required on the output. In this example A_4 is used for the Display Chip Select signal and B_4 is used for the Display On/Off control signal.

In order to update and display the contents of the Display then both terminals A_4 and B_4 need to be set to a logic "1" therefore A_4,B_4 are configured in mode 3 with ZYXW = 0011.

To change the state of the switch terminals A_2,B_2 and A_3,B_3 follow the flowchart in *Figure 6*.

SOFTWARE TO INTERFACING THE NM95C12 TO THE COP820 MICROCONTROLLER

This section includes a number of subroutines to interface to a NM95C12 as described in the design example above. There are subroutines to implement each of the basic instructions together with routines for configuring and controlling the switch logic. These subroutines can be used as the basis for a design and be tailored to meet the individual application requirements.

CONCLUSION

The NM95C12 is an extremely versatile and inexpensive device which allows simple interfacing to all popular microcontrollers and microprocessors via a 4-wire serial bus. The complete operation of the NM95C12 can be controlled by a few simple instructions.

The design outlined offers an inexpensive solution for industrial control applications with the key benefits of:

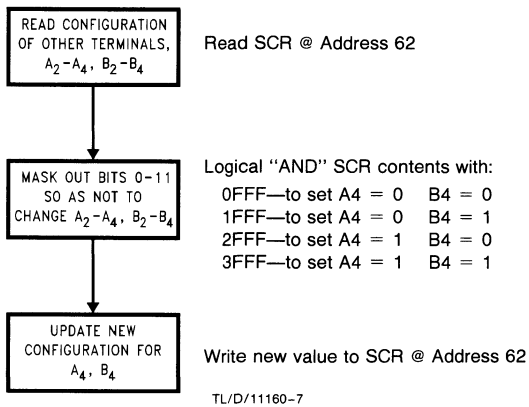
— simple interfacing between microcontroller, EEPROM, "ADC"

— low part count

— fully software controlled and changeable

This has highlighted the flexibility of the NM95C12 and how the switch terminals can be configured for a wide range of applications including: mechanical switch replacement, programmable Address Decoder, programmable I/O expander and a programmable interrupt controller. The NM95C12 offers greater reliability than mechanical switches with the benefits of software control and lower cost.

Plus you still get the 1K-bit EEPROM memory as well!; together with the 8 switch terminals it forms a truly remarkable device.



TL/D/11160–7

FIGURE 6. Controlling Switch Terminals A_4,B_4

```
;
; THIS FILE PREDECLARES ITEMS FREQUENTLY USED BY THE
; COP820 PROGRAMMER.
; REGISTER NAMES, CONTROL BITS, ETC. ARE NAMED THE SAME WAY
; AS IN THE DATA-SHEETS.
;
; ***********************************************
; * PORT~ , CONFIGURATION- AND CONTROL REGISTERS  *
; ***********************************************
      PORTLD   =   0D0          ; PORT L DATA
      PORTLC   =   0D1          ; PORT L CONFIGURATION
      PORTLP   =   0D2          ; PORT L PIN
;
      PORTGD   =   0D4          ; PORT G DATA
      PORTGC   =   0D5          ; PORT G CONFIGURATION
      PORTGP   =   0D6          ; PORT G PIN
      PORTD    =   0DC          ; PORT D
      PORTI    =   0D7          ; PORT I
;
      SIOR     =   0E9          ; SID SHIFT REGISTER
      TMRLO    =   0EA          ; TIMER LOW BYTE
      TMRHI    =   0EB          ; TIMER HIGH BYTE
      TAULO    =   0EC          ; TIMER REGISTER LOW BYTE
      TAUHI    =   0ED          ; TIMER REGISTER HIGH BYTE
;
      CNTROL   =   0EE          ; CONTROL REGISTER
      PSW      =   0EF          ; PSW REGISTER
;             ********************
;             * CONSTANT DECLARE *
;             ********************
;              ---- CNTRL - REGISTER BITS ----

;
      TSEL     =   7
      CSEL     =   6
      TEDG     =   5
      TRUN     =   4
      MSEL     =   3
      IEDG     =   2
      S1       =   1
      SO       =   0
;
;             ---- PSW- REGISTER BITS ----
      HCARRY   =   7
      CARRY    =   6
      TPND     =   5
      ENTI     =   4
      IPND     =   3
      BUSY     =   2
      ENI      =   1
      GIE      =   0

;             I/O - SIGNALS ----
;
      TMRINP   =   3
      INTR     =   0
      TIO      =   3
      SO       =   4
      SK       =   5
      SI       =   6
      CKO      =   7
;
      .CHIP    820
      LD       SP,#02F          ; DEFAULT INITIALIZATION OF SP
```

```
;INCLD COP820.INC
;
; This program provides in the form of subroutines, the ability to enable,
; disable, read and write to the NM95C12 EEPROM with DIP switches.
;
;       ************************************************
;       * PROGRAM VARIABLE MEMORY LOCATION DEFINITIONS  *
;       ************************************************
;
SNDBUF  = 0 ;CONTAINS THE COMMAND BYTE TO BE WRITTEN TO NM95C12
RDATL   = 1 ;LOWER BYTE OF THE NM95C12 REGISTER DATA READ
RDATH   = 2 ;UPPER BYTE OF THE NM95C12 REGISTER DATA READ
WDATL   = 3 ;LOWER BYTE OF THE DATA TO BE WRITTEN TO NM95C12 REGISTER
WDATH   = 4 ;UPPER BYTE OF THE DATA TO BE WRITTEN TO NM95C12 REGISTER
ADDRESS = 5 ;THE LOWER 6-BITS OF THIS LOCATION CONTAIN THE ADDRESS
            ;OF THE NM95C12 REGISTER TO BE READ/WRITTEN
FLAGS   = 6 ;USED FOR SETTING UP FLAGS
            ;
            ; FLAG VALUE        ACTION
            ; ----------        ------
            ;    00             WRITE ENABLE,DISABLE,WRITE ALL
            ;    01             READ CONTENTS OF NM95C12 REGISTER
            ;    03             WRITE TO NM95C12 MEMORY REGISTER
            ;    07             WRITE NM95C12 SCR REGISTER
            ;    OTHERS         ILLEGAL COMBINATION
;
; THE INTERFACE BETWEEN THE COP820C/840C AND THE NM95C12 (1024-BIT EEPROM)
; CONSISTS OF FOUR LINES. THE G1 (CHIP SELECT LINE), G4 (SERIAL OUT SO),
; G5 (SERIAL CLOCK SK) AND G6 (SERIAL IN SI).
;
; ANOTHER PINS USED BY THIS DESIGN IS G0 (INTERRUPT INTR)
;
;
;       ******************
;       * INITIALIZATION *
;       ******************
;
        LD PORTGC,032       ;Setup G1,G4,G5 as outputs
        LD PORTGD,00        ;Initialize G data reg to zero
        LD CNTROL,08        ;Enable MSEL, select MW rate of 2tc
        LD B,fPSW           ;Load B with address of PSW
        LD X,fSIOR          ;Load X with address of Serial I/O Register
;
;
;
;       ********************
;       * WEN INSTRUCTION  *
;       ********************
;
;
; THIS ROUTINE ENABLES PROGRAMMING OF THE NM95C12. PROGRAMMING MUST
; BE PRECEDED ONCE BY A PROGRAMMING ENABLE (WEN).
;
WEN:
        LD  SNDBUF,f030    ; LOAD OP CODE AND 'ADDRESS'
        LD  FLAGS,f0
        JSR INIT
        RET
;
;
;       ********************
;       * WDS INSTRUCTION  *
;       ********************
;
```

```
;
; THIS ROUTINE DISABLES PROGRAMMING OF THE NM95C12.
;
WDS:
        LD  SNDBUF,f0       ; LOAD OP CODE AND 'ADDRESS'
        LD  FLAGS,f0
        JSR INIT
        RET
;
;
;
;       ********************
;       * READ INSTRUCTION *
;       ********************
;
;
; THIS ROUTINE READS THE CONTENTS OF THE NM95C12 REGISTER.
; THE NM95C12 ADDRESS IS SPECIFIED IN THE LOWER 6-BITS OF
; LOCATION "ADDRESS". THE UPPER 2-BITS SHOULD BE SET TO ZERO.
; THE 16-BIT CONTENTS OF THE NM95C12 REGISTER ARE STORED IN
; RDATL AND RDATH.
;
READ:   LD  A,ADRESS     ; LOAD ADDRESS A5-A0 INTO ACCUMULATOR
        OR  A,f080       ; SET OP CODE BITS TO '10'
        X   A,SNDBUF     ; TRANSFER COMMAND BYTE TO SERIAL I/O VARIABLE
        LD  FLAGS,f1
        JSR INIT
        RET
;
;
;
;       *********************
;       * WRITE INSTRUCTION *
;       *********************
;
;
; THIS ROUTINE WRITES A 16-BIT VALUE STORED IN WDATL AND WDATH
; TO THE NM95C12 REGISTER WHOSE ADDRESS IS CONTAINED IN THE
; LOWER 6-BITS OF THE LOCATION "ADDRESS". THE UPPER 2-BITS OF
; ADDRESS LOCATION SHOULD BE SET TO ZERO.
;
WRITE:  LD  A,ADRESS      ; LOAD ADDRESS A5-A0 INTO ACCUMULATOR
        OR  A,f040        ; SET OP CODE BITS TO '01'
        X   A,SNDBUF      ; TRANSFER COMMAND BYTE TO SERIAL I/O VARIABLE
        LD  FLAGS,f3
        JSR INIT
        RET
;
;
;
;       *********************
;       * WRALL INSTRUCTION *
;       *********************
;
;
;
; THIS ROUTINE WRITES A 16-BIT VALUE STORED IN WDATL AND WDATH
; TO ALL THE NM95C12 REGISTERS
;
WRALL:  LD  SNDBUF,f040   ; LOAD OP CODE AND ADDRESS'
        LD  FLAGS,f3
        JSR INIT
        RET
;
;
```

```
;       **********************
;       * WRSCR 'INSTRUCTION' *
;       **********************
;
;
;
; THIS ROUTINE WRITES A 16-BIT VALUE STORED IN WDATL AND WDATH
; TO THE NM95C12 SCR (SWITCH CONTROL REGISTER) WHOSE ADDRESS IS 62 DECIMAL
; WHICH EQUALS f03E HEXADECIMAL. OP CODE = '01'
; A WRITE TO THE SCR DOES NOT REQUIRE A PROGRAMMING CYCLE
;
WRSCR:  LD  SNDBUF,f07E  ; LOAD OP CODE AND ADDRESS
        LD  FLAGS,f7
        JSR INIT
        RET
;
;
;
;       **********************************
;       * EXECUTE INSTRUCTION SUBROUTINES *
;       **********************************
;
;
;
; THIS ROUTINE SENDS OUT THE START BIT AND THE COMMAND BYTE.
; IT ALSO DECIPHERS THE CONTENTS OF THE FLAG LOCATION AND TAKES
; A DECISION REGARDING WRITE, WRITE SCR, READ OR RETURN TO THE
; CALLING_ROUTINE.
;
INIT:   SBIT 1,PORTGD     ;SET CHIP SELECT HIGH
        LD SIOR,f001      ;LOAD SIOR WITH START BIT
        SBIT BUSY,[B]     ;SEND OUT THE START BIT
PUNT1:  IFBIT BUSY,[B]
        JP PUNT1
        LD A,SNDBUF
        X  A,[X]          ;LOAD SIOR WITH COMMAND BYTE
        SBIT BUSY,[B]     ;SEND OUT COMMAND BYTE
PUNT2:  IFBIT BUSY,[B]
        JP PUNT2
        IFBIT 0,FLAGS     ;ANY FURTHER PROCESSING?
        JP NOTDON         ;YES
        RBIT 1,PORTGD     ;NO, RESET CS AND RETURN
        RET
;
NOTDON:
        IFBIT 1,FLAGS     ;READ OR WRITE?
        JP WR95C12        ;JUMP TO WRITE ROUTINE
        LD SOIR,f000      ;NO, READ NM95C12
        SBIT BUSY,PSW     ;DUMMY CLOCK TO READ ZERO
        RBIT BUSY,[B]
        SBIT BUSY,[B]
PUNT3:  IFBIT BUSY,[B]
        JP PUNT3
        X   A,[X]
        SBIT BUSY,[B]
        X   A,RDATH
```

```
PUNT4:  IFBIT BUSY,[B]
        JP PUNT4
        LD A,[X]
        X  A,RDATL
        RBIT 1,PORTGD
        RET
;
WR95C12:
        LD  A,WDATH
        X   A,[X]
        SBIT BUSY,[B]
PUNT5:  IFBIT BUSY,[B]
        JP  PUNT5
        LD  A,WDATL
        X   A,[X]
        SBIT BUSY,[B]
PUNT6:  IFBIT BUSY,[B]
        JP  PUNT6         ; FINISHED CLOCKING OUT DATA
        RBIT 1,PORTGD     ; RESET CHIP SELECT
        IFBIT 2, FLAGS    ; WRITE/WRALL OR WRSCR?
        RET               ; WRSCR , NO PROGRAMMING TIME SO RETURN
        SBIT 1,PORTGD     ; SET CHIP SELECT TO ALLOW TO POLL DO FOR BUSY/READY
POLL:   IFBIT SI,PORTGP   ; IS NM95C12 DO = SI LOW?
        JP ENDTWP         ; DO HAS GONE HIGH, SO END PROGRAMMING CYCLE
        JP POLL           ; DO IS STILL LOW, SO KEEP POLLING
;
ENDTWP;
        BIT 1, FLAGS      ; RESET CHIP SELECT
        RET
;
        .END
```

Using National's MICROWIRE™ EEPROM

National Semiconductor
Application Note 758
Paul Bryant

National Semiconductor manufactures a wide range of low density serial EEPROMs that use the MICROWIRE interface as a means of communication. Although all of these devices use the MICROWIRE interface, there are slight variations in interfacing due to differences in memory sizes, features, and technology used to implement the device. Additionally, the MICROWIRE interface does not specifically define any protocol, it only defines a basic set of signal lines to interconnect two or more devices. Due to these reasons, additional information is necessary to fully understand how to best interface to National's family of MICROWIRE EEPROM.

The goal of this application guide is to cover a diversity of information in regard to basic timing, interfacing options, and functionality of different EEPROMs. I will use an outline approach, so the appropriate heading can be located easily. Each section attempts to be stand alone so the information can be easily extracted. The outline appears below:

OUTLINE

1.0 Description of EEPROM Families
- 1.1 CMOS EEPROM
- 1.2 NM93C Family
- 1.3 NM93CS Family
- 1.4 NM93CxxA Family

2.0 HARDWARE CONNECTIONS
- 2.1. INTERFACE PIN DESCRIPTIONS
  - 2.1.1 Chip Select (CS)
  - 2.1.2 Serial Clock (SK)
  - 2.1.3 Data-In (DI)
  - 2.1.4 Data-Out (DO)
  - 2.1.5 Program Enable (PE)
  - 2.1.6 Protect Register Enable (PRE)
  - 2.1.7 Organization (ORG)
  - 2.1.8 Status (RDY/BUSY)
- 2.2. FOUR WIRE BUS
- 2.3. THREE WIRE BUS

3.0 TIMING CONSIDERATIONS
- 3.1 BUS TIMING
- 3.2 INSTRUCTION SEQUENCE DESCRIPTIONS
  - 3.2.1 Read Cycle
  - 3.2.2 Sequential Read
  - 3.2.3 Erase and Erase All
  - 3.2.4 Write and Write All
  - 3.2.5 Program Enable and Program Disable
  - 3.2.6 Protect Register Read
  - 3.2.7 Protect Register Enable
  - 3.2.8 Protect Register Disable
  - 3.2.9 Protect Register Clear
  - 3.2.10 Protect Register Write
- 3.3. INTERFACING SOLUTIONS

4.0 CONCLUSION

1.0 Description of EEPROM Families

1.1 CMOS EEPROM

National builds a range of MICROWIRE CMOS EEPROMs in memory sizes ranging from 256-bit to 16,384-bit. The NM93C family is the base family and the NM93CS is a similar family with additional features, there are also other devices with slight variations on the interface. All these devices are available with certain "standard" options such as operating temperature ranges and operating voltage ranges, packaging options and test options. These options being fairly standard variations for semiconductor devices, will not be addressed beyond this. The purpose of this article is to address basic functionality and interfacing, including various tricks to simplify or modify the interface.

1.1 NM93C Family

The NM93C family of EEPROM is available in 256-, 1024-, 2048-, 4096-bit, and 16,384-bit sizes. All of these are internally organized in 16-bit words, therefore all data transactions deal with 16 bits. This family of EEPROMs has 7 instructions that deal with read, write, and a basic level of data protection. The instructions are listed in Table I. It is important to note that there is a basic difference in length of the instruction between the NM93C06 or NM93C46 and the NM93C56 or NM93C66. This is due to the larger devices needing additional address bits.

The NM93C family of EEPROM, like all of National's serial EEPROMs have a basic level of write protection that can be turned on or off by the use of the ERASE/WRITE DISABLE (EWDS) and ERASE/WRITE ENABLE (EWEN) instructions. Although there are two erase instructions included in the NM93C family, these are included only for compatibility with older EEPROMs that require erase before write. These EEPROMs don't require erase before write and it is recommended that in application the erase not be used as this adversely affects endurance.

1.3 NM93CS Family

The NM93CS EEPROMs are available in 256-, 1024-, 2048-, and 4096-bit sizes. Making them different, they have two additional functions, sequential read and user configurable write protection, and don't have either of the erase functions, ERASE and ERASE-ALL as they are not needed. Like all of the CMOS EEPROMs, these have self timed programming cycles and operate from a single external supply of either 4.5V to 5.5V or 2.7V to 5.5V. In these devices it is necessary to eliminate the erase cycles from the code as they may adversely affect the performance of the device.

As these have additional functions, the instruction set includes a total of 10 instructions, 3 that operate on the memory array, 2 that deal with the basic write protection and 5 that deal with the user configurable write protection. Refer

to the NM93CS instruction set table (Table II) for definitions of these instructions. As with the NM93C family, there is a basic difference in instruction length depending on memory size.

To further increase data security in these EEPROMs there are also two additional input signals defined, Program Enable (PE) and Protect Register Enable (PRE). These signals are on pins that are unused on the NM93C family providing upward compatibility to the NM93CS devices.

TABLE I. NM93C Family Instruction Set Table

Instruction	SB	Op Code	Address	Data	Comments
READ	1	10	A7/A5–A0		Reads data stored in memory.
EWEN	1	00	11XXXX		Write enable must precede all programming modes.
ERASE	1	11	A5–A0		Erase register A5A4A3A2A1A0.
WRITE	1	01	A5–A0	D15–D0	Writes register.
ERAL	1	00	10XXXX		Erase all registers.
WRAL	1	00	01XXXX	D15–D0	Writes all registers.
EWDS	1	00	00XXXX		Disables all programming instructions.

TABLE II. NM93CS Family Instruction Set Table

Instruction	SB	Op Code	Address	Data	PRE	PE	Comments
READ	1	10	A5–A0		0	X	Reads data stored in memory, starting at specified address.
WEN	1	00	11XXXX		0	1	Write enable must precede all programming modes.
WRITE	1	01	A5–A0	D15–D0	0	1	Writes register if address is unprotected.
WRALL	1	00	01XXXX	D15–D0	0	1	Writes all registers. Valid only when Protect Register is cleared.
WDS	1	00	00XXXX		0	X	Disables all programming instructions.
PRREAD	1	10	XXXXXX		1	X	Reads address stored in Protect Register.
PREN	1	00	11XXXX		1	1	Must immediately precede PRCLEAR, PRWRITE, and PRDS instructions.
PRCLEAR	1	11	111111		1	1	Clears the Protect Register so that no registers are protected from WRITE.
PRWRITE	1	01	A5–A0		1	1	Program address into Protect Register. Thereafter, memory addresses $\geq$ the address in Protect Register are protected from WRITE.
PRDS	1	00	000000		1	1	One time only instruction after which the address in the Protect Register cannot be altered.

1.4 NM93CxxA Family

There are two variations on the standard implementation of the Microwire bus. Both variations can be viewed as enhancements. The first enhancement is a Organization (ORG) input that allows the user to select the internal configuration of the memory as either 8 bits wide or 16 bits wide. When the input is high or unconnected, the device is configured as 16 bits wide, when the ORG input is at a low level, the memory is configured as 8 bits wide, but twice as deep. The feature is present on the NM93C46A, NM93C56A, NM93C66A, and NM93C86A.

The second variation is the STATUS output. This is the Busy/Ready polling to indicate programming status. All other devices have this feature on the Data-Out (DO) output, the NM59C11 alone has status available as a separate output and not on the Data-Out output. This can simplify interfacing to a bidirectional data bus.

2.0 Hardware Connection

2.1 INTERFACE PIN DESCRIPTIONS

In this section, each possible input or output will be described followed by the most popular variations of bus connections. Not all devices have all of the described I/Os. The I/Os are available according to Table III, I/O Functionality.

2.1.1 CHIP SELECT (CS)

Chip Select is used to differentiate between various devices on the same Microwire bus. In the case of EEPROM it cannot be tied high even if it is the only device on the bus as it performs several additional functions. As it applies to any of the Microwire EEPROMs, the rising edge resets the internal circuitry of the device, a function necessary prior to initiating any new cycle. As shown in the functional block diagram *(Figure 1)* chip select also gates the data input and clock input, thus disabling these functions.

During the course of clocking in the start bit, op-code address and data-in or data-out, chip select must be held high continuously, otherwise the internal circuits will be reset and the cycle will have to be started again with a new start bit.

During programming cycles chip select initiates the internal programming cycle. The falling edge of chip select will start the internal programming cycle when a programming opcode has been entered (Erase, Write, Erase All, Write All) and then, in conjunction with Data-Out (DO), will indicate if programming is complete (except the NMOS NMC9306). If programming is complete, Data-Out will drive high, if incomplete it will drive low. In the case of the NMC9306, the user must provide the programming time and in this case chip select must be held low for a minimum of 10 ms, then brought high and clocked to end the programming cycle.

Several additional notes in regard to chip select:

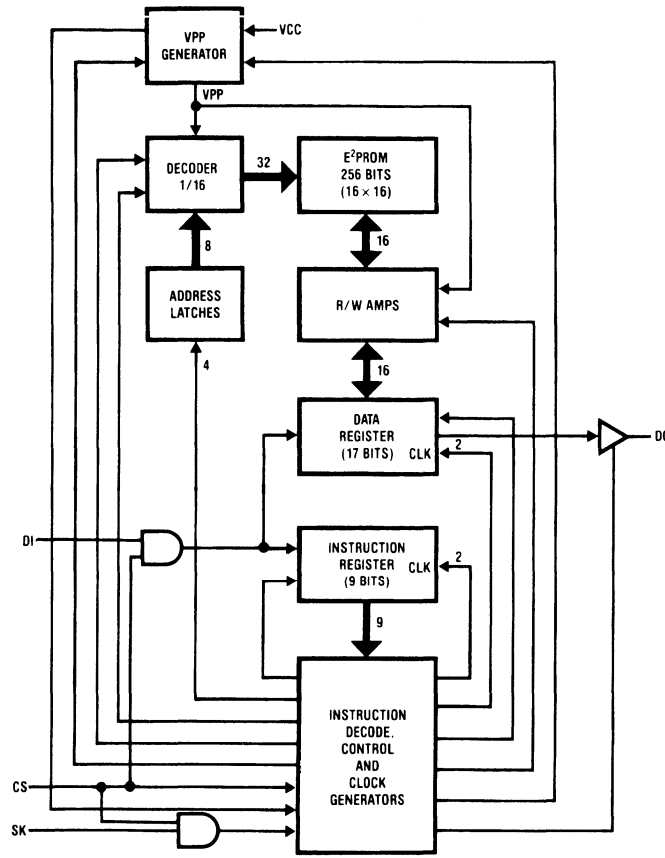
If a programming cycle is partially clocked in and then chip select dropped, the EEPROM may enter into a programming mode. This is determined by how many bits have been clocked in when chip select is dropped. If the start bit, opcode, and all of the address has been clocked in, a programming cycle will be initiated with no or partial data. If less than a complete address has been clocked in, the programming cycle will not be initiated. Refer to *Figure 2*, reference line 1.

In the case of the NM59C11, a programming cycle will not be entered unless a full data field has been clocked in. A full data field may be either 8 or 16 bits depending on the logic level present at the ORG input. A programming cycle will be entered at reference line 2 in *Figure 2* for the NM59C11.

Chip select hold time at the end of a cycle is referenced to the last rising edge of clock (SK). The hold time from the rising edge is the same as the minimum SK high time for the particular device. This is stated in the datasheets as 0 ns hold time from the falling edge of SK which assumes that SK high time is always minimum. In this case SK can be left in the high state or taken low at a later time. Internally chip select gates SK, therefore SK is not critical.

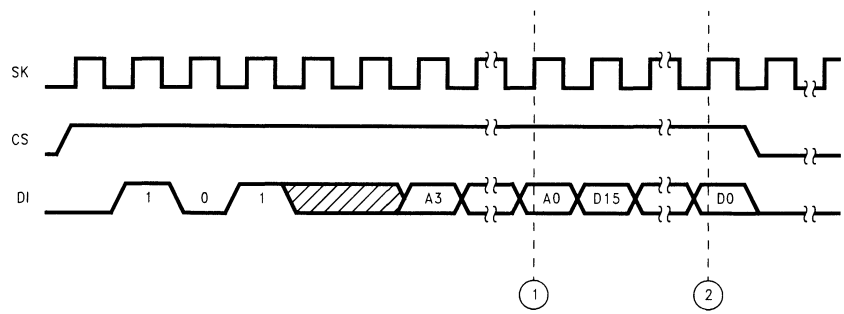
TABLE III. I/O Functionality by Device

	CS	SK	DI	DO	PE	PRE	ORG	STAT
NMC9306	X	X	X	X				
NMC9346	X	X	X	X				
NM93C Family	X	X	X	X				
NM93CS Family	X	X	X	X	X	X		
NM93C46A	X	X	X	X			X	
NM59C11	X	X	X	X			X	X



TL/D/11169–1

FIGURE 1. Block Diagram



TL/D/11169–2

FIGURE 2. Programming Cycle Point of No Return

2.1.2 SERIAL CLOCK (SK)

The clock input is used to clock all data, address, op-code, and start bits into or out of the EEPROMs. SK clocks both input and output on the rising edge only, the falling edge has no effect on the devices. The only function it is not necessary for is the Busy/Ready Polling which is an asynchronous function.

Since SK is gated by ship select, it is a "Don't Care" any time chip select is low. It is also don't care prior to a start bit being clocked in and during Busy/Ready Polling. During these conditions Data-In (DI) must be held at a low level, otherwise a start bit will be interpreted.

If it is desirable to insert additional clock cycles during a instruction sequence for the purpose of byte aligning the data, there are several places in the data stream they may be inserted as described below:

— On any instruction, zeros can be clocked into the DI input before the start bit. Any number of clock cycles may be added if Data-In (DI) is held at zero. The first 1 clocked in will be interpreted as the start bit. This requires special precautions if a bidirectional data bus is used (Data-In tied to Data-Out) as the Busy/Ready Polling will interfere with the Data-In if it is not cleared out at the end of each programming cycle. See Section 2.3, THREE WIRE BUS, for more information.

— During a Read instruction, it is allowable to continue to clock the device after the 16 bits of data has been clocked out. In the case of the NM93CS family this will cause the memory to increment to the next register and present its contents on the Data-Out pin. In the case of all other devices, whatever was present on the Data-In pin will become present on the Data-Out pin (Fall thru). Refer to *Figure 1,* Block Diagram.

— During a Write or Write-All, additional clock cycles may be added after address A0 and before the valid data. The EEPROM will write into the memory the most recent 16 bits, or in the case of the NM93C46A, the most recent 8 bits or 16 bits depending on the status of the ORG input. Adding additional clocks after the valid data will cause the data to be misaligned. In the case of the NM59C11, the device counts the data bits clocked in and automatically enters the programming mode when it receives a full data field, therefore bits cannot be inserted between A0 and valid data.

— During the EWEN, Erase, Erase All, EWDS, WEN, WDS cycles, it is not necessary to clock in a data field, although it is mandatory to clock in a complete address field, even if the addresses are "Don't Care". Additional clocks can be added after the address field.

2.1.3 DATA-IN (DI)

The Data-In input receives the Start-Bit, Address, and input data in a serial stream, each bit clocked in on the rising edge of SK. DI is gated by the chip select to provide a high degree of noise immunity. As shown in the block diagram, Data-In is routed to both the instruction shift register and the data shift register. When the start bit is clocked into the last bit of the instruction register, the clock is switched to the data register to receive input data and clock data out simultaneous. The Data-Out remains in high impedance unless a read cycle or Busy/Ready status is being done. The safest state is to keep the Data-In pin in a low level as a start bit is a high level.

2.1.4 DATA-OUT (DO)

The Data-Out (DO) output sends read data onto the microwire bus and is clocked out on the rising edge of SK. It also carries the programming status after a programming cycle which is an asynchronous function that does not require the clock. At all other times the Data-Out is in the high impedance state. During a Read cycle, the Data-Out output begins to drive actively after the last address bit (A0) is clocked in. During the Busy/Ready polling it begins to drive active after chip select is raised to a high level.

During the Busy/Ready Polling, the Data-Out output drives low while the device is still in the internal programming cycle. After the EEPROM has completed the internal programming cycle, the Data-Out pin will drive high when chip select is high. Subsequently, if chip select is brought high again, Data-Out will again drive high indicating it has completed the programming cycle. To clear the Busy/Ready Polling it is necessary to raise chip select and clock in a start bit. Once the start bit is clocked in, Data-Out will return to the high impedance state. It is not necessary to continue with a cycle after this start bit has been clocked in, although it is permissible to start a new cycle with this start bit. This clearing of the Busy/Ready status may be necessary if a bidirectional data bus is used (Data-In tied to Data-Out) as the Data-Out output will interfere with the new data being presented on the Data-In input.

2.1.5 PROGRAM ENABLE (PE)

The program enable (PE) input will enable all programming cycles when it is held at a high level during the duration of a programming cycle. Conversely, it will disable all programming, including programming of the protect register, while it is held low. This input has no affect on any other cycle, so it may be permanently tied high or low, or may be used in an active mode. This input is available on the NM93CS family only.

2.1.6 PROTECT REGISTER ENABLE (PRE)

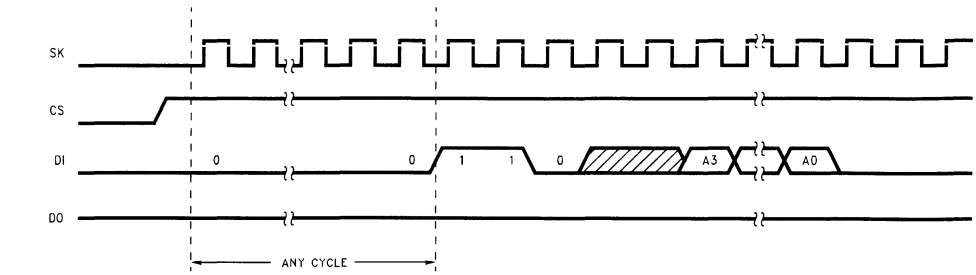
The protect register enable (PRE) input is used to switch between memory operations and protect register operations since the same op-codes are used for both. With the PRE input high, the op-codes define operations in the protect register, with the PRE input low, the op-codes define operations in the memory. This pin may be tied high or low, or used in the active mode. This input is available on the NM93CS family only.

2.1.7 ORGANIZATION (ORG)

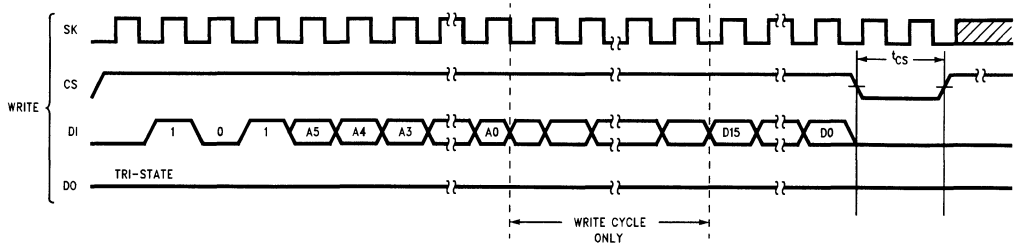
The Organization input (ORG) is used to control the internal organization of the memory. The two selectable organizations are 16-bit words and 8-bit words. Simply by holding the ORG pin at a high level, 16-bit words are selected, by holding the input at a low level 8-bit words are selected. When in the 8-bit mode, one additional address bit is required in the instruction sequence since the depth of the memory is doubled. This input is available only on certain device types, refer to the individual datasheets.

2.1.8 STATUS (RDY/BUSY)

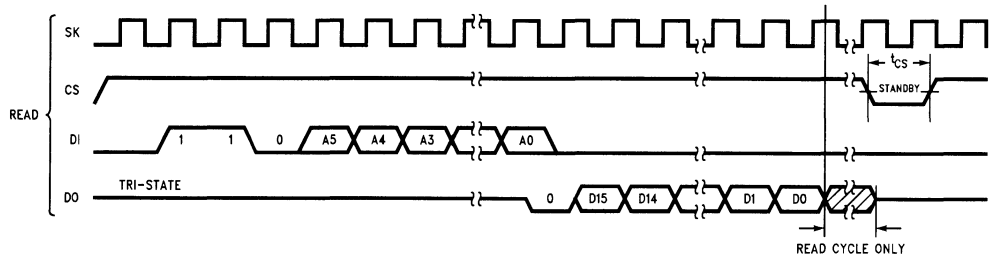
The status output indicates the programming cycle status after a programming cycle. When the device is in the programming mode and therefore cannot accept any other cycles, this pin will be low. After completion of the cycle the STATUS pin will be driven high. When this function is present, the Busy/Ready Polling is not available on the Data-Out



TL/D/11169–3



TL/D/11169–4



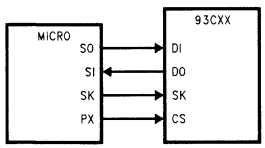
TL/D/11169–5

FIGURE 3. Possible Locations for Additional SK Cycles

output. In some systems, particularly those using a bi-directional data bus, this can simplify interfacing by eliminating the possible contention between the Ready indication and the incoming data from the host device. This output is available only on certain device types, refer to the individual datasheets.

2.2 FOUR WIRE BUS

The 4 wire bus is the simplest interconnection between the EEPROM and the host device. In most cases the only signals necessary to provide are clock, chip select, Data-In and Data-Out as shown in *Figure 5.* The PRE, PE, ORG, and STATUS pins are not shown as they are variations on this and the 3 wire bus connection. Multiple devices can be connected to the microwire bus, the only limitations being loading and available chip select means. In some systems it is necessary to have a bi-directional data line as described below in 3 wire bus.



TL/D/11169–7

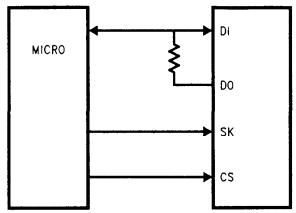
FIGURE 5. Four Wire Connection

2.3 THREE WIRE BUS

The 3 wire bus operates in the same mode as the 4 wire bus with the exception that the Data-In and Data-Out pins on the EEPROM are tied together. When using this connection, there are two precautions that need to be observed.

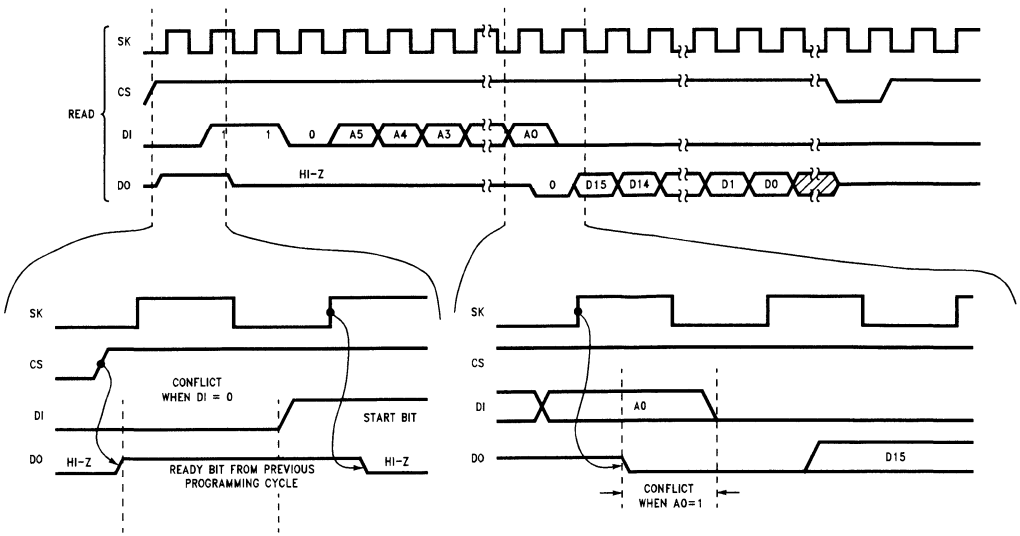
— When Data-In is tied to Data-Out, there is a possible conflict between address A0 in the instruction sequence and the dummy bit. This only occurs during a READ cycle. This is not harmful to the device and the internal circuitry of the EEPROM guarantees that the device will function properly under this condition. To decrease the noise created by the condition, a resistor may be placed in the locations indicated in *Figure 6.* The timing diagram in *Figure 7* shows the bus conflict.

— The second possible area of conflict occurs when the Busy/Ready status is on the Data-Out output. Since the device will continue to indicate a Ready status indefinitely after a programming cycle (until a start-bit is clocked in), this can conflict with the beginning of the next cycle if leading zeros are clocked in (See *Figure 7*). The solution is to either use a separate cycle to clear the Ready bit or to eliminate any leading zeros from the instruction sequence. If the Busy/Ready Polling is not used in the application, the easiest solution is to use the NM59C11 that does not have the polling on Data-Out but has it on a separate output.



TL/D/11169–8

FIGURE 6. Three Wire Connection Showing Optional Resistor



TL/D/11169–9

FIGURE 7. Three Wire Connection Bus Conflict Areas

7

3.0 Timing Considerations

The following information describing the Microwire bus timing must be used in conjunction with the datasheet as it is an expansion and clarification of the datasheet. First, the basic timings with respect to the clock (SK) will be described, followed by instruction sequence timing, and finally, specific information in each instruction sequence.

3.1. BUS TIMING

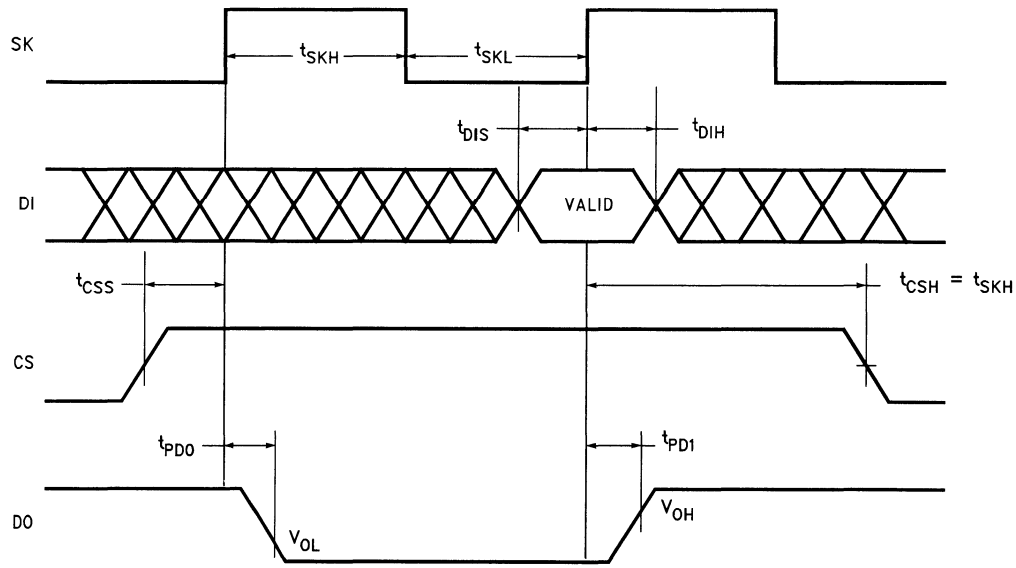
The synchronous data timing shown in *Figure 8* is similar to that shown in the various datasheets. There is one significant modification to the timing specification though, the chip select (CS) hold time is referenced to the rising edge of the clock rather than the falling edge. With this modification, the hold time specification must be changed to be the same as the minimum clock (SK) high time. Other significant points are:

— The only active edge of the clock is the rising edge.

— The only time the clock is necessary is when clocking data into or out of the EEPROM. It is not necessary during Busy/Ready Polling.

— The clock may be left in either the high state or low state between cycles. It is safer to leave the clock in the low state.

— When chip select (CS) is high, clock (SK) is a critical signal. With the exceptions noted in Section 2.1.2 tilted SERIAL CLOCK (SK), no additional clock cycles or noise that crosses the V_{IH} or V_{IL} thresholds can be tolerated.



TL/D/11169–11

FIGURE 8. Synchronous Timing

3.2 INSTRUCTION SEQUENCE DESCRIPTIONS

3.2.1 READ CYCLE

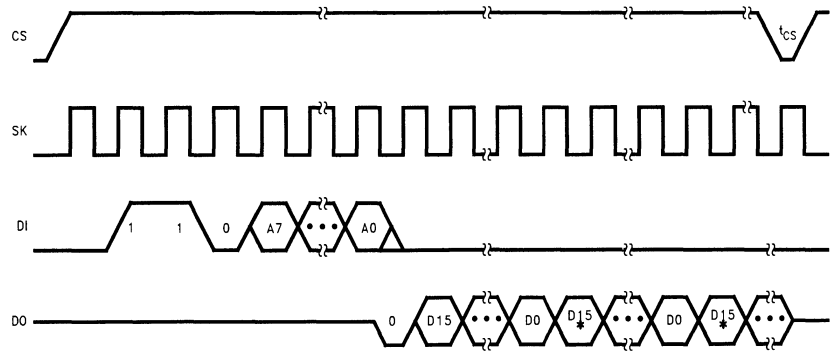
The READ cycle requires the host to raise chip select (CS) and then clock in thru the Data-In (DI) pin a start-bit, op-code, and address. Following clocking in the last address bit, the Data-Out (DO) output comes out of the high impedance state and drives a low level on the output. This is referred to as the dummy bit and is a good indication that a READ mode has been successfully entered if difficulty is encountered during initial debug of a system. The dummy bit is clocked out of the EEPROM on the same rising edge of SK that clocks in the last address bit, A0. This is shown in *Figure 9*.

3.2.2 SEQUENTIAL READ

Sequential read is a read mode available only on the NM93CS family. It is entered by entering a READ cycle and clocking out the first 16-bit word. After reading the first 16-bit word if chip select (CS) is kept high, address A + 1 may be clocked out followed by address A + 2 and so on. When the maximum address is reached, the memory continues in the sequential read mode at address 0. In this manner, the host may operate the memory in a continuous loop read. When initiating a SEQUENTIAL READ, the first data word is proceeded by a dummy bit as in a standard READ, although the dummy bit is supressed in all subsequent data words as shown in *Figure 9*.

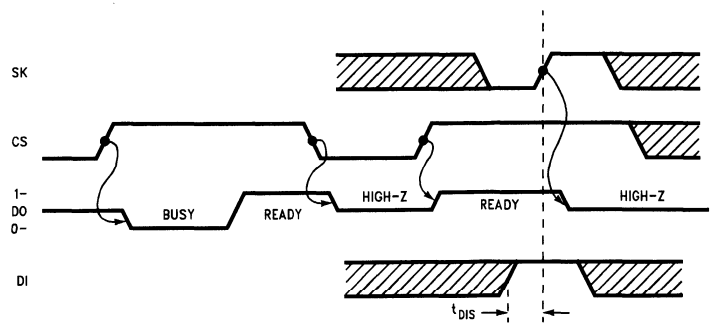
3.2.3 ERASE AND ERASE ALL

The ERASE cycles return the contents of the EEPROM to a clear state which is read as 1's. It is not necessary for any of the CMOS EEPROM described in this article, and is included in the NM93C family, NM93C46A, and NM59C11 only for compatibility with older devices that require erasing. It is recommended that the erase cycles be eliminated from the instructions to simplify the code, speed up writing and to improve the endurance obtained in the application. These modes are entered by clocking in a start-bit, op-code, and address. It is not necessary to clock in the data field as it is assumed to be all 1's. It is necessary to clock in the address, even in the case of ERASE-ALL where it is "don't care" in all except the first two bits of the address field which is used as additional op-code bits. After the full address field has been clocked in, chip select must be returned to a low level in initiate the erase cycle. In all devices, except the NMC9306, programming completion can be determined by Polling as shown in *Figure 4,* or a simple 10 ms timeout will guarantee programming is complete if polling is not used.



TL/D/11169–12

FIGURE 9. Sequential Read Sequence



TL/D/11169–6

FIGURE 4. Busy/Ready Polling Sequence

3.2.4 WRITE AND WRITE ALL

The Write and Write All cycles will write a specified data word into the specified address, or in the case of Write All, the same data pattern will be written into all locations. In all devices a new data pattern may be directly written over an existing data pattern without erasing the first data pattern. The write mode is entered by clocking in a start-bit, op-code, address, and data. The full address field must be clocked in for the Write All even though it is don't care in all but the first 2 bits. It is also necessary to clock in a full data field to assure correct alignment of data. The write cycle will be initiated after 8- or 16-bit have been clocked into the device in some of the devices and in other devices after chip select is brought low regardless of how many data bits have been clocked in. Refer to the specific datasheets to determine which method is used.

3.2.5 PROGRAM ENABLE AND PROGRAM DISABLE

Program enable and program disable are the instructions that enable or disable writing and, where included, erasing. The instruction name varies depending on the specific device but includes EWEN, EWDS, WEN, and WDS. These instructions enable or disable the entire memory array with a single instruction. All devices power up in the disable mode and once placed in the enabled mode remain enabled until a disable instruction is performed or V_{CC} is cycled. These instructions provide the most basic level of data protection. Although since most lost data is the result of the host device becoming uncontrolled and performing the "Program Subroutine" it may be helpful to structure the software such that the enable command is not included in the "Program Subroutine" but is in a separate subroutine. If a greater degree of data security is needed, a NM93CS family device is recommended, or other more elaborate schemes involving redundant data storage and polling.

3.2.6 PROTECT REGISTER READ

The protect register read (PRREAD) command is the same as a word read command except the input PRE must be held at a high level and the address is don't care. In spite of the address being don't care, the entire address field must be clocked in. On the Data-Out pin the contents of the protect register will be clocked out MSB first descending to LSB.

3.2.7 PROTECT REGISTER ENABLE

Similar to the programming enable instructions described above, the PREN instruction is necessary to perform any programming instruction the affects the Protect Register. Unlike the enable instructions described above, a PREN must immediately proceed each programming instruction that involves the protect register. The Protect Register programming instructions are PRCLEAR, PRWRITE, and PRDS.

3.2.8 PROTECT REGISTER DISABLE

The protect register disable instruction permanently disables any further programming instructions to the protect register. Therefore it can only be performed once in the lifetime of a NM93CS device. The purpose of it is to permanently configure a portion of the EEPROM as true ROM and a portion as Read/Write EEPROM. Great caution should be exercised prior to executing this instruction as there is no second chance. It is performed by sending a start-bit, op-code and an address field of all 0's while both the PRE and PE inputs are at a high level. This instruction must be immediately proceeded by a PREN instruction.

3.2.9 PROTECT REGISTER CLEAR

The protect register clear instruction will clear the contents of the Protect Register making the entire contents of the EEPROM alterable only if the PRDS instruction has not previously been executed. This is done by clocking in a start-bit, op-code, and address field of all ones. This instruction must be immediately proceeded by PREN instruction and requires that both PRE and PE inputs be held at a high level.

3.2.10 PROTECT REGISTER WRITE

The Protect Register write command (PRWRITE) allows the host to write the protect register with the address where the memory is to be segmented into ROM and EEPROM. The defined address is the first ROM address and the ROM field then continues to the top of memory. To execute this command a start-bit, op-code, and address must be clocked in, the address field containing the memory address that defines the ROM/EEPROM boundary. The PRE and PE inputs must be held at a high level.

3.3 INTERFACING SOLUTIONS

When interfacing serial microwire EEPROMs to microcontrollers there is an apparent conflict that occurs when selecting clock polarity and phase. This can be easily overcome in most situations, although when using some microcontrollers that do not allow selection of either clock polarity or clock phase, the only solution may be to resort to bit set and bit reset instructions to interface to the EEPROM rather than use of the serial interface provided on the microcontroller.

In the instance where there is a dedicated serial interface provided, the conflict typically occurs as follows. *Figure 10* demonstrates an EEPROM READ as this involves data being transferred from the micro to the EEPROM (Start bit, op-code, and address) and data transferred from the EEPROM to the micro (address contents). The conflict occurs in this example when the micros clock sets data up on the falling edge of SK and expects the EEPROM to accept it on the rising edge, but then expects the EEPROM to do the same when it sends data back to the micro.

1. The micro sets up a data bit. A propagation delay after the falling edge the data bit is valid at the EEPROM DI pin.
2. The EEPROM uses the rising edge of SK to clock the data bit into its internal register.
3. When the data direction changes the EEPROM sets the data up starting at the rising edge of SK.
4. The micro attempts to clock the data bit in that was set up on clock edge 3.

This example will work if the micro requires 20 ns or less data hold time after edge 4. If greater than 20 ns is required, an alternate strategy is needed.

1a. The micro sets up the data bit on the rising edge and a propagation delay later it is valid at the EEPROM.

2a. The EEPROM clocks the data into its internal register. The EEPROM requires only 10 ns data hold time, which can normally be guaranteed.

3a. The EEPROM sets the Data-Out up on the rising edge.

4a. The micro clocks the data into it's internal registers on the falling edge of the clock and a minimum data setup and hold time is guaranteed for the micro based on the minimum high and low time of the SK clock used in the application.

It should be noted that in the second example, CS (chip select) is asserted when SK is low. If this cannot be done, the DI input should be low when CS is asserted. If both DI and SK are high when CS is asserted the EEPROM will recognize this as a rising edge of SK. To accommodate this in a design, it is allowable to clock in any number of logic zeros prior to the start bit.

4.0 Conclusion

The serial EEPROM offered by National all share a common structure. Separating them are various features that give benefit to various applications such as the need for a bi-directional data bus or need for one byte word width. There are a number of "tricks" that may simplify interfacing to these which can easily be understood with the help of a functional block diagram. Given this information the overall job of using a serial interface EEPROM will be simpler.

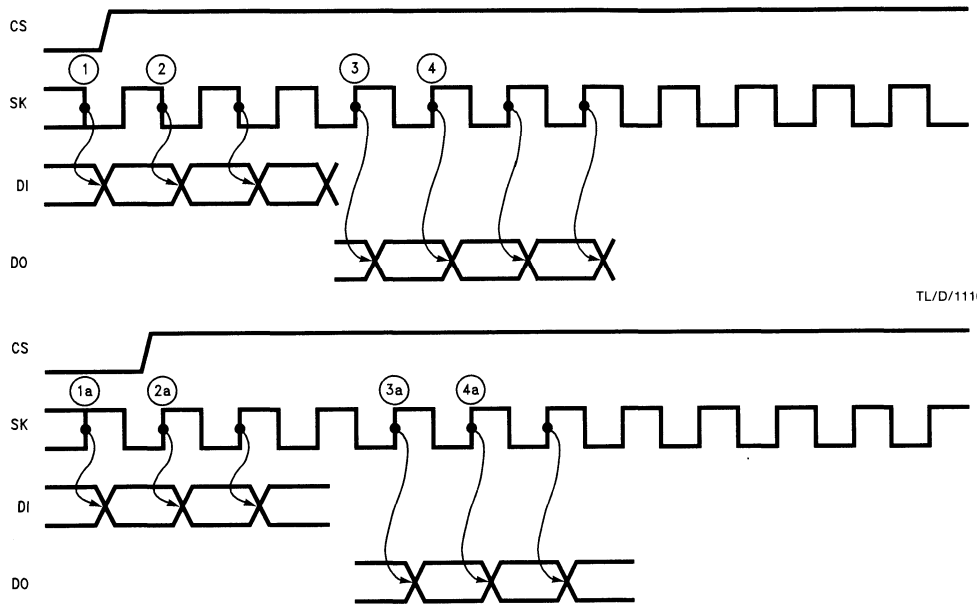


FIGURE 10

Using an EEPROM— I²C™ Interface NM24C02/03/04/05/08/09/16/17

National Semiconductor
Application Note 794
Paul Bryant

INTRODUCTION

National Semiconductor's NM24C EEPROMs are designed to interface with Inter-Integrated Circuit (I²C) buses and hardware. NSC's electrically erasable programmable read only memories (EEPROMs) offer valuable security features (write protection), two write modes, three read modes and a wide variety of memory sizes. Applications for the I²C bus and NM24C memories are included in SANs (small-area networks), stereos, televisions, automobiles and other scaled-down systems that don't require tremendous speeds but instead cost efficiency and design simplicity.

I²C BACKGROUND

The I²C bus configuration is an amalgam of microcontrollers and peripheral controllers. By definition: a device that transmits signals onto the I²C bus is the "transmitter" and a device that receives signals is the "receiver"; a device that controls signal transfers on the line in addition to controlling the clock frequency is the "master" and a device that is controlled by the master is the "slave". The master can transmit or receive signals to or from a slave, respectively, or control signal transfers between two slaves, where one is the transmitter and the other is the receiver. It is possible to combine several masters, in addition to several slaves, onto an I²C bus to form a multimaster system. If more than one master simultaneously tries to control the line, an arbitration procedure decides which master gets priority. The maximum number of devices connected to the bus is dictated by the maximum allowable capacitance on the lines, 400 pF, and the protocol's addressing limit of 16k; typical device capacitance is 10 pF. Up to eight E²PROMs can be connected to an I²C bus, depending on the size of the memory device implemented.

Simplicity of the I²C system is primarily due to the bidirectional 2-wire design, a serial data line (SDA) and serial clock line (SKL), and to the protocol format. Because of the efficient 2-wire configuration used by the I²C interface compared to that of the MICROWIRE™ and SPI interface, reduced board space and pin count allows the designer to have more creative flexibility while reducing interconnecting cost.

OPERATING NATIONAL SEMICONDUCTOR'S NM24Cs

The NM24C E²PROMs require only six simple operating codes for transmitting or receiving bits of information over the 2-wire I²C bus. These fields are explained in greater detail below and briefly described hereafter: a start bit, a 7-bit slave address, a read/write bit which defines whether the slave is a transmitter or receiver, an acknowledge bit, message bits divided into 8-bit segments and a stop bit.

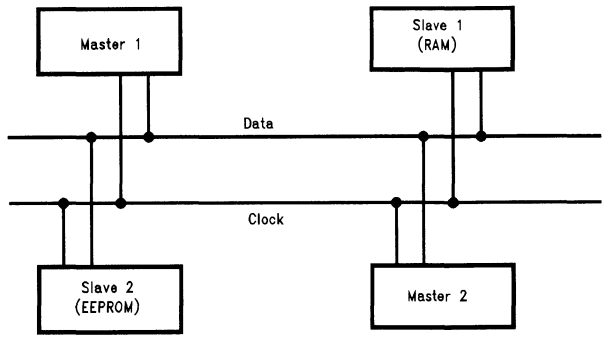
For efficient and faster serial communication between devices, the NM24C Family features page write and sequential read.

The NM24C03/C05/C09/C16/C17 Family offers a security feature in addition to standard features found in the NM24C02/C04/C08/C16 Family. The security feature is beneficial in that it allows Read Only Memory (ROM) to be implemented in the upper half of the memory to prevent any future programming in that particular chip section; the remaining memory that has not been write protected can still be programmed. The security feature in the NM24C03/C05/C09/C17 Family does not require immediate implementation when the device is interfaced to the I²C bus, which gives the designer the option to choose this feature at a later date. Table I displays the following parameters: memory content, write protect and the maximum number of individual I²C E²PROMs allowed on an I²C bus at one time if the total line capacitance is kept below 400 pF.

Code used to interface the NM24Cs with National Semiconductor's COP8™ Microcontroller Family is listed in a latter section of this application note for further information to the reader.

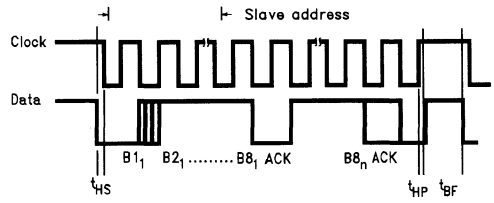
TABLE I

Part No.	Number of 256x8 Page Blocks	Write Protect Feature	Max. Parts
NM24C02	1	No	8
NM24C03	1	Yes	8
NM24C04	2	No	4
NM24C05	2	Yes	4
NM24C08	4	No	2
NM24C09	4	Yes	2
NM24C16	8	No	1
NM24C17	8	Yes	1



TL/D/11268–1

FIGURE 1. I²C-Bus Configurations



TL/D/11268–2

FIGURE 2. I²C Bus Timing

Start Condition

— Clock and Data line high (Bus free)

— Change Data line from high to low

— After $t_{HS(Min)}$ = 4 μs the master supplies the clock

Acknowledge

— Transmitting device releases the Data line

— The receiving device pulls the Data line low during the ACK-clock if there is no error

— If there is no ACK, the master will generate a Stop Condition to abort the transfer

Stop Condition

— Clock line goes high

— After $t_{HP(Min)}$ = 4.7 μs the Data lines go high

— The master maintains the Data and Clock line high

— Next Start Condition after $t_{FB(Min)}$ = 4.7 μs is possible

7

START/STOP CONDITIONS

If both the data and clock lines are HIGH, the bus is not busy. To attain control of the bus, a start condition is needed from a master; and to release the lines, a stop condition is required.

Start Condition: HIGH-to-LOW transition of the data line while the clock line is in a HIGH state.

Stop Condition: LOW-to-HIGH transition of the data line while the clock line is in a HIGH state.

The master always generates the start and stop conditions. After the start condition the bus is in the busy state. The bus becomes free after the stop condition.

DATA BIT TRANSFER

After a start condition "S" one databit is transferred during each clock pulse. The data must be stable during the HIGH-period of the clock. The data line can only change when the clock line is at a LOW level.

Normally each data transfer is done with 8 data bits and 1 acknowledge bit (byte format with acknowledge).

ACKNOWLEDGE

Each data transfer needs to be acknowledged. The master generates the acknowledge clock pulse. The transmitter releases the data line (SDA = HIGH) during the acknowledge clock pulse. If there was no error detected, the receiver will pull down the SDA-line during the HIGH period of the acknowledge clock pulse.

If a slave receiver is not able to acknowledge, the slave will keep the SDA line HIGH and the master can then generate a STOP condition to abort the transfer.

If a master receiver keeps the SDA line HIGH, during the acknowledge clock pulse the master signals the end of data transmission and the slave transmitter release the data line to allow the master to generate a STOP-condition.

ARBITRATION

Only in multimaster systems.

If more than one device are potential masters and more than one desires access to the bus, an arbitration procedure takes place: if a master transmits a HIGH level and another master transmits a LOW level, the master with the LOW level will get the bus and the other master will release the bus; and the clock line switches immediately to the slave receiver mode. This arbitration could carry on through many bits (address bits and data bits are used for arbitration).

FORMATS

There are three data transfer formats supported:

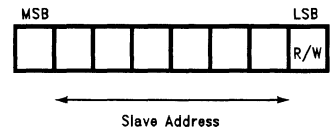
— Master transmitter writes to slave receiver; no direction change

— Master reads immediately after sending the address byte

— Combined format with multiple read or write tranfers.

ADDRESSING

The 7-bit address of an I²C device and the direction of the following data is coded in the first byte after the start condition:

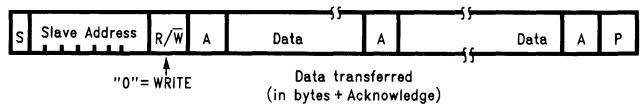


TL/D/11268–3

A "0" on the least significant bit indicates that the master will write information to the selected Slave address device; a "1" indicates that the master will read data from the slave.

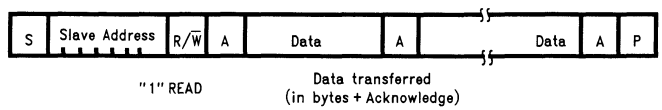
Some slave addresses are reserved for future use. These are all addresses with the bit combinations 1111XXX and 0000XXX. The address 00000000 is used for a general call address, for example, to initialize all I²C devices (refer to I²C bus specification for detailed information).

Master Transmits to Slave, No Directon Change



TL/D/11268–4

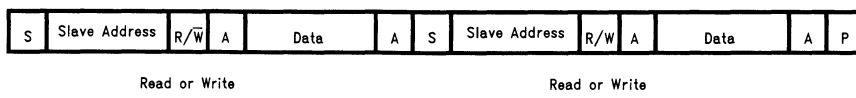
Master Reads Slave Immediately after First Byte



TL/D/11268–5

The master becomes a master receiver after first ACK

Combined Formats



TL/D/11268–6

n bytes Data + ACK n bytes Data + ACK

S = Start Condition A = Acknowledge P = Stop Condition

FIGURE 3. I²C-Bus Transfer Formats

TIMING

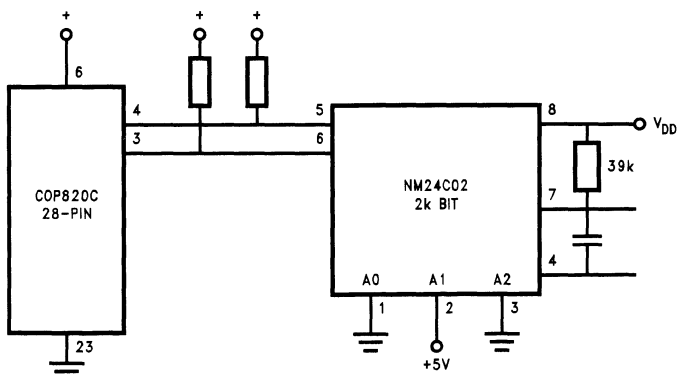
The master can generate a maximum clock frequency of 100 KHz. The minimum LOW period is defined as 4.7 μs; the minimum HIGH period width is 4 μs; the maximum rise time on SDA and SCL is 1 μs; and the maximum fall time on SDA and SCL is 300 ns.

Figure 4 shows the detailed timing requirements.

Symbol	Parameter	Min	Max	Units
f_{SCL}	SCL Clock Frequency	0	100	kHz
t_{BUF}	Time the Bus Must Be Free before a New Transmission Can Start	4.7		μs
t_{HD}; STA	Hold Time Start Condition. After this Period the First Clock Pulse is Generated	4.0		μs
t_{LOW}	The LOW Period of the Clock	4.7		μs
t_{SU}; STA	Setup Time for Start Condition (Only Relevant for a Repeated Start Condition)	4.7		μs
t_{HD}; DAT	Data in Hold Time	5 0*		μs μs
t_{SU}; DAT	Setup Time Data	250		ns
t_r	Rise Time of Both SDA and SCL Lines		1	μs
t_f	Fall time of Both SDA and SCL Lines		300	ns
t_{SU}; STO	Setup Time for Stop Condition	4.7		μs

*Note that a transmitter must internally provide at least a hold time to bridge the undefined region (max. 300 ns) of the falling edge of SCL.

FIGURE 4. I²C-Bus Timing Requirements



TL/D/11268–7

FIGURE 5. I²C Bus EEPROM/μController Configuration Used for Sample Code

SOFTWARE TASKS

I. Write fixed values to E2PROM cells

II. Read values back from E2PROM and save in RAM locations from COP

Note: I²C Bus Modes Used:

Master Transmitter $\overset{SDA}{SCL} \rightarrow$ Slave Receiver

Master Receiver $\overset{SDA}{SCL} \rightarrow$ Slave Receiver

REMARKS

— The I²C bus, 2-wire serial interface generally requires a pull-up resistor on the SDA line and the SCL line, depending on whether TTL or CMOS hardware interfacing exists.

— I²C bus compatible μC's or peripherals have OPEN DRAIN outputs at SDA and SCL.

— COP800 does not have OPEN DRAIN outputs, but the "bus requirements" can be met by switching SDA and SCL connections into TRI-STATE® for the following cases:

The bus is not accessed

A slave has to send an acknowledge bit.

— MICROWIRE can not be used for I²C bus operations.

— Current sink capability on SDA and SCL must be 3 mA to maintain "Low Level" (an I²C bus spec.).

```
.TITLE IIC - EEPROM  ROUTINES'
.INCLD COP800.INC
.CHIP 840
.LIST X '21

          * * *TASK RELATED RAM - DECLARE* * *

EEADR                                    = 002      ;ADDRESS OF EEPROM
EEWRD                                    = 003      ;WORD ADDRESS EEPR.
EEDAT1                                   = 004      ;DATA TO EECELL
EEDAT2                                   = 005      ;SECOND BYTE
FLAG                                     = 010      ;FLAG–WORD
EEREAD                                   = 012      ;READ–DATA FROM EE
                                         = 013      ;SECOND BYTE
                                         = 014      ;THIRD BYTE
                                         = 015      ;FOURTH BYTE
BITCO                                    = 0F0      ;COUNTER FOR BITSHFT

INIT:
   LD SP,                                #06F
   LD B,                                 PORTLD     ; INIT LS, L3 FOR EE-
   LD  [B+],                             #00C       ;OPERATIONS
   LD  [B],                              #00C
   LD B,                                 #EEDAT2    ; INIT RAMS
   LD  [B-],                             #034       ;FIXEED VALUES FOR
   LD  [B-],                             #012       ; EEWRITE (2 BYTES)
   LD  [B-],                             #0A0       ; MIRROR OF #05
   LD [B]                                #025       ; MIRROR OF "A5"

      ;* * * * * * * * * * * * * * * * * * * * * * * * * * * * *
      ; EXAMPLE: IF ADDRESS BYTES IS    "1010 010X THEN     *
      ; STORE:                           "X010 0101          *
      ; INTO RAM (X=0/1; WRITE/READ)                         *
      ;* * * * * * * * * * * * * * * * * * * * * * * * * * * * *
      ;

   LD PSW,                               #00        ;LOAD PSW
   LD CNTRL,                             #00        ;AND CNTRL REG.
   LD FLAG,                              #0
   .FORM

            ;  ******************************
            ;  ****   DO WRITE TO EE-PROM   ****
            ;  ******************************
            ;

         ; (2 BYTE SUCCESSIVE WRITE)

   SBIT 0,                               FLAG       ;SET FLAG FOR WRITE
   LD B,                                 PORTLD     ;POINT LPORT DAT REG.
                                                    ;TO MODIFY "SDA, SCL"
   RBIT 2 [B],                                      ;PREPARE FOR START
   JSR STACON                                       ;CONDITION.
   JSR WAIT                                         ;AFTER WRITE TO EE.
                                                    ;WAIT FOR > THAT 40
```

TL/D/11288–6

```
                ************************
              ; * * DO THE START CONDITION * *
              ; * * AND SHIFT OUT ADRESS - * *
              ; * * BYTE AND WORD–ADRESS * *
                ************************

STACON:
    RBIT 3,                          PORTLD      ; FINISH START COND.
    LD B,                            #EEADR      ; PREPARE TO CLOCK
                                                 ; OUT ADDRESS.
LOPA:
    LD BITCO,                        #0 0 8      ; DO SETS OF 8 BITS

LOPA 1:
    IFBIT 0, [B]                                 ; SWITCH SDA BEFORE
    JP ONE,                                      ; SCL
    RBIT 2,                          PORTLD      ; SET BIT LEVLE "0"
    JP CLK

ONE:
    SBIT 2,                          PORTLD      ; SET BIT LEVEL "1"
    JP CLK                                       ; ENSURE SAME BIT
                                                 ; LENGTH

CLK:
    SBIT 3,                          PORTLD      ; DO CLOCK PULSE
    NOP
    NOP
    RBIT 3,                          PORTLD      ; ENSURE> 4USEC
    RBIT 2,                          PORTLD      ; SWITCH ALSO SDA LOW
    .FORM

    LD A, [B]                                    ; ROTATE BYTE ONE
    RRC A,                                       ; BIT POS. RIGHT
    X A, [B]                                     ; AND SAVE
    DRSZ BITCO                                   ; CHECK IF 8 BITS
    JP LOPA1,                                    ; SHIFTED
    LD A, [B+],                                  ; DECREMENT 8
    IFBIT 1,                         FLAG        ; CHECK IF READ
    JMP,                             GETDAT      ; 3RD BYTE IS NEXT?
                                                 ; IF SO, THEN READ.
    JSR ACK,                                     ; GET ACKNOWLEDGED
                                                 ; WHEN 8 BITS ARE
                                                 ; SHIFTED.
    IFBIT 0,                         FLAG        ; CHECK IF READ
    JP CEC1,                                     ; OR WRITE OPERATION.
    IFBNE                            #0 4        ; ON READ (HERE)
    JMP LOPA,                                    ; IF NOT 2 BYTES YET
    RET                                          ; AFTER EE-ADDRESS AND
                                                 ; WORD ADDRESS ARE SHFT.

CEC1:
    IFBNE,                           #0 6        ; 1ST AND 2ND DATA-
    JMP LOPA,                                    ; BYTE (3RD + 4TH)
```

TL/D/11268–9

```
                                                    ;NSEC TO PROPERLY
                                                    ; ERASE WRITE.

LD B,                                   #EEDAT2     ; INIT RAMS
LD  [B-],                               #078        ;ANOTHER 2 BYTES
LD  [B-],                               #056        ;OF FIXED DATA
LD  [B-],                               #0E0        ; MIRROR OF #07
LD  [B],                                #025        ; MIRROR OF "A5"
LD B,                                   PORTLD      ; POINT LPORT DAT REG.
                ; TO MODIFY "SDA, SCL"
RBIT 2, [B],                                        ;PREPARE FOR START
JSR STACON,                                         ;CONDITION.
JSR WAIT,                                           ;AFTER WRITE TO EE.
                                                    ; WAIT FOR > THAN 40
                                                    ;MSEC TO PROPERLY
                                                    ; ERASE WRITE.
.FORM

        ; ****************************
        ; **** DO READ FROM EE-PROM ****
        ; ****************************

            (READ 4 SUCCESSIVE BYTES)
RBIT 0                                  FLAG        ;INDICATE READ
LD B,                                   #EEWRD      ; INIT RAMS
LD  [B-],                               #0A0        ; MIRROR OF #05
LD  [B],                                #025        ; MIRROR OF "A5"

        ; ****************************
        ; ** FIRST 2 BYTES SAME AS IF WRITE **
        ; ****************************

                                (IN TERMS OF TRNSMIT)
LD B,                                   #PRTLD      ;PREPARE
RBIT 2 [B]                                          ;FOR
JSR STACON,                                         ;START COND.
                                                    ;AND SHIFT 1ST
                                                    ;2 BYTES.
SBIT 2,                                 PORTLD      ;PREPARE FOR
NOP,                                                ;ANOTHER START-
NOP,                                                ;CONDITION,
SBIT 3,                                 PORTLD      ;SDA HIGH FIRST.
SBIT 1,                                 FLAG        ;INDICATE THAT
                                                    ;3RD BYTE IS NEXT
LD B,                                   #EEWRD      ; INIT RAMS
LD  [B-],                               #0A0        ; MIRROR OF #05
LD  [B],                                #0A5        ; MIRROR OF "A5"
                                                    ;PERFORM ANOTHER
RBIT 2, [B],                            PORTLD      ;START
JSR STACON
RBIT 1,                                 FLAG
JMP INIT                                            ;CLOSE THE LOOP WHEN
.FORM                                               ;FINISHED
```

TL/D/11268–14

```
STP:
    SBIT 3,                          PORTLD      ; ESTABLISH STOP-
    NOP,                                         ; CONDITION
    SBIT 2,                          PORTLD
    RET,
    .FORM

        ********************************
        ;** GET 8BIT OF DATA FROM EE-PROM **
        ********************************

GETDAT:
    JSR ACK,                                     ; GET ACKNOWLEDGMENT
    LD B,                            #EEREAD     ; POINT FIRST READ RAM
    JP                               GETDT1      ; AND READ IN

GETDAT:
    JSR ACK,                                     ; ACKNOWLEDGMENT TO EE-
                                                 ; PROM WHEN 8 BITS
                                                 ; ARE SHIFTED IN.

GETDAT1:
    LD BITCO,                        #008        ; INIT BIT COUNTER
    RBIT 2,                          PORTLC      ; BEFORE READING, PUT
    RBIT 2,                          PORTLD      ; 'SDA' INTO HIGH-Z.

LOPB:
    SBIT 3,                          PORTLD      ; DO CLOCK HIGH
    RBIT 7, [B]                                  ; READ IN EEDATA
    IFBIT 2,                         PORTLD      ; IN SETS OF 8 BITS
    SBIT 7, [B]
    RBIT 3,                          PORTLD      ; DO CLOCK LOW
    DRSZ BITCO,                                  ; CHECK IF 8 BITS
    JP SHFT                                      ; ARE SHIFTED
    LD A, [B+],                                  ; INCREMENT B
    IFBNE                            #06         ; CHECK IF 4 BYTES
    JMP GETDT,                                   ; ARE SHIFTED IN?
    SBIT 2,                          PORTLC      ; PUT L2=0
    JMP STP                                      ; WHEN TRUE, DO STOP
                                                 ; CONDITION AND
                                                 ; RETURN

    .FORM

SHFT:
    LD A, [B],                                   ; ROTATE BITS ONE
    RRC A                                        ; POSITION RIGHT
    X A, [B]
    JP LOPB

        ************************************
        ; ** SIMPLE ROUTING TO DO 40 MSEC DELAY **
        ************************************
```

TL/D/11268–12

7

```
WAIT:
    LD 0F1              #0.20       ; SIMPLE WAIT LOOP

LOPD:

                                    ; TO PRODUCE>40MSEC
    LD 0F2,             #OFF        ; TIMEOUT

LOPC:

                                    ; TO PROPERLY PROGRAM
    DRSZ 0F2,                       ; EEPROM. TIME REQUIRED
    JP LOPC,                        ; TO ERASE/WRITE
    DRSZ0F1,                        ; THE EEPART.
    JP LOPD
    RET

ACK1:
    SBIT 2,             PORTLC      ; INDICATE TO EE-PROM
    JP ACLK,                        ; (PUT DATA LINE LOW)

ACK:
    RBIT 2,             PORTLC      ; PUT DATA-LINE HI-Z

ACLK:

                                    ; AND GET ACKNOWLEDGE
    SBIT 3,             PORTLD      ; 8 BITS ARE SHIFTED,
    NOP                             ; DO A DUMMY CLOCK
    NOP
    NOP
    RBIT 3,             PORTLD      ; (FOR ACKNOWLEDGE)

    SBIT 2,             PORTLC
    RET
    .END
```

TL/D/11268–13

Timekeeping Using a COP800 Microcontroller

National Semiconductor
Application Note 823
Mike Werstlein

ABSTRACT

Many applications require the time of day be available to the system so that a time stamp can be associated with an event. This application note discusses how National Semiconductor's COP820C microcontroller can be used to implement a timekeeping function. In addition, this paper demonstrates how two COP472 LCD display drivers are cascaded together to drive an 8 digit LCD display, which is used to display the time of day.

INTRODUCTION

Most often applications which use microcontrollers require some way of obtaining the time of day. Examples of these such applications are security systems, automatic sprinkler systems, telephone answering machines, automatic coffee makers, automatic thermostats, etc. One way of keeping track of the time and date is to use a dedicated real time clock IC. Although this is an easy and accurate option to implement, it is not very cost effective. The most cost effective method of timekeeping is to have the microcontroller itself perform the timekeeping function.

Timekeeping using the COP820C microcontroller is a relatively simple task. The system described here consists of a COP820C microcontroller, two COP472 LCD display drivers, and a Hamlin 8 digit LCD display (model #4216). The COP820C uses a 32.768 kHz crystal.

The timekeeping system described in this application note has many capabilities. The display can be toggled between two modes. One mode displays the time in the following format: HH-MM-SS (hours-minutes-seconds). The other mode displays the date in the following format: YR-MTH-DAY (year-month-day). Each part of the time or date can be individually set. In other words, one doesn't have to increment the seconds past 59 in order to increment the minutes by 1. Also, the real time clock automatically determines leap years and adjusts itself accordingly.

The following sections will detail the hardware and software portions of the real time clock.

HARDWARE

As previously stated, the hardware consists of a COP820C microcontroller, two COP472 LCD display drivers, a Hamlin 8 digit LCD display, and four normally open pushbuttons. These buttons are used to set the time/date and to select the display mode, either time or date. The schematic for the system is shown in *Figure 1*.

The four pushbuttons are interfaced to the COP820C via the lower four pins of the I port. 1 kΩ pullup resistors are used to keep the I lines high until a button is pressed, in which case the corresponding pin is pulled low.

G0 and G1 are used as chip select lines for the two COP472's. G0 is used for the slave while G1 is used for the master COP472. The technique used to cascade two COP472's together is detailed in the COP472 data sheet.

SOFTWARE

The software portion of the real time clock consists of several subroutines used to perform special functions. The program uses approximately 600 bytes of ROM. Included in the program are the routines to display the time/date to the LCD. There are also some overhead routines used for the LCD. This means that the clock portion of the program is actually less than 600 bytes.

The program relies on the 32.768 kHz crystal oscillator as its time base. The 16-bit timer is loaded with a value that will correspond to one underlow per second. This underflow will cause an interrupt to occur that will increment the seconds value by one.

The beginning of the program is used to initialize certain RAM locations and also to configure the proper I/O ports and timers.

The first subroutine to be called is labeled SYNC. This routine is used to synchronize two COP472 LCD displays. One COP472 can drive 4½ digits. Therefore, in order to display more data, additional COP472's can be cascaded together to drive more digits.

The next section of the program is used to set the time/date information. The first step is to read the value of port I. Since only I0–I3 are used, the value of port I is AND'ed with 0F hex to mask out the upper four bits. If a button is pressed, a status bit in the CLKSTAT register is checked to see if the clock is displaying the time or the date. The correct part of the time/date is then set accordingly. The program keeps scanning the buttons until one of them is pressed or a timer interrupt is encountered, at which time the appropriate routines are executed.

When the time underflow interrupt occurs (once per second) the program jumps to location 0FF hex. Here the program jumps to the timer interrupt service routine if the timer interrupt pending bit is set. The timer inerrupt service routine simply increments the seconds value by one. It also takes care of incrementing the minutes, hours, days, months, and years as needed. Leap years are also accounted for during this routine.

The next subroutine to be called is labeled ADJUST and is used to account for the clock inaccuracies. This is necessary because the instruction cycle time is 1/3276.8 seconds instead of the ideal time of 1/3277 seconds. An integer value for the instruction cycle time is considered ideal. The reason it is ideal is because if this value is loaded into the timer, then the underflow will occur exactly once per second. This inaccuracy can be fixed easily. By using a generic counter, one second can be added to the time whenever the clock becomes one second behind.

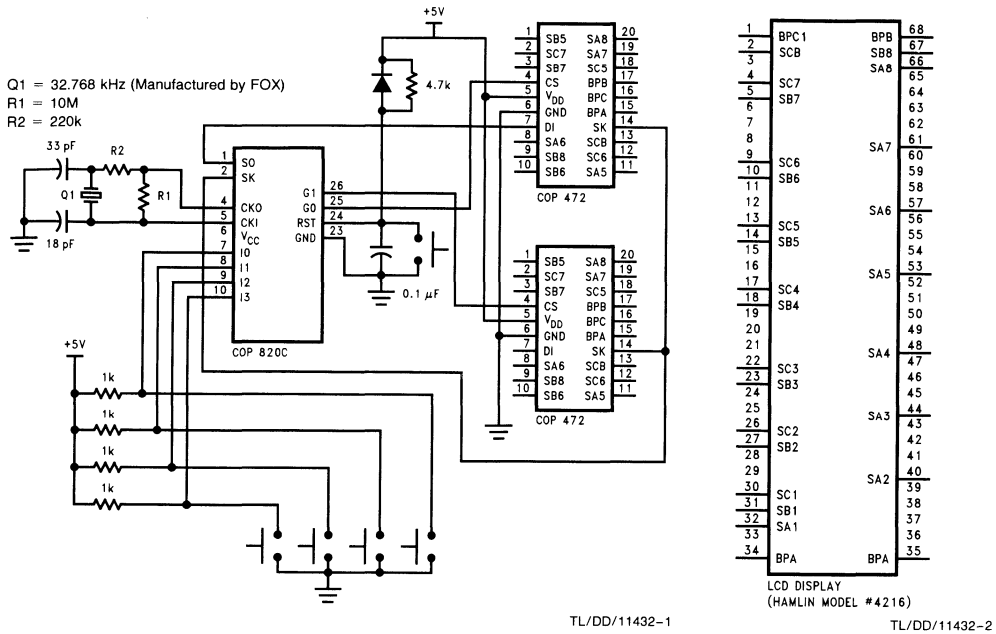
The next subroutine to be called is BCDDIG. This routine is used to convert the time/date information into a format that is usable by the COP472's. Memory locations 01–0A contain the COP472's digit data.

The subroutine CONVRT is then called. This routine is used to convert the data contained in memory locations 01–0A into the proper format for the Hamlin LCD. This conversion routine might vary depdending on the particular display being used.

The final subroutine to be called is DISP. This routine is used to send the contents of memory locations 01–0A to the COP472's so that the new time/date information can be updated on the display. The program then returns to the main loop at the beginning.

CONCLUSION

As can be seen, the COP820C can easily incorporate time-keeping functions into virtually any existing program. This approach can save the user the cost and time of adding additional circuit components to the overall system. Although this program takes about 600 bytes of memory, a significant portion of the program is overhead for the COP472's and the LCD display. Various parts of the program might not be necessary in a specific application.



TL/DD/11432-1

TL/DD/11432-2

Note: For simplification purposes connections between the LCD and the COP472's are not shown.

FIGURE 1. System Schematic

8-Channel 8-Bit PWM Controller

National Semiconductor
Application Note 824
Patrick Furlan

INTRODUCTION

This application note discusses a cost effective implementation of an 8-channel DAC to replace potentiometers.

TECHNICAL OVERVIEW

The COP822C was considered for the application. At the outset since the DACs were replacing pots, speed of conversion was not an issue. The issue became in that how fast a frequency with 8-bits of resolution on eight channels could be implemented in software. This would then determine the response time and therefore the filtering components to convert the varying duty cycle squarewave to a DC voltage. A simple RC can be used or for better response a pie filter can be used. Depending on the load, buffering may be required. In preliminary testing ripple was less than 1-bit.

IMPLEMENTATION

Software was then written to determine the time required to execute one loop of the program that determined the resolution that could be achieved for 8 separate channels. The routine is basically a small loop that decrements 8 registers or counters and reloads these counters after 8-bits of resolution. It was determined that the loop could be done in 40 μs. This is the limiting factor. From this 40 μs (100 Hz instruction cycle frequency) per bit for 8 bits of resolution, the period turns out to be 10 ms. Therefore, in 10 ms all 8 channels are updated with their on/off times.

Since the outputs are constantly running, interrupts are not used so that the PWM outputs stay more stable. Also, this provides a faster throughput. Interface to the chip can be done in either a serial (MICROWIRE/PLUS™) or parallel fashion, depending on best fit for the application. For a serial implementation the Microwire busy bit can be polled each loop. If parallel interface is required, there are enough pins on the device to implement a simple handshake exchange; i.e., have 3 address lines, 4 data lines and a chip select. In either case, it requires a two byte protocol: address and data. Data is the PWM "on time" to determine duty cycle.

CONCLUSION

This low cost implementation of an 8-channel 8-bit PWM controller has multiple features. Besides a low speed DAC, PWM control in conjunction with NSC DMOS power products could also be a cost effective peripheral for power drive applications. It should be noted that using this approach, there is no CPU time for doing other tasks. One last item to note is the COP800 output structure. Depending on application the outputs (G and L) can be configured in TRI-STATE® mode, thereby putting the external filter in a holding pattern or low leakage state. In this way other small routines i.e., interface, could be accomplished.

Due to the software implementation methodology, there is flexibility, i.e., in the number of channels, resolution and the interface. Also, since it is based around a COP800 solution, packing (pins) and operating frequency including crystal options are also flexible.

The following pages show the code used in evaluating the concept as well as the filter components. Basically, eight register with varying "on times" were loaded so that the PWM outputs could be analyzed along with software performance. The remaining code for MICROWIRE/PLUS and the exact filter components are not finalized.

```
; COP822 - 8-Channel 8-Bit PWM Output
          .CHIP       820

INIT:     LD          0EE,#00       ;clear control reg.
          LD          0EF,#00        ;clear psw, int, etc.
          LD          SP,#02F        :TOP OF STACK ??

          LD          008,#05        ;LOAD 8 AUTO RELOAD RESCNT"ERS
          LD          009,#25        ;RAM ADDR 8 THROUGH OFH
          LD          00A,#50        ;TEST ONLY, IN REAL LIFE THESE
          LD          00B,#90        ;GET LOADED THROUGH MICROWIRE
          LD          00C,#125
          LD          00D,#160
          LD          00E,#210
          LD          00F,#250

; PLACE TO TRANSFER RELOAD COUNTERS TO RESCNTERS

          JSR         RELOAD         ;AUTO RELOAD COUNT TO RESCNT'ERS

          LD          0D1,#0FF       ;L CONFIG. REG TO PUSH PULL ONE OUT
          LD          0D0,#0FF       ;L ports to all 1's

PERIOD:   LD          0F0,#255       ;255 * THROUGH LOOP = 8-BIT RES.

RESCNT:   LD          B,#00          ;START OF RAM MAP FOR RESCNT'ERS
          LD          A,[B]          ;DEC "ON TIME" COUNTERS
          DEC         A
          X           A,[B+]         ;PUT BACK FOR NEXT TIME
          IFEQ        A,#00          ;WHEN CNT = 0, PORT LOW
          RBIT        0,0D0           ;D0 = MEMORY MAP FOR PORT L

;2ND PWM OUTPUT
          LD          A,[B]          ;DEC "ON TIME" COUNTERS
          DEC         A
          X           A,[B+]         ;PUT BACK FOR NEXT TIME
          IFEQ        A,#00          ;WHEN CNT = 0, PORT LOW
          RBIT        1,0D0           ;D0 = MEMORY MAP FOR PORT L
;3RD PWM OUTPUT
          LD          A,[B]          ;DEC "ON TIME" COUNTERS
          DEC         A
          X           A,[B+]         ;PUT BACK FOR NEXT TIME
          IFEQ        A,#00          ;WHEN CNT = 0, PORT LOW
          RBIT        2,0D0           ;D0 = MEMORY MAP FOR PORT L
;4TH PWM OUTPUT
          LD          A,[B]          ;DEC "ON TIME" COUNTERS
          DEC         A
          X           A,[B+]         ;PUT BACK FOR NEXT TIME
          IFEQ        A,#00          ;WHEN CNT = 0, PORT LOW
          RBIT        3,0D0           ;D0 = MEMORY MAP FOR PORT L
```

```
;5TH PWM OUTPUT
          LD          A,[B]           ;DEC "ON TIME" COUNTERS
          DEC         A
          X           A,[B+]          ;PUT BACK FOR NEXT TIME
          IFEQ        A,#00           ;WHEN CNT = 0, PORT LOW
          RBIT        4,0D0            ;DO = MEMORY MAP FOR PORT L
;6TH PWM OUTPUT
          LD          A,[B]           ;DEC "ON TIME" COUNTERS
          DEC         A
          X           A,[B+]          ;PUT BACK FOR NEXT TIME
          IFEQ        A,#00           ;WHEN CNT = 0, PORT LOW
          RBIT        5,0D0            ;DO = MEMORY MAP FOR PORT L
;7TH PWM OUTPUT
          LD          A,[B]           ;DEC "ON TIME" COUNTERS
          DEC         A
          X           A,[B+]          ;PUT BACK FOR NEXT TIME
          IFEQ        A,#00           ;WHEN CNT = 0, PORT LOW
          RBIT        6,0D0            ;DO = MEMORY MAP FOR PORT L
;8TH PWM OUTPUT
          LD          A,[B]           ;DEC "ON TIME" COUNTERS
          DEC         A
          X           A,[B+]          ;PUT BACK FOR NEXT TIME
          IFEQ        A,#00           ;WHEN CNT = 0, PORT LOW
          RBIT        7,0D0            ;DO = MEMORY MAP FOR PORT L

;  NOTE 255 TIMES IS NOW IN LOOP, SOON TO BE INTERRUPT.
          DRSZ        0F0             ;PERIOD TERMINATOR = 0F0
          JMP         RESCNT          ;FINISH 255 TIMES

;DEC 0F0 IF ZERO RESET RES COUNTERS AND PUT PORT L HI.

          LD          0D0,#0FF        ;PORT L HI
          JSR         RELOAD
                              ;This is place Microwire should be checked.
          JMP         PERIOD          ;START PERIOD OVER WITH NEW COUNTS

; RELOAD WILL PUT RAM FROM ADDR 8 TO F IN 0 TO 7.
RELOAD
          LD          X,#008          ;1ST RELOAD COUNTER
          LD          B,#00           ;ST RESCNT
          LD          A,[X+]
          X           A,[B+]
          LD          A,[X+]
          X           A,[B+]
          LD          A,[X+]
          X           A,[B+]
          LD          A,[X+]
          X           A,[B+]
          LD          A,[X+]
          X           A,[B+]
          LD          A,[X+]
          X           A,[B+]
          LD          A,[X+]
          X           A,[B+]
          LD          A,[X+]
          X           A,[B+]
          RET
          .END
```

Software for Interfacing the COP8™ Family Microcontrollers to National's MICROWIRE™ EEPROMs

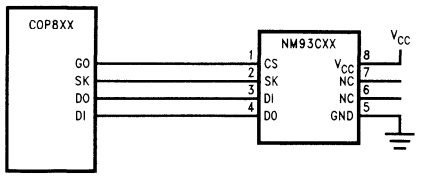
National Semiconductor
Application Note 841
Paul Bryant

ABSTRACT

National's NM93Cxx and NM93CSxx family of serial EEPROMs have MICROWIRE "slave" interfaces that directly connect to the MICROWIRE "master" interfaces on the COP8™ family of 8-bit microcontrollers. Peak data transfer rates are as high as 1 MB on the 4 wire MICROWIRE bus. This application note includes the essential assembly language software to address MICROWIRE EEPROMs.

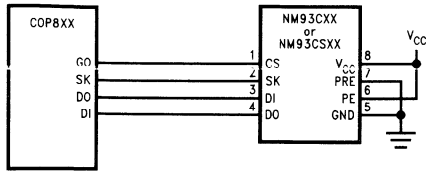
HARDWARE INTERFACE

A schematic for connecting a COP8 family microcontroller to one of Nationals NM93Cxx family MICROWIRE EEPROMs via the microcontrollers dedicated MICROWIRE port is shown in *Figure 1*. National makes two basic families of MICROWIRE EEPROMs, the classic NM93**C**xx series and the newer full featured NM93**CS**xx series. The NM93CSxx series EEPROMs have a sequential read capability and the "S" in "CS" stands for sequential (the "C" implies CMOS). As can be seen the "CS" series devices have two additional pins which control write protect features (consult a data sheet for information on their function). By connection these pins to V_{CC} and GND as shown in *Figure 2*, it becomes possible to use either "C" or "CS" family parts in the same physical socket. This would be desirable if a change from a "C" series part to a "CS" series part is possible for reasons of product upgrades or simply manufacturing inventory control. Pins 6 and 7 on the "C" series parts are true no connects and thus can be tied to V_{CC} or ground harmlessly.



TL/D/11491–1

FIGURE 1. Basic National MICROWIRE Interconnection to a COP8 Family Part



TL/D/11491–2

FIGURE 2. This schematic allows either a NM93Cxx or NM93CSxx part to be used in the same socket. Software can then be adjusted to take advantage of the "CS" series advantages without making changes to the board.

If it's desirable to use both types in the same socket without being forced to make software changes, one must be careful not to use the sequential read capability of the "CS" series. Both types of parts should be tested in the socket before the software is frozen.

NM93C06 to COP8 Family Software Details

Always consult the latest data sheets for information about timing variables mentioned in the text that follows. These numbers were correct at the time that this application note was written but are subject to change.

1. The SK clock frequency must not exceed 1 MHz. Consult the processor data sheet for details.
2. The CS low time following a write must exceed 250 ns. This starts the internally timed write operation. The DO line will leave the high impedance state if CS goes high again and will drive low until the internal write cycle is complete. After DO returns high, indicating "ready" the first rising edge of SK with CS high and DI high will return the DO pin to the high impedance condition. This condition is normally the start bit of the next instruction.

   The DO pin will be low for up to 10 ms and then go high to indicate that the write is complete. If a new instruction is attempted before the DO pin returns high it will be ignored and the DO pin will not go tristate. The DO pin will always go to the tristate condition when CS is low.
3. Opcodes are either 2-bits or 4-bits long depending on the instruction type and are always preceded by a "start-bit" of a logic one. Any number of leading zeros can be clocked in before the start-bit (the sample assembly code inserts seven). Addresses are either 6 or 8-bits long depending on the density of the device. The combined opcode and address field is 8-bits for the smaller devices (93C06 and 93C46) and 10-bits for the larger devices (93C56 and 93C66). On the opcode types that do not use addresses, all of the "dummy" address bits must be clocked anyway (the combined opcode/address field is constant number of clock cycles).
4. On read operations the data out stream starts with a dummy zero. On NM93Cxx family EEPROMs, it is acceptable but not required to have extra clocks after the 16th actual data bit. On NM93CSxx family EEPROMs, extra clocks after the 16th actual data bit will begin to read the next data word.

Notes on the Assembly Code:

The subroutines that follow are adequate to quickly pilot the programmers task of addressing a serial EEPROM of the NM93Cxx family. Additional subroutines can very easily be adapted from these to handle the additional opcode types of the NM93CSxx series parts. Enough code has been included to allow the code to operate in a stand-alone fashion. However, when integrating the routines in to another program, initialization statements affecting global variables such as initializing the stack point or the X or B registers will need to be moved, deleted or replaced by statements in the main program.

The assembly code uses a software timer loop to time out the write time of the EEPROM. The programmer should be aware that it is possible to use the EEPROMs own internal timer to accomplish this task. This is done by monitoring the EEPROMs DO line after taking the EEPROMs CS line low to start a write and then setting CS high again to re-enable the DO output. The write is complete when the DO (of the EEPROM) drives high. Using the EEPROMs internal timer will allows the microcontroller time to accomplish some other task in the 10 ms that the write or erase operation requires. If the DO line is to be used to indicate that the write is complete, other MICROWIRE components on the bus must wait for the EEPROM writes to time out before being accessed (the DO line is in use).

The code was tested on a COP820 device via a Metalink In Circuit Emulator. The code should translate to other COP8 devices with little or no modification.

NM93C06 and NM93C46 Opcodes and Address Fields*

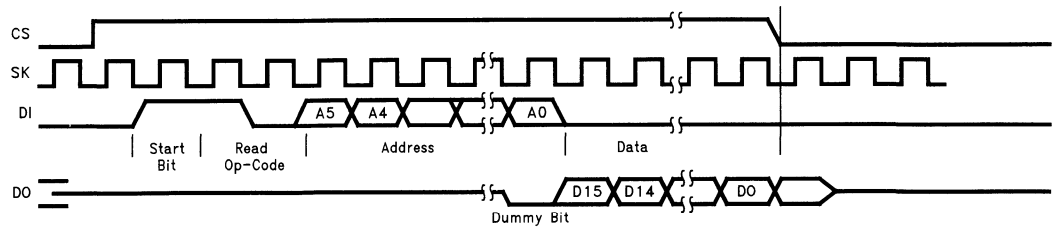
WREN	Write Enable	0	0	1	1	X	X	X	X
WRDI	Write Disable	0	0	0	0	X	X	X	X
ERAL	Erase All	0	0	1	0	X	X	X	X
WRAL	Write All	0	0	0	1	X	X	X	X
READ	Read	1	0	A5	A4	A3	A2	A1	A0
WRITE	Write	0	1	A5	A4	A3	A2	A1	A0

NM93C56 and NM93C66 Opcodes and Address Fields*

WREN	Write Enable	0	0	1	1	X	X	X	X	X	X
WRDI	Write Disable	0	0	0	0	X	X	X	X	X	X
ERAL	Erase All	0	0	1	0	X	X	X	X	X	X
WRAL	Write All	0	0	0	1	X	X	X	X	X	X
READ	Read	1	0	A7	A6	A5	A4	A3	A2	A1	A0
WRITE	Write	0	1	A7	A6	A5	A4	A3	A2	A1	A0

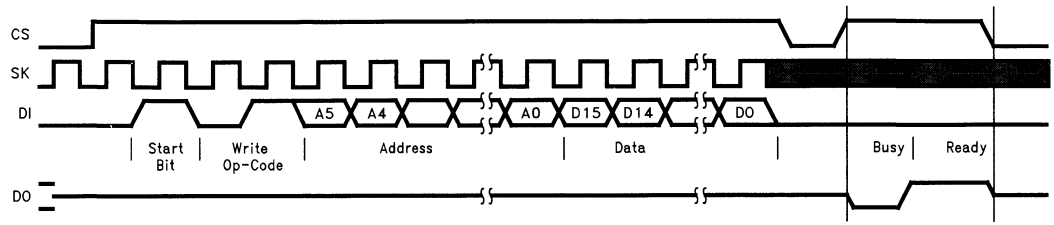
***Note:** All Opcode/Address Fields must be preceded with a leading "1" as a start-bit.

Read Cycle



TL/D/11491–3

Write Cycle



TL/D/11491–4

FIGURE 3. Read and Write cycle waveforms. Notice that one leading zero is shown before the start-bit. The actual code inserts seven.

```
;*********************************************************
; THIS PROGRAM PROVIDES SUBROUTINES TO HANDLE COP820 OPERATIONS ON
; THE NM93C06 EEPROM I.E., WRITES, READS, ERASES, ENABLES AND DISABLES
;*********************************************************
;
 .INCLD COP820.INC
;Reserving RAM locations for key variables
 RDATL = 1   ;LOWER BYTE OF THE NM93C06 MEMORY DATA READ
 RDATH = 2   ;UPPER BYTE OF THE NM93C06 MEMORY DATA READ
 WDATL = 3   ;LOWER BYTE OF THE DATA TO BE WRITTEN TO NM93C06
 WDATH = 4   ;UPPER BYTE OF THE DATA TO BE WRITTEN TO NM93C06
 ADRESS = 5  ;THE LOWER 4-BITS OF THIS LOCATION CONTAINS THE ADDRESS
             ;OF THE NM93C06 MEMORY LOCATIONS TO BE READ/WRITTEN
             ;THE UPPER NIBBLE MUST BE ZEROS
 SNDBUF = 0  ;USED FOR THE COMMAND BYTE TO BE WRITTEN (Local Scratch Pad)
 DLYH = 0F0  ;LOCATIONS RESERVED FOR WRITE TIMEOUT VALUES
 DLYL = 0F1
 FLAGS = 6   ;USED FOR PROGRAM FLAGS (Local Scratch Pad)
             ;
             ;FLAG VALUE DEFINITIONS
             ;00         ERASE, ENABLE, DISABLE, ERASE ALL
             ;01         READ CONTENTS OF NM93C06 REGISTER
             ;03         WRITE TO NM93C06 REGISTER
             ;OTHERS     ILLEGAL COMBINATION
```

```
;THE INTERFACE BETWEEN THE COP820C/840C AND THE NM93C06 (256-BIT EEPROM)
;CONSISTS OF FOUR LINES. THE GO(CHIP SELECT LINE), G4(SERIAL OUT SO),
;G5(SERIAL CLOCK SK) AND G6(SERIAL IN SI).
;****************************************
;
;INITIALIZATION, MODIFY MOVE OR DELETE WHEN INTEGRATING INTO MAIN PROGRAM
                    ;USE ONLY IF SP WAS NOT PREVIOUSLY INITIALIZED
       LD  SP,#02F     ;INITIALIZE STACK POINTER
       LD  PORTGC,#031 ;SETUP GO, G4, G5 AS OUTPUT
       LD  PORTGD,#000 ;INITIALIZE G DATA REG TO ZERO
       LD  CNTRL,#008  ;ENABLE MSEL, SELECT MW RATE OF 2TC

       LD  X,#SIOR     ;SET THE X REGISTER TO POINT TO SIOR
       LD  B,#PSW      ;SET THE B REGISTER TO POINT TO PSW

;EXAMPLE SUBROUTINE CALLS ONLY, DO NOT INCLUDE IN FINAL CODE LOAD
;ADDRESS IN LOCATION "ADRESS" HIGH AND LOW BYTE TO BE WRITTEN INTO
;WDATH AND WDATL AND CALL THE SUBROUTINE,
       JSR EWEN
       JSR WRITE
       JSR EWDS
       JSR READ
DONE:  JP  DONE

;THIS ROUTINE ERASES THE MEMORY LOCATION POINTED TO BY THE ADDRESS
;CONTAINED IN THE LOCATION "ADRESS". THE LOWER NIBBLE OF THE VALUE
;IN THE LOCATION "ADRESS" IS THE NM93C06 REGISTER ADDRESS. THE UPPER
;NIBBLE SHOULD BE SET TO ZERO.
;

ERASE: LD  A,ADRESS
       OR  A,#0C0
       X   A,SNDBUF
       LD  FLAGS,#00
       JSR INIT
       RET
;
;THIS ROUTINE ENABLES PROGRAMMING THE NM93C06 (EWEN).
;
EWEN:  LD  SNDBUF,#030
       LD  FLAGS,#00
       JSR INIT
       RET
;
;THIS ROUTINE DISABLES PROGRAMMING OF NM93C06.
;
EWDS:  LD  SNDBUF,#00
       LD  FLAGS,#00
       JSR INIT
       RET
;
;THIS ROUTINE ERASES ALL REGISTERS OF NM93C06.
;
ERAL:  LD  SNDBUF,#020
       LD  FLAGS,#00
       JSR INIT
       RET
```

```
;
;THIS ROUTINE READS THE CONTENTS OF THE NM93C06 REGISTER. THE ADDRESS
;IS SPECIFIED IN THE LOWER NIBBLE OF LOCATION "ADRESS". THE UPPER
;NIBBLE SHOULD BE SET TO ZERO. THE 16-BIT CONTENTS OF NM93C06 REGISTER ARE
;STORED IN RDATL AND RDATH.
;
READ:  LD    A,ADRESS
       OR    A,#080
       X     A,SNDBUF
       LD    FLAGS,#01
       JSR   INIT
       RET
;
;THIS WRITES A 16-BIT VALUE STORED IN WDATL AND WDADTH TO THE EEPROM
;REGISTER WHOSE ADDRESS IS CONTAINED IN THE LOWER NIBBLE OF THE
;LOCATION "ADRESS". THE UPPER NIBBLE OF THE ADDRESS SHOULD BE SET TO ZERO.
;
WRITE: LD    A,ADRESS
       OR    A,#040
       X     A,SNDBUF
       LD    FLAGS,#03
       JSR   INIT
       RET
;
;THIS ROUTINE SENDS OUT THE START BIT AND COMMAND BYTE. IT ALSO
;DECIPHERS THE CONTENTS OF THE FLAG LOCATION AND MAKES A DECISION
;REGARDING WRITE, READ OR RETURN TO THE CALLING ROUTINE.
;
INIT:  SBIT  0,PORTGD     ;SET CHIP SELECT HIGH
       LD    SIOR,#001    ;LOAD SIOR WITH START BIT
       SBIT  BUSY,[B]     ;SEND OUT THE START BIT
PUNT1: IFBIT BUSY,[B]
       JP    PUNT1
       LD    A,SNDBUF
       X     A,[X]        ;LOAD SIOR WITH COMMAND BYTE
       SBIT  BUSY,[B]     ;SEND OUT COMMAND BYTE
PUNT2: IFBIT BUSY,[B]
       JP    PUNT2
       IFBIT 0,FLAGS      ;ANY FURTHER PROCESSING?
       JP    NOTDON       ;YES
       RBIT  0,PORTGD     ;NO, RESET CS AND RETURN
       RET
;
NOTDON:IFBIT 1,FLAGS      ;READ OR WRITE?
       JP    WR93C        ;JMP TO WRITE ROUTINE
       LD    SIOR,#000    ;NO READ NM93C06
       SBIT  BUSY,PSW     ;DUMMY CLOCK TO READ ZERO
       RBIT  BUSY,[B]
       SBIT  BUSY,[B]
PUNT3: IFBIT BUSY,[B]
       JP    PUNT3
       X     A,[X]
       SBIT  BUSY,[B]
       X     A,RDATH
```

```
PUNT4: IFBIT BUSY,[B]
       JP    PUNT4
       LD    A,[X]
       X     A,RDATL
       RBIT  0,PORTGD
       RET
WR93C: LD    A,WDATH
       X     A,[X]
       SBIT  BUSY,[B]
PUNT5: IFBIT BUSY,[B]
       JP    PUNT5
       LD    A,WDATL
       X     A,[X]
       SBIT  BUSY,[B]
PUNT6: IFBIT BUSY,[B]
       JP    PUNT6
       RBIT  0,PORTGD
       JSR   TOUT
       RET
;
;ROUTINE TO GENERATE DELAY FOR WRITE
;*************************************
;
TOUT:  LD    DLYH,#007    ;CHECK YOUR OSCILLATOR--PROCESSOR COMBINATION
                          ;TUNE FOR 10 MS DELAY
WAIT:  LD    DLYL,#OFF
WAIT1: DRSZ  DLYL
       JP    WAIT1
       DRSZ  DLYH
       JP    WAIT
       RET
       .END
```

Selling National's Write Protected "CS" Series EEPROMs to High Volume OEMs

National Semiconductor
Application Note 871
Robert Stodieck

ABSTRACT

National's 93CSxx family of serial EEPROMs offer sophisticated protection against accidental overwrite. Unfortunately, understanding why and when this protection is needed often requires an equally sophisticated design engineer. This application note sheds light on the causes of accidental overwrite and the solutions that the "CS" series provide for controlling the problem.

GETTING SOPHISTICATED

EEPROMs are an important refinement to digital electronic products of many types. As consumers become more sophisticated about features, the advantages of non-volatile and cost effective serial EEPROMs frequently become important. Digital systems from cordless phones to VCRs, and from keyless entry systems to thermostats, benefit from the ability to store small stashes of data that do not go away when the power goes down or the batteries die.

The designers first reflex is to grab the cheapest generic solution on the market and run with it. After all, serial EEPROMs are all really the same aren't they?

NO!

In the beginning National Semiconductor married its MICROWIRE™ serial bus to a small EEPROM and created a classic 8-pin device that has since been copied by dozens of makers and has become the industry standard. Its descendants are available now from National in densities from 256 bits to 16 kbits.

So buy the industry standard device (every one else does) and run with it right?

NO!

National Semiconductor's classic 8-pin device is clearly the correct choice for some, but not all systems, and not knowing the difference can be expensive. Designs that shine through prototyping and test may fail in the field.

Serial EEPROMs are connected to microprocessors, microcontrollers and ASICs. Under a variety of error conditions the controlling system can accidentally overwrite data in the EEPROM. Application environments vary widely as does the importance of the data the EEPROMs are used to retain. The bit of information retained in the non-volatile memory may control the shape of the graphic sprite used in a child's game or something of importance to a nuclear power plant. It is be to expected that there is a need for more than one type of EEPROM.

The economy of serial EEPROMs and their utility explains why they are often used in high volume consumer items. Quality here is paramount because a flaw in the design can lead to very expensive returns and even recalls.

WHAT'S THE PROBLEM?

Volatile DRAM and SRAM memory forget past mistakes on reset or power down. A malfunctioning computer or video game can usually be fixed by turning the unit off and back on again. In contrast, non-volatile memories retain data even after the power goes off. They also retain memory of past single event mistakes such as unintended write operations.

Accidental overwrites to EEPROM can be caused by power supply noise, electrical interface noise, errant software, or "crashing" microcontrollers and processors. Such an event may be rare indeed but needs to occur ONLY ONCE in a product's lifetime to cause a problem and . . . such events are not always so rare.

Overwrite problems are well understood in the parallel EEPROM world. Overwrite protection schemes have long been central to the parallel EEPROM's architecture. Parallel EEPROMs usually have V_{CC} voltage sensing circuitry that helps to prevent unintended write events on power-up or power-down and V_{CC} transients. In addition, fairly complex write protection algorithms are used to reduce the probability of accidental overwrite.

Serial EEPROMs are far less susceptible to accidental overwrites than are parallel devices. The long sequence of serial data required to trigger a write in a serial device makes many of the precautions used on parallel devices unnecessary. Random noise on power-up is not likely to imitate the 19 plus serial bit sequence required to initiate a write operation.

However, the difficult problem of overwrite protection during a processor "crash" remains. There are some applications where there is no definitive solution to this problem. However, in many other situations the write protect features of the NM93CSxx can limit or cleanly eliminate the problem.

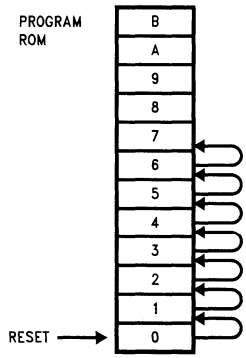
SO WHAT IS THE PROBLEM?

The fact that a long serial bit sequence is required to initiate a write operation into a serial non-volatile memory solves most accidental overwrite problems. But there is one persistent type of failure pathology that remains and can defeat this type of protection. It is particularly likely to occur in applications using small microcontrollers.

The problem is that no matter what input sequence is required to trigger a write operation, it has to be known to and be embedded in the software. The code segment that "knows" the write input sequence can be executed by unplanned jumps in the program sequencing induced by electrical noise, in practice usually V_{CC} noise or low V_{CC}. It can also be caused by imperfect software.

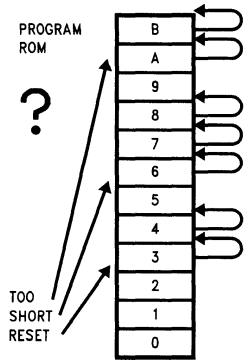
DOES THAT REALLY HAPPEN?

Yes. Understanding the problem requires a little reflection on how a processor works. A program counter is the device that generates addresses for program memory accesses. Remember that the program counter in a processor is basically a re-loadable binary counter. As long as V_{CC} is steady, and the clock and control signals to the control are noise free and within specified timing, the count (and program execution) normally advances one step at a time (see *Figure 1a*).



TL/D/11782–1

FIGURE 1a. Normal Program Flow after Power-On Reset



TL/D/11782–2

FIGURE 1b. Disturbed Reset Operation Following a Too Short Reset Pulse. This is a Common Scenario after a Short Interruption of V_{CC}.

If V_{CC} is not steady or is too low, and the clock or control signals are not noise free or are not within specified timings, the count may skip randomly. If V_{CC} or the offending control signals then return to normal, the count (and program execution) again begins to advance one increment at a time. But now the program is at a completely random location. If the program counter has skipped to the program segment that writes to the EEPROM, it will tend to write garbage to random locations. This is a particularly serious problem for microcontrollers which have relatively small program memory spaces. Thus, any skipping that occurs is more likely to overrun the serial write code.

THE SHORT RESET EXAMPLE

Normal processor reset is illustrated in *Figure 1a.* Faulty reset, possibly caused by short V_{CC} interruptions, is illustrated in *Figure 1b.* If the processor starts operation near the "WRITE_EEPROM" code segment it may trigger an accidental overwrite. It is difficult to prevent this type of misequencing in all cases because of the large spectrum of V_{CC} transients that may be experienced in the products lifetime. Some combination of low V_{CC} and V_{CC} interruption timing can usually be found that prevents the power up reset circuit from resetting correctly.

The slowly decaying V_{CC} level resulting from a discharging battery will also cause the program counter to begin skipping at some point. This is a common problem with the myriad of small electronic devices powered by unregulated batteries. Learning remote control units are a practical example of a system that uses EEPROM and suffers from this problem.

In large microprocessors with large memory spaces and various traps to stop errant operation, the possibility of accidental overwrite is not particularly high. But, in microcontrollers that may have only 1 kbyte to 2 kbytes of program code and few trapping techniques, this is a serious problem. Furthermore, in some applications conditions that can trigger this type of problem are repetitive and frequent.

FOR EXAMPLE

Room thermostats and refrigerators have historically used bi-metallic switches to maintain temperature control. If the line power dipped or dropped out there was no problem. Operating temperatures would soon return to normal when the power came back on. These kinds of controls are, of course, frequently being replaced with digital units, but these have a few problems. The temperature setting must be stored in battery backed-up RAM, or it must be stored in EEPROM. If not it will be volatile and lost after a power outage.

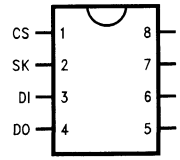
Serial EEPROM is by far the most desirable and cost effective way of providing non-volatility for this type of application, but care must be taken to guarantee data integrity in the EEPROM to prevent a change in temperature settings caused by an accidental overwrite.

Fortunately, the write security features of the NM93CSxx family of EEPROMs can not only guarantee security for the temperature data under the worst conditions, but at the same time has space for temperature calibration data, a serial number, and a date of manufacture, if required. Furthermore, the temperature calibration and serial number data can be rendered absolutely permanent. *Figure 3* shows how the write Protect Register of a NM93CSxx part can be used to protect a zone of memory from writes. This protection can be made permanent, if desired, by executing a one time only instruction that disables all writes to the protect register itself.

The write protect features are controlled by two new pin functions that replace two no connect pins on the standard MICROWIRE pinout *(Figure 3):* the Program Enable pin, PE, and the Protect Register Enable pin, PRE. The PE pin must be high to enable any write to the EEPROM. The PRE pin must be high to write to allow writes to the internal Write Protect Register.

Note: The words write and program are used interchangably.

Dual-In-Line Package (N)
Small Outline Gull Wing (M) (0.050" C-C)

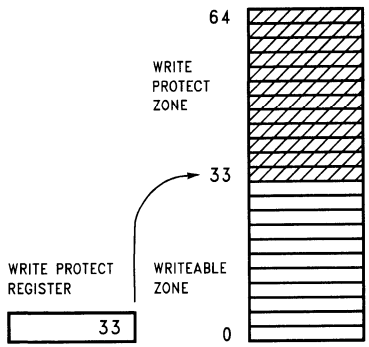


TL/D/11782–3

Top View

NM93CXX	NM93CSXX
V_{CC}	→
NC	PRE (Protect Register Enable)
NC	PE (Program Enable)
GND	→

FIGURE 2. The NM93CSxx Series Replaces No Connect Pins on the Standard Serial EEPROM Pinout with Two New Pins to Control Write Enable Functions.



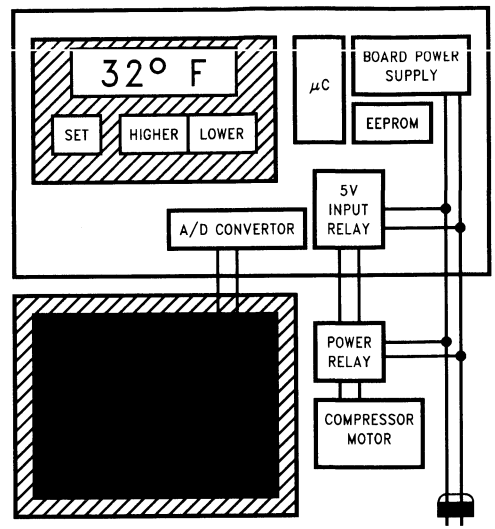
TL/D/11782–4

FIGURE 3. With the Write Protect Register Programmed to 33, EEPROM Locations 33 to the End of Memory are Not Writable.

Figure 4 is a block diagram of a refrigerator controller example. *Figure 5* shows how a factory-only accessible jumper can be used to control the logic level of the Protect Register Enable pin, and a user accessible push button switch can be used to control the logic level of the Program Enable pin.

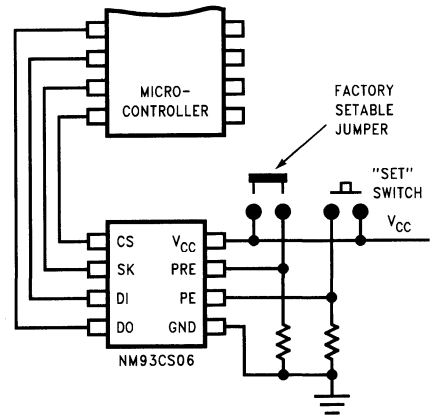
This arrangement allows the factory to enter and protect temperature calibration data in the EEPROM. The calibration data is located in the area protected by the protect register defined area. This data can only be altered by a trained technician who knows about the jumper.

At the same time, this arrangement allows the user to change the temperature settings which are located in the "writeable" area. Keep in mind though that this area is writeable only when the Program Enable (PE) pin is held high by the "SET" push button. The temperature setting is located in the "unprotected" area, where overwrites are controlled by using the PE pin. This example illustrates the use of two levels of write protection available with the "CS" series EEPROMs.



TL/D/11782–5

FIGURE 4. Refrigerator Temperature Controller. The Main Power Relay, Connected Directly to the Same power Line as the Controller, Cycles Off and On Frequently to Maintain Refrigerator Temperature, Creating an Ample Capacity for Generating Short Destructive V_{CC} Transients.



TL/D/11782–6

FIGURE 5. A Combination of a User-Accessible Switch, and a Factory Only Accessible Jumper, Allows Two Different Zones of Over-Write Protection in the Same EEPROM.

Most importantly though, the arrangement of the user accessible switch prevents accidental writes to the EEPROM from any source, and the write protection control is not software dependent. The EEPROM may not be altered in any way except when the user is actually pressing a front panel button labeled SET (*Figures 4* and *5*). This makes the probability of unintended overwrite vanishingly slight.

MORE EXAMPLES

Many contemporary applications have requirements similar to the refrigerator example. In all cases where an EEPROM is called for, data loss is expensive or even dangerous and the power supply cannot be guaranteed to be transient free. In these cases either the write protection required can be made permanent or a non-software dependent method is available to temporarily disable write protection when a data change is required, i.e., a switch or jumper allows the write protect pins to be toggled under controlled conditions. Alternatively, the part can be removed from the PC board for reprogramming.

AUTOMOTIVE APPLICATIONS

Automotive applications subject auto accessories to dead batteries, battery replacements, brown outs caused by cranking the engine on low batteries, etc. Thus, automotive applications are a prime environment for write protected serial EEPROMs. In the following applications and many others, hardware over-write protection is desirable and practical.

— **Auto Stereo Anti-Theft Codes**

The transients that the cassette deck's motors and relays generate in operation contribute to the normal automotive V_{CC} transients that the microcontroller must weather. Unless overwrite protection is provided for the EEPROM, one "crash" event could provoke an unintended overwrite and the deck may fail to work for its correct owner. Code changes are rare and overwrite protection can be controlled by a push button switch.

— **Automotive Engine Controls**

Engine control adjustments are retained in the EEPROM and must not be disturbed or the car may not operate correctly. Settings change only when the car is serviced and overwrite protection in this application can be controlled by the service equipment.

— **Automotive Keyless Entry Systems**

A keyless entry system must not forget its security codes and lock out its owner. Code changes are rare and overwrite protection can be controlled by a push button switch.

HOUSEHOLD APPLICATIONS

Household V_{CC} transients tend to come from brownouts, blackouts, and most commonly, simply plugging in or unplugging an appliance. Some contemporary applications include:

— Keyless Entry Systems

— Digital Room Thermostats

— Refrigerator Temperature Controls

TELECOM SWITCH APPLICATIONS

It is now a requirement that all boards found in common telecom switching systems have an electronically readable serial number. The serial number of any board in a switch must be readable from an operators console. This is an aid to repair. Reliability is a paramount consideration. With the NM93CSxx series serial EEPROMs the serial number can be recorded either permanently or alterably, changeable only by replacing the part in a socket or by a technician fitting a jumper during a rare, factory only, reprogramming event.

IN SUMMARY

The multiple write protect features of the NM93CSxx parts allow the parts to retain data such as serial numbers, date-of-manufacture codes, and calibration data, permanently and unalterably. The same part can provide application alterable EEPROM. In these applications one NM93CSxx series part takes the place of two devices. In other applications, just the fact that the data in the device can be rendered permanent may justify their use.

How to Use the NM93C86A Serial EEPROM as a PC/ Laptop Detachable Printer File Memory Card (DPFMC)

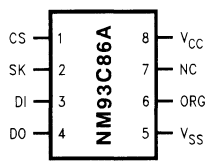
National Semiconductor
Application Note 936
Charles Watts

INTRODUCTION

This applications note describes how to build a DPFMC. The card will be designed around a COP888CG microcontroller and four NM93C86A serial EEPROMs. The card will be designed to plug into any standard IBM-PC/Laptop parallel port. Once the card has been installed, the user can download any document (text, graphic, or combination) and print out that document at a later time. The DPFMC will be designed to make the computer think a printer is actually connected by simulating the printer's input port. Once all documents have been sent, the user needs to press the SEND-DOC button once to save the pointer address. Next the user can remove the card and turn its power off. The documents contained in the card's EEPROMs can be stored for hours, days, months, or even years if needed. However, hours will probably be a more realistic time frame. When the user is ready to print-out the documents saved, the card can be plugged into a stand along printer by either using the printer's DB-25 cable or (with an appropriate adapter) the printer itself. After switching on the cards power, all the user has to do is press the SEND DOC button and the printer will begin to print out the saved documents.

NM93C86A DESCRIPTION

The NM93C86A is a 16,384-bit CMOS non-volatile serial EEPROM that can be configured to have a 1024 x 16 or a 2048 x 8 architecture. The configuration is determined by the state of the ORG pin. If the ORG pin is tied low the NM93C86A is configured as a 2048-byte-wide memory. If the ORG pin is left floating or tied to V_{CC}, the 1024 word wide configuration is enabled. An internal pull-up resistor to V_{CC} assures that a floating ORG pin is pulled high. *Figure 1* shows the NM93C86A pin arrangement.



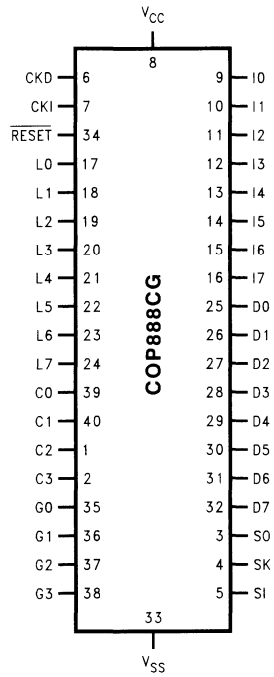
TL/D/12033–1

FIGURE 1. NM93C86A Pin Out

The NM93C86A has 7 instructions that can be performed. The instructions are: Read a byte/word (READ), Enable programming (EWEN), Disable programming (EWDS), Erase a byte/word (ERASE), Write a byte/word (WRITE), Erase all bytes/words (ERAL), and write a data pattern to all bytes/words (WALL). The NM93C86A uses the industry standard MICROWIRE™ interface.

COP888CG DESCRIPTION

The COP888CG is an 8-bit microcontroller. Its a fully static CMOS device containing RAM, ROM, and Microwire interface. The microcontroller contains 4,096 bytes of ROM used to store program code and 192 bytes of RAM used to store register data. It contains an 8-bit input, an 8-bit output, and two 8-bit bi-directional ports. The microcontroller also has a Microwire interface which will be used to connect the NM93C86A to it. These attributes make the COP888CG a good choice for this particular application.



TL/D/12033–2

FIGURE 2. COP888CG Microcontroller

CIRCUIT DESCRIPTION

Figure 3 shows the complete schematic diagram of the DPFMC. The L-port of the microcontroller is connected to the D0–D7 data lines of the computer's bidirectional data port. The LS-nibble of C-port is connected to the STROBE, ACKNLG, BUSY, and the PE pins of the computer's parallel port. Pin 13 is tied HIGH and all of the other pins are tied LOW. All of the bi-directional I/O lines are pulled HIGH through two 10 kΩ SIP resistors.

The MICROWIRE Interface of the COP888CG uses pins 3(SO), 4(SK), and 5(SI). These pins are connect to the equivalent pins of the EEPROM. The MS-nibble of the D-port is used as chip select outputs to the four EEPROMs. Since the data the DPFMC will be processing is 1-byte length data, the NM93C86A will be configured to deal with 8-bit chunks instead of 16-bit chunks of data.

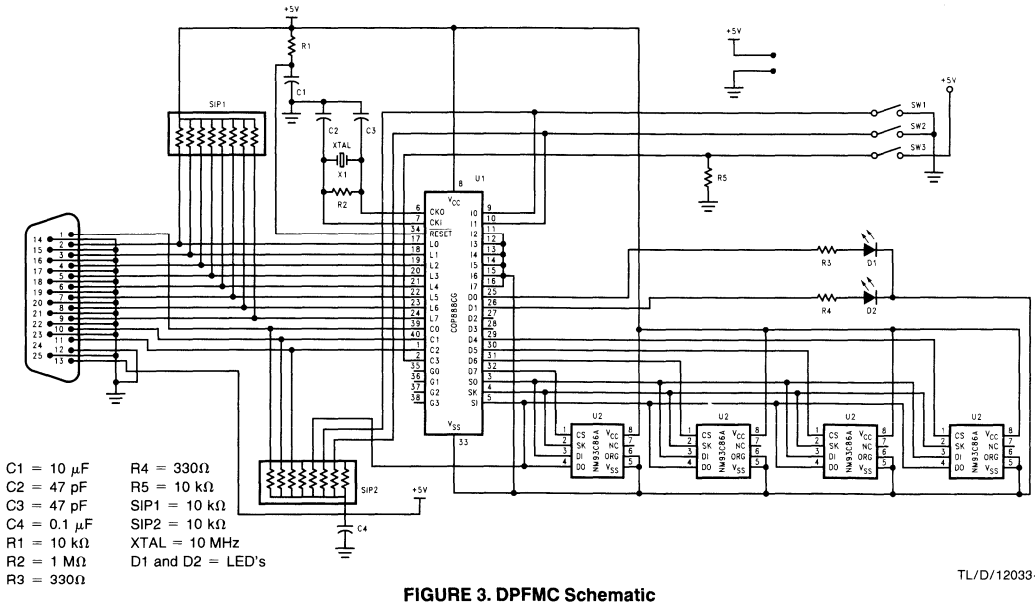


FIGURE 3. DPFMC Schematic

The DPFMC will have three inputs. Two are tied HIGH through 10 kΩ resistors. The third is tied LOW. The first input when LOW (pin 9) will indicate RECEIVE MODE. A LOW on the second input (pin 10) will indicate WRITE MODE. A HIGH on the third input (pin 11) will tell the microcontroller to save the current pointer address and begin sending the document data to the printer.

Their are also two LED outputs. LED1 will inform the user that the DPFMC cannot accept any more data. LED1 also informs the user that the pointer address has been saved. If LED1 flashes continuously the DPFMC is out of memory. If LED1 flashes irregular the DPFMC has saved the pointer address and is ready for power to be removed. LED2 informs the user of the current mode. Two flashes indicates the READ MODE and one flash indicates the WRITE MODE.

SOFTWARE DESCRIPTION

For simplicity and structure the software part of this application will be divided into three parts. The first or main block will monitor the three inputs and decide which mode the DPFMC will enter. The second block will control the interface logic between the computer's parallel port and the DPFMC.

In the case of a read operation, this block of code will start by configuring the STROBE and BUSY pins as inputs and the ACKNLG pin as an output. The routine then waits for the BUSY pin to fall LOW. When the BUSY pin falls low, the routine will begin monitoring the STROBE pin for a 0.5 μs LOW. When this happens, the routine reads the data port and stores that data into accumulator A. After the data is safely stored into accumulator A, the ACKNLG pin will be pulsed LOW for 5 μs to inform the computer that the data was received. Control is now passed to the final routine. This routine takes the data from ACC A stores the data into the EEPROMs. This routine basically will control the memory matrix part of the card. After the data is stored into the EEPROMs, control is passed back to the interface routine and the loop continues.

The write sequence will be just the opposite. The BUSY and STROBE pins are first configured to be outputs and the ACKNLG pin configured to be a input. The routine will then wait for the SEND DOC input to go low. When this happens the BUSY will go LOW to indicate the card is about to send a byte to the printer. The memory matrix routine the then stores the first byte of data into accumulator A. After that the interface routine sends that data to the port. The STROBE pin is pulsed LOW for 5 μs. From this point on the routine monitors the ACKNLG pin for a 5 μs LOW. When a LOW has occurred the routine loops back to the top, fetches the next byte, and starts a write loop.

```
; ASSEMBLY CODE FOe THE DETACHABLE PRINTER FILE MEMORY CARD(DPFMC)
; By Charles Watts
.INCLD COP888CG.INC
.SECT CODE,ROM,ABS=0
; -------------------- INITIALIZE PORT & REGISTER DATA ---------
DLYL    = 0F0
DLYH    = 0F1
ADDL    = 0F2
ADDH    = 0F3
BYT     = 00
HLD     = 01
STOLO   = 02
STOHI   = 03
STOHLD  = 04
;
START:  LD      PORTD, #00
        LD      PORTGC, #030
        SBIT    MSEL, CNTRL
        SBIT    S0,     CNTRL
        LD      PORTLC, #00
        LD      PORTCC, #0B
        LD      PORTCD, #02
        LD      B, #PORTCP
        JSR     LAST
; ------------------------- MAIN ROUTINE ------------------
MAIN:   LD      A, PORTI        ;
        IFEQ    A, #06          ;
        JSR     READ            ;
        IFEQ    A, #05          ;
        JSR     WRITE           ;
        JP      MAIN            ;
; ------------------------ SUBROUTINES WILL FOLLOW --------
READ:   LD      PORTLC, #00     ;CONFIGURE PORT
        LD      PORTCC, #06     ;TO READ MODE
        LD      PORTCD, #02     ;
        JSR     FLSH2           ;BLINK LED 2 TIMES
        JSR     EWEN            ;
LP1:    IFBIT 3, [B]            ;
        JSR     SAVE            ;
        IFBIT 0, [B]            ;WAIT FOR STROBE TO GO LOW
        JP      LP1             ;
        LD      PORTCD, #06     ;BRING BUSY HIGH
        LD      A, PORTLP       ;READ PORT AND SAVE IN ACCA
        X       A, BYT          ;
        JSR     PUT             ;STORE ACCA IN NVM
        LD      PORTCD, #04     ;PULSE ACKNLG
        NOP                     ;
        NOP                     ;
        LD      PORTCD, #06     ;
        NOP                     ;
        NOP                     ;
        LD      PORTCD, #02     ;
        JP      LP1             ;
;
WRITE:  LD      PORTLC, #0FF    ;CONFIGURE PORT
        LD      PORTCC, #01     ;TO WRITE MODE
        LD      PORTCD, #01     ;
        JSR     FLSH1           ;
LP2:    IFBIT 3, [B]            ;
        JP      LP3             ;
```

TL/D/12033–4

```
        JP    LP2           ;
LP3:    IFBIT 2, [B]        ;WAIT FOR BUSY LOW
        JP    LP3           ;
        JSR   GET           ;GET BYTE FROM
        LD    A, BYT        ;
        X     A, PORTLD     ;NVM.
        NOP                 ;
        NOP                 ;
        LD    PORTCD, #00   ;PULSE STROBE
        NOP                 ;
        NOP                 ;
        LD    PORTCD, #01   ;
        NOP                 ;
        NOP                 ;
        JP    LP3           ;
;
GET:    LD    A, HLD        ;SET CS HIGH
        X     A, PORTD      ;
        LD    A, ADDH       ;SEND OPCOCE AND
        OR    A, #030       ;HI ADDRESS
        X     A, SIOR       ;
        SBIT  BUSY, PSW     ;
LP4:    IFBIT BUSY, PSW     ;
        JP    LP4           ;
        LD    A, ADDL       ;SEND LOW ADD
        X     A, SIOR       ;
        SBIT  BUSY, PSW     ;
LP5:    IFBIT BUSY, PSW     ;
        JP    LP5           ;
        LD    SIOR, #000    ;
        SBIT  BUSY, PSW     ;RECEIVE BYTE
        RBIT  BUSY, PSW     ;
        SBIT  BUSY, PSW     ;
LP6:    IFBIT BUSY, PSW     ;
        JP    LP6           ;
        X     A, SIOR       ;
        X     A, BYT        ;
        LD    PORTD, #00    ;
        JSR   COUNT         ;
        LD    A, HLD        ;
        IFEQ  A, STOHLD     ;
        JP    SKIP1         ;
        RET                 ;
SKIP1:  LD    A, ADDL       ;
        IFEQ  A, STOLO      ;
        JP    SKIP2         ;
        RET                 ;
SKIP2:  LD    A, ADDH       ;
        IFEQ  A, STOHI      ;
        JP    ZD            ;
        RET                 ;
;
PUT:    LD    A, HLD        ;SET CS HIGH
        X     A, PORTD      ;
        LD    A, ADDH       ;SEND OPPCOCE AND
        OR    A, #028       ;HI ADDRESS
        X     A, SIOR       ;
        SBIT  BUSY, PSW     ;
LP7:    IFBIT BUSY, PSW     ;
        JP    LP7           ;
```

TL/D/12033–5

```
        LD    A, ADDL       ;SEND LOW ADD
        X     A, SIOR       ;
        SBIT  BUSY, PSW     ;
LP8:    IFBIT BUSY, PSW     ;
        JP    LP8           ;
        LD    A, BYT        ;SEND BYTE
        X     A, SIOR       ;
        SBIT  BUSY, PSW     ;
LP9:    IFBIT BUSY, PSW     ;
        JP    LP9           ;
LP10:   IFBIT SI, PORTGP    ;
        JP    LP10          ;
LP11:   IFBIT SI, PORTGP    ;
        JP    LP12          ;
        JP    LP11          ;
LP12:   LD    PORTD, #00    ;
        LD    A, HLD        ;
        X     A, PORTD      ;
        LD    SIOR, #0FF    ;
        SBIT  BUSY, PSW     ;
        RBIT  BUSY, PSW     ;
        LD    PORTD, #00    ;
        JSR   COUNT         ;
        RET                 ;
;
EWEN:   LD    PORTD, #0F0   ; Enable EE
        LD    SIOR, #026    ;
        SBIT  BUSY, PSW     ;
LP13:   IFBIT BUSY, PSW     ;
        JP    LP13          ;
        LD    SIOR, #00     ;
        SBIT  BUSY, PSW     ;
LP14:   IFBIT BUSY, PSW     ;
        JP    LP14          ;
        LD    PORTD, #00    ;
        RET                 ;
;
COUNT:  LD    A, ADDL       ; Address Counter
        IFEQ  A, #0FF       ;
        JP    ZA            ;
        INC   A             ;
        X     A, ADDL       ;
        RET                 ;
ZA:     LD    A, ADDH       ;
        IFEQ  A, #07        ;
        JP    ZC            ;
        INC   A             ;
        X     A, ADDH       ;
        LD    ADDL, #00     ;
        RET                 ;
ZC:     LD    A, HLD        ;
        IFEQ  A, #080       ;
        JP    ZE            ;
        LD    A, HLD        ;
        ADD   A, HLD        ;
        X     A, HLD        ;
        LD    ADDL, #00     ;
        LD    ADDH, #00     ;
        RET                 ;
ZE:     JSR   SAVE          ;
```

TL/D/12033–6

```
ZD:     LD    PORTD, #02     ;
        JSR   DELAY          ;
        LD    PORTD, #00     ;
        JSR   DELAY          ;
        JP    ZD             ;
;
SAVE:   LD    A, ADDL        ; Save Pointer
        X     A, STOLO       ;
        LD    ADDL, #0FD     ;
        LD    A, ADDH        ;
        X     A, STOHI       ;
        LD    ADDH, #07      ;
        LD    A, HLD         ;
        X     A, STOHLD      ;
        LD    HLD, #084      ;
        LD    A, STOLO       ;
        X     A, BYT         ;
        JSR   PUT            ;
        LD    A, STOHI       ;
        X     A, BYT         ;
        JSR   PUT            ;
        LD    A, STOHLD      ;
        X     A, BYT         ;
        JSR   PUT            ;
        IFBIT 3, [B]         ;
        JSR    GOTIT         ;
        RET

GOTIT:  LD    PORTD, #02     ; LED Flashing sequence
        JSR   DELAY          ;
        LD    PORTD, #00     ;
        JSR   DELAY          ;
        LD    PORTD, #02     ;
        JSR   DELAY          ;
        LD    PORTD, #00     ;
        JSR   DELAY          ;
        JSR   DELAY          ;
        JSR   DELAY          ;
        JP    GOTIT          ;
;
LAST:   LD    ADDL, #0FD     ; Get pointer
        LD    ADDH, #07      ;
        LD    HLD, #084      ;
        JSR   GET            ;
        LD    A, BYT         ;
        X     A, STOLO       ;
        JSR   GET            ;
        LD    A, BYT         ;
        X     A, STOHI       ;
        JSR   GET            ;
        LD    A, BYT         ;
        X     A, STOHLD      ;
        LD    ADDL, #00      ;
        LD    ADDH, #00      ;
        LD    HLD, #010      ;
        RET                  ;
;
FLSH2:  LD     PORTD, #001   ;FLASH LED 2 TIMES
        JSR    DELAY         ;
        LD     PORTD, #00    ;
```

TL/D/12033-7

```
        JSR   DELAY          ;
        LD    PORTD, #001    ;
        JSR   DELAY          ;
        LD    PORTD, #00     ;
        RET                  ;
;
FLSH1:  LD    PORTD, #001    ;FLASH LED ONCE
        JSR   DELAY          ;
        LD    PORTD, #00     ;
        JSR   DELAY          ;
        RET                  ;
;
DELAY:  LD   DLYH, #040      ;
LP15:   LD   DLYL, #0FF      ;
LP16:   DRSZ DLYL            ;
        JP   LP16            ;
        DRSZ DLYH            ;
        JP   LP15            ;
        RET                  ;
        .END  START          ;
```

TL/D/12033–8

Low Cost A/D Conversion Using COP800

National Semiconductor
Application Note 952
Robert Weiss

INTRODUCTION

Many microcontroller applications require a low cost analog to digital conversion. In most cases the controller applications do not need high accuracy and short conversion time.

This appnote describes a simple method for performing analog to digital conversion by reducing external elements and costs.

PRINCIPLE OF A/D CONVERSION

The principle of the single slope conversion technique is to measure the time it takes for the RC network to charge up to the threshold level on the port pin, by using Timer T1 in the input capture mode. The cycle count obtained in Timer T1 can be converted into voltage, either by direct calculation or by using a suitable approximation.

Figure 1 shows the block diagram for the simple A/D conversion which measures the temperature.

BASIC CIRCUIT IMPLEMENTATION

Usually most applications use a comparator to measure the time it takes for a RC network to charge up to the voltage level on the comparator input. To reduce cost, it is possible to switch both inputs as shown in *Figure 2*.

Port G3 is the Timer T1 input. Ports G2/G1 are general purpose I/O pins that can be configurated using the I/O configurations (push-pull output/tristate). All Port G pins are Schmitt Trigger inputs. R_{LIM} is required to reduce the discharge current.

GENERAL IMPLEMENTATION

The temperature is measured with a NTC which is linearized with a parallel resistor. Using a parallel resistor, a linearization in the range of 100 Kelvin can be reached. The value of the resistor can be calculated as follow:

$$R_P = R_{tm} * (B - 2T_m)/(B + 2T_m)$$

R_{tm} Value of the NTC at a medium temperature
T_m Medium Temperature
B NTC–material constant

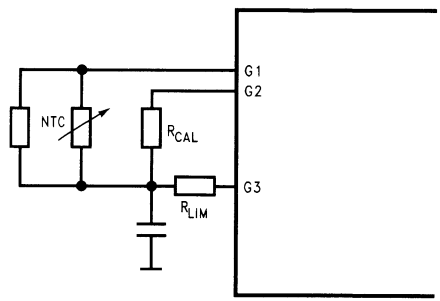
The linearization reduces the code, improves the accuracy and the tolerance of the NTC-R network (e.g. NTC = 100 kΩ ±10%, R = 12 kΩ ±1%, NTC//R ±2%). Using that method the useful range does not cover the whole operating temperature range of the NTC.

GENERAL ACCURACY CONSIDERATIONS

Using a single slope A/D conversion the accuracy is dependent on the following parameters:

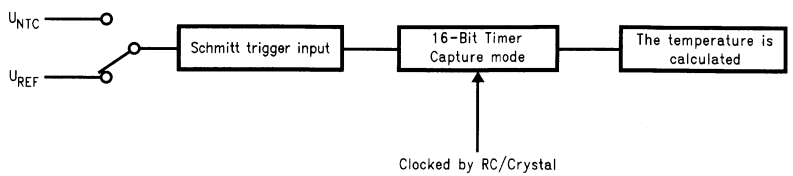
— Stability of the Clock frequency
— Time constant of the RC network
— Accuracy of the Schmitt Trigger level
— Non-linearity of the RC-network

Figure 3. The maximum failure that appears when a sawtooth is generated without using a current source. In the current application the maximum failure would be more than 15% without using methods for reducing the non-linearities of RC-network/NTC-network.



TL/DD/12075–2

FIGURE 2. Basic Circuit Implementation



TL/DD/12075–1

FIGURE 1. Simple A/D Conversion

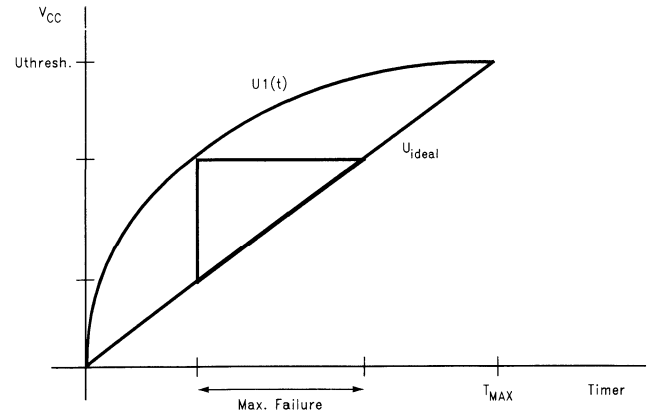


FIGURE 3. Single Slope A/D Conversion

The maximum error occurs when the gradient of the exponential function (RC) equals the gradient of the straight line (counter).

To reduce the error that is caused by the non-linearity of the RC-network a offset should be added to the calculated value. The offset reduce the failure to the middle.

Further, the accuracy can be improved by using a relative measurement method. The following diagram shows the method.

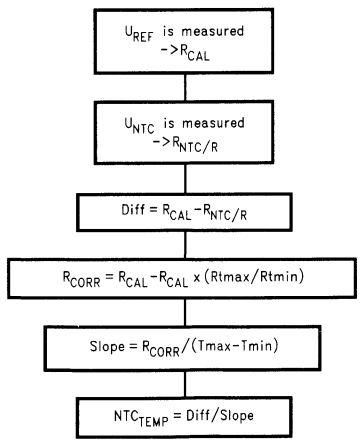


FIGURE 4. Accuracy Improvement

Measurement:

- — Timer Capture mode: R_{CAL} * C is measured
- — Timer Capture mode: $R_{NTC//R}$ * C is measured

Calculation:

- — Build the vertical-component (R_{TMIN} − R_{TMAX}) of the triangle
- — Calculate the slope
- — Calculate the actual temperature

Using this method the accuracy is primarily dependent on the accuracy of R_{TMIN} and R_{TMAX} and independent of the stability of the system clock, the capacitor and the threshold of the Schmitt Trigger level. The variation of the capacitor only leads to variation of the resolution.

The following diagram shows the ideal resistance/temperature characteristic of a NTC which is linearized with a parallel resistor.

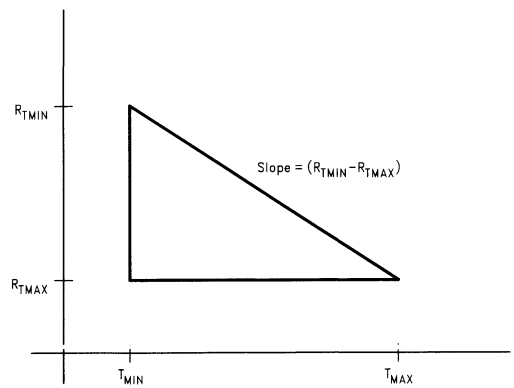


FIGURE 5. Resistance vs Temperature Characteristics

APPLICATION EXAMPLE

The following application example for temperature measurement demonstrates the procedure. The temperature is measured from 20° to 100° and is displayed on a Triplex LCD display.

NTC_{20} = 100 kΩ ±10%
R_P = 12 kΩ ±1%
T_m = 333 Kelvin → 60 Degrees
B = 4800 Kelvin
$NTC_{20}//R_P$ = 10.7 kΩ ±2%
R_{CAL} = 10.7 kΩ ±1%
T_{MIN} = 20 Degree
R_{TMIN} = 10.7 kΩ
T_{MAX} = 100 Degree
R_{TMAX} = 2.8 kΩ
C = 1 µF
RC-Clock = 2 MHz → 200 kHz instruction cycle, 5 µs
Timeconst. = R_{CAL} * C → 0.0107s
Resolution = 2140 → 11 byte, depends which Cap. value is used
Accuracy = ±2 Degree

This temperature measurment example shows a low cost technique ideally suited for cost sensitive applications which do not need high accuracy.

Figure 6 shows the complete circuit of the demoboard using the Triplex LCD method and the low cost A/D conversion technique.

The Triplex LCD drive technique is documented in a separate application note.

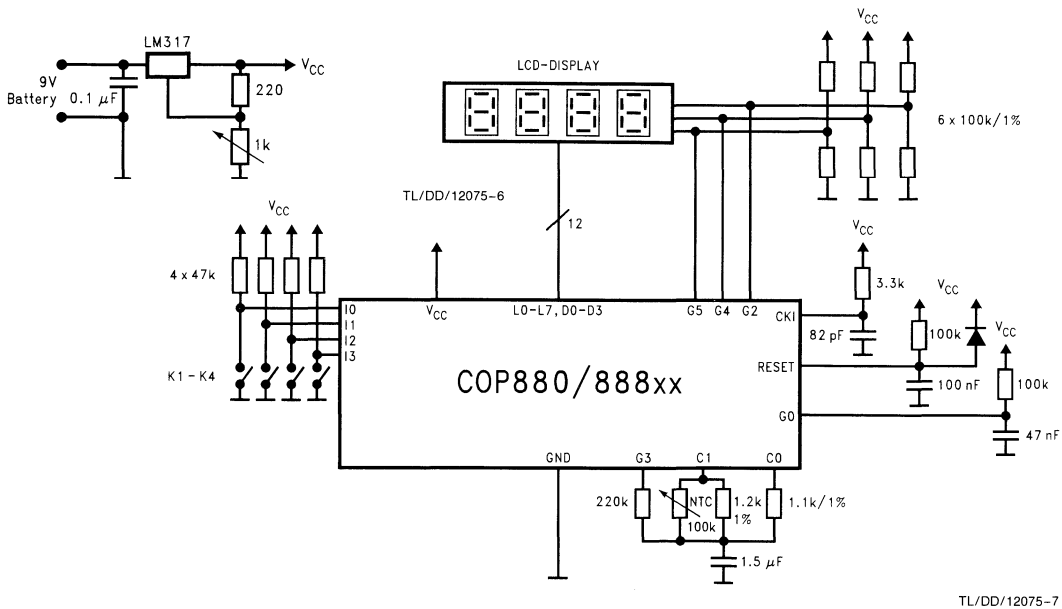
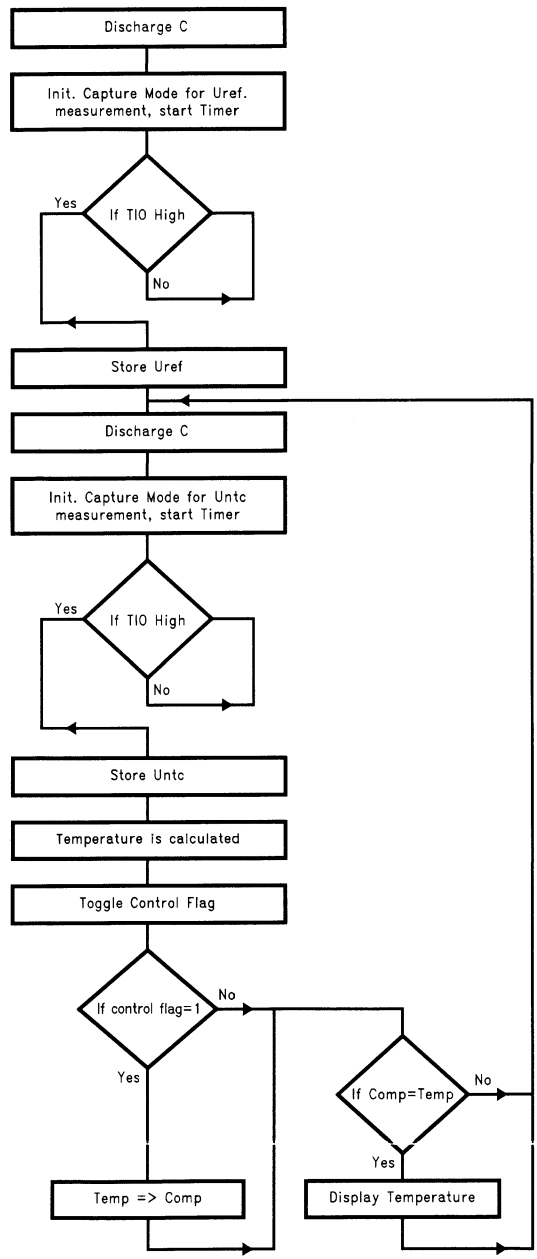


FIGURE 6. Circuit Diagram

Pressing key 1, key 2 the temperature is displayed in Degree/Fahrenheit.
Pressing key 3, key 4 Up/Down counter is displayed.

SOURCE CODE

Figure 7 shows the flow chart of the program.



TL/DD/12075–8

FIGURE 7. Flow Chart

The following code is required to implement the function. It does not include the code for the Triplex LCD drive.

```
RAM = 17 Byte;
ROM = 450 Byte; Optimization is possible about 50 byte if the B - pointer consistent is used!
;*****************************A/D-CONVERSION************************************
;
;
;*****************************VAR. DECLARIATION*********************************
.SECT REGPAGE,REG
COUNT1:     .DSB 1
COUNT2:     .DSB 1
;
.SECT BASEPAGE,BASE
ZL:         .DSB 1      ;TEMPORARY
YL:         .DSB 1      ;TEMPORARY
;
.SECT RAMPAGE,RAM
CALIBLO:    .DSB 1      ;CALIBRATION-VALUE
CALIBHI:    .DSB 1
NTCLO:      .DSB 1      ;NTC-VALUE
NTCHI:      .DSB 1
TEMP:       .DSB 2      ;TEMP.-VALUE
KORRL:      .DSB 2
COMPL:      .DSB 1
COMPH:      .DSB 1
CONTROL:    .DSB 1      ;STATUS REGISTER
;*********************************START MAIN PROGRAM*****************************
MAIN: LD    SP,#06F     ;INIT SPACKPOINTER
      JSR   DISCH       ;DISCHARGE C (A/D-CONVERSION)
      JSR   CALB        ;INIT CAPTURE MODE FOR UREF. MEASURMENT
POLL: IFBIT 3,PORTGP    ;POLL - MODE (TIO - PORT)
      JP    CAL
      JP    POLL
CAL:  LD    B,#CALIBLO
      JSR   CAPTH       ;STOP TIMER, STORE CAPTURE VALUE
      JSR   CALCR       ;SLOPE IS CALCULATED
NEW:  JSR   DISCH       ;DISCHARGE C (A/D-CONVERSION)
      JSR   NTC         ;INIT CAPTURE MODEFOR UNTC MEASURMENT
POLL1: IFBIT 3,PORTGP   ;POLL-MODE
      JP    CAL1
      JP    POLL1
CAL1: LD    B,#NTCLO
      JSR   CAPTH       ;STOP TIMER, STORE CAPTURE VALUE
      JSR   CALCN       ;TEMPERATURE IS CALCULATED
      JSR   DISCH       ;DISCHARGE C (A/D-CONVERSION)
      JSR   DCHECK      ;REDUCE THE DISPLAY FLICKERING
      JMP   NEW
.ENDSECT
;*****************************************************************************
```

TL/DD/12075–9

```
;*************************************************************************
.SECT CODE1,ROM
;THIS ROUTINE IS REQUIRED TO REDUCE THE NOICE ON THE LINE AND THE
; DISPLAY FLICKERING.
.SECT CODE1,ROM
DCHECK:                 ;COMPARE TWO VALUES, IF EQUAL THEN
        LD   A,CONTROL  ;DISPLAY IT, OTHERWISE THE OLD VALUE
        XOR  A,#080     ;IS DISPLAYED
        X    A,CONTROL
        IFBIT 7,CONTROL
        JSR  SAVE       ;TEMP. SAVE
        JSR  COMP       ;COMPARE
        RET
;*************************************************************************
; HANDLER FOR CAPTURE MODE
CAPTH: RBIT TPND,PSW    ;RESET TIMER PENDING
        RBIT TRUN,PSW   ;STOP TIMER
        LD  A,#0FF
        SC
        SUBC A,TAULO
        X   A,[B+]      ;STORE THE CAPTURED VALUE
        LD  A,#0FF
        SUBC A,TAUHI
        X   A,[B+]      ;STORE THE CAPTURED VALUE
        RET
;*************************************************************************
; CALIBRATION SUBROUTINE, UREF IS MEASURED
CALB:
        RBIT 3,PORTGD
        RBIT 3,PORTGC   ;TRISTATE TIO
        LD PORTCD,#00
        LD PORTCC,#00   ;TRISTATE PORT C
        T1CAP HIGH      ;INIT CAPTURE MODE, HIGH SENSITIVE (MACRO)
        LD B,#CALIBLO
        SBIT 0,PORTCD   ;CONFIGURE C0 TO OUTPUT HIGH
        SBIT 0,PORTCC   ;CHARGE CAP.
        SBIT TRUN,CNTRL ;START TIMER CAPTURE MODE
        RET
;*************************************************************************
; NTC SUBROUTINE, UNTC IS MEASURED
NTC:
        RBIT 3,PORTGD
        RBIT 3,PORTGC   ;TRISTAT TIO
        LD PORTCD,#00
        LD PORTCC,#00   ;TRISTATE PORT C
        T1CAP HIGH      ;INIT CAPTURE MODE, HIGH SENSITIVE (MACRO)
        LD B,#NTCLO
        SBIT 1,PORTCD   ;CONFIGURE C1 TO OUTPUT HIGH
        SBIT 1,PORTCC   ;CHARGE CAP.
        SBIT TRUN,CNTRL ;START TIMER CAPTURE MODE
        RET
;*************************************************************************
```

TL/DD/12075-10

```
;*************************************************************************
;DISCHARGE - ROUTINE
DISCH:
        LD PORTCD,#000
        LD PORTCC,#000
        RBIT TIO,PORTGD     ;DISCHARGE CAP.
        SBIT TIO,PORTGC
        LD COUNT1,#H(500)  ;DISCHARGE TIME
        LD COUNT2,#L(500)
        JSR C1              ;DELAY ROUTINE FOR DISCHARGE TIME
        RET
;*************************************************************************
;THIS SUBROUTINE CALCULATES THE SLOPE
;THE FOLLOWING CALCULATIONS ARE DONE
;KORR=CALIB/11KOHM (RCALIB.=11KOHM)
;KORR=KORR*2,8KOHM (T=100 DEGREE, RNTC=2,8KOHM)
;CALIB=CALIB-KORR
;DIV=CALIB\80    (TEMPRANGE=80 DEGREE,100-20), SLOPE IS CALCULATED
CALCR:
;KORR=CALIB/11KOHM
        LD ZL,#L(110)
        LD ZL+1,#H(110)
        LD A,CALIBLO
        X  A,YL
        LD A,CALIBHI
        X  A,YL+1
        JSR DIVBIN16        ;SUBROUTINE BINARY DIVIDE 16 BIT BY 16 BIT
        LD A,YL
        X  A,KORRL
;*************************************************************************
;KORR=KORR*28
        LD A,KORRL
        X  A,ZL
        LD A,#28
        X  A,YL
        JSR MULBIN8         ;SUBROUTINE MULTIPLY TWO 8 BIT VALUES
        LD A,YL
        X  A,KORRL
        LD A,YL+1
        X  A,KORRL+1
;*************************************************************************
;KORR=CALIB-KORR
        LD B,#CALIBLO
        LD A,[B+]
        SC
        SUBC A,KORRL
        X A,KORRL
        LD A,[B]
```

TL/DD/12075–11

```
        SUBC A,KORRL+1
        X A,KORRL+1
;*****************************************************************************
;DIV=KORR/80
        LD ZL,#L(80)
        LD ZL+1,#H(80)
        LD A,KORRL
        X  A,YL
        LD A,KORRL+1
        X  A,YL+1
        JSR DIVBIN16        ;SUBROUTINE BINARY DIVIDE 16 BIT BY 16 BIT
        LD A,YL
        X  A,DIV
        RET
;*****************************************************************************
;THIS SUBROUTINE CALCULATES THE TEMPERATURE
;THE FOLLOWING CALCULATIONS ARE DONE
;TEMP=CALIB-NTC
;TEMP=TEMP/DIV
;ADD OFFSET 20 DEGREE
;CONVERSION FROM HEX TO BCD
;*****************************************************************************
;TEMP=CALIB-NTC
CALCN: LD B,#CALIBLO
        LD A,[B+]
        SC
        SUBC A,NTCLO
        X  A,TEMP
        LD A,[B]
        SUBC A,NTCHI
        IFNC
        JMP ERR
        X  A,TEMP+1
;*****************************************************************************
;TEMP=TEMP/DIV
        LD A,TEMP
        X  A,YL
        LD A,TEMP+1
        X  A,YL+1
        LD A,DIV
        X  A,ZL
        CLRA
        X  A,ZL+1
        JSR DIVBIN16        ;SUBROUTINE BINARY DIVIDE 16 BIT BY 16 BIT
        LD A,YL
        ADD A,#20           ;ADD TEMPERATURE OFFSET
        IFGT A,#56          ;IF TEMPERATURE IS HIGER THAN 56 DEGREE THEN
        JSR CORR            ;ADD CORRECTION. OFFSET
;*****************************************************************************
```

TL/DD/12075–12

```
;*************************************************************************
;HEX TO BCD CONVERSION
        X  A,ZL
        LD A,ZL
        IFGT A,#100          ;IF TEMPERATURE IS MORE THAN 100 DEGREE THEN
        JP  ERR              ;ERROR
        JSR BINBCD           ;SUBROUTINE BINARY TO BCD CONVERSION;
        LD A,BCDLO
        X  A,TEMP
        LD A,BCDLO+1
        X  A,TEMP+1
        RET
ERR:    LD  A,#00E           ;ERROR MESSAGE IS DISPLAYED
        X  A,TEMP
        CLR A
        X  A,TEMP+1
        RET
;*************************************************************************
COMP:LD     A,COMPL     ;IF THE LAST BOTH MEASURMENTS ARE EQUAL
        SC              ;THEN DISPLAY
        SUBC  A,TEMP
        IFEQ  A,#0
        JP    DISPLAY
        RET             ;OTHERWISE DISPLAY THE OLD VALUE
DISPLAY:LD  A,TEMP
        X     A,PB+2
        LD    A,TEMP+1
M1:     X     A,PB+3
        JSR   LCDDR     ;UPDATE THE DISPLAY
        JSR   DEL       ;DELAY TIME
        RET
;*************************************************************************
SAVE: LD    A,TEMP      ;TEMPORARY SAVE
        X     A,COMPL
        LD    A,TEMP+1
        X     A,COMPH
        RET
;*************************************************************************
```

TL/DD/12075–13

LCD Triplex Drive with COP820CJ

National Semiconductor
Application Note 953
Klaus Jaensch and
Siegfried Rueth

INTRODUCTION

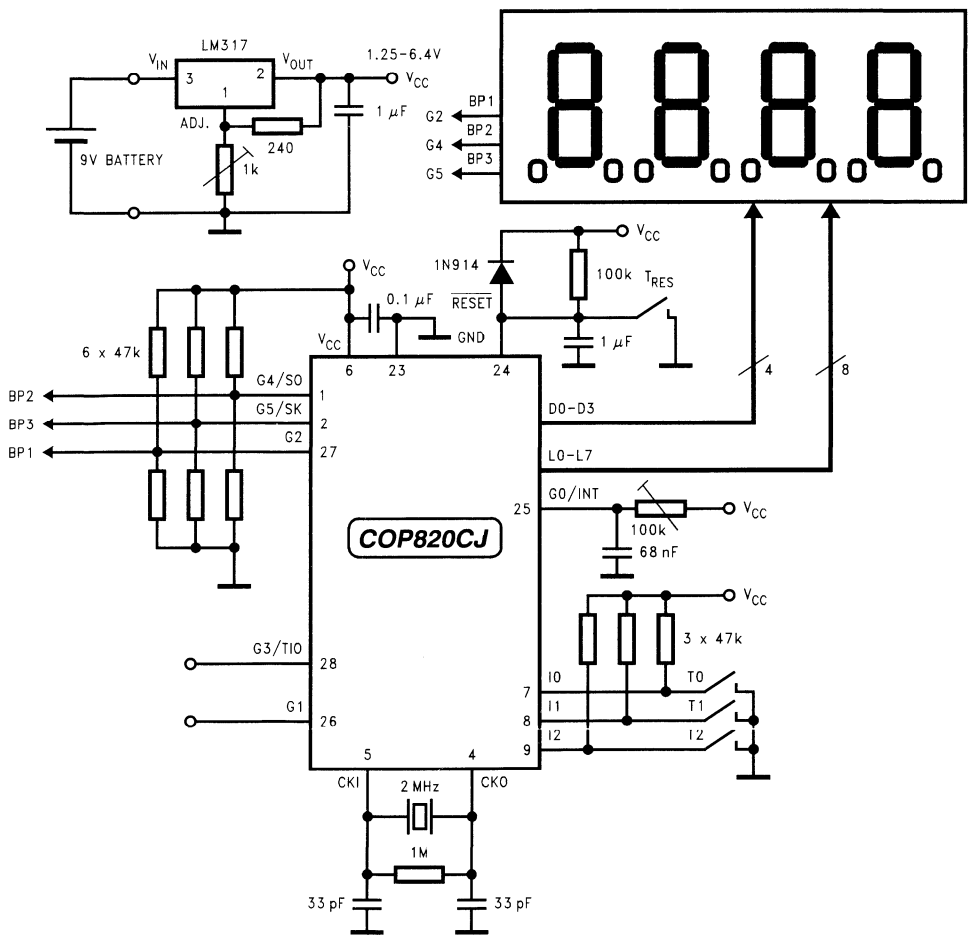
There are many applications which use a microcontroller in combination with a Liquid Crystal Display. The normal method to control a LCD panel is to connect it to a special LCD driver device, which receives the display data from a microcontroller. A cheaper solution is to drive the LCD directly from the microcontroller. With the flexibility of a COP8 microcontroller the multiplexed LCD direct drive is possible. This application note shows a way how to drive a three way multiplexed LCD with up to 36 segments using a 28-pin COP800 device.

ABOUT MULTIPLEXED LCD'S

There is a wide variety of LCD's, ranging from static devices to multiplexed versions with multiplex rates of up to 1:256.

The multiplex rate of a LCD is determined by the number of its backplanes (segment-common planes). The number of segments controlled by one line (with one segment pin) is equal to the number of backplanes on the LCD. So, a three way multiplexed LCD has three backplanes and three segments are controlled with one segment pin. For example in a three way multiplexed LCD with three segment inputs (SA, SB, SC) one can drive a 7-segment digit plus two special segments.

These are 3 × 3 = 7 + 2 = 9 segments. The special segments can have an application specific image. ("+", "−", ".", "mA", . . . etc).



TL/DD/12076-1

FIGURE 1. Schematic for LCD Triplex Driver

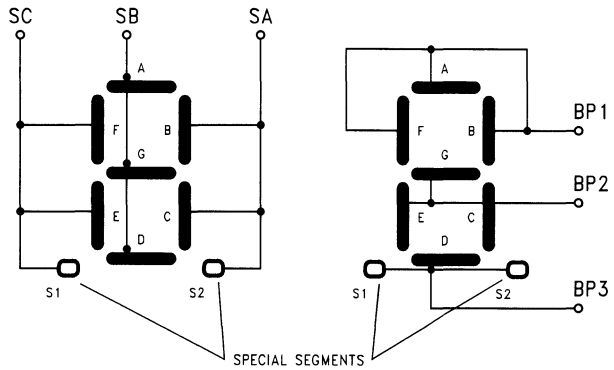


FIGURE 2. Example: Backplane-Segment Arrangement

A typical configuration of a triplex LCD is a four digit display with 8 special segments (thus having a total of 36 segments). Fifteen outputs of the COP8 are needed; **4 × 3 segment** pins and **3 backplane** pins.

Common to all LCD's is that the voltage across backplane(s) and segment(s) has to be an AC-voltage. This is to avoid electrochemical degradation of the liquid crystal layer. A segment being "off" or "on" depends on the **r.m.s.** voltage across a segment.

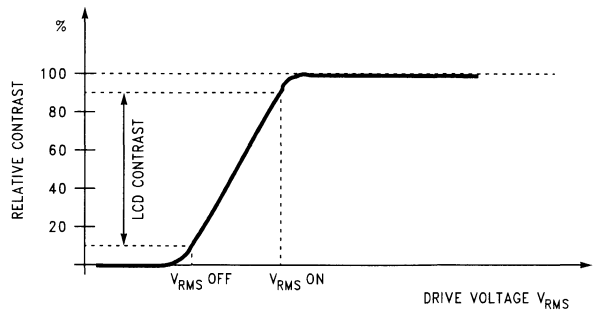
The maximum attainable ratio of "on" to "off" r.m.s. voltage (discrimination) is determined by the multiplex ratio. It is given by:

$(V_{ON}/V_{OFF})max = SQR((SQR(N) + 1)/(SQR(N) - 1))$

N is the multiplex ratio.

The maximum discrimination of a 3 way multiplexed LCD is 1.93, however, it is also possible to order a customized display with a smaller ratio. With the approach used in this application note, it may not be possible to acheive the optimum contrast acheived with a standard 3 way muxed driver. As a result of decreased discrimination (1.93 to 1.73) the user may have to live with a tighter viewing angle and a tighter temperature range.

In this application you get a **VrmsOFF** voltage of 0.408*Vop and a **VrmsON** voltage of 0.707*Vop. Vop is the operating voltage of the LCD. Typical Vop values range from 3V–5V. With the optoelectrical curve of the LCD you can evaluate the maximum contrast of the LCD by calculating the difference between the relative "OFF" contrast and the relative "ON" contrast.



In this example:

VrmsON = 0.707*Vop

VrmsOFF = 0.408*Vop

FIGURE 3. Example Curve: Contrast vs r.m.s. Drive Voltage

The backplane signals are generated with the voltage steps **0V, Vop/2** and **Vop** at the backplanes, also see *Figure 4*.

Two resistors are necessary for each backplane to establish all these levels.

The backplane connection scheme is shown in *Figure 1*.

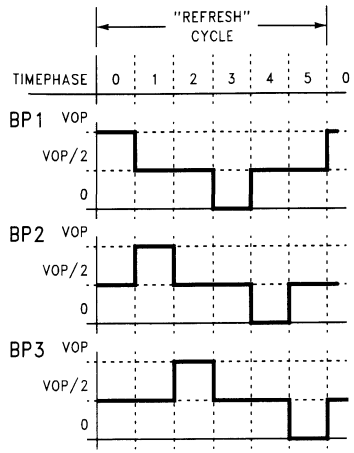
The Vop/2 level is generated by switching the appropriate COP's port pin to Hi-Z.

The following timing considerations show a simple way how to establish a discrimination ratio of 1,732.

TIMING CONSIDERATIONS

A Refresh cycle is subdivided in 6 timephases. *Figure 4* shows the timing for the backplanes during the equal distant timephases 0 . . . 5.

Backplane Control



TL/DD/12076–4

Note: After timephase 5 is over the backplane control timing starts with timephase 0 again.

FIGURE 4. Backplane Timing

While the backplane control timing continuously repeats after 6 timephases, the segment control depends on the combination of segments just being activated.

TABLE I. Possible Segment ON/OFF Variations

Tiphtab Address	Segment A	Segment B	Segment C
0	off	off	off
1	on	off	off
2	off	on	off
3	on	on	off
4	off	off	on
5	on	off	on
6	off	on	on
7	on	on	on

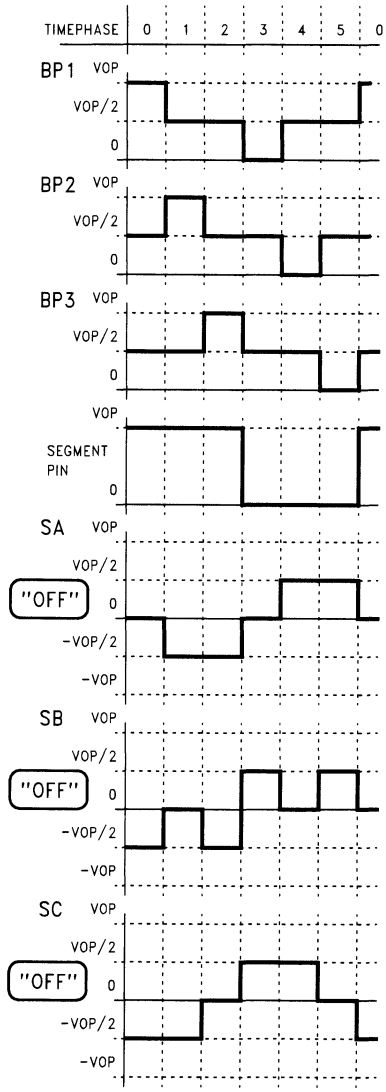
Figure 5 through *Figure 12* below show all possible combinations of controlling a **"Segment Triple"** with help of the 3 backplane connections and one segment pin. The segment switching has to be done according to the ON/OFF combination required (see also Table I).

Each figure shows in the first 3 graphs the constant backplane timing.

The 4th graph from the top shows the segment control timing necessary to switch the 3 segments (SA/SB/SC), activated from one pin, in the eight possible ways.

The 3 lower graphs show the resulting r.m.s. voltages across the 3 segments (SA, SB, SC).

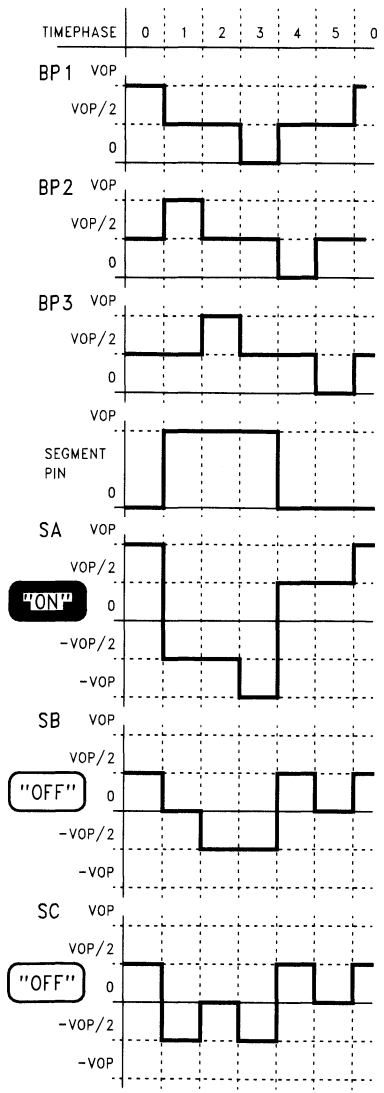
Segment/Backplane Control-Timing



TL/DD/12076-5

tiphtab address = 0

FIGURE 5

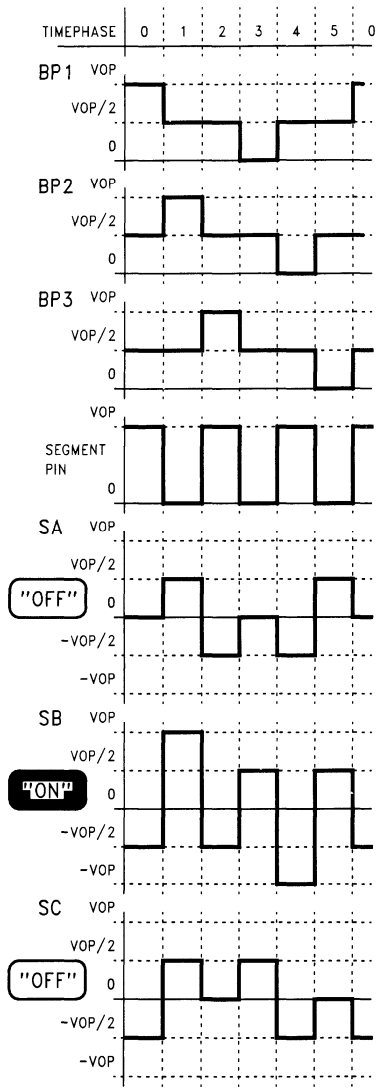


TL/DD/12076-6

tiphtab address = 1

FIGURE 6

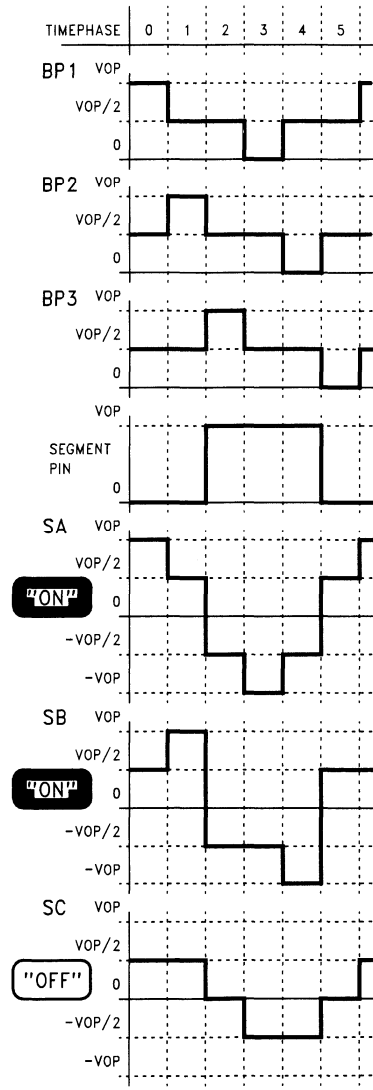
Segment/Backplane Control-Timing



TL/DD/12076-7

tiphtab address = 2

FIGURE 7

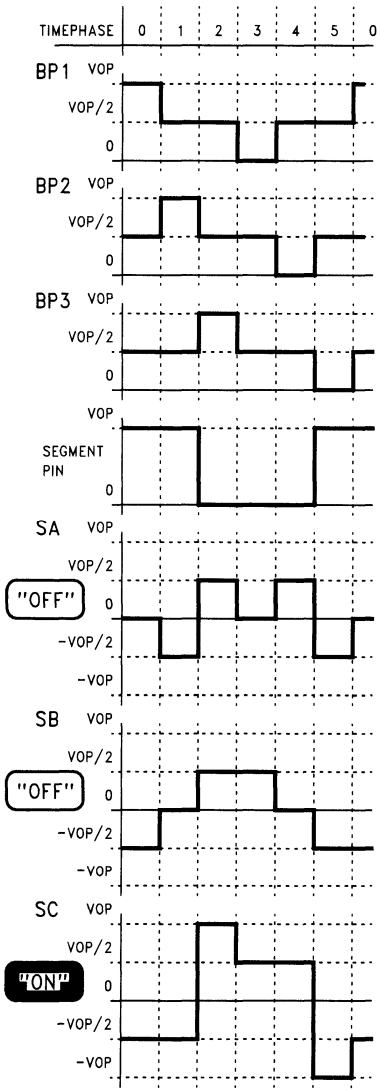


TL/DD/12076-8

tiphtab address = 3

FIGURE 8

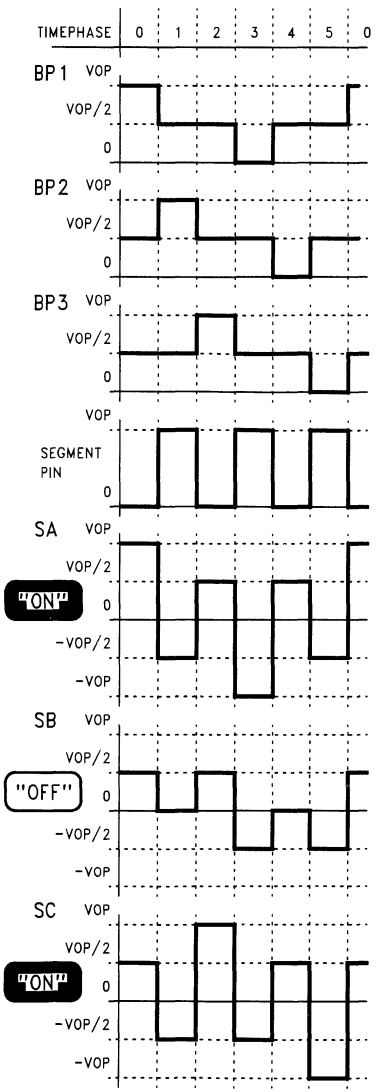
Segment/Backplane Control-Timing



TL/DD/12076–9

tiphtab address = 4

FIGURE 9

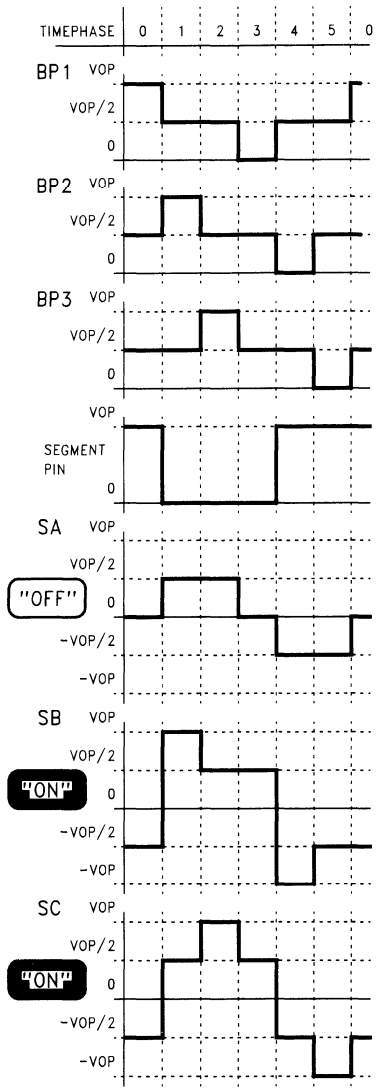


TL/DD/12076–10

tiphtab address = 5

FIGURE 10

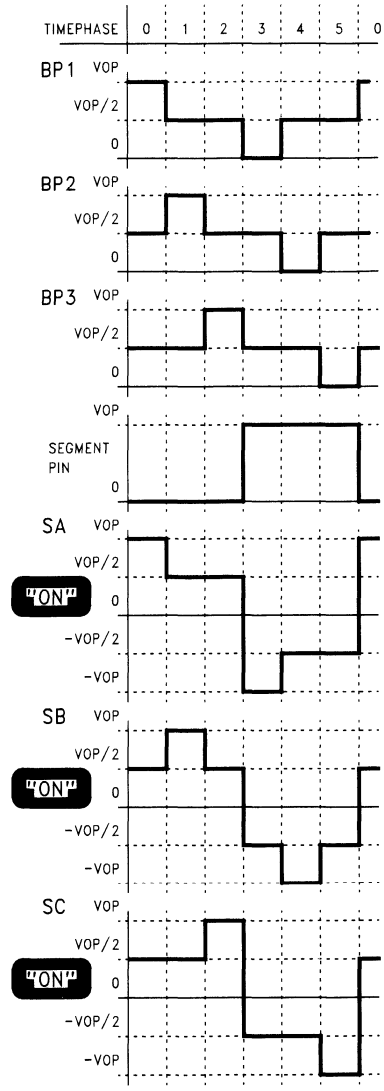
Segment/Backplane Control-Timing



TL/DD/12076-11

tiphtab address = 6

FIGURE 11



TL/DD/12076-12

tiphtab address = 7

FIGURE 12

REFRESH FREQUENCY

One period with six timephases is called a **refresh cycle** (also see *Figure 4*).

The refresh cycle should be in a frequency range of 30 . . . 60 Hz. A frequency below 30 Hz will cause a flickering display. On the other hand, current consumption increases with the LCD's frequency. So it is also recommended to choose a frequency below 60 Hz.

In order to periodically update the μC's port pins (involved in backplane or segment control) at the beginning of a new timephase, the COP8 needs a timebase of typ. 4 ms which is realized with an external RC-circuit at the G0/INT pin.

The G0 pin is programmable as input (Schmitt Trigger). The conditions for the external interrupt could be set for a low to high transition on the G0 pin setting the IPND-flag (external interrupt pending flag) upon an occurrence of such a transition. The external capacitor can be discharged, with the G0 pin configured as Push/Pull output and programmed to "0". When, switching G0 as input the Cap. will be charged through the resistor, until the threshold voltage of the Schmitt-Trigger input is reached. This triggers the external interrupt. The first thing the interrupt service routine has to do is to discharge the capacitor and switch G0 as input to restart the procedure.

This timing method has the advantage, that the timer of the device is free for other tasks (for example to do an A/D conversion).

The time interval between two interrupts depends on the RC circuit and the threshold of the G0 Schmitt Trigger V_{TH}.

The refresh frequency is independent of the clock frequency provided to the COPs device.

The variations of "threshold" levels relative to V_{CC} (over process) are as follows:

$$(V_{TH}/V_{CC})\ \text{min} = 0.376$$

$$(V_{TH}/V_{CC})\ \text{max} = 0.572$$

at V_{CC} = 5V

Charge Time:

$$T = -(\ln(1-V_{TH}/V_{CC})*RC)$$

To prevent a flickering display one should aim at a minimum refresh frequency of f_{refr} = 30 Hz. This means an interrupt frequency of f_{int} = 6 × 30 Hz = 180 Hz. So, the maximum charge up time T_{max} must not exceed 5.5 ms (T_{min} = 2.78 ms).

With the formula:

$$RC_{max} = T_{max}/(-\ln(1-(V_{TH}/V_{CC})\text{max})) = 5.5\ \text{ms} \times 0.849$$

$$RC_{max} = 6.48\ \text{ms}$$

$$(RC_{min} = 5.98\ \text{ms})$$

The maximum RC time-constant is calculated. The minimum RC time constant can be calculated similarly.

A capacitor in the nF-range should be used (e.g. 68 nF), because a bigger one needs too much time to discharge. To discharge a 68 nF Cap., the G0 pin of the device has to be low for about 40 μs.

On the other hand the capacitor should be large enough to reduce noise susceptibility.

When the RC combination is chosen, one can calculate the maximum refresh frequency by using the minimum values of the RC constant and the minimum threshold voltage:

$$T_{min} = RC_{min}*(-\ln(1-(V_{TH}/V_{CC})\text{min} = RC_{min}*0.472$$

and

$$f_{refr,max} = f_{int,max}/6 = 1/(T_{min}*6)$$

In the above example one timephase would be minimum 2.82 ms long. This means that about 250 instructions could be executed during this time.

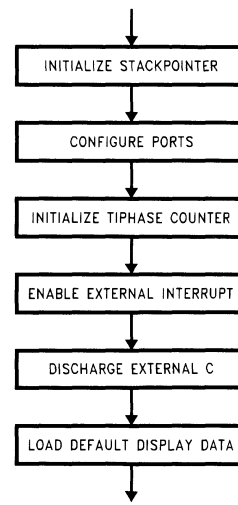
SOFTWARE

The software for the triplex LCD drive-demo is composed of three parts:

1. The initialization routine is executed only once after resetting the device, as part of the general initialization routine of the main program. The function of this routine is to **configure the ports, set the timephase counter (tiphase) to zero, discharge the external capacitor and enable the external interrupt.**

The initialization routine needs 37 bytes ROM.

Figure 13 shows the flowchart of this routine.



TL/DD/12076–13

FIGURE 13. Flowchart for Initialization Routine

2. The update routine calculates the port-data for each timephase according to the BCD codes in the RAM locations **'digit1' . . . 'digit4'** and the **special segments**. This routine is only called if the display image changes.

The routine converts the BCD code to a list **1st**, which is used by the refresh routine. *Figure 14* gives an overview and illustrates the data flow in this routine.

In *Figure 15* the data flow chart is filled with example data according to the display image in *Figure 16*.

First the routine creates the **seg1st** (4 bytes long), which contains the "on/off" configuration of each segment of the display. The display has 36 segments but the 4 bytes have only 32 bits, so the four special segments **S1** are stored in the **specbuf** location. The **bcdsegtab** table (in ROM) contains the LOOK-UP data for all possible Hex numbers from **0 to F**.

The routine takes three bits at the beginning of each time-phase from the **seg1st**.

These 3 bits address the 8 bytes of the **tiphtab** table in ROM. Each byte of this table contains the **time curve** for a segment pin (only 6 bits out of 8 are used).Using this information, the program creates the lists **for port D and port L (pod1st, pol1st)**. Every byte of this list contains the **timing representatives** for the pins D0–D3 and L0–L7, to allow an easy handling of the refresh routine.

The external interrupt has to be disabled while the **copy** routine is working, because the mixed data of two different display images would result in improper data on the display. *Figure 17* shows the flowchart of the **update** routine. The Flowchart of the **convert** subroutine is shown in *Figure 18*.

MEMORY REQUIREMENTS

ROM: 152 bytes incl. look up tables

RAM: 43 bytes (*Figure 15* illustrates the RAM locations)

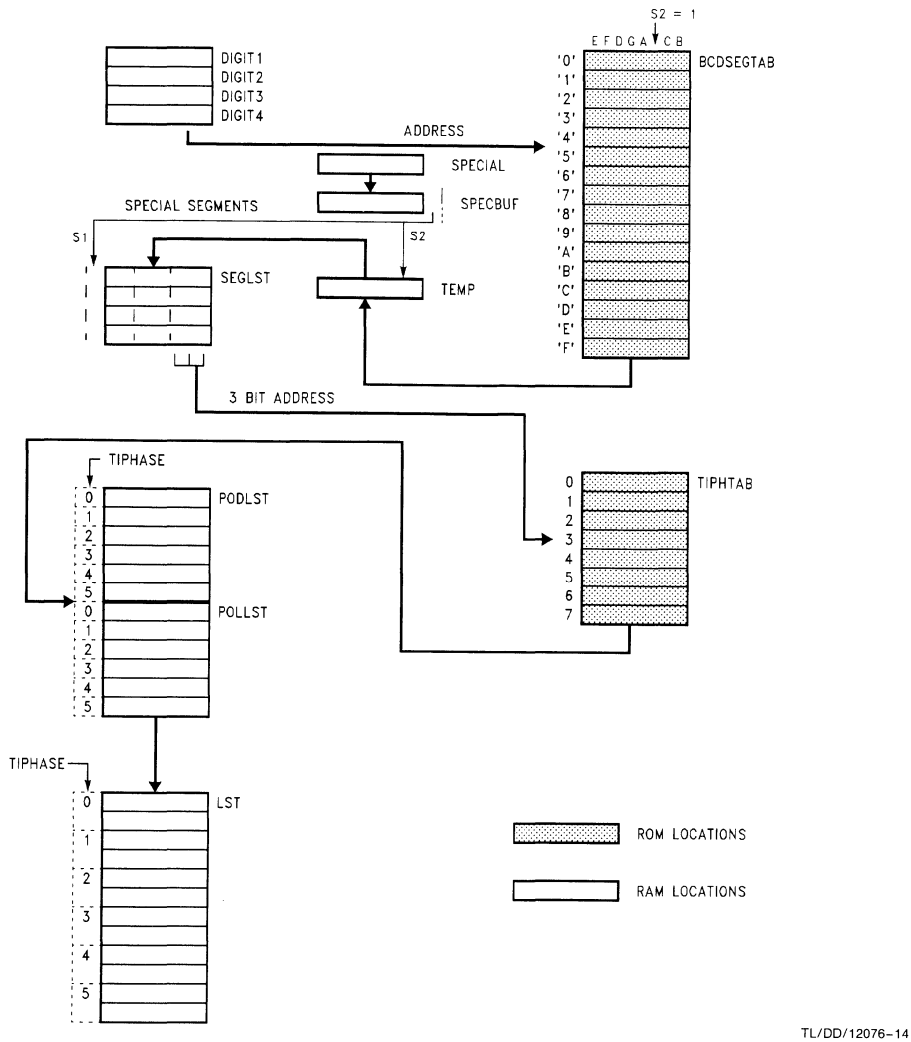


FIGURE 14. Data Flow Chart for Update Routine

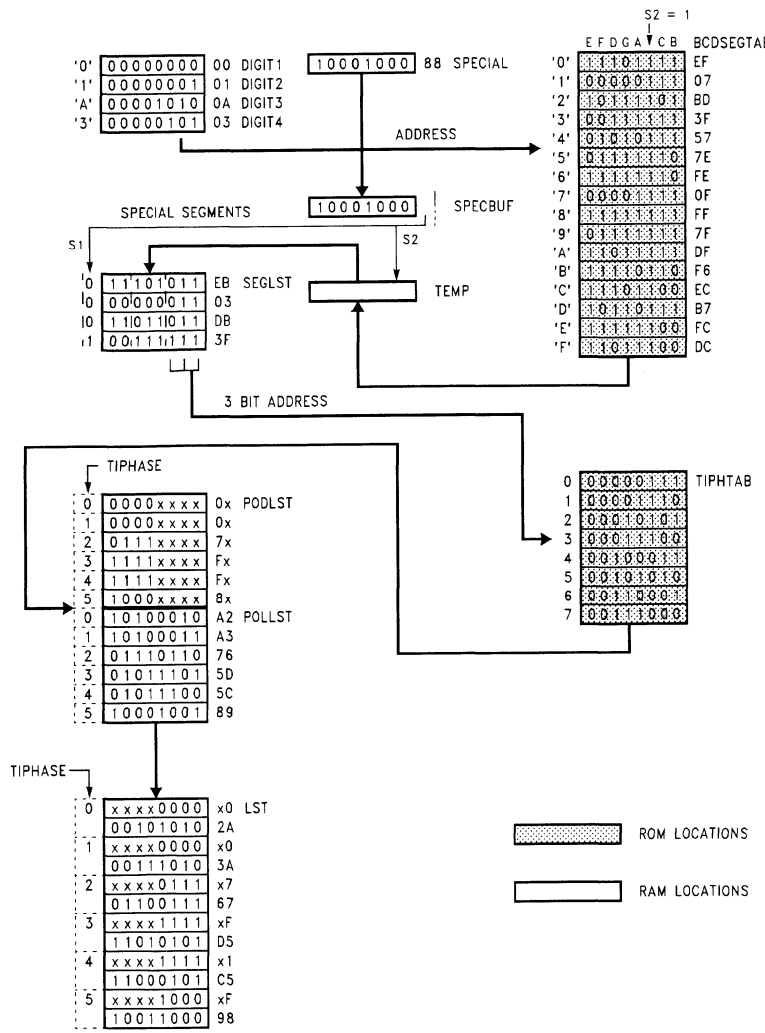
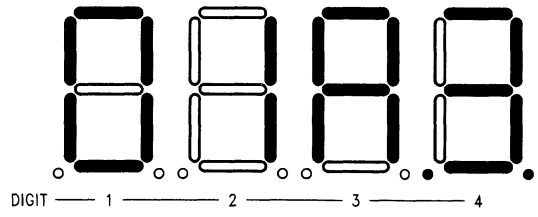


FIGURE 15. Data Flow Chart for Update Routine



TL/DD/12076-16

FIGURE 16. Display Example

3. The refresh routine is the interrupt service routine of the external interrupt and is invoked at the beginning of a new timephase. First the routine discharges the external capacitor and switches the G0/INT pin back to the input mode, to initialize the next timephase. The backplane ports G2, G4 and G5 and the segment pin ports D and L are updated by this routine according to the actual timephase. For the backplanes the data are loaded from the **bptab** table in ROM.

Table II shows how the **bptab** values are gathered. *Figure 20* shows the flowchart for the refresh routine.

TIME REQUIREMENTS

The routine runs max. 150 cycles.

For a non flickering display, the refresh frequency must be 30 Hz minimum. One refresh cycle has six timephases and is max. 33 ms long. So each timephase is 5.5 ms long. With an oscillator (CKI) frequency of 2 MHz, one instruction cycle takes 1/(2 MHz/10) = 5 μs to execute. During one timephase the controller can execute:

5.5 ms/5 μs = 1100 cycles. So the refresh routine needs 134/1100 = 0.122 = 12.2% of the whole processing time (in this case).

With a refresh frequency of 50 Hz the routine needs about 20.1% of the whole processing time.

The refresh routine needs about **103** ROM bytes.

TABLE II. Phase Values

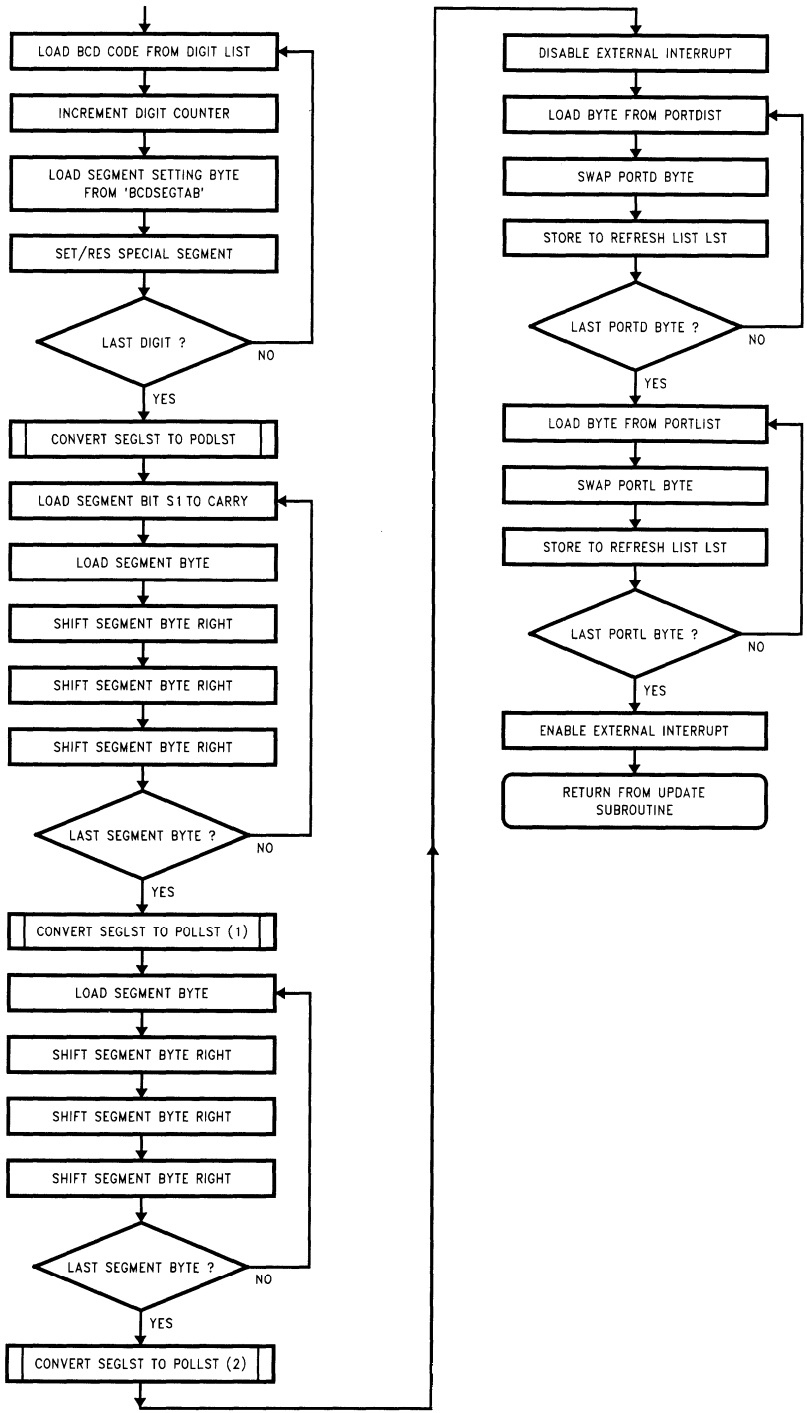
Tiphase	G5	G4	G2	Portg Data	Hex	Portg Config.	Hex
0	0/0	0/0	1/1	XX00X1XX	04	XX00X1XX	04
1	0/0	1/1	0/0	XX01X0XX	10	XX01X0XX	10
2	1/1	0/0	0/0	XX10X0XX	20	XX10X0XX	20
3	0/0	0/0	0/1	XX00X0XX	00	XX00X1XX	04
4	0/0	0/1	0/0	XX00X0XX	00	XX01X0XX	10
5	0/1	0/0	0/0	XX00X0XX	00	XX10X0XX	20

data/configuration register of portg

0/0 : Hi-Z input
0/1 : output low
1/1 : output high

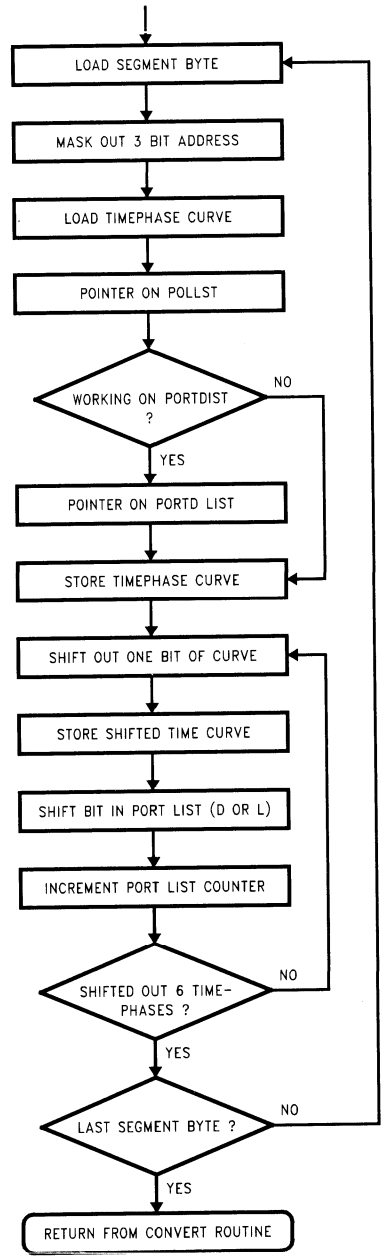
SUMMARY OF IMPORTANT DATA

LCD type:	3 way multiplexed
Amount of segments:	36
$V_{OP} = (V_{CC})$ (range):	2.5V to 6V
Oscillator frequency:	2 MHz (typ.)
Instruction cycle time:	5 μs
ROM requirements:	
init routine:	37 bytes
update routine:	152 bytes
refresh routine:	103 bytes
total:	292 bytes
RAM requirements:	
permanent use:	25 bytes
temporary use:	18 bytes
stack:	6 bytes
total:	49 bytes
	(also see *Figure 19*)
Timer:	not used
External interrupt:	with RC circuit used as time-base generator
Ports D, L:	used for LCD control
Port G:	3 G-pins are still free for other purposes +
Port I:	can be used as key-inp.



TL/DD/12076–17

FIGURE 17. Flowchart for Update Routine



TL/DD/12076-18

FIGURE 18. Flowchart for Convert Subroutine

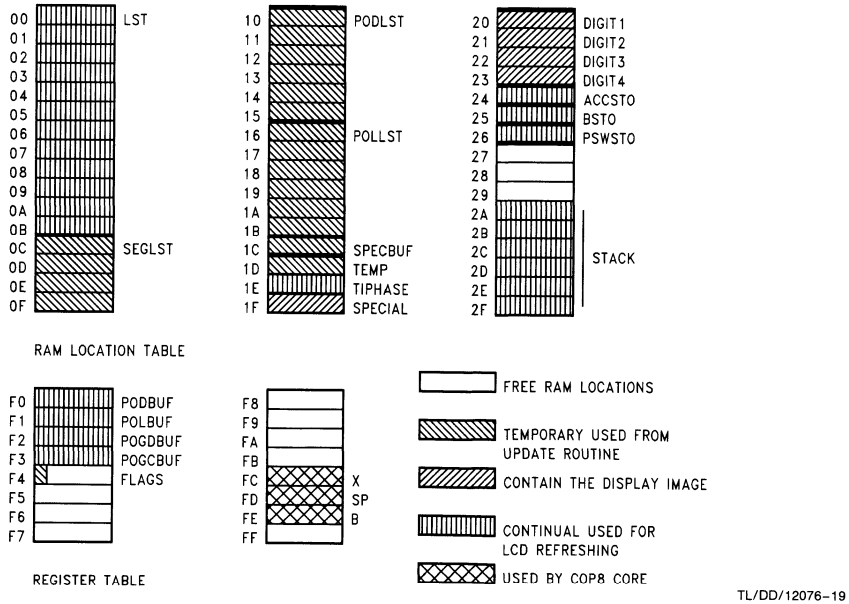
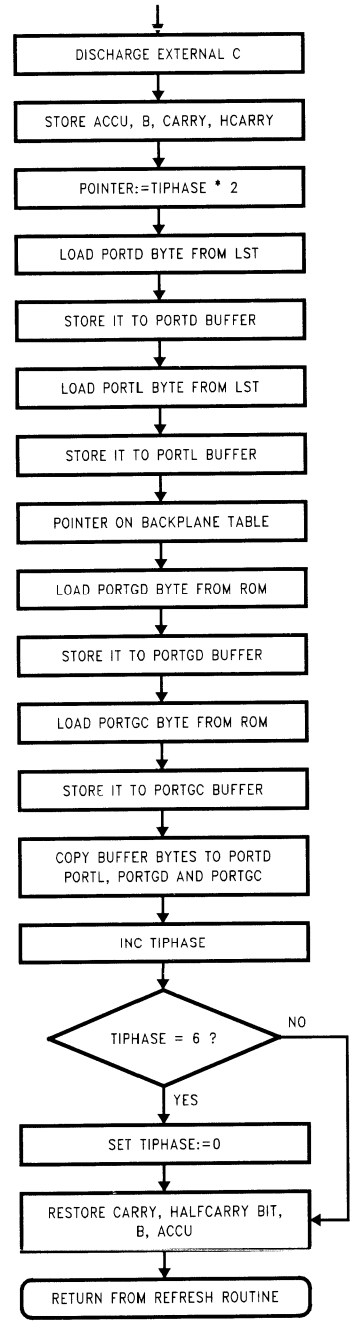


FIGURE 19. RAM Assignment



TL/DD/12076-20

FIGURE 20. Flowchart for Refresh-Routine

Listing

```
;  DEMO FOR COP820CJ:
;  3 WAY MULTIPLEXED LCD DRIVER DEMO
;  CONSTANT DISPLAY "01A3" and two special segments on

                .incld cop820cj.inc

;RAM assignments

                tiphase=01E
                special=01F             ;this byte must contain the
                                        ;on/off configuration of
                                        ;the extra segments
                                        ;('-','low bat',etc.)

                digit1=020              ;in these RAM locations the
                digit2=021              ;BCD code of the display
                digit3=022              ;digits are stored.
                digit4=023              ;

                accsto=024              ;accu buffer used during
                                        ;interrupt service routine
                bsto=025                ;b buffer
                pswsto=026              ;psw buffer

;register definition:

                podbuf=0f0              ;portd buffer
                polbuf=0f1              ;portl buffer
                pogdbuf=0f2             ;portgd buffer
                pogcbuf=0f3             ;portgc buffer
                flags=0f4               ;flag byte for podfla

;flag definition in flags byte

                podfla=07

;************** initialization routine ******************************

init:

                ld sp,#02f              ;initialize stackpointer

                ld portlc,#0ff          ;port l output
                ld portgc,#037          ;port g:G1,G2,G4,G5  are
                                        ;outputs
                ld portgd,#00           ;all outputs low, all
                                        ;inputs Hi-Z
                                        ;C at G0 is discharged
                ld tiphase,#00          ;begin with timephase 0
                ld psw,#002             ;ext. interrupt enable
```

TL/DD/12076–21

```
begin:          sbit #gie,psw           ;interrupts are welcome now
                rbit #00,portgc         ;now the external C can be
                                        ;charged

                ld b,#special
                ld [b+],#088            ;two special segments
                                        ;are 'ON'

                                        ;display:"01A3"
                ld [b+],#00             ;digit1
                ld [b+],#001            ;digit2
                ld [b+],#00A            ;digit3
                ld [b],#003             ;digit4

;************** main program ************************************
loop:
                jsr update
                jp loop

;************** update subroutine *********************************

;RAM definitions:

                specbuf=01C             ;buffer for 'special'
                temp=01D                ;temporary used

;pointer on tables:

                podlst=010              ;adress of list for port d
                pollst=016              ;adress of list for port l
                lst   =000              ;main list for display
                                        ;routine to refresh
                                        ;port d,l each timephase

                seglst=00C              ;this list contains the
                                        ;on/off configuration of
                                        ;the segments

                .=0200
                .local
update:
                ld a,special            ;load 'special' register
                x a,specbuf             ;to the buffer 'specbuf'
                ld x,#seglst            ;x points the segmentlist
                ld b,#digit1            ;b points digitlist

nxtdig:         ld a,[b+]               ;load BCD code of
                                        ;current digit
                add a,#L(bcdsegtab)     ;set pointer on look up
                                        ;table for segment setting
                laid                    ;load segment data of
                                        ;current digit
                x a,temp                ;store it to RAM
                ld a,specbuf            ;load special bit
                rrc a                   ;to carry
```

TL/DD/12076–22

```
                x a,specbuf             ;prepare for next
                                        ;special segment
                ifnc                    ;special bit not set ?
                rbit #2,temp            ;then reset it in the
                                        ;temp byte
                ld a,temp               ;store temp
                x a,[x+]                ;to the seglst list
                ifbne #04               ;if not last digit
                jp nxtdig               ;load data for next digit

                sbit #podfla,flags      ;set flag for working at
                                        ;port d list
                jsr convert             ;convert 3 bits from the
                                        ;segment bytes to the
                                        ;timephaselist for portd


;shift with carry

shwc:
                ld b,#seglst            ;b points seglst
nxtshwc:        ld a,specbuf            ;load special segment bit
                rrc a                   ;to carry
                x a,specbuf             ;prepare for next
                                        ;special segment
                ld a,[b]                ;shift the segmentbyte
                rrc a                   ;three positions right
                rrc  a                  ;and append the special
                                        ;segment bit
                rrc  a                  ;
                x a,[b+]                ;store shifted byte
                ifbne #00               ;end of segment list
                                        ;not reached ?
                jp nxtshwc              ;then shift the next
                                        ;segment byte

                rbit #podfla,flags      ;reset flag for working
                                        ;at port 1 list
                jsr convert             ;convert 3 bits of the
                                        ;segment bytes to the
                                        ;timephaselist for port 1
;shift (without carry)

shift:          ld b,#seglst            ;b points segmnet list
nxtshift:       ld a,[b]                ;load segment byte
                rrc a                   ;shift the segmentbyte
                rrc a                   ;three positions right
                rrc a                   ;
                x a,[b+]                ;store shifted byte
                ifbne #00               ;end of segment list
                                        ;not reached ?
                jp nxtshift             ;then shift the next
                                        ;segment byte
```

TL/DD/12076-23

```
                jsr convert             ;convert 3 bits of the
                                        ;segment bytes to the
                                        ;timephaselist for port 1

;copy portdata to the list on which the refresh routine will access

copy:
                rbit #eni,psw           ;disable interrupt to
                                        ;prevent fail display
                ld b,#podlst            ;b points podlst
                ld x,#lst               ;x points refresh list
nxtd:           ld a,[b+]               ;load portbyte
                swap a                  ;swap it
                x a,[x+]                ;store it to refresh list
                ld a,[x+]               ;increment x
                ifbne #06               ;if the end of the podlst
                                        ;is not reached
                jp nxtd                 ;then next timephase
                ld b,#pollst            ;b points pollst
                ld x,#lst               ;x points refresh list
nxtl:           ld a,[x+]               ;increment x
                ld a,[b+]               ;load portbyte
                swap a                  ;swap it
                x a,[x+]                ;store it to refresh list
                ifbne #0C               ;if the end of the pollst
                                        ;is not reached
                jp nxtl                 ;then next timephase
                sbit #eni,psw           ;refresh routine allowed
                                        ;again

                ret                     ;end of update routine

;subroutines for update routine:

convert:
                ld x,#seglst            ;x points segment list
nxtsgl:         ld a,[x+]               ;load segment byte
                and a,#007              ;mask out first three bits
                add a,#L(tiphtab)       ;pointer on timephase table
                laid                    ;load timephase curve for
                                        ;one segment pin
                ld b,#pollst            ;b points list for portd
                ifbit #podfla,flags     ;working at podlst ?
                ld b,#podlst            ;then b points on podlst

;shift timephase data according to 3 bits ( 8 combinations are
;possible with 3 segments)

tipsh:
                x a,temp                ;store timephase curve to
                                        ;temp buffer
nxtphsh:
                ld a,temp               ;load timephase curve again
                rrc a                   ;shift out one bit into
```

TL/DD/12076–24

```
                                        ;carry bit
                x a,temp                ;store shifted curve
                ld a,[b]                ;load portbyte
                rrc a                   ;shift in one bit from
                                        ;carry bit
                x a,[b+]                ;store shifted portbyte
                                        ;again
                ld a,#pollst            ;end of podlst ?
                ifeq a,b                ;
                jp eplst                ;then return
                ifbne #0C               ;else end of pollst
                jp nxtphsh              ;
eplst:
                ld a,#L(seglst+4)       ;if the end of the segment
                ifgt a,x                ;list is not reached
                jp nxtsg1               ;work at next segment byte
                ret


bcdsegtab:

;in this bytes are the on/off configuration of the segments
;for a digit are stored. there are only 7 bits of each byte
;the configuration of the 2 special segments is stored
;in the 'special' byte.

                .BYTE 0EF,007,0BD,03F   ;'0'...'3'
                .BYTE 057,07E,0FE,00F   ;'4'...'7'
                .BYTE 0FF,07F,0DF,0F6   ;'8'...'B'
                .BYTE 0EC,0B7,0FC,0DC   ;'C'...'F'
tiphtab:

;one pin controls 3 segments. there are 8 possible
;combinations. for each combination there is one byte.
;6 bits of one byte control the pin for each timephase.

                .BYTE 007,00E,015,01C,023,02A,031,038



;************** interrupt service routine *************************

                .=0ff
refresh:
                x a,accsto              ;store accu
                ld a,b                  ;store b
                x a,bsto                ;

                ld b,#portgd
                                        ;discharge C
                rbit #00,[b]
                ld a,[b+]               ;increment b (b=#portgc)
                sbit #00,[b]            ;by switching G0 to a
                                        ;low output
```

TL/DD/12076–25

```
rbit #00,[b]            ;C can be charged again
ld b,#psw
rbit #ipnd,[b]          ;reset ext. interrupt
                        ;pending flag

ld a,[b]                ;load psw
x a,pswsto              ;store psw

ld a,tiphase            ;accu:=tiphase*2
add a,tiphase           ;

x a,b                   ;store accu in b
ld a,[b+]               ;load portbyte from
                        ;refresh list('lst')
x a,podbuf              ;store it to port d buffer
ld a,[b+]               ;load portbyte
x a,polbuf              ;store it to port l buffer
ld a,b                  ;accu:=timephase*2+2
add a,#L(bptab)-2       ;accu points on
                        ;backplane table
x a,b                   ;store pointer
ld a,b                  ;
laid                    ;load port g data byte
x a,pogdbuf             ;store it to port g data
                        ;buffer
ld a,[b+]               ;increment b
ld a,b                  ;load pointer
laid                    ;load portg conf. byte
x a,pogcbuf             ;store it to buffer


ld b,#podbuf            ;b points buffer list
ld a,[b+]               ;
x a,portd               ;refresh port d
ld a,[b+]               ;
x a,portld              ;refresh port l

ld portgc,#00           ;all backplane wires on
                        ;Vop/2 level to prevent
                        ;spikes


ld a,[b+]               ;
x a,portgd              ;refresh port g data
ld a,[b+]               ;
x a,portgc              ;refresh port g config.

ld a,tiphase            ;update timephase counter
inc a                   ;
ifeq a,#06              ;tiphase = 0..5
ld a,#00                ;
x a,tiphase             ;
ld b,#pswsto
rc                      ;restore carry bit
ifbit #07,[b]           ;
```

TL/DD/12076-26

```
                sbit #07,psw
                ifbit #06,[b]           ;restore halfcarry bit
                sbit #06,psw            ;

                ld a,bsto               ;restore b
                x a,b                   ;
                ld a,accsto             ;restore accu


                reti                    ;return from lcd
                                        ;refresh routine

bptab:          .BYTE 004,004,010,010,020,020
                .BYTE 000,004,000,010,000,020

                .END
```

TL/DD/12076–27

COP888GW Features and Applications

National Semiconductor
Application Note 982
Martin Embacher

INTRODUCTION

The COP888GW is a member of National Semiconductor's COP888 family of microcontrollers fabricated in M²CMOS™ technology. The device was established using CCM (Configurable Controller Methodology) design techniques which provide a fast and reliable way to create new derivatives for a fast growing and demanding controller market. In addition to the COP888 feature core this controller includes a math unit to allow fast multiply/divide operations. The controller has special timers for motor control allowing it to be used in applications where enhanced processing power and motor control capabilities are required (as in plotters, robot-arms or information systems.)

This note describes the COP888GW microcontroller emphasizing the powerful new features. The reader will be given easy to use macros to take full advantage of the new CCM blocks.

COP888GW KEY FEATURES

- Multiply/Divide Unit
- Two 16-bit capture modules with 8-bit prescalers
- Four pulse train generators with 16-bit prescalers
- Full duplex UART
- MICROWIRE/PLUS™ serial port with interrupt
- 16 kbytes of on-board ROM
- 512 bytes of on-board RAM
- Multi-Input Wakeup pins (8) optionally usuable hardware interrupts either on falling or rising edge
- Two 16-bit timers, each with two 16-bit autoreload/capture registers supporting:
  — Processor independent PWM mode
  — External Event counter mode
  — Input Capture mode
- Low EMI
- Two power saving modes: HALT and IDLE
- Fourteen multi-source vectored interrupts
- Software selectable I/O options
- Package:
  — 68 PLCC with 56 I/O pins

CAPTURE TIMERS

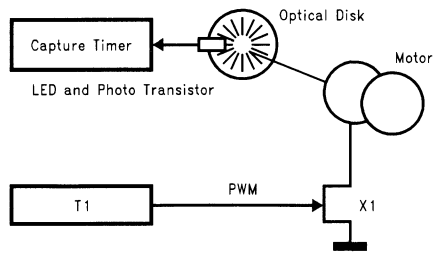
With its two high speed, high resolution capture timers, the COP888GW is ideally suited high resolution frequency measurement. To give a feeling for the capabilities, Table I shows the capture timers resolution versus the measured signal freuency with a processor clock (CKI) of 10 MHz. The sampling rate is CKI divided by a 8-bit prescaler. In the table, the prescaler is set to 1 to allow maximum performance. This results in a maximum resolution of 100 ns.

TABLE I. Capture Timer Resolution vs Signal Frequency

Resolution/ Bits	16	14	12	10	8
Max. Frequency	152 Hz	610 Hz	2.4 kHz	9.6 kHz	39 kHz

The capture timer can be used for a closed loop control in a DC motor. One of the standard COP888 timers (T1 or T2) generates a PWM signal to control the motor. One of the capture timer modules measures the actual RPM of the motor. A simple algorithm is used to adjust the motor's speed to the desired value.

Figure 1 shows a block diagram of a closed loop DC motor control with one capture and one PWM timer. The MOSFET X1 acts as a low-side driver for the motor and controls the current flow. The current through the motor is proportional to the duty cycle of the PWM signal, generated by T1, at the gate of X1. An optical disk, in conjunction with an LED/photo transistor senses the motor's RPM. The pulses generated by the photo transistor are directly fed to the capture timer's input. The capture timer inputs are alternate functions of Port-L. This I/O-port features integrated schmitt trigger inputs eliminating the need for external signal conditioning of the input signals.



TL/DD/12376–1

FIGURE 1. Closed Loop DC Motor Control Block Diagram

SOFTWARE MODULES

The following section provides a guide to general purpose macros and subroutines to set-up the capture timer and the standard timers T1 to T2. The fuctions are written as of macros so the user can easily include them in the application software.

CAPTURE TIMERS

Capture Timers Setup Routine

This routine sets the capture timer module CCM1 or CCM2 to allow input capture of a signal on the respective port pin. Additionally the I/O pins Port L6 for CCM1 and Port L7 for CCM2 are configured as inputs. This macro should be placed in a separate macro file and included in the source code after the processor specific include file. *Figure 2* shows the source code to include the macros in the applications program.

```
.incld cop888gw.inc
.incld gw_macros.inc
.sect main_prog, rom
..
..
.end
```

FIGURE 2. Including a Macro Into Source Code

The macro is called in the source code as:

__CT__SETUP, __MODULE, __PRESCALER, __EDGE, __INTR

__MODULE is the capture module number

1 = capture module 1
2 = capture module 2

__PRESCALER is the prescaler value the range is 0 to 255.

EDGE selects the input capture edge

0 = rising edge
1 = falling edge

__INTR select if capture interrupt is used

0 = disable capture interrrupt
1 = enable capture interrrupt

The macro will check that all values are in the acceptable range and warn the user if this is not the case.

An example of the generated code is shown below. The macro is called to setup capture timer 1 with a prescaler of 100 (dec) to capture signals on a falling edge without the generation of an interrupt.

```
_CT_SETUP 1 100 1 0

ld      b, #CCMR1 ; point to capture con
ld      [b], #0   ; stop capture, clear
rbit    6, portlc ; cm 1 input on port L
ld      cm1psc, #100 ; load prescaler of c
ld      [b], #ZZ0000 ; start capture
```

TL/DD/12376–2

FIGURE 3. __CT__SETUP Resulting Code

Special attention is needed if the interrupts are enabled in the macro by setting the __INTR directive to one. An appropriate interrupt handler must be placed elsewhere in the code. The macro will allow the capture timer to generate interrupts, however it will not provide the respective interrupt service routine.

Capture Timer Read Macro

__CT__READ __MODULE __VARIABLE

__MODULE is the capture module number

1 = capture module 1
2 = capture module 2

__VARIABLE is a variable to hold the contents of the capture register

Standard Timer Setup for PWM Output

__TIM__SETUP __NUMBER __CYCLE __TON __INIT

__NUMBER is the timer number

1 = timer 1
2 = timer 2
3 = timer 3

__CYCLE is the cycle time of the PWM signal the range is 0 to 65535

__TON is on time of the PWM signal

__INIT is the initialization value

0 = start with low
1 = start with high

Standard Timer Glitch Free PWM Output

This macro is used to provide a glitch free switched PWM output with the standard COP8 PWM timer. In the macro interrupts are disabled to wait until a PWM cycle is completed and then reload the registers are re-written. This produces—in difference to an interrupt driven routine—a glitch free output of the PWM signal.

__TIM__PWM __NUMBER __CYCLE __TON

__NUMBER is the timer number

1 = timer 1
2 = timer 2
3 = timer 3

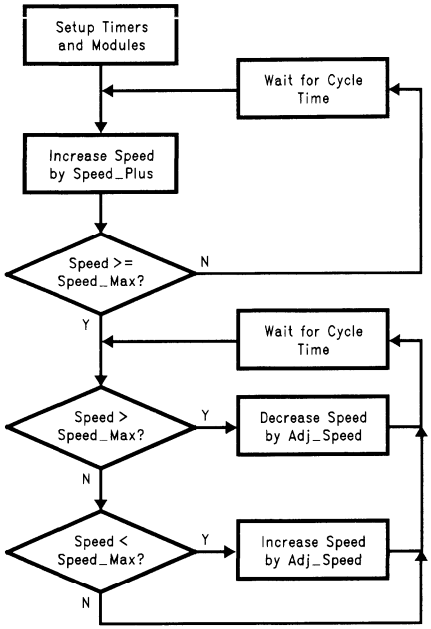
__CYCLE is the cycle time of the PWM signal the range is 0 to 65535

__TON is on the time of the PWM signal

Note: The source of the macros is found in Section Capture Module Macros on page 5.

DC Motor Control Example

An example for the application of the macros is basic DC motor closed loop control. The motor drive consists of the phases start and run. During start, the motor's speed increases up to the maximum speed set by speed__max. During run, the speed is kept around speed __max. The slope of speed increase is set by speed__plus and the maximum allowable speed increase or decrease during run is set by adj__speed. The speed update cycle time may be controlled by the IDLE timer, generating an interrupt every 4096 t_C. *Figure 4* shows the flowchart of a simple closed loop DC-motor control. The macros can be used to setup the times and to read the current speed value.



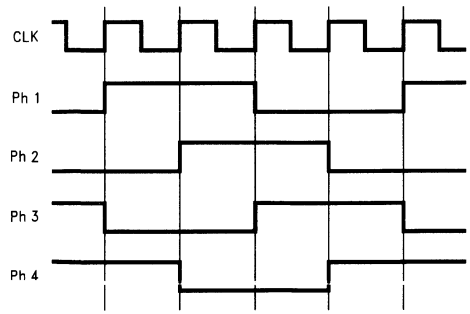
TL/DD/12376–3

FIGURE 4. Basic DC Motor Control Loop Flowchart

Pulse Train Generators

The COP888GW contains four independent pulse train generators designed for stepper motor control. The pulse train generator provides a programmable number of 50% duty cycle pulses on the output pin. Depending on the external sequencer logic these timers can be used for regular step control or for microstep driving.

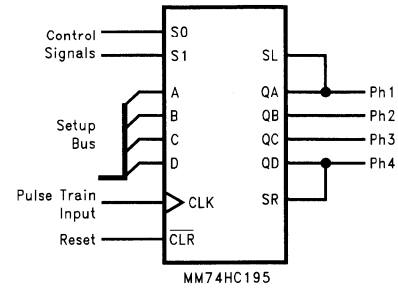
If the user does not require pulse train outputs on all pins the pulse train generators can be used as four independent timers with interrupt capability.



TL/DD/12376–4

FIGURE 5. Standard Stepper Motor Control Waveform (Two Phases On)

Microstepping is used in applications where the step sizes have to be very small to allow smooth transitions from one state to another. Special microstep drivers and motors have been developed. These devices turn the applied pulses into current steps to run the motor. The standard input of these drivers is a pulse train as generated by the COP888GW microcontrollers. To show the basic stepper motor control circuit a standard stepper motor drive circuit is discussed based on a four-phase motor. *Figure 5* shows the waveform required to control a standard motor. CLK is the pulse train generated by the pulse train generator, PH1–PH4 are the outputs to the motor drivers. These signals can be generated by using a simple shift/rotate register.



TL/DD/12376–5

FIGURE 6. Simple Stepper Motor Sequencer

Figure 6 shows an application circuit for a bidirectional stepper motor. The shift register is loaded under software control and then the pulse train is started. More than one motor can be connected in the same manner. Each motor gets its own sequencer the setup bus signals are common for every sequencer. Two control signals, unique for every motor, select the direction and the mode of the sequencer. Mode can be load new pattern, shift left and shift right. The circuit shown above is capable of controlling low current stepper motors used in some applications, if a higher output drive is required appropriate driver stages should be used.

As stepper motors usually have some inertia the pulse train starts up slowly and increases its frequency, under software control, up to full step speed. The same scheme as shown in *Figure 4* can be applied if increase speed is replaced by increase pulse train frequency and decrease speed accordingly to decrease pulse train frequency.

Pulse Train Generator Routines

The first routine sets up the pulse train generators PT1–PT4. Additionally the respective output pins Port E0 for PT1, Port E1 for PT2, Port E2 for PT3 and Port E3 for PT4 can be configured if the user needs to output the data.

Pulse Train Generator Setup

__PT__SETUP __MODULE, __PRESCALER, __PULSES, __OUT, __INTR, __RUN

__MODULE is the pulse train module number

1 = pulse train module 1
2 = pulse train module 2
3 = pulse train module 3
4 = pulse train module 4

__PRESCALER is the prescaler value the range is 0 to 65535.

__PULSES selects the number of pulses: 0 to 65535.

__OUT enables or disables pulse output

0 = output disabled
1 = output enabled

__INTR select if an interrupt is generated on completion of the pulse generation

0 = disable interrupt
1 = enable interrupt

__RUN selects if the pulses are output directly after initialization or at a later time.

0 = don't start pulse train generator

1 = start pulse train generator

The macro will check that all values are in the acceptable range and warn the user if this is not the case.

Pulse train output:

The second routine loads a new number of pulses into the pulse train registers and starts the generation. Setting regarding interrupts and output are not affected. The macro is called in the source code as:

__PT__CHANGE__MODULE,__PULSES

__MODULE is the pulse train module number

1 = pulse train module 1
2 = pulse train module 2
3 = pulse train module 3
4 = pulse train module 4

__PULSES selects the number of pulses: 0 to 65535.

MULTIPLE/DIVIDE

The device constains a multiply/divide block to speed up the multiply/divide operations. To give an idea about the speed increase, Table II shows a comparison of the standard multiply/divide operations done in software with the same operation performed using the MUL/DIV unit.

TABLE II. Multiply/Divide Operation Speed

Type of Operation	MUL/DIV Operation in Software	MUL/DIV Block
Multiply 8 x 8	129 t_C	11 t_C
Multiply 8 x 16	243 t_C	11 t_C
Divide 8 x 8	208 t_C	14 t_C
Divide 24 x 8	827 t_C	14 t_C

Note: Comparisons include loading the registers for the MUL/DIV block and accordingly the RAM cells used for software multiply/divide on standard COP8 microcontrollers. The actual 16 x 8 multiply operation takes one t_C and a 24 x 16 divide takes 2 t_C. However for accurate comparison the register setup times have to taken into account.

The following section provides software modules used to set up the multiply/divide unit and to load the respective registers. Although the MUL/DIV block does not allow signed arithmetic it can be used with some additional software to perform signed operations. The signed operations check for a sign in every number to be divided or multiplied—performs a sign conversion if negative—does the operation and adjusts the output accordingly.

Note: These software modules use constants to show the basic operation example of loading the MUL/DIV registers. These need to be customized in the application software.

Macro for unsigned 8 x 16 multiply

__MUL816U __MUL1 __MUL2
__MUL1 is the 8-bit unsigned multiplier
__MUL2 is the 16-bit unsigned multiplicand

Macro for unsigned 16 x 16 divide

__DIV1616U __DIV1 __DIV2
__DIV1 is the 16-bit unsigned dividend
__DIV2 is the 16-bit unsigned divisor

Macro for signed 6 x 16 multiply

__MUL816S __MUL1 __MUL2
__MUL1 is the 8-bit signed multiplier
__MU21 is the 16-bit signed multiplicand

Macro for signed 16 x 16 divide

__DIV1616S __DIV1 __DIV2
__DIV1 is the 16-bit unsigned dividend
__DIV2 is the 16-bit unsigned divisor

FURTHER APPLICATION HINTS

It was shown that the COP888GW with its new and enhanced functions can operate as a system contoller in various applications. Due to the large number of I/O lines additional features can be included in the system with minimal external hardware effort. For example it was shown in application note AN-673 how a simple two way multiplexed LCD control can be done with a COP8 microcontroller. The feature can be easily adopted to the COP888GW allowing system e.g. a simple plotter device to be controlled. Another application example is a climate control system, where COP888GW can be used to control the heating process. Control algorithms can be generated with National's NeuFuz™ software package. User interfaces can be done as described above with LCD drive. Diagnostics is possible with the on-chip UART.

CAPTURE MODULE MACROS

```
; This macro sets up the cm on the COP888xW including
; prescaler, edge select, interrupt and I/O port configuration
;
.macro  _CT_SETUP, _MODULE, _PRESCALER, _EDGE, _INTR
.mloc   _CMSETUP        ; local variable


.ifstr _MODULE NE 1     ; check for correct use of _MODULE
.ifstr _MODULE NE 2
.error capture module must be selected 1 or 2
.endif
.endif

.if _PRESCALER > 255    ; check for correct use of _PRESCALER
.error capture prescaler out of range -  must be 0 to 255
.endif

.if _EDGE > 1           ; check for correct use of _EDGE
.error select rising (0) or falling (1) edge
.endif

.if _INTR > 1           ; check for correct use of _INTR
.error select interrupt disabled (0) or enabled (1)
.endif

.ifstr _MODULE EQ 1
        ld      b, #CCMR1       ; point to capture control 1
.else
        ld      b, #CCMR2       ; point to capture control 2
.endif
        ld      [b], #0         ; stop capture, clear pending

.ifstr _MODULE EQ 1
        rbit    6, portlc       ; cm 1 input on port L6
        ld      cm1psc, #_PRESCALER     ; load prescaler of cm 1
.else
        rbit    7, portlc       ; cm 2 input on port L7
        ld      cm1psc, #_PRESCALER     ; load prescaler of cm 2
.endif



        _CMSETUP = ((1) OR (_EDGE*010) OR (_INTR*002))
        ld      [b], #_CMSETUP  ; start capture
.endm
```

TL/DD/12376–6

```
;
; macro for capture module (cm) read
;
; This macro reads the contents of the cm on the COP888xW
; to the RAM location of the specified variable
;
.macro  _CT_READ, _MODULE, _VARIABLE
.mloc   _CTREAD

.ifstr _MODULE NE 1     ; check for correct use of _MODULE
.ifstr _MODULE NE 2
.error capture module must be selected 1 or 2
.endif
.endif

.if _VARIABLE > 127     ; check for correct use of _VARIABLE
.error capture variable out of range -  must be 0 to 127
.endif

_CTREAD = _MODULE

        ld      b, #_VARIABLE
        ld      x, #_CTREAD
        ld      a, [x+]
        x       a, [b+]
        ld      a, [x]
        x       a, [b]
.endm
```

TL/DD/12376-7

STANDARD TIMER PWM SETUP MACRO

```
; macro for standard timer pwm setup
;
; This macro configures a standard timer of the COP888xW
; including PWM cycle on-time and init value

.macro  _TIM_SETUP, _NUMBER, _CYCLE, _TON, _INIT

.ifstr _NUMBER NE 1     ; check for correct use of _NUMBER
.ifstr _NUMBER NE 2
.error timer must be selected 1 or 2
.endif
.endif

.if _CYCLE < 48         ; check min cycle time
.warning timer _NUMBER cycle time to low for glitch free PWM output
.endif
.if _CYCLE > 65535      ; check max cycle time
.error timer _NUMBER cycle time too high must be below 2^16
.endif

.if _TON > _CYCLE       ; check on time
.error on time can`t be longer than cycle
.endif

.if _INIT > 1           ; check init value
.error init value can be only 0 or 1
.endif

.if _NUMBER = 1         ; timer 1
        ld      b, #CNTRL
        ld      a, [b]
        and     a, #00f         ; stop all timer activity
        x       a, [b]
        ld      b, #TMR1LO
        ld      [b+], #low(_TON)                ; on-time low
        ld      [b+], #high(_TON)               ; on-time high
        ld      [b+], #low(_CYCLE - _TON)       ; off-time low
        ld      [b], #high(_CYCLE - _TON)       ; off-time high
        ld      b, #T1RBLO
        ld      [b+], #low(_TON)                ; on-time low
        ld      [b], #high(_TON)                ; on-time high

        ld      b, #portgd
.if     _INIT = 0
        rbit    3, [b]
```

TL/DD/12376–8

```
.else
        sbit    3, [b]
.endif

        ld      a, [b-]                         ; dummy read to point
                                                ; to portgc
        sbit    3, [b]                          ; port as output

        ld      b, #CNTRL
        ld      a, [b]
        or      a, #b`10110000                  ; PWM and start
        x       a, [b]
.endif          ; end timer 1 setup

.if _NUMBER = 2         ; timer 2
        ld      b, #T2CNTRL
        ld      [b], #0                 ; stop all activity

        ld      b, #TMR2LO
        ld      [b+], #low(_TON)                ; on-time low
        ld      [b+], #high(_TON)               ; on-time high
        ld      [b+], #low(_CYCLE - _TON)       ; off-time low
        ld      [b+], #high(_CYCLE - _TON)      ; off-time high
        ld      [b+], #low(_TON)                ; on-time low
        ld      [b], #high(_TON)                ; on-time high

        ld      b, #portld
.if     _INIT = 0
        rbit    4, [b]
.else
        sbit    4, [b]
.endif

        ld      a, [b-]                         ; dummy read to point
                                                ; to portlc
        sbit    4, [b]                          ; port as output

        ld      b, #T2CNTRL
        ld      [b], #b`10110000                ; PWM and start
.endif          ; end timer 2 setup
.endm
```

TL/DD/12376-9

STANDARD TIMER PWM OUTPUT MACRO

```
.macro  _TIM_PWM, _NUMBER, _CYCLE, _TON
.mloc   _WAIT0, _WAIT1, WAIT2, WAIT3
.mloc   _MORE0, _MORE1, MORE2, MORE3

.ifstr _NUMBER NE 1     ; check for correct use of _NUMBER
.ifstr _NUMBER NE 2
.error timer must be selected 1 or 2
.endif
.endif

.if _CYCLE < 48         ; check min cycle time
.warning timer _NUMBER cycle time to low for glitch free PWM output
.endif
.if _CYCLE > 65535      ; check max cycle time
.error timer _NUMBER cycle time too high must be below 2^16
.endif

.if _TON > _CYCLE       ; check on time
.error on time can`t be longer than cycle
.endif

        rbit    GIE, PSW        ; disable interrupts !
.warning interupts are disabled for at least _CYCLE tC

.if     _ON > (_CYCLE / 2)

.if     _NUMBER = 1
        ld      [b], ICNTRL
        rbit    T1PNDB, [b]
_WAIT0:
        ifbit   T1PNDB, [b]
        jp      _MORE0
        jp      _WAIT0
_MORE0:
        ld      b, #T1RALO
        ld      [b+], #low(_CYCLE - _TON)       ; off-time low
        ld      [b], #high(_CYCLE - _TON)       ; off-time high
        ld      b, #T1RBLO
        ld      [b+], #low(_TON)                ; on-time low
        ld      [b], #high(_TON)                ; on-time high
.else
        ld      [b], T2CNTRL
        rbit    T2PNDB, [b]
_WAIT1:
        ifbit   T2PNDB, [b]
```

TL/DD/12376-10

```
        jp      _MORE1
        jp      _WAIT1
_MORE1:
        ld      b, #T2RALO
        ld      [b+], #low(_CYCLE - _TON)       ; off-time low
        ld      [b+], #high(_CYCLE - _TON)      ; off-time high
        ld      [b+], #low(_TON)                ; on-time low
        ld      [b], #high(_TON)                ; on-time high
.endif

.else
        ld      [b], PSW
        rbit    T1PNDA, [b]
_WAIT2:
        ifbit   T1PNDA, [b]
        jp      _MORE2
        jp      _WAIT2
_MORE2:
        ld      b, #T1RBLO
        ld      [b+], #low(_TON)                ; on-time low
        ld      [b], #high(_TON)                ; on-time high
        ld      b, #T1RALO
        ld      [b+], #low(_CYCLE - _TON)       ; off-time low
        ld      [b], #high(_CYCLE - _TON)       ; off-time high
.else
        ld      [b], T2CNTRL
        rbit    T2PNDA, [b]
_WAIT3:
        ifbit   T2PNDA, [b]
        jp      _MORE3
        jp      _WAIT3
_MORE3:
        ld      b, #T2RBHI
        ld      [b-], #high(_TON)               ; on-time high
        ld      [b-], #low(_TON)                ; on-time low
        ld      [b-], #high(_CYCLE - _TON)      ; off-time high
        ld      [b],  #low(_CYCLE - _TON)       ; off-time low
.endif

        sbit    GIE, PSW        ; enable interrupts
.endm
```

TL/DD/12376-11

7

PULSE TRAIN GENERATOR SETUP MACRO

```
; macro for pulse train generator (ptg) setup
;
; This macro sets up the ptg on the COP888xW including
; prescaler, # pulses, interrupt and I/O port configuration
;
.macro  _PT_SETUP, _MODULE, _PRESCALER, _PULSES, _OUT, _INTR, _RUN
.mloc _PTSETUP

.if _MODULE < 1          ; check for correct use of _MODULE
.error pulse train generator must be selected 1 to 4
.endif
.if _MODULE > 5
.error pulse train generator must be selected 1 to 4
.endif

.if _PRESCALER > 65535  ; check for correct use of _PRESCALER
.error pulse train prescaler out of range -  must be 0 to 2^16 - 1
.endif

.if _PULSES > 65535  ; check for correct use of _PULSES
.error pulse train: number of pulses out of range -  must be 0 to 2^16 - 1
.endif

.if _INTR > 1            ; check for correct use of _INTR
.error select interrupt disabled (0) or enabled (1)
.endif

.if _RUN > 1           ; check for correct use of _RUN
.error select run later (0) or immedately (1)
.endif

.if _MODULE = 1
        _PTSETUP = 1
        ld      b, #C1PRL       ; point to counter 1 prescaler low
.endif

.if _MODULE = 2
        _PTSETUP = 5
        ld      b, #C2PRL       ; point to counter 1 prescaler low
.endif

.if _MODULE = 3
        _PTSETUP = 1
        ld      b, #C3PRL       ; point to counter 1 prescaler low
.endif
```

TL/DD/12376–12

```
.if _MODULE = 4
        _PTSETUP = 5
        ld      b, #C4PRL       ; point to counter 1 prescaler low
.endif

        ld      [b+], #low(_PRESCALER)  ; load prescaler low
        ld      [b+], #high(_PRESCALER) ; load prescaler high
        ld      [b+], #low(_PULSES)     ; load # pulses low
        ld      [b],  #high(_PULSES)    ; load # pulses high

.if _OUT = 1
        sbit    (_MODULE-1), portec     ; counter _MODULE output on port E3
.endif

.if _MODULE < 3
        ld      b, #CCR1        ; point to counter control 1
.else
        ld      b, #CCR2        ; point to counter control 2
.endif

        rbit    (_PTSETUP+2), [b] ; make sure test bit = 0
        rbit    (_PTSETUP+1), [b] ; clear intr. pending bit

.if _INTR = 1
        sbit    (_PTSETUP+0), [b] ; enable interrupt
.else
        rbit    (_PTSETUP+0), [b] ; disable interrupt
.endif

.if _RUN = 1
        sbit    (_PTSETUP-1), [b] ; start pulse train output
.else
        rbit    (_PTSETUP-1), [b] ; don't start pulse train output
.endif

.endm
```

TL/DD/12376-13

PULSE TRAIN GENERATOR CHANGE MACRO

```
; macro for pulse train generator (ptg) change
;
; This macro changes the ptg on the COP888xW including to
; output a number of pulses and starts the ptg. It does not
; change the output configuration or the interrupt configuration.
;
; Note:
; this routine should only be used after the CxIPND bit was
; set or a pulse train interrupt occured
;

.macro  _PT_CHANGE, _MODULE, _PULSES

.if _MODULE < 1         ; check for correct use of _MODULE
.error pulse train generator must be selected 1 to 4
.endif
.if _MODULE > 5
.error pulse train generator must be selected 1 to 4
.endif

.if _PULSES > 65535  ; check for correct use of _PULSES
.error pulse train: number of pulses out of range -  must be 0 to 2^16 - 1
.endif

.if _MODULE = 1
        ld      b, #C1CTL               ; point to counter 1 register low
        ld      [b+], #low(_PULSES)     ; load # pulses low
        ld      [b],  #high(_PULSES)    ; load # pulses high
        ld      b, #CCR1                ; point to counter control 1
        rbit    2, [b]                  ; reset interrupt pending bit
        sbit    0, [b]                  ; start counter
.endif

.if _MODULE = 2
        ld      b, #C2CTL               ; point to counter 1 register low
        ld      [b+], #low(_PULSES)     ; load # pulses low
        ld      [b],  #high(_PULSES)    ; load # pulses high
        ld      b, #CCR1                ; point to counter control 1
        rbit    6, [b]                  ; reset interrupt pending bit
        sbit    4, [b]                  ; start counter
.endif
```

TL/DD/12376-14

```
.if _MODULE = 3
        ld      b, #C3CTL               ; point to counter 1 register low
        ld      [b+], #low(_PULSES)     ; load # pulses low
        ld      [b],  #high(_PULSES)    ; load # pulses high
        ld      b, #CCR2                ; point to counter control 2
        rbit    2, [b]                  ; reset interrupt pending bit
        sbit    0, [b]                  ; start counter
.endif

.if _MODULE = 4
        ld      b, #C4CTL               ; point to counter 1 register low
        ld      [b+], #low(_PULSES)     ; load # pulses low
        ld      [b],  #high(_PULSES)    ; load # pulses high
        ld      b, #CCR2                ; point to counter control 2
        rbit    6, [b]                  ; reset interrupt pending bit
        sbit    4, [b]                  ; start counter
.endif
.endm
```

TL/DD/12376-15

MULTIPLY/DIVIDE MACROS

```
; macro for 8 x 16 multiplication

.macro  _MUL816U, _MUL1, _MUL2

.if _MUL1 > 255
;.error multiplier out of range
.endif

.if _MUL2 > 65535
.error multiplicant out of range
.endif

        ld      b, #mdr2
        ld      [b+], #_MUL1
        ld      a, [b+]                 ; dummy read to inc b
        ld      [b+], #low(_MUL2)
        ld      [b+], #high(_MUL2)
        ld      [b], #001               ; multiply
.endm

; macro for 16 / 16 unsigned division

.macro  _DIV1616U, _DIV1, _DIV2

.if _DIV1 > (256*256-1)
.error divident out of range
.endif

.if _DIV2 > 65535
.error divisor out of range
.endif

        ld      b, #mdr1
        ld      [b+], #low(_DIV1)
        ld      [b+], #high(_DIV1)
        ld      [b+], #0
        ld      [b+], #low(_DIV2)
        ld      [b+], #high(_DIV2)
        ld      [b], #002               ; divide
        nop                             ; wait one cycle
.endm
```

TL/DD/12376–16

```
; macro for 8 x 16 signed multiplication

.macro  _MUL816S, _MUL1, _MUL2
#SIG
.if     (_MUL1 > 0) & (_MUL2 > 0)       ; positive multiply
#NOSIG
        _MUL816U _MUL1 _MUL2
.endif

#SIG
.if     (_MUL1 > 0) & (_MUL2 < 0)       ; negative multiply
#NOSIG
        _MUL816U _MUL1 -(_MUL2)

        ld      b, #mdr1                ; build one`s complemet
        ld      a, [b]
        xor     a, #0ff
        rc
        adc     a, #1
        x       a, [b+]
        ld      a, [b]
        xor     a, #0ff
        adc     a, #0
        x       a, [b]
.endif

#SIG
.if     (_MUL1 < 0) & (_MUL2 > 0)       ; negative multiply
#NOSIG
        _MUL816U -(_MUL1) _MUL2

        ld      b, #mdr1                ; build one`s complemet
        ld      a, [b]
        xor     a, #0ff
        rc
        adc     a, #1
        x       a, [b+]
        ld      a, [b]
        xor     a, #0ff
        adc     a, #0
        x       a, [b]
.endif
```

TL/DD/12376–17

```
#SIG
.if     (_MUL1 < 0) & (_MUL2 < 0)       ; positive multiply
#NOSIG
        _MUL816U -(_MUL1) -(_MUL2)
.endif
.endm


; macro for 16 / 16 signed division

.macro  _DIV1616S, _DIV1, _DIV2

#SIG
.if     (_DIV1 > 0) & (_DIV2 > 0)
#NOSIG
        _DIV1616U _DIV1 _DIV2
.endif

#SIG
.if     (_DIV1 > 0) & (_DIV2 < 0)
#NOSIG
        _DIV1616U _DIV1 -_DIV2
        ld      b, #mdr1                ; build one`s complemet
        ld      a, [b]
        xor     a, #0ff
        rc
        adc     a, #1
        x       a, [b+]
        ld      a, [b]
        xor     a, #0ff
        adc     a, #0
        x       a, [b]
.endif

#SIG
.if     (_DIV1 < 0) & (_DIV2 > 0)
#NOSIG
        _DIV1616U -_DIV1 _DIV2
        ld      b, #mdr1                ; build one`s complemet
        ld      a, [b]
        xor     a, #0ff
        rc
        adc     a, #1
        x       a, [b+]
```

TL/DD/12376-18

```
        ld      a, [b]
        xor     a, #0ff
        adc     a, #0
        x       a, [b]
.endif

#SIG
.if     (_DIV1 < 0) & (_DIV2 < 0)
#NOSIG
        _DIV1616U -_DIV1 -_DIV2
.endif
.endm
```

TL/DD/12376-19

Simple, Cost Effective A/D Conversion Using COP888EK

National Semiconductor
Application Note 983
Siegfried Rueth

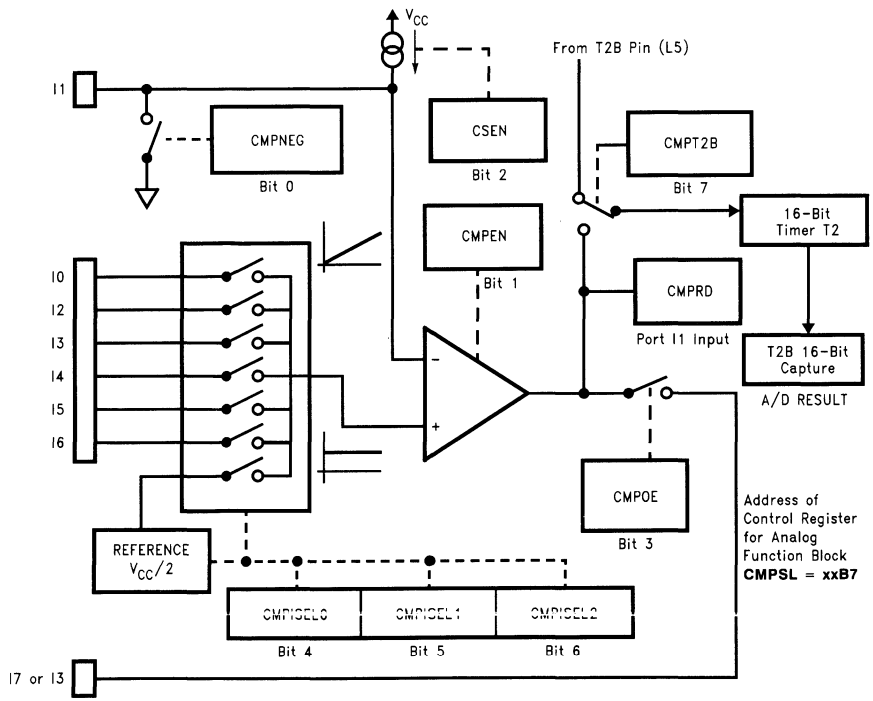
INTRODUCTION

The COP888EK is a member of National Semiconductor's COP888 feature family of microcontrollers fabricated in M²CMOS™ technology. The device was designed using CCM (Configurable Controller Methodology) design techniques, a fast and reliable way to create new derivatives for a fast growing and demanding controller market.

This Application Note describes the device emphasizing the new and powerful features.

KEY FEATURES

- Analog Function Block composed of a multiplexer with a total of 6 external analog inputs, a constant current source, an analog comparator, a software selectable $V_{CC}/2$ reference and a dedicated 16-bit timer/16-bit capture register pair (see below).
- MICROWIRE/PLUS™ serial port with interrupt.
- 8 kbytes of on-board ROM
- 256 bytes of on-board RAM
- Multi-Input Wakeup (8 pins) optionally usable as hardware interrupts with programmable transition
- Three 16-bit timers with two I/O pins assigned and two 16-bit autoreload/capture registers supporting:
  - — Processor independent PWM mode
  - — External Event counter mode
  - — Input Capture mode
- Quiet design (low radiated emissions)
- WATCHDOG™/Clock monitor and Idle Timer
- Two power saving modes, HALT and IDLE
- Twelve multi-source vectored interrupts
- Software selectable I/O options
- Packages:
  - — 28 SO or DIP with 23 I/O's each
  - — 40 DIP with 35 I/O's
  - — 44 PLCC with 39 I/O's



TL/DD/12377–1

FIGURE 1. COP888EK Analog Function Block

THE ANALOG FUNCTION BLOCK

The analog function block allows single slope (integration) type of A/D conversion on 6 analog multiplexed input pins. With this alternate function on the 8-bit I-port (which is an "input only" port in other devices) this derivative received a powerful enhancement.

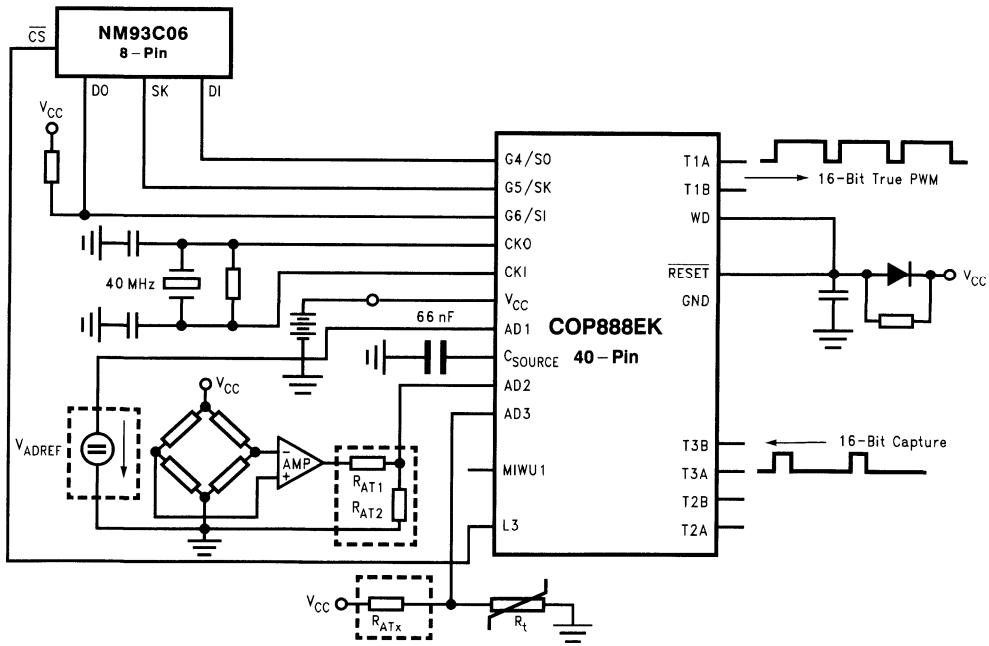
Figure 1 illustrates the principles of the whole block. One of the six I-pins (I0, I2, . . ., I6) or an internally available $V_{CC}/2$ reference can be multiplexed onto the positive terminal of an internal analog comparator. A ramp can be fed to the negative comparator terminal by simply applying a capacitor on pin I1 and initializing the on-chip constant current source. The 16-bit "T2B" capture register completes the analog function block and will be directly triggered from the output of the on-chip comparator. T2B is one of the two 16-bit autoreload/capture registers assigned to Timer 2.

A/D CONSIDERATIONS

Single slope (integration) A/D converters have sources of inaccuracy affecting the quality of the digital result of the conversion. In addition, this serial approach is slower than successive approximation type A/D's and can not be applied in cases where fast conversion is required. The following considerations may help the user to judge if this technique is adequate for the particular A/D conversion task(s) the microcontroller has to perform.

A resolution of 11 bits can be established with a CKI frequency of 2 MHz and a 66 nF capacitor. The linearity of the ramp is dependent on the constant current source, the common mode range of the comparator and the type of capacitor. The optimum linearity can be obtained if the voltage level on the selected input channel does not exceed a maximum of V_{CC} − 1.5V. Therefore it may be necessary to attenuate the input levels as shown in *Figure 2*. In order to maximize the accuracy of the A/D conversion there are some guidelines the user should follow:

- — An accurate reference is required. This can be a fixed reference (for example a reference diode) directly applied to one of the analog inputs. In many applications there is an EEPROM present which could hold a digital reference "burned in" during the manufacturing process. This can eliminate the need for an accurate and expensive crystal oscillator and a much cheaper RC clock can be used instead.
- — The user should limit the levels at the analog inputs to values less than or equal to V_{CC} − 1.5V.
- — Relating all measurements to the on-chip $V_{CC}/2$ reference will help to compensate for temperature variations.
- — Capacitors with values higher than 100 nF may not be useful because of decreasing linearity. For values higher than 33 nF, it is recommended to use styroflex capacitors or similar quality.



TL/DD/12377–2

FIGURE 2. COP888EK with Strain Gauge Transducer

The constant current delivered from the on-chip current source can vary in the range of 10 μA–30 μA. To calculate the minimum resolution, the following equation can be used:

$$X_{RES} = \frac{C \bullet dV}{I_{MAX} \bullet t_c}$$

with:

C = Value of capacitance in [nF]
dV = Range of channel input voltage in [V]
I_{MAX} = Max. current delivered from current source [μA]
t_c = Instruction cycle rate of microcontroller [10/CKI]
X_{RES} = Resolution of Digital value (found in T2B).

APPLICATIONS EXAMPLES

Frequently used sensors in industrial and consumer applications are "NTC's" and strain gauge transducers. *Figure 2* shows a configuration with both types connected. Using the above equation, the minimum resolution at 2 MHz CKI clock input and a 66 nF capacitor, is 10-bits with an input signal varying between 0V and 3V and a V_{CC} voltage of 5V. The typical resolution with these parameters is more than 11-bit and the achievable accuracy is already better than the accuracy of an 8-bit successive approximation type of A/D converter. Doubling the CKI frequency will increase the resolution by 1-bit. A typical software initialization sequence for an A/D conversion (measuring the on-chip $V_{CC}/2$ ref) looks as follows:

```
RESET:
                .
                .
                .
CONVERT:
   LD   CMPSL,#b'11110011   discharge Capacitor connected
                            to 'Csource' (I1); enable Comp.,
                            select timer 'T2B' input to be
                            driven from comparator output;
                            connect internal Vcc/2 ref. as a
                            positive input to the comparator;
                            disable constant current source.
DELAY:
   NOP                      give time to discharge the
   NOP                      capacitor.
                .
                .
   LD   TMR2LO,#0FF         Preset Timer T2
   LD   TMR2HI,#0FF         with #0FFFF, initialize capture
   LD   T2CNTRL,#0E0        mode and start timer.
   LD   CMPSL,#b'11110110   enable constant current source
                            and start charging the capacitor
                            on I1.
ADFIN:
   IFBIT 1,T2CNTRL          Testloop to check if T2B pending
   JP    DONE               flag is set. If this flag is set then
   JP    ADFIN              the A/D conversion is done. The
                            result can be picked up from the
                            T2B capture register.
                .
                .
                .
                                              TL/DD/12377-3
```

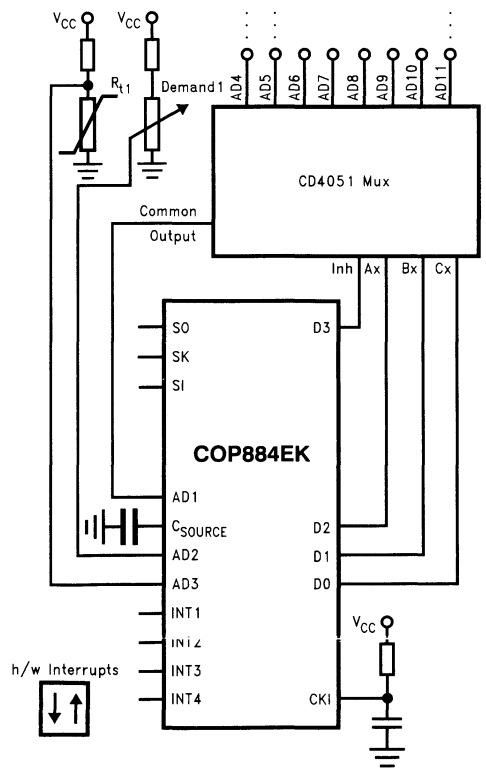
Similarly, the other (external) sources can be measured by appropriately setting bits 4, 5 and 6 in the **CMPSL** register, discharging the capacitor, setting the timer, and charging the capacitor.

A/D CONVERSION TIME

The minimum specified constant current fed into the capacitor connected to pin C_{SOURCE} (i1) is 10 μA and causes the longest possible conversion time. With the values given, the X_{RESmax} = 3960 counts. A 2 MHz crystal will yield a t_c-cycle of 5 μs in a COP800 microcontroller. This means the conversion time to measure one channel is maximum:

$$T = 3960 \times 5\ \mu s = 18.8\ ms$$

To ensure secure and stable measurements however it is recommended to do a minimum of 2 or 3 conversions on a particular input channel and compare subsequent results with each other. Taking the worst case with 3 A/D-cycles plus additional 3 cycles measuring the internal $V_{CC}/2$ reference will result in a total conversion time of 6 × 18.8 ms which gives 113 ms. The processing time to find the final digital value representing the analog level at the A/D channel will not contribute a significant period of time so that a total conversion time of 120 ms is a reasonable assumption for a worst case scenario. Likewise the total conversion time will be half with a CKI frequency of 4 MHz and a capacitance at pin C_{SOURCE} equal to half the original value.



TL/DD/12377-4

FIGURE 3. Expansion of A/D Channels

EXPANDING THE NUMBER OF A/D CHANNELS

Figure 3 shows an inexpensive way to establish additional A/D channels by using a CD4051 analog multiplexer. The COP884EK is a 28-pin version of the COP888EK and provides 3 analog inputs. One input must be used to connect the common output of the external mux which carries the channel selected via the pins (Ax, Bx, Cx). An additional control pin is required to enable the desired channel. The whole setup is capable of handling a total of 10 A/D channels. In many automotive and industrial applications, a controller has to scan temperature sensors, monitor fluid-levels and measure pots which hold "demand-values" for open- or closed-loop control systems. In these applications, timing is typically not a critical requirement. Also, the internal $V_{CC}/2$ reference is most likely adequate to gain sufficient accuracy.

TABLE I. Resolution and Accuracy vs dV and CKI

	Ex1	Ex2	Ex3	Ex4	Ex5
V_{CC}[V]	5	5	5	5	3
dV[V]	3	3	3	3	1.8
CKI[MHz]	1	2	4	5	2
t_c[μs]	10	5	2.5	2	5
X_{RESmin}	8-bit	9-bit	11-bit	12-bit	9-bit
X_{REStyp}	>9-bit	>10-bit	>12-bit	14-bit	>10
Acc	6-bit	7-bit	8-bit	>9-bit	7-bit
C[nF]	33	33	66	100	66

Table I shows some examples of achievable minimum resolution (row: X_{RESmin}) and accuracy (row: Acc) dependent on the V_{CC} voltage, the CKI frequency and the capacitor value (connected to pin I1).

OTHER FEATURES

I. Timers

The COP888EK has three 16-bit timers with two I/O pins assigned and two autoreload/capture registers associated with each.

Figure 4 shows one of the timers configured for PWM mode. The contents of the 2 autoreload registers are copied alternately into the 16-bit timer (upon underflow) one holding the high time and the other holding the low time of the pulses presented to the Timer I/O pin TxA (with x = 1, 2 or 3), thus being able to establish duty-cycles from 1/65535 to 65535/1.

In capture mode both I/O's assigned to a timer can be incorporated into time or frequency measurements. Inputs on TxA trigger measurements in capture register TxRA and inputs to pin TxB trigger measurements in capture register TxRB.

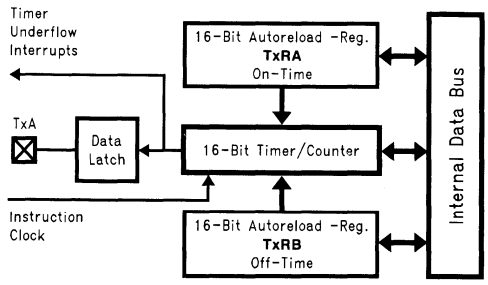


FIGURE 4. Timer in PWM-Mode

Thereby all possible combinations of transitions can be programmed.

There are many interrupt sources associated with all timer modes such as timer underflow, autoreloads on TxRA and TxRB and positive or negative transitions on the timer pins TxA and TxB thus allowing a very flexible use.

In addition, there is the free running 16-bit timer T0 that can be used for time base and to establish the low current idle-mode.

II. Low EMI and Low Current Oscillator

Low radiated emissions are achieved by gradual turn-on output drivers and internal I_{CC} filters on the chip logic and crystal oscillator. All precautions taken lowered the radiated noise by about 20 dB compared to older COP888 family devices.

III. Multi-Input Wakeup/Hardware Interrupts

The 8 pins of the 8-bit L-port can alternately be configured as wakeup inputs. The trigger condition can be either programmed on the "high to low" or "low to high" transition. The Multi-Input Wake-up can be used with the power saving Halt- (with Halt currents less than 10 μA) or Idle mode, to wakeup the device via one of the eight wakeup pins. Alternately these pins can serve as additional hardware interrupts with programmable transition polarity.

CONCLUSIONS

— With the performance and the combination of functions provided (low EMI, multi-channel high resolution A/D, powerful timer structure . . .) the COP888EK is ideally suited for many applications in automotive and industrial/consumer markets.

— The analog function block can be configured to perform integration type A/D conversions with resolutions of up to 14 bits and achievable accuracies significantly better than the 8-bit approximation type deliver.

— The strengths of the COP888EK can be supplemented from National's NeuFuz4™ development environment which supports the generation of intelligent control algorithms for open- and closed loop control systems delivering the final COP8 assembler code as a result.

COP8™ Instruction Set Performance Evaluation

National Semiconductor
Application Note 1042

1.0 OVERVIEW

National offers two families of 8-bit COP8 microcontrollers: Basic family and Feature family. The Feature family offers more on-chip peripheral and nine additional instructions compared to the Basic family. In this report, the COP8 Basic family instruction set performance is evaluated versus that of three competitive microcontrollers, the Motorola M68HC05, the Intel 80C51, and the Microchip PIC16C5X. The architecture, addressing modes, instruction sets and salient features of these microcontrollers are compared. Eight benchmark programs are developed, with each microcontroller programmed with full documentation for each of the benchmarks. Summary tables compare the results relative to both code efficiency and execution time.

The report examines only the instruction set efficiency and speed of execution of the selected microcontrollers. Factors related to on-chip hardware features (RAM/ROM sizes, interrupts, etc.) are not considered. Manufactures offer a variety of options of hardware features, but the instruction set efficiency and execution speed typically remain identical across a manufacturer's microcontroller product line.

When comparing a microcontroller to the competition, it is customary for manufacturers to select benchmark programs which particularly highlight the instruction set of their device. To avoid this phenomenon, general purpose commonly used benchmark routines were selected.

One word of caution—this benchmark report, like all others of its kind, relies on a set of small program fragments. The operations performed by each of these fragments may or may not correspond to the operations required for a particular application. When evaluating a microcontroller for a demanding application, it is important to examine how an individual microcontroller will perform in that particular case.

ARCHITECTURE

Three of the four microcontrollers have a modified Harvard architecture, while a fourth (the Motorola M68HC05) has a Von Neumann architecture (named after John Von Neumann, an early pioneer in the computer field at Princeton). With a Von Neumann architecture, a CPU (Central Processing Unit) and a memory are interconnected by a common address bus and a data bus. Positive aspects of this approach include convenient access to tables stored in ROM and a more orthogonal instruction set. The address bus is used to identify which memory location is being accessed, while the data bus is used to transfer information either from the CPU to the selected memory location or vice versa. Von Neumann was the first to realize that this architectural model could have the memory serve as either program memory or data memory. In earlier computers (both electronic and electromechanical), the program store (often a programmed patchboard) had been completely separate from the data store (often a bank of vacuum tubes or relays).

The single address bus of the Von Neumann architecture is used sequentially to access instructions from program memory and then execute the instructions by retrieving data from and/or storing data in data memory. This means that instruction fetch cannot overlap data access from memory.

A Harvard architecture (names after the Harvard Mark 1 and the early electromechanical computers developed at Harvard by Howard Aiken—another computer pioneer) has separate program memory and data memory with a separate address bus and data bus for each memory. One of the benefits of the Harvard architecture is that the operation of the microcontroller can be controlled more easily in the event of corrupted program counter. A modified Harvard architecture allows accessing data tables from program memory. This is very important with modern day computers, since the program memory is usually ROM (Read Only Memory) while the data memory is usually RAM (Random Access Memory). Consequently, data tables usually need to be in ROM so that they are not lost when the computer is powered down.

The obvious advantage of a Von Neumann architecture is the single address and single data bus linking memory with the CPU. A drawback is that code can be executed from data memory opening up the possibility for undesired operation due to corruption of the program counter or other registers. Alternatively, the advantage of a modified Harvard architecture is that instruction fetch and memory data transfers can be overlapped with a two stage pipeline, which means that the next instruction can be fetched from program memory while the current instruction is being executed using data memory. A drawback is that special instructions are required to access RAM and ROM data values making programming more difficult.

The instructions which cause the "Modified" Harvard architectures of the three microcontrollers are the instructions that provide a data path from program memory to the CPU. These instructions are listed below:

COP8	LAID	Load Accumulator indirect from program memory
80C51	MOVC A, @A + DPTR	Move Constant - Load Accumulator with a "fixed constant" from program memory
PIC16C5X	RETLW	Return Literal to W from program memory

COP8 offers the single-byte LAID instruction which uses the contents of the accumulator to point to a data table stored in the program memory. The data accessed from the program memory is transferred to the accumulator. This instruction can be used for table lookup operations and to read the entire program memory contents for checksum calculations.

The 80C51 family offers a similar instruction. The MC68HC05 family has a Von Neumann architecture where CPU and program memory are interconnected by a common address and data bus.

Microchips's PIC16C5x family offers the RETLW instruction. To do table lookup, the table must contain a string of RETLW instructions. The first instruction just in front of the table of the RETLW instructions, calculates the offset into the table. The table can only be used as a result of a CALL. This instruction certainly does not offer the flexibility of the COP8 LAID instruction and cannot be used to perform a checksum calculation on the entire program memory contents.

The use these instructions is demonstrated in the fourth benchmark program, which entails a three byte table search from a data table in program memory.

2.0 SELECTED MICROCONTROLLERS

COP8

The COP8 has a modified Harvard architecture with memory mapped input/output. The CPU registers include an 8-bit accumulator (A), a 16-bit program counter (PC), two 8-bit data pointers (B and X), an 8-bit stack pointer (SP), an 8-bit processor status word (PSW), and an 8-bit control register (CNTRL). The data memory includes a bank of 16 registers (including the three pointers) which have special attributes. All RAM, I/O ports, and registers (except A and PC) are mapped into the data memory address space. The timer section includes a 16-bit timer, and an associated 16-bit autoreload register. The generic COP8 I/O section includes two 8-bit I/O ports, each with an associated 8-bit configuration register and an associated 8-bit data register. The I/O section also contains one dedicated 8-bit output port with an associated 8-bit data register, one dedicated 8-bit input register, and one special purpose 8-bit I/O port with associated 8-bit configuration register and 8-bit data register.

M68HC05

The M68HC05 has a Von Neumann architecture with memory mapped input/output. The CPU registers include an 8-bit accumulator (A), a 16-bit program counter (PC), an 8-bit index register (X), a stack pointer (SP), and an 8-bit condition code register (CCR). The timer section includes a 16-bit free running counter, two 16-bit counter registers to read the counter, a 16-bit timer input capture register, a 16-bit counter output compare register, an 8-bit timer control register, and an 8-bit timer status register. The I/O section includes three bidirectional 8-bit I/O ports, each with an associated 8-bit data register and an 8-bit direction register. The I/O section also includes a 7-bit input port with an associated input register.

80C51

The 80C51 has a modified Harvard architecture with memory mapped input/output. The CPU registers include an 8-bit accumulator (A), an 8-bit auxilliary register (B) for multiply and divide, a 16-bit program counter (PC), and 8-bit program status word (PSW), an 8-bit stack pointer (SP), and a 16-bit data pointer (DPTR). The register bank consists of eight special working registers R0–R7. Registers R0 and R1 can be used as indirect address pointers to data memory. The timer section includes two 16-bit timers, an 8-bit timer mode register (TMOD), and an 8-bit timer control register (TCON). The I/O section includes four 8-bit I/O ports, each with an associated 8-bit register.

PIC16C5X

The PIC15C5X has a modified Harvard architecture with memory mapped input/output. The PIC16C5X also has a RISC (Reduced Instruction Set Computer) type architecture in that there are only 33 single word basic instructions. Actually these 33 instructions should be expanded to a total of 47 for comparison purposes with the other microcontrollers. This is necessary since 14 of the 33 instructions have a programmable destination bit, which selects one of two destinations for the result of the instruction. Consequently, each of these 14 instructions should be counted as dual instructions. The CPU registers include an 8-bit working register (W) which serves as a pseudo accumulator in that it holds the second operand, receives the literal in the immediate type instructions, and also can be program selected as the destination register. However, a bank of 31 file registers serve as the primary accumulators in that they represent the first operand and also may be program selected as the destination register. The first eight file registers include the real time clock/counter register (RTCC) mapped as F1, the 9-bit program counter (PC) mapped as F2, the 8-bit status word register (SWR) mapped as F3, and the 8-bit I/O port registers mapped as F5–F7. The 8-bit File Select Register (FSR) is mapped as F4, whose low order 5 bits select one of the 31 file registers in the indirect addressing mode. Calling for file F0 in any of the file oriented instructions selects indirect addressing and will use the File Select Register (FSR). It should be noted that file register F0 is not a physically implemented register. The CPU also contains a two level 12-bit hardware push/pop stack for subroutine linkage. The PIC16C5X also has a WATCHDOG™ timer as well as the real time clock/counter register (RTCC), but it does not have any hardware interrupts. Consequently, the counter register RTCC must be program sampled for any overflow.

The number of instructions in a program must also be calibrated differently with the PIC16C5X since the instruction word is 12 bits in length. Consequently, twenty 12-bit word instructions will contain the byte equivalent of thirty COP8 8-bit byte instructions. The larger size instruction word is instrumental in implementing the RISC architecture. It should also be noted that the upper members of the PIC family expand to having a 16-bit instruction word.

3.0 INSTRUCTION SET ANALYSIS

Addressing Modes

COP8

1. Direct
2. Register Indirect
3. Register Indirect with Post Increment/Decrement
4. Immediate
5. Immediate Short
6. Indirect from Program Memory
7. Jump Relative
8. Jump Absolute
9. Jump Absolute Long
10. Jump Indirect

68HC05

1. Inherent
2. Immediate
3. Extended
4. Direct
5. Indexed with no offset
6. Indexed with 8-bit offset
7. Indexed with 16-bit offset
8. Relative

80C51

1. Register
2. Direct
3. Indirect
4. Immediate
5. Relative
6. Absolute
7. Long
8. Indexed

PIC16C5X

1. Data Direct
2. Data Indirect
3. Immediate
4. Program Direct
5. Program Indirect
6. Relative

Instruction Types

COP8

Total basic instructions: 49
Total instructions including addressing modes: 87

1. Arithmetic
2. Load and Exchange
3. Logical
4. Bit Manipulation
5. Conditional
6. Transfer of Control

68HC05

Total basic instructions: 62
Total instructions including addressing modes: 210

1. Register/Memory
2. Read/Modify/Write
3. Branch
4. Control

80C51

Total basic instructions: 51
Total instructions including addressing modes: 111

1. Arithmetic
2. Logical
3. Data Transfer
4. Boolean Variable
5. Program Branching

PIC16C5X

Total basic instructions: 33
Total instructions with 14 dual destination instructions counted: 47

1. Byte-oriented File Register
2. Bit-oriented File Register
3. Literal and Control

COP8 Instruction Set Features

1. Majority of single byte opcode instructions to minimize program size.
2. One instruction cycle for the majority of single byte instructions to minimize program execution time.
3. Many single byte multiple function instructions such as DRSZ.
4. Three memory mapped pointers: Two data pointers (B and X) for register indirect addressing, and one program memory stack pointer (SP) for the software stack.
5. Sixteen memory mapped registers which allow an optimized implementation of certain instructions.
6. Ability to set, reset, and test any individual bit in data memory address space, including the memory mapped I/O ports and associated registers.
7. Register indirect LOAD and EXCHANGE instructions with optional automatic post-incrementing or post-decrementing of the register pointers (Both B and X pointers). This allows for greater efficiency (both in throughput time and program code) in both loading and processing fields in data memory.
8. Unique instructions to optimize program size and throughput efficiency. Some of these instructions are:

DRSZ	IFBNE	DCOR
RETSK	RRC	LAID

9. Forty nine basic instructions.
10. Ten addressing modes provide great flexibility.

M68HC05 Instruction Set Features

1. Very flexible branch structure with 21 different branch instructions.
2. Twelve different read/modify/write instructions.
3. Five bit manipulation instructions (Clear, Set, Branch if Bit Clear, Branch if Bit Set, Bit Test Memory with Accumulator) provide great flexibility.
4. Indexed addressing with options of no offset, 8-bit offset, or 16-bit offset.
5. Data Tables located in page 0 address space (0000–00FF) take advantage of the direct addressing mode for optimal code.
6. Multiply instruction (unsigned, 8 × 8).
7. Sixty two basic instructions.
8. Eight addressing modes provide great flexibility.

80C51 Instruction Set Features

1. Optimized for 8-bit control applications.
2. Provides a variety of fast compact addressing modes for accessing data memory to facilitate operations on small data structures.
3. Offers extensive support for one bit variables, allowing direct bit manipulation in control and logic systems that require Boolean processing.
4. Eight working registers (R0–R7) selected by three bits allow a function code and register address to be combined in one single byte instruction.
5. Register R0 and R1 serve as indirect addressing data memory pointers.
6. Four banks of working registers, with only one bank active at a time, permit fast and effective "context switching".
7. Several register specific instructions referring to the accumulator (A), the accumulator and auxiliary register (B), register pair (AB), the carry flag (C), the data pointer (DPTR), and the program counter (PC) provide great efficiency.
8. Indexed addressing using a base register (either the program counter (PC) or the data pointer (DPTR) and an offset in the accumulator (A) provide great flexibility.
9. Multiply and Divide instruction (using A and B registers).
10. Fifty one basic instructions.
11. Eight addressing modes provide great flexibility.

PIC16C5X Instruction Set Features

1. All single word (12-bit) instructions for compact code efficiency.
2. All instructions are single cycle except the jump type instructions (GOTO, CALL) and failed test instructions (DECFSZ, INCFSZ, BTFSC, BTFSS) which are two cycle.
3. 32 File registers can be addressed directly or indirectly, and serve as accumulators to provide first operand, Working register (W) serves as pseudo accumulator, providing second operand.
4. Working register (W) also serves as destination for literal from program memory in MOVLW (Move Literal to W) and RETLW (Return Literal to W) instructions.
5. Many instructions include a destination bit which selects either the register file or the accumulator as the destination for the result.
6. Four bit manipulation instructions (Set, Clear, Test and Skip if Set, Test and Skip if Clear) provide great flexibility.
7. Status word register (SWR) memory mapped as register file F3 allows testing of status bits (carry, digit carry, zero, power down, and time-out).
8. Program counter (PC) memory mapped as register file F2 allows W to be used as offset register for indirect addressing of program memory.
9. Indirect addressing mode data pointer FSR (file select register) memory mapped as register file F4. Addressing F0 causes FSR to be used to select file register.
10. Literal in RETLW (Return Literal to W) instruction combined with file register mapped program counter allow data tables to be accessed from program memory.
11. Two level 12-bit push/pop hardware stack for subroutine linkage using the CALL and RETLW instructions.
12. WATCHDOG timer.
13. Thirty three basic instructions.
14. Six addressing modes provide great flexibility.

COMPARISON OF SALIENT INSTRUCTION SET FEATURES

Addressing

The three types of indexed addressing (no offset, single byte offset, and dual byte offset) in the M68HC05 provide great flexibility in walking across two or three data fields (two operands, result) simultaneously. The three other microcontrollers achieve the same result with indirect addressing and pointers. The COP8 and 80C51 both have dual pointers for indirect data addressing (B and X pointers in the COP8, R0 and R1 in the 80C51). The PIC16C5X only has one indirect data pointer F4, which is called by using F0. Consequently, processing two multiple byte operands with the PIC16C5X requires alternate loading of the operand addresses into the F4 pointer as the multiple byte data fields are processed.

All four microcontrollers have some form of indirect addressing for program memory, which is very useful in producing jump tables. The COP8 uses the JID (jump indirect) instruction, while the PIC16C51X can address the program counter as file register F2 and add an offset to it. The M68HC05 and 80C51 both use their indexed addressing modes to achieve indirect addressing of program memory. Both jump tables and lookup tables are easily created with indexed addressing. The COP8 uses the LAID (load accumulator indirect) instruction to implement lookup tables, while the PIC16C5X uses the RETLW (return literal to W) instruction as the table entries preceded by a table header which produces the table offset address.

Bit Manipulation

All four microcontrollers have instructions to set, reset, and test individual bits in data memory. However, the 80C51 allows bit manipulation in only 210-bit addressable locations of data memory (16 general purpose byte locations 20–2F, and the rest in the special function registers including the I/O ports). The other three microcontrollers are capable of bit addressing anywhere in data memory. Three of the four microcontrollers have direct bit tests for a bit being either set or reset, while the COP8 test is for bit set only. The 80C51 has two instructions (MOV C, bit and MOV bit, C) to either transfer a bit of data memory to or from the carry. This is very handy for saving a carry and restoring it for later use. The two bit test instructions (BRCLT and BRSET) in the M68HC05 also set the carry to the state of the bit tested. The 80C51 also has four logical (two logical AND, two logical OR) bit manipulation instructions, with the carry and a selected memory bit serving as the two operands. These four instructions are described more fully with the logical instructions.

Input/Output

The four microcontrollers all have mulitple I/O ports, with some of them (COP8, M68HC05) also having dedicated input and/or dedicated output ports. The COP8 and M68HC05 both have configuration registers for each I/O port, with each configuration bit determining whether the associated port bit is selected as input or output. The PIC16C5X contains a TRISTATE® instruction (TRIS) whose operand determines whether or not each associated port bit is put in the TRISTATE condition to serve as an input pin. The 80C51 configures a port by writing ones to TRISTATE the port bit positions selected as inputs. Otherwise the associated 80C51 port bit is selected as an output bit. The COP8 has three addresses associated with each I/O port. The first two addresses select the associated data and configuration register for the port, while the third address selects the actual port pins. If a COP8 port bit is configured as an input, then the associated bit in the data register selects whether the associated input is Hi-Z or weak pull-up. Input/Output manipulation with the I/O ports is demonstrated in the fifth benchmark program.

Increment/Decrement

All four micrcontrollers have accumulator increment and decrement instructions. The register file of the PIC16C5X serves as multiple accumulators. The COP8 has a DRSZ (decrement register and skip if zero) instruction for its sixteen registers located at data memory addresses 0F0–0FF. This register bank includes the B and X data memory pointers and also the SP stack pointer. The PIC16C5X also has both INCFSZ (increment file and skip if zero) and DECFSZ (decrement file and skip if zero) instructions. The M68HC05 increment and decrement instructions can be selected for the accumulator, the index register, or any memory location. The 80C51 increment and decrement instructions can be selected for the accumulator, the eight R registers including the R0 and R1 data pointers, or any memory location (selected either directly or indirectly). The 80C51 also has a DJNZ (decrement and jump if not zero) instruction which can be selected for any of the eight R registers including the R0 and R1 data pointers, or for any memory location selected with a relative address.

Load Memory Immediate

Only the COP8 and the 80C51 have instructions that can load memory with an immediate value. The memory location to be loaded can be selected with either direct or indirect addressing for both microcontrollers.

Post Incrementation or Decrementation of Data Pointers

Only the COP8 has instructions that can post increment, post decrement, or leave the data pointer unchanged. This feature applies to both the COP8 load and exchange instructions, and can be used with either of the two pointers B and X. The feature is very useful when processing multiple byte fields, especially with two operands (additions, subtractions) or an operand and a result (block move). Examples of the usage of this feature can be found in the first three COP8 benchmark programs.

Loop Counting and Data Pointing Testing

Three of the four microcontrollers (excluding M68HC05) have specific instructions to facilitate loop counting. The COP8 DRSZ (decrement register and skip if zero) instruction tests one of sixteen registers (including the two data pointers B and X) and skips the next instruction (a branch back to loop) if the result is zero. The PIC16C5X DECFSZ (decrement file register and skip if zero) is analogous to the COP8 DRSZ instruction. The 80C51 DJNZ (decrement and jump if not zero) instruction combines both the test and jump in a single instruction. These three instructions (COP8 DRSZ, PIC16C5X DECFSZ, 80C51 DJNZ) all operate on a loop counter, but uses the standard BNE (branch if not equal) branch instruction following the decrement to test if the counter is zero. The BNE instruction tests the M68HC05 zero status flag which is active with the decrement instruction. Only the COP8 contains a test data pointer uction (IFBNE—If B pointer not equal). This instruction is used to sense the end of a program loop where multiple byte fields are being processed.

The 80C51 CJNE (compare and jump if not equal) instruction used in the immediate addressing mode provides an alternative method of loop counting. This instruction is analogous to the COP8 IFBNE instruction but combines the test with the jump in a single instruction.

It should be noted that neither the COP8 IFBNE instruction nor the 80C51 CJNE instruction require the use of a counter, but rather depend on an end of field comparison for loop termination. Consequently these instructions are used in loops where a data pointer is being used to walk across a mulitple byte field. Examples of the usage of these two instructions can be found in the first three benchmarks.

Branch and Subroutine Call Instructions

All four micrcontrollers have absolute address branching, and three of the four (excluding the PIC16C5X) also have relative address branching. Only the COP8 and the M68HC05 have single byte branching. The COP8 single byte branching is with relative addressing, where six bits provide relative addressing of −31 to +32. The M68HC05 single byte branching is with no offset indexed addressing, where the index register X provides the absolute address. All of the myriad conditional branch instructions of the M68HC05 use two bytes, with the second bytes providing relative addressing of −127 to +128.

Three of the four microcontrollers have a software stack (excluding the PIC16C5X) where the return address is stored when a subroutine is called. Consequently, multiple levels of subroutine and interrupt nesting are possible with the software stack. The PIC16C5X has a two level hardware stack where the return address is stored with a subroutine call. Thus only two levels of subroutine nesting are possible with the PIC16C5X. The other three microcontrollers have absolute address subroutine calls, with the 80C51 also having a relative address call. Only the M68HC05 provides a single byte subroutine call, using no offset indexed addressing (with the index register providing the absolute address.

Return Instructions

All four microcontrollers have return from subroutine instructions and three of the four (excluding the PIC16C5X) have return from interrupt instructions. The PIC16C5X does not have an interrupt.

The COP8 has two return from subroutine instructions, RET and RETSK. The RET is the normal return from subroutine instruction, while the RETSK instruction returns from the subroutine and then skips the next instruction. These two instructions are very useful in passing back flag information from the subroutine in lieu of actually using a flag such as the carry. This is demonstrated in the fourth benchmark (table search) where a flag needs to be passed back from the subroutine to indicate whether or not the search was successful. The COP8 simply uses the RETSK instruction to indicate a successful search, whereas each of the other three microcontrollers need to set up the carry as the flag.

The PIC16C5X return from subroutine instruction is RETLW (return and place literal in W). The literal serves as an immediate data value from memory. The use of this instruction is demonstrated in the fourth benchmark (table search), where the program memory data table consists of a block of RETLW instructions preceded by the table header.

Comparison and Conditional Branch Instructions

Three of the four microcontrollers (excluding the PIC16C5X) have comparison instructions. The COP8 has two comparison instructions, IFEQ (if equal) and IFGT (if greater than). These comparison instructions compare the accumulator versus an immediate value or a number from memory (selected with either direct or indirect addressing). The M68HC05 has two comparison instructions, CMP (compare) and BIT (bit test memory with accumulator). The CMP instruction compares the accumulator versus an immediate value or a number from memory (selected with either direct or indexed addressing). The BIT instruction performs a logical AND comparison between the accumulator and an immediate value or a number from memory (selected with either direct or indexed addressing). The 80C51 has one comparison instruction CJNE (compare and jump if not equal) which is combined with a branch in the same instruction. All four microcontrollers have bit test instructions (including test carry.)

The M68HC05 has a myriad of test conditions and branch instructions, including BCC (branch if carry clear), BCS (branch if carry set), BRCLR (branch if memory bit clear), BRSET (branch if memory bit set), BEQ (branch if equal), BHI (branch if higher), BHS (branch if higher or same), BLO (branch if lower), BLS (branch if lower or same), BMI (branch if minus), BNE (branch if not equal), and BPL (branch if plus).

The 80C51 includes seven conditional branch instructions, consisting if JC (jump if carry), JNC (jump if no carry), JNZ (jump if not zero), JZ (jump if zero), JB (jump if memory bit set), JNB (jump if memory bit not set), and JBC (jump if memory bit set and clear bit).

The COP8 includes seven conditional test instructions, IFBIT (test if memory bit is set), IFC (if carry), IFNC (if no carry), IFEQ (if equal), IFGT (if greater than), IFBNE (if B pointer not equal), and DRSZ (decrement register and skip if zero). These seven instructions will all cause the next instruction to be skipped if the test is successful. The skipped instruction can either be a branch or some function (such as setting a bit flag or returning from a subroutine) that can be contained in one instruction.

The PIC16C5X includes four test instructions, consisting of DECFSZ (decrement file register and skip if zero), INCFSZ (increment file register and skip if zero), BTFSC (bit test file register and skip if bit clear), and BTFSS (bit test file register and skip if bit set). These four instructions will all cause the next instruction to be skipped if the test is successful.

Logical Instructions (AND, OR, Exclusive OR)

All four microcontrollers have a full complement of the logical instructions, with the accumulator and a selected memory location (using either direct or indirect addressing) serving as the two operands. Three of the four microcontrollers (excluding the PIC16C5X) can also perform the logical instructions with the accumulator and an immediate value serving as the operands. The 80C51 contains four logical bit instructions which perform either the logical AND or logical OR between the carry bit and a selected memory bit, with either polarity of the memory bit being selectable. The M68HC05 contains a logical compare instruction, which performs a logical AND comparison between the accumulator and an immediate value or a number from memory (selected with either direct or indexed addressing).

Shift and Rotate Instructions

Only the M68HC05 has shift instructions, both logical left shift and logical right shift. All four microcontrollers have a right rotate instruction through carry (none bit loop), and three of the four (excluding the COP8) also have a left rotate through carry. The 80C51 also has both a right and left rotate direct (eight bit loop).

Complement Instructions

Three of the four microcontrollers (excluding the COP8) have a complement instruction for the accumulator. The 80C51 also has two complement bit instructions, with the first being used to toggle the carry and the second to complement selected accumulator bits.

BDC Decimal Correct

Only the COP8 and the 80C51 provide instructions to expedite BCD processing. The DCOR (decimal correct) instruction of the COP8 is used following an add or subtract instruction to achieve the correct BCD result. Note that a hex 66 must be added to the first operand to start the addition process. The DA (decimal adjust) instruction of the 80C51 is used following an add instruction to achieve the correct BCD result. The decimal adjust instruction does not work following subtraction. Consequently, BCD subtraction in the 80C51 must be implemented by adding the complement of the subtrahend (second operand) and then using the decimal adjust instruction. BCD subtraction is demonstrated in the third benchmark program.

Exchange and Swap

Only the COP8 and the 80C51 have exchange instructions between the accumulator and memory. Both microcontrollers can select either direct or indirect addressing for the associated exchange memory selection. The 80C51 also has a XCHD A, @Ri (exchange digit) instruction which exchanges the low order nibble (digit) of the accumulator with the low order nibble of the indirectly addressed memory location selected.

All of the microcontrollers except the M68HC05 have a SWAP instruction which interchanges the low and high order nibbles of the accumulator. With the PIC16C5X, the SWAPF instruction can be selected for any of the register bank accumulators.

Push and Pop Instructions

Only the 80C51 has PUSH and POP instructions to augment operations with the software stack. Consequently, only the 80C51 has the ability to store temporary data in the software stack as well as subroutine and interrupt return addresses.

4.0 BENCHMARK PROGRAMS

1. FIVE BYTE BLOCK MOVE

This benchmark program moves a block of five data byte from one location to another.

2. FOUR BYTE BINARY ADDITION

This benchmark subroutine adds two four byte binary numbers and replaces the first operand with the result, like an adding machine (A + B replaces A). The carry flag is used to indicate an overflow.

3. FOUR BYTE PACKED BCD SUBTRACTION

This benchmark subroutine subtracts two eight digit packed BCD numbers and replaces the first operand with the result, like an adding machine. (A − B replaces A). The carry flag is used to indicate a negative result.

4. THREE BYTE TABLE SEARCH

This benchmark subroutine searches a program memory data table for a three byte match. A flag indicates whether or not the search was successful, with the address of the first byte of a matched string being returned.

5. INPUT/OUTPUT MANIPULATION

This benchmark compares two 8-bit I/O ports P1 and P2. If they are equal, a nine is output as the least significant digit of a third port P3. If P1 is greater than P2, then P2 data is output on P1. If P1 is < P2, then the higher order digit of P1 is copied to the lower order digit position of P3.

6. SERIAL INPUT/OUTPUT WITH OFFSET TABLE

This benchmark subroutine inputs a sequence of byte couplets, each of which consists of an address followed by a data byte. The address is used to access a data table to produce an offset value which is added to the data byte. The updated data byte is output while the next couplet's address byte is input.

7. TIMEKEEPING

This timekeeping benchmark program emulates a real time clock, keeping track of hours, minutes, and seconds in packed BCD format. The program is interrupt driven, using a timer interrupt.

8. SWITCH ACTIVATED FIVE SECOND LED

This benchmark samples a switch input to activate a five second output for turning on an LED. The switch is debounced with a 50 ms program delay both on opening and closure.

5.0 BENCHMARK DATA

This section provides the benchmark programs for each microcontroller. For each instruction, the byte count and instruction cycle count is given. The total cycle count is multiplied by the fastest cycle time of the selected microcontroller to yield the total benchmark execution time.

Microcontrollers	Instruction Cycle Time	XTAL
COP8	1.0 μs	10 MHz
M68HC05	0.5 μs	4 MHz
80C51	1.0 μs	12 MHz
PIC16C5X	0.5 μs	8 MHz

COP8 BENCHMARK # 1—FIVE BYTE BLOCK MOVE

This benchmark moves only a block of five data bytes from a specific source location to a specific destination location.

```
MOVE:   LD      B, #14      ; 1/1  Source address to B pointer
        LD      X, #50      ; 2/3  Destination address to X pointer
LOOP:   LD      A, [B+]     ; 1/2  Load A with source byte
        X       A, [X+]     ; 1/3  Source byte to destination
        IFBNE   #3          ; 1/1  Test (modulo 16) if block move finished (19-16=3)
        JP      LOOP        ; 1/3  Loop back if not finished

                             7 BYTES
                            47 CYCLES
```

68HC05 BENCHMARK # 1—FIVE BYTE BLOCK MOVE

This benchmark moves only a block of five data bytes from a specific source location to a specific destination location.

```
MOVE:   LDX     #5          ; 2/2  Load X with block length
LOOP:   LDA     24, X       ; 2/4  Load A with source byte (start at block top)
        STA     54, X       ; 2/5  Store A at destination (start at block top)
        DECX                ; 1/3  Decrement X
        BNE     LOOP        ; 2/3  Loop back if block transfer not finished

                             9 BYTES
                            76 CYCLES
```

80C51 BENCHMARK # 1—FIVE BYTE BLOCK MOVE

This benchmark moves only a block of five data bytes from a specific source location to a specific destination location.

```
MOVE:  MOV     R0, #20        ;  2/1  Load R0 pointer with source address
       MOV     R1, #50        ;  2/1  Load R1 pointer with destination address
LOOP:  MOV     A, @R0         ;  1/1  Load A with source byte
       MOV     @R1, A         ;  1/1  Store source byte at destination
       INC     R1             ;  1/1  Increment destination pointer
       INC     R0             ;  1/1  Increment source pointer
       CJNE    R0, #25, LOOP; 3/2  Compare & loop back if block transfer not finished

                              11 BYTES
                              32 CYCLES
```

PIC16C5X BENCHMARK # 1—FIVE BYTE BLOCK MOVE

This benchmark moves only a block of five data bytes from a specific source location to a specific destination location.

```
       TEMP    EQU F29        ;
       BASE    EQU F30        ;       Source @ F20-F24 Destination @ F25-F29
       CNTR    EQU F31        ;
MOVE:  MOVLW   5              ;  1/1  Load W with length of block
       MOVWF                  ;  1/1  Store block length in CNTR
       MOVLW   20             ;  1/1  Load W with address of source
LOOP:  MOVF    BASE, W            1/1  Source address to W
       MOVWF   F4             ;  1/1  Source address to F4 (indirect pointer)
       MOVF    F0, W          ;  1/1  Source byte to W
       MOVWF   TEMP           ;  1/1  Source byte to TEMP
       MOVLW   5              ;  1/1  Destination/source address difference to W
       ADDWF   F4             ;  1/1  Add to pointer to get destination address
       MOVF    TEMP, W        ;  1/1  Source byte to W
       MOVWF   F0             ;  1/1  Source byte to destination
       INCF    BASE           ;  1/1  Increment BASE
       DECFSZ  CNTR           ;  1/1  Test if block move finished
       GOTO    LOOP           ;  1/2  Loop back if not finished

                              14 WORDS (Equivalent to 21 BYTES)
                              62 CYCLES
```

COP8 BENCHMARK #2—FOUR BYTE BINARY ADDITION

This benchmark adds two four byte binary numbers and replaces the first operand with the result. This emulates an adding machine addition, where A + B replaces A. The benchmark is programmed as a subroutine, with the carry flag indicating an overflow.

```
ADDITION:  LD        B, #10        ; 1/1  Set up address of 1st operand in B pointer
           LD        X, #20        ; 2/3  Set up address of 2nd operand in X pointer
           RC                      ; 1/1  Reset carry
LOOP:      LD        A, [X+]       ; 1/3  Load 2nd operand to A
           ADC       A, [B]        ; 1/1  Add 1st operand to 2nd operand
           X         A, [B+]       ; 1/2  Result replaces 1st operand
           IFBNE     #14           ; 1/1  Test if addition finished
           JP        LOOP          ; 1/3  Loop back if not finished
           RET                     ; 1/5  Return from subroutine

                                   10 BYTES
                                   48 CYCLES
```

M68HC05 BENCHMARK #2—FOUR BYTE BINARY ADDITION

This benchmark adds two four byte binary numbers and replaces the first operand with the result. This emulates an adding machine addition, where A + B replaces A. The benchmark is programmed as a subroutine, with the carry flag indicating an overflow.

```
ADDITION:  LD        X, #16        ; 2/2  Load index register with 1st operand address
           CLC                     ; 1/2  Clear carry
LOOP:      LDA       X             ; 1/3  Load A with 1st operand
           ADC       4, X          ; 2/4  Add 2nd operand to A
           STA       X             ; 1/4  Replace 1st operand with result
           INCX                    ; 1/3  Increment index register
           CPX       #20           ; 2/2  Compare X with end of field
           BNE       LOOP          ; 2/3  Loop back if addition not finished
           RTS                     ; 1/6  Return from subroutine

                                   13 BYTES
                                   85 CYCLES
```

80C51 BENCHMARK #2—FOUR BYTE BINARY ADDITION

This benchmark adds two four byte binary numbers and replaces the first operand with the result. This emulates an adding machine addition, where A + B replaces A. The benchmark is programmed as a subroutine, with the carry flag indicating an overflow.

```
ADDITION:  MOV     R1, #20          ;  2/1  Load R1 pointer with address of 2nd operand
           MOV     R0, #16          ;  2/1  Load R0 pointer with address of 1st operand
           CLR     C                ;  1/1  Clear carry
LOOP:      MOV     A, @R0           ;  1/1  Load 1st operand to A
           ADD     A, @R1           ;  1/1  Add 2nd operand to A
           MOV     @R0, A           ;  1/1  Replace 1st operand with result
           INC     R1               ;  1/1  Increment 2nd operand pointer
           INC     R0               ;  1/1  Increment 1st operand pointer
           CJNE    R0, #20, LOOP    ;  3/2  Test if finished and loop back if not
           RET                      ;  1/2  Return from subroutine

                                    14 BYTES
                                    33 CYCLES
```

PIC16C5X BENCHMARK #2—FOUR BYTE BINARY ADDITION

This benchmark adds two four byte binary numbers and replaces the first operand with the result. This emulates an adding machine addition, where A + B replaces A. The benchmark is programmed as a subroutine, with the carry flag indicating an overflow.

```
           STATUS  EQU F3           ;       Status register
           BASE    EQU F26          ;       1st operand @ F16-F19 2nd operand @ F20-F23
           CNTR    EQU F25          ;
           TEMP    EQU F24          ;
ADDITION:  MOVLW   4                ;  1/1  Load W with byte count (length of field)
           MOVWF   CNTR             ;  1/1  Store byte count in CNTR
           MOVLW   20               ;  1/1  Load W with address of 2nd operand
           MOWWF   BASE             ;  1/1  Store address of 2nd operand in BASE
           BCF     STATUS, 0        ;  1/1  Clear carry (carry is bit 0 of F3 register)
LOOP:      MOVF    BASE, W          ;  1/1  2nd operand address to W
           MOVWF   F4               ;  1/1  2nd operand address to F4 (indirect pointer)
           MOVF    F0, W            ;  1/1  2nd operand to W
           MOVWF   TEMP             ;  1/1  2nd operand to TEMP
           MOVLW   4                ;  1/1  Byte count to W
           SUBWF   F4               ;  1/1  Subtract from pointer to get 1st operand address
           MOVF    TEMP, W          ;  1/1  2nd operand to W
           ADDWF   F0               ;  1/1  Add 2nd operand to 1st operand
           MOVLW   4                ;  1/1  Byte count to W
           ADDWF   F4               ;  1/1  Add to pointer to restore 2nd operand address
           INCF    BASE             ;  1/1  Increment BASE
           DECFSZ  CNTR             ;  1/1  Test if addition finished
           GOTO    LOOP             ;  1/2  Loop back if addition not finished
           RETLW   0                ;  1/2  Return from subroutine

                                    19 WORDS (Equivalent to 28 1/2 BYTES)
                                    62 CYCLES
```

COP8 BENCHMARK #3—FOUR BYTE PACKED BCD SUBTRACTION

This benchmark subtracts two eight digit packed BCD numbers (four bytes each) and replaces the minuend (first operand) with the result. This emulates an adding machine subtraction, where A - B replaces A. The benchmark is programmed as a subroutine, with the carry flag being used to indicate a positive or negative result (carry set for negative result). The BCD decimal correct (DCOR) command is used following the subtraction to achieve the correct BCD result.

```
BCDSUBT:  LD      B, #14        ;  1/1  Set up address of 1st operand in B pointer
          LD      X, #20        ;  2/3  Set up address of 2nd operand in X pointer
          SC                    ;  1/1  Set carry for no input borrow to subtraction
LOOP:     LD      A, [X+]       ;  1/3  Load 2nd operand to A
          X       A, [B]        ;  1/1  Exchange 1st and 2nd operands
          SUBC    A, [B]        ;  1/1  Subtract 2nd from 1st operand
          DCOR    A             ;  1/1  Decimal correct result of subtraction
          X       A, [B+]       ;  1/2  Result replaces 1st operand
          IFBNE   #2            ;  1/1  Test (modulo 16) if subtraction finished (18-16=2)
          JP      LOOP          ;  1/3  Loop back if not finished
          IFNC                  ;  1/1  Test if result negative (borrow=no carry)
          JP      NEGR          ;  1/3  Jump if result negative
          RC                    ;  1/1  Reset carry to indicate positive result
          RET                   ;  1/5  Return from subroutine
NEGR:     SC                    ;  1/1  Set carry for no input borrow to subtraction
          LD      B, #14        ;  1/1  Reinitialize B pointer to start of result
LUP:      CLR     A             ;  1/1  Clear A to set up subtraction from zero
          SUBC    A, [B]        ;  1/1  Subtract result from zero
          DCOR    A             ;  1/1  Decimal correct new result
          X       A, [B+]       ;  1/2  Store new result
          IFBNE   #2            ;  1/1  Test (modulo 16) if subtraction finished (18-16=2)
          JP      LUP           ;  1/3  Loop back if not finished
          SC                    ;  1/1  Set carry to indicate negative result
          RET                   ;  1/5  Return from subroutine

                                25 BYTES
                                97 CYCLES
```

M68HC05 BENCHMARK #3—FOUR BYTE PACKED BCD SUBTRACTION

This benchmark subtracts two eight digit packed BCD numbers (four bytes each) and replaces the minuend (first operand) with the result. This emulates an adding machine subtraction, where A - B replaces A. The benchmark is programmed as a subroutine, with the carry flag being used to indicate a positive or negative result (carry set for negative result). Note in the M68HC05 subtraction that the carry represents the borrow, unlike some microcontrollers where the carry represents the absence of borrow in subtraction. The M68HC05 does not have any decimal correct or decimal adjust instructions, so the BCD correction must be program implemented. The correction algorithm for BCD subtraction consists of subtracting six whenever a nibble borrow occurs from the BCD subtraction result for each digit.

```
          TEMP    EQU $91          ;
          FLAG    EQU $92          ;
BCDSUBT:  JSR     INIT             ;  2/5  Call initialization subroutine
LOOP:     LDA     X                ;  1/3  Load A with 1st operand
          SBC     4, X             ;  2/4  Subtract 2nd operand from A
          JSR     CORRECT          ;  2/5  Call correction subroutine
          BNE     LOOP             ;  2/3  Loop back if subtraction not finished
          BCS     NEGR             ;  2/3  Branch to NEGR (neg. result) if carry
          RTS                      ;  1/6  Return from subroutine
NEGR:     JSR     INIT             ;  2/5  Call initialization subroutine
LUP:      CLRA                     ;  1/3  Clear A
          SBC     X                ;  1/3  Subtract result of previous subtraction from zero
          JSR     CORRECT          ;  2/5  Call correction subroutine
          BNE     LUP              ;  2/3  Loop back if subtraction from zero not finished
          SEC                      ;  1/2  Set carry to indicate negative result
          RTS                      ;  1/6  Return from subroutine
INIT:     LD      X, #16           ;  2/2  Load index register with 1st operand address
          CLC                      ;  1/2  Clear carry (reset borrow)
          CLRA                     ;  1/2  Clear A
          STA     FLAG             ;  2/4  Clear FLAG register
          RTS                      ;  1/6  Return from initialization subroutine
CORRECT:  STA     X                ;  1/4  Replace 1st operand with result
          CLRA                     ;  1/3  Clear A
          BCC     BYP1             ;  2/3  Branch if carry set (no borrow)
          BSET    0, FLAG          ;  2/5  Set flag if carry (save carry value)
          LDA     #$60             ;  2/2  Load A with packed BCD 60
BYP1:     BHCC    BYP2             ;  2/3  Branch if half carry set (no half borrow)
          ADD     #$06             ;  2/2  Add packed BCD 06 to A
BYP2:     STA     TEMP             ;  2/4  Store A in TEMP
          LDA     X                ;  1/3  Load A with result of first subtraction
          SUB     A, TEMP          ;  2/3  Subtract TEMP from previous result
          BRSET   0, FLAG, NXT     ;  3/5  Restore carry (flag value to carry)
NXT:      STA     X                ;  1/4  Store new result
          INCX                     ;  1/3  Increment index register
          CPX     #20              ;  2/2  Compare X with end of field
          RTS                      ;  1/6  Return from correction subroutine

                                   54 BYTES
                                   567 CYCLES
```

80C51 BENCHMARK # 3—FOUR BYTE PACKED BCD SUBTRACTION

This benchmark subtracts two eight digit packed BCD numbers (four bytes each) and replaces the minuend (first operand) with the result. This emulates an adding machine subtraction, where A - B replaces A. The benchmark is programmed as a subroutine, with the carry flag being used to indicate a positive or negative result (carry set for negative result). It should be noted in the 80C51 subtraction that the carry represents the borrow, unlike some microcontrollers where the carry represents the absence of borrow in subtraction.

The BCD decimal adjust (DA) command only works following addition, not subtraction. Consequently, the BCD subtraction must be implemented as an addition by adding the complement of the subtrahend (2nd operand) to the 1st operand. This complement is achieved by subtracting the subtrahend from a packed BCD 99 (binary 10011001), and then adding one to the result. The following are examples of this complementation procedure:

1. 61−49 = 61 + (99−49) + 1 = 61 + 50 + 1 = 12 (mod 100) (Output borrow shows positive result)
2. 49−61 = 49 + (99−61) + 1 = 49 + 38 + 1 = 88 (No output borrow indicates negative result)
   Negative result correction: - (99−88 + 1) = −12

```
BCDSUBT:  MOV   R2, #4          ;  2/1  Load R2 with byte count
          MOV   R1, #20         ;  2/1  Load R1 with address of 2nd operand
          MOV   R0, #16         ;  2/1  Load R0 with address of 1st operand
          CLR   C               ;  1/1  Clear carry for no input borrow to subtraction
LOOP:     MOV   A, #099H        ;  2/1  Load accumulator with packed BCD 99
          SUBB  A, @R1          ;  1/1  Subtract 2nd operand from packed BCD 99
          ADD   A, #01          ;  2/1  Add one to result
          DA    A               ;  1/1  Decimal correct result
          ADD   A, @R0          ;  1/1  Add 1st operand to result
          DA    A               ;  1/1  Decimal correct new result
          MOV   @R0, A          ;  1/1  Replace 1st operand with result
          INC   R1              ;  1/1  Increment 2nd operand pointer
          INC   R0              ;  1/1  Increment 1st operand pointer
          DJNZ  R2, LOOP        ;  2/2  Test if finished and loop back if not
          JC    NEGR            ;  2/2  Test and jump if negative result
          RET                   ;  1/2  Return from subroutine (no carry: positive result)
NEGR:     CLR   C               ;  1/1  Clear carry for no input borrow to subtraction
          MOV   R0, #16         ;  2/1  Load R0 with address of result
LUP:      MOV   A,#099H         ;  2/1  Load accumulator with packed BCD 99
          SUBB  A, @R0          ;  1/1  Subtract result from packed BCD 99
          ADD   A, #01          ;  2/2  Add one to result
          DA    A               ;  1/1  Decimal correct result
          MOV   @R0, A          ;  1/1  Store result
          INC   R0              ;  1/1  Increment pointer
          CJNZ  R0, #20, LUP    ;  3/2  Test if finished and loop back if not
          SETB  C               ;  1/1  Set carry to indicate negative result
          RET                   ;  1/2  Return from subroutine (carry: negative result)

                                39 BYTES
                                91 CYCLES
```

PIC16C5X BENCHMARK #3—FOUR BYTE PACKED BCD SUBTRACTION PAGE 1 OF 2

This benchmark subtracts two eight digit packed BCD numbers (four bytes each) and replaces the minuend (first operand) with the result. This emulates an adding machine subtraction, where A - B replaces A. The benchmark is programmed as a subroutine, with the carry flag being used to indicate a positive or negative result (carry set for negative result). The PIC16C5X does not have any decimal adjust or decimal correct instructions, so the BCD correction must be program implemented. The correction algorithm for BCD subtraction consists of subtracting six whenever a nibble borrow occurs from the BCD subtraction result for each digit.

```
            STATUS  EQU F3          ;       Status register
            FLAG    EQU F27         ;
            BASE    EQU F26         ;       1st operand @ F16-F19 2nd operand @ F20-F23
            CNTR    EQU F25         ;
            TEMP    EQU F24         ;
BCDSUBT:    MOVLW   4               ; 1/1   Load W with byte count (length of field)
            MOVWF   CNTR            ; 1/1   Store byte count in CNTR
            MOVLW   20              ; 1/1   Load W with address of 2nd operand
            MOVWF   BASE            ; 1/1   Store address of 2nd operand in BASE
            BSF     STATUS, 0       ; 1/1   Set carry (reset borrow) (STATUS register bit 0)
LOOP:       MOVF    BASE, W         ; 1/1   2nd operand address to W
            MOVWF   F4              ; 1/1   2nd operand address to F4 (indirect pointer)
            MOVF    F0, W           ; 1/1   2nd operand to W
            MOVWF   TEMP            ; 1/1   2nd operand to TEMP
            MOVLW   4               ; 1/1   Byte count to W
            SUBWF   F4              ; 1/1   Subtract from pointer to get 1st operand address
            CALL    CORRECT         ; 1/2   Call CORRECT subroutine
            MOVLW   4               ; 1/1   Byte count to W
            ADDWF   F4              ; 1/1   Add to pointer to restore 2nd operand address
            DECFSZ  CNTR            ; 1/1   Test if subtraction finished
            GOTO    LOOP            ; 1/2   Loop back if subtraction not finished
            BTFSS   STATUS, 0       ; 1/1   Test if output borrow (no carry)
            GOTO    NEGR            ; 1/2   Branch to NEGR (neg. result) if output borrow
            BCF     STATUS, 0       ; 1/1   Clear carry to indicate positive result
            RETLW   0               ; 1/2   Return from subroutine
NEGR:       MOVLW   4               ; 1/1   Load W with byte count (length of field)
            MOVWF   CNTR            ; 1/1   Store byte count in CNTR
            MOVLW   16              ; 1/1   Load W with address of result (old 1st operand)
            MOVWF   BASE            ; 1/1   Store address of result in BASE
            BSF     STATUS, 0       ; 1/1   Set carry (reset borrow)
LUP:        MOVF    BASE, W         ; 1/1   Result address to W
            MOVWF   F4              ; 1/1   Result address to F4 (indirect pointer)
            MOVF    F0, W           ; 1/1   Result to W
            MOVWF   TEMP            ; 1/1   Result to TEMP
            CLRF    F0              ; 1/1   Zero to result (old 1st operand)
            CALL    CORRECT         ; 1/2   Call CORRECT subroutine
            DECFSZ  CNTR            ; 1/1   Test if subtraction finished
            GOTO    LUP             ; 1/2   Loop back if subtraction not finished
            BSF     STATUS, 0       ; 1/1   Set carry to indicate negative result
            RETLW   0               ; 1/2   Return from subroutine

                                    35 WORDS
```

PIC16C5X BENCHMARK #3—FOUR BYTE BCD SUBTRACTION (Continued)

```
CORRECT:  CLRF      FLAG            ;  1/1  Clear FLAG register
          MOVF      TEMP, W         ;  1/1  Move TEMP to W
          SUBWF     F0              ;  1/1  Subtract W from register
          CLRF      TEMP            ;  1/1   Clear TEMP
          BTFSC     STATUS, 0       ;  1/1  Test if carry (no borrow)
          GOTO      BYP1            ;  1/2  Branch if carry (no borrow)
          MOVLW     60H             ;  1/1  Load W with packed BCD 60
          ADDWF     TEMP            ;  1/1  Add W to TEMP
BYP1:     BSF       FLAG, 0         ;  1/1  Carry to FLAG bit 0
          BTFSC     STATUS, 1       ;  1/1  Test if digit carry (no digit borrow)
          GOTO      BYP2            ;  1/2  Branch if digit carry (no digit borrow)
          MOVLW     06H             ;  1/1  Load W with packed BCD 06
          ADDWF     TEMP            ;  1/1  Add W to TEMP
BYP2:     MOVF      TEMP, W         ;  1/1  Move TEMP to W
          BSF       STATUS, 0       ;  1/1  Set carry (reset borrow)
          SUBWF     F0              ;  1/1  Subtract W from previous result
          BTFSC     FLAG, 0         ;  1/1  Test if FLAG bit 0 reset
          BCF       STATUS, 0       ;  1/1  Clear carry (set borrow)
          INCF      BASE            ;  1/1  Increment BASE
          RETLW     0               ;  1/2  Return from CORRECT subroutine

                                    20 WORDS

                    TOTAL: 55 WORDS (Equivalent to 82 1/2 BYTES)
                                    274 CYCLES
```

COP8 BENCHMARK #4—THREE BYTE TABLE SEARCH

This benchmark searches a 200 byte table (resident in program memory) for a three byte character string, which may be resident anywhere in the lookup table (not necessarily on three byte boundaries). The type of return from subroutine (RETSK versus RET) indicates the success or failure of the search, with the address of the first byte of a matched string being returned in BASE. The benchmark is programmed as a subroutine, which should be located following the table since the LAID instructions in the subroutine must be located in the same 256 byte page of program memory.

```
          CHAR1 = 00          ;       Three bytes of character string
          CHAR2 = 01          ;       resident in registers 00, 01, 02
          CHAR3 = 02          ;
          SIZE = 0F0          ;
          BASE = 0F1          ;
          TEMP = 0F2          ;
TBLSRCH:  LD      B, #SIZE    ;  2/3  Address of SIZE to B pointer
          LD      [B+], #198  ;  2/2  Table size (200) minus 2 to SIZE
          LD      [B], #0     ;  2/2  Table base address of 0 to BASE
SEARCH:   LD      A, [B+]     ;  1/2  1st byte of table address to A
          X       A, [B]      ;  1/1  Save 1st byte address in TEMP
          LD      A, [B]      ;  1/1  Restore 1st byte table address to A
          LAID                ;  1/3  Load 1st of 3 test bytes from prog. memory table
          IFEQ    A, CHAR1    ;  3/4  Test if 1st byte match
          JP      SEARCH2     ;  1/3  First byte match
          JP      FAIL        ;  1/3  Fail if mismatch
SEARCH2:  LD      A, [B]      ;  1/1  Restore 1st byte address to A
          INC     A           ;  1/1  Increment address to get 2nd byte table address
          X       A, [B]      ;  1/1  Save 2nd byte address in TEMP
          LD      A, [B]      ;  1/1  Restore 2nd byte table address to A
          LAID                ;  1/3  Load 2nd of 3 test bytes from prog. memory table
          IFEQ    A, CHAR2    ;  3/4  Test if 2nd byte match
          JP      SEARCH3     ;  1/3  Second byte match
          JP      FAIL        ;  1/3  Fail if mismatch
SEARCH3:  LD      A, [B]      ;  1/1  Restore 2nd byte address to A
          INC     A           ;  1/1  Increment address to get 3rd byte table address
          X       A, [B]      ;  1/1  Save 3rd byte address in TEMP
          LD      A, [B]      ;  1/1  Restore 3rd byte table address to A
          LAID                ;  1/3  Load 3rd of 3 test bytes from prog. memory table
          IFEQ    A, CHAR3    ;  3/4  Test if 3rd byte match
SUCCESS:  RETSK               ;  1/5  RETURN and SKIP if three byte match found
FAIL:     LD      A, [B-]     ;  1/2  Decrement B pointer to select BASE
          LD      A, [B]      ;  1/1  Restore 1st byte address to A from BASE
          INC     A           ;  1/1  Increment 1st byte address
          X       A, [B]      ;  1/1  Save new 1st byte address in BASE
          DRSZ    SIZE        ;  1/3  Decrement SIZE and skip if table search finished
          JMP     SEARCH      ;  2/3  Continue table search
          RET                 ;  1/5  RETURN if three byte match not found
```

*First search iteration fails with first byte 42 BYTES
 mismatch, second search iteration successful 77 CYCLES*

68HC05 BENCHMARK #4—THREE BYTE TABLE SEARCH

This benchmark searches a 200 byte table (resident in program memory) for a three byte character string, which may be resident anywhere in the lookup table (not necessarily on three byte boundaries). The status of the carry bit indicates the success or failure of the search, with the address of the first byte of a matched string being returned in BASE. The benchmark is programmed as a subroutine.

```
TBLSRCH:  LDA     #198          ;  2/2  Set up table size (200) minus 2
          STA     SIZE          ;  2/4  Save table size
          CLX                   ;  1/3  Set up table base address of 0
          STX     BASE          ;  2/4  Save table base address
SEARCH:   LDA     X             ;  2/4  Get first byte from table
          CMP     CHAR1         ;  2/3  Compare 1st byte with CHAR1
          BNE     FAIL          ;  2/3  Fail
          INCX                  ;  1/3  Set up address of 2nd byte from table
          LDA     X             ;  2/4  Get second byte from table
          CMP     CHAR2         ;  2/3  Compare 2nd byte with CHAR2
          BNE     FAIL          ;  2/3  Fail
          INCX                  ;  1/3  Set up address of 3rd byte from table
          LDA     X             ;  2/4  Get third byte from table
          CMP     CHAR3         ;  2/3  Compare 3rd byte with CHAR3
          BNE     FAIL          ;  2/3  Fail
SUCCESS:  SEC                   ;  1/2  Set carry to indicate three byte match found
          RTS                   ;  1/6  Return
FAIL:     LDX     BASE          ;  2/3  Restore base address
          INCX                  ;  1/3  Increment base address to get new 1st byte address
          STX     BASE          ;  2/4  Save new base address
          CPX     SIZE          ;  2/3  Compare new base address with table size
          BNE     SEARCH        ;  2/3  Continue table search
          CLC                   ;  1/2  Reset carry to indicate no three byte match found
          RTS                   ;  1/6  Return

                                40 BYTES
                                80 CYCLES*
```

*First search iteration fails with first byte mismatch, second search iteration successful

80C51 BENCHMARK #4—THREE BYTE TABLE SEARCH

This benchmark searches a 200 byte table (resident in program memory) for a three byte character string, which may be resident anywhere in the lookup table (not necessarily on three byte boundaries). The status of the carry bit indicates the success or failure of the search, with the address of the first byte of a matched string being returned in DPTR. The benchmark is programmed as a subroutine.

```
TBLSRCH:  MOV     DPTR, #TBLBASE  ;  3/2  Set up table starting address
          MOV     R2, #198        ;  2/1  Set up table size (200) minus 2
SEARCH:   CLR     A               ;  1/1  Clear accumulator
          MOVC    A, @A + DPTR    ;  1/2  Get 1st byte from program memory table
          CJNE    A, CHAR1, FAIL  ;  3/2  Compare 1st byte with CHAR1
          MOV     A, #1           ;  2/1  Load accumulator with offset of 1
          MOVC    A, @A + DPTR    ;  1/2  Get 2nd byte from table
          CJNE    A, CHAR2, FAIL  ;  3/2  Compare 2nd byte with CHAR2
          MOV     A, #2           ;  2/1  Load accumulator with offset of 2
          MOVC    A, @A + DPTR    ;  1/2  Get 3rd byte from table
          CJNE    A, CHAR3, FAIL  ;  3/2  Compare 3rd byte with CHAR3
SUCCESS:  SET     C               ;  1/1  Set carry to indicate three byte match found
          RET                     ;  1/2  Return from subroutine
FAIL:     INC     DPTR            ;  1/2  Increment data pointer to get new
                                          1st byte address
          DJNZ    R2, SEARCH      ;  2/2  Decrement size and test if
                                          table search finished
          CLR     C               ;  1/1  Clear carry to indicate no three
                                          byte match found
          RET                     ;  1/2  Return from subroutine

                                  29 BYTES
                                  30 CYCLES*
```

*First search iteration fails with first byte mismatch, second search iteration successful

PIC16C5X BENCHMARK #4—THREE BYTE TABLE SEARCH

This benchmark searches a 200 word table (resident in program memory) for a three byte character string, which may be resident anywhere in the lookup table (not necessarily on three word boundaries). The status of the carry bit indicates the success or failure of the search, with the address of the first byte of a matched string being returned in OFFSET. The benchmark is programmed as a subroutine, which in turn calls the table header (TABLE) as a subroutine. This table header should immediately precede the 200 word table.

```
TBLSRCH:  MOVLW   198                    ;  1/1  Set up table size (200) minus 2
          MOVWF   SIZE                   ;  1/1  Save table size
          MOVLW   1                      ;  1/1  Initialize table offset from table header
          MOVWF   OFFSET                 ;  1/1  Save offset
SEARCH:   MOVF    W                      ;  1/1  Load offset
          CALL    TABLE                  ;  1/2  Get 1st byte from table
          SUBWF   CHAR1                  ;  1/1  Subtract 1st test byte from CHAR1
          BTFSS   STATUS, Z              ;  1/1  Test if result is zero (F3, bit 2)
          GOTO    FAIL1                  ;  1/2  Fail if mismatch
          INCF    OFFSET                 ;  1/1  Increment offset to set up 2nd byte
          MOVF    W                      ;  1/1  Load offset
          CALL    TABLE                  ;  1/2  Get 2nd byte
          SUBWF   CHAR2                  ;  1/1  Subtract 2nd test byte from CHAR2
          BTFSS   STATUS, Z              ;  1/1  Test if result is zero (F3, bit 2)
          GOTO    FAIL2                  ;  1/2  Fail if mismatch
          INCF    OFFSET                 ;  1/1  Increment offset to set up 3rd byte
          MOVF    W                      ;  1/1  Load offset
          CALL    TABLE                  ;  1/2  Get 3rd byte
          SUBWF   CHAR3                  ;  1/1  Subtract 3rd test byte from CHAR3
          BTFSS   STATUS, Z              ;  1/1  Test if result is zero (F3, bit 2)
          GOTO    FAIL3                  ;  1/2  Fail if mismatch
SUCCESS:  DECF    OFFSET                 ;  1/1  Decrement offset to return to 2nd byte
          DECF    OFFSET                 ;  1/1  Decrement offset to return to 1st byte
          BSF     STATUS, C              ;  1/1  Set carry (F3, bit 0) to indicate match
                                                 found
          RETLW   0                      ;  1/2  Return
FAIL3:    DECF    OFFSET                 ;  1/1  Decrement offset to set up 2nd byte
          DECF    OFFSET                 ;  1/1  Decrement offset to set up 1st byte
FAIL1:    INCF    OFFSET                 ;  1/1  Increment offset to set up new 1st byte
FAIL2:    DECFSZ  SIZE                   ;  1/1  Decrement size and skip if table search
                                                 finished
          GOTO    SEARCH                 ;  1/2  Continue table search
          BCF     STATUS, C              ;  1/1  Clear carry (F3, bit 0) to indicate no
                                                 match found
          RETLW   0                      ;  1/2  Return

TABLE:    ADDWF   PC                     ;  1/1  Add offset to PC
          RETLW   Data 1                 ;  1/2  1st entry of data table
          RETLW   Data 2                 ;  -/-  2nd entry of data table

*First search iteration fails with first byte 34 WORDS (Equivalent to 51 BYTES)
mismatch, second search iteration successful  52 CYCLES*

          RETLW   Program table entry 1  ;  1/2  This 200 word table contains 300 bytes
          RETLW   Program table entry 2  ;  1/2
          ----    ---------              ----
          RETLW   Program table entry 200;  1/2
```

COP8 BENCHMARK #5—INPUT/OUTPUT MANIPULATION

This benchmark compares two 8-bit I/O ports P1 and P2. If they are equal, a nine is output as the least significant digit (lower nibble) of a third port P3. If port P1 is greater than port P2, then port P2 is output on port P1. If port P1 is less than port P2, then the most significant digit of Port P1 is copied to the least significant digit of Port P3.

```
            PORTLD = 0D0                ;
            PORTLC = 0D1                ;
            PORTLI = 0D2                ;
            PORTGD = 0D4                ;
            PORTGC = 0D5                ;
            PORTGI = 0D6                ;
            PORTD = 0DC                 ;

PORTCMP:    LD          B, #PORTLI      ; 2/3  Load B pointer (PORTL is P1)
            LD          X, #PORTGI      ; 2/3  Load X pointer (PORTG is P2)
            SC                          ; 1/1  Initialize carry (no borrow) for subtraction
            LD          A, [X]          ; 1/1  Load P2
            SUBC        A, [B]          ; 1/1  Subtract P1 from P2
            IFC                         ; 1/1  Test if P2 greater or equal to P1
            JP          POS             ; 1/3  Branch if result positive
NEG:        LD          A, [B-]         ; 1/2  Decrement B pointer
            LD          [B-], #0FF      ; 2/2  Configure PORTL (P1) as output port
            LD          A, [X]          ; 1/3  Load P2 to A
            X           A, [B]          ; 1/1  Output P2 to P1
            JP          FIN             ; 1/3  Jump to finish
POS:        IFEQ        A, #0           ; 2/2  Test if result zero
            JP          EQUAL           ; 1/3  Branch if zero
            LD          A, [B]          ; 1/1  Load P1 to A
            SWAP        A               ; 1/1  Swap nibbles of A
            X           A, PORTD        ; 2/3  Result to P3 (PORTD)
            JP          FIN             ; 1/3  Jump to finish
EQUAL:      LD          PORTD, #09      ; 3/3  Digit 9 to P3 (PORTD)
FIN:                    ---

                                        26 BYTES

                        P1 < P2         24 CYCLES
                        P1 = P2         21 CYCLES
                        P1 > P2         22 CYCLES
```

68HC05 BENCHMARK #5—INPUT/OUTPUT MANIPULATION

This benchmark compares two 8-bit I/O ports P1 and P2. If they are equal, a nine is output as the least significant digit (lower nibble) of a third port P3. If Port P1 is greater than port P2, then port P2 is output on Port P1. If Port P1 is less than Port P2, then the most significant digit of Port P1 is copied to the least significant digit of Port P3.

```
            PORTA       EQU $00          ;       Port P1
            PORTB       EQU $01          ;       Port P2
            PORTC       EQU $02          ;       Port P3
            DDRA        EQU $04          ;       PORTA configuration register
            DDRB        EQU $05          ;       PORTB configuration register
            DDRC        EQU $06          ;       PORTC configuration register
            TEMP        EQU $9F          ;       Temporary register

PORTCMP:    CLA                          ; 1/3   Clear A
            STA         DDRA             ; 2/4   Configure Port A as input port (P1)
            STA         DDRB             ; 2/4   Configure Port B as input port (P2)
            DECA                         ; 1/3   Decrement A to all ones
            STA         DDRC             ; 2/4   Configure Port C as output port (P3)
            LDA         PORTB            ; 2/3   Load A with Port P2 data
            CMP         A, PORTA         ; 2/3   Compare Port P1 data with Port P2 data
            BPL         POS              ; 2/3   Branch if comparison result (P2-P1) positive
NEG:        LDA         #$FF             ; 2/2   Load A with all ones
            STA         DDRA             ; 2/4   Configure Port A as output port (P1)
            LDA         PORTB            ; 2/3   Port P2 data to A
            STA         PORTA            ; 2/4   Output Port P2 data to Port P1
            BRA         FIN              ; 2/3   Branch to FIN
POS:                    BEQ EQUAL        ; 2/3   Branch if comparison result equal
            LDA         PORTA            ; 2/3   Port P1 data to A
            AND         A, #$F0          ; 2/2   Extract high order nibble of Port P1 data
            LSRA                         ; 1/3   Logical shift A right one bit four times
            LSRA                         ; 1/3   in order to shift upper nibble
            LSRA                         ; 1/3   down into lower nibble position
            LSRA                         ; 1/3
            STA         PORTC            ; 2/4   Output result to Port P3
            BRA         FIN              ; 2/3   Branch to FIN
EQUAL:      LDA         #9               ; 2/2   Load A with digit 9
            STA         PORTC            ; 2/4   Output digit 9 to Port P3
FIN:        ---                          ; -/-

                                         42 BYTES

                        P1 < P2          53 CYCLES
                        P1 = P2          36 CYCLES
                        P1 > P2          42 CYCLES
```

80C51 BENCHMARK #5—INPUT/OUTPUT MANIPULATION

This benchmark compares two 8-bit I/O ports P1 and P2. If they are equal, a nine is output as the least significant digit (lower nibble) of a third port P3. If Port P1 is greater than Port P2, then Port P2 is output on Port P1. If Port P1 is less than Port P2, then the most significant digit of Port P1 is copied to the least significant digit of Port P3.

```
PORTCMP:    MOV         P1, #0FF        ;  3/2  Configure Port P1 for input
            MOV         P2, #0FF        ;  3/2  Configure Port P2 for input
            CLR         C               ;  1/1  Clear carry
            MOV         A, P2           ;  2/1  Load A with Port P2 data
            SUBB        A, P1           ;  2/1  Subtract Port P1 data from A
            JNC         POS             ;  2/2  Jump TO POS if no carry from subtract
NEG:        MOV         A, P2           ;  2/1  Load A with Port P2 data
            MOV         P1, A           ;  2/1  Output Port P2 data to Port P1
            AJMP        FIN             ;  2/2  Jump to FIN
POS:        JZ          EQUAL           ;  2/2  Jump to EQUAL if subtraction result zero
            MOV         A, P1           ;  2/1  Load A with Port P1 data
            SWAP        A               ;  1/1  Swap nibbles of A
            ANL         A, #00F         ;  2/1  Extract lower nibble of A
            MOV         P3, A           ;  2/1  Output Port P1 upper nibble to Port P3
            AJMP        FIN             ;  2/2  Jump to FIN
EQUAL:      MOV         P3, #9          ;  3/2  Output digit 9 to Port P3
FIN:        ---                         ;  -/-

                                        33 BYTES

                        P1 < P2         17 CYCLES
                        P1 = P2         11 CYCLES
                        P1 > P2         13 CYCLES
```

PIC16C5X BENCHMARK #5—INPUT/OUTPUT MANIPULATION

This benchmark compares two 8-bit I/O ports P1 and P2. If they are equal, a nine is output as the least significant digit (lower nibble) of a third Port P3. If Port P1 is greater than Port P2, then Port P2 is output on Port P1. If Port P1 is less than Port P2, then the most significant digit of Port P1 is copied to the least significant digit of Port P3.

```
            STATUS      EQU F3          ;       Status register
            PORTA       EQU F5          ;       Serves as Port P3
            PORTB       EQU F6          ;       Serves as Port P2
            PORTC       EQU F7          ;       Serves as Port P1
            TEMP        EQU F8          ;       Temporary register

PORTCMP:    MOVLW       FFH             ; 1/1   Load W with all ones
            TRIS        PORTB           ; 1/1   Tristate PORTB
            TRIS        PORTC           ; 1/1   Tristate PORTC
            MOVF        PORTB, W        ; 1/1   Load W with PORTB input
            MOVWF       TEMP            ; 1/1   Store PORTB input in TEMP
            MOVF        PORTC, W        ; 1/1   Load W with PORTC input
            BSF         STATUS, 0       ; 1/1   Set carry (bit 0 of STATUS)
            SUBWF       TEMP            ; 1/1   Subtract P1 from P2
            CLRW                        ; 1/1   Clear W
            TRIS        PORTA           ; 1/1   Configure PORTA as output port
            BTFSC       STATUS, 0       ; 1/1   Test if P2 less than P1
            GOTO        POS             ; 1/2   Branch to POS if test fails
NEG:        TRIS        PORTC           ; 1/1   Change PORTC to output port
            MOVF        PORTB, W        ; 1/1   Load W with PORTB (P2) input
            MOVWF       PORTC           ; 1/1   P2 input data output to PORTC (P1)
            GOTO        FIN             ; 1/2   Branch to FIN
POS:        BTFSC       STATUS, 2       ; 1/1   Test if subtraction result non-zero
            GOTO        EQUAL           ; 1/2   Branch to EQUAL if test fails
            MOVF        PORTC, W        ; 1/1   Load W with PORTC (P1) input
            MOVWF       TEMP            ; 1/1   P1 input to TEMP
            SWAPF       TEMP, W         ; 1/1   Swap nibbles of TEMP with result to W
            MOVWF       PORTA           ; 1/1   Output result from W to PORTA (P3)
            GOTO        FIN             ; 1/2   Branch to FIN
EQUAL:      MOVLW       9               ; 1/1   Digit 9 to W
            MOVWF       PORTA           ; 1/1   Digit 9 output to PORT A (P3)
FIN:        ---                         ; -/-

                                        25 WORDS (Equivalent to 37 1/2 BYTES)

                        P1 < P2         21 CYCLES
                        P1 = P2         18 CYCLES
                        P1 > P2         17 CYCLES
```

COP8 BENCHMARK #6—SERIAL INPUT/OUTPUT WITH OFFSET TABLE

This benchmark inputs a sequence of byte couplets, with each couplet consisting of an 8-bit offset address followed by a data byte. The address is used to access a program memory data table to produce an offset value which is added to the second byte of the input couplet. The updated data byte is output while the address byte of the next couplet is being input. The benchmark is written as a subroutine, with the size of the input stream being an input parameter.

The program is written using the MICROWIRE/PLUS fast burst technique, which utilizes a 500 kHz burst clock SK (instruction cycle clock divided by 2). Each byte of the I/O stream utilizes nine burst clock periods, yielding an effective byte rate of 55.6 kHz.

*** Setting the Microwire busy bit to initialize the Microwire must occur at the same time in each alternating 18 cycle loop for this program to work. Alternate cycle definitions are as follows:

```
                    Even cycle:    New address input for table lookup offset value
                                   Previous data result output

                    Odd cycle:     New data input added to offset value from table lookup
                                   New data input returned as output

        PORTGD = 0D4                ;       PORTG configuration register
        PORTGC = 0D5                ;       PORTG data register
        SIOR = 0E9                  ;       Serial Input/Output register
        CNTRL = 0EE                 ;       Control register
        PSW = 0EF                   ;       Program Status Word
        SIZE = 0F0                  ;       Size of data stream
        BUSY = 2                    ;       Microwire busy bit in PSW register

UPDATE: LD      PORTGC, #030    ; 3/3  Configure G4, G5 as outputs SO, SK to
        LD      PORTGD, #0      ; 3/3    select uwire master mode (SI is G6)
        INC     A               ; 1/1  Increment byte count to compensate for one
        X       A, SIZE         ; 2/3    byte throughput and store result in SIZE
        LD      B, #CNTRL       ; 2/3  Set up B pointer for CNTRL register
        LD      [B+], #8        ; 2/2  Select uwire master with divide by 2 clock
        RC                      ; 1/1  Initialize carry
LOOP:   IFNC                    ; 1/1  1 1  Test if no carry
        JP      BYP1            ; 1/3  1 3  Branch if no carry
        NOP                     ; 1/1  1 -  NOP for delay compensation for 18 cycle loop
        ADD     A, SIOR         ; 3/4  4 -  Add uwire input data to table offset value
        SBIT    BUSY, [B]       ; 1/1  1 -  ***Set uwire BUSY bit to start microwire
        RC                      ; 1/1  1 -  Reset carry
        JP      BYP2            ; 1/3  3 -  Branch
BYP1:   X       A, SIOR         ; 2/3  - 3  New address input & previous result output
        SBIT    BUSY, [B]       ; 1/1  - 1  ***Set uwire BUSY bit to start microwire
        LAID                    ; 1/3  - 3  Offset table lookup from program memory
        SC                      ; 1/1  - 1  Set carry
BYP2:   DRSZ    SIZE            ; 1/3  3 3  Decrement SIZE and test if result zero
        JP      LOOP            ; 1/3  3 3  Loop back if zero test fails
        RET                     ; 1/5    Return from subroutine

                                31 BYTES
                                18 CYCLES per Loop
```

M68HC05 BENCHMARK #6—SERIAL INPUT/OUTPUT WITH OFFSET TABLE

This benchmark inputs a sequence of byte couplets, with each couplet consisting of an 8-bit offset address followed by a data byte. The address is used to access a data table to produce an offset value which is added to the second byte of the input couplet. The updated data byte is output while the address byte of the next couplet is being input. The benchmark is written as a subroutine, with the size of the input stream being an input parameter.

The program is written using the SPI with the divide by 2 fast clock option selected, which yields a 1 MHz burst clock SCK (instruction cycle clock divided by 2). Each byte of the I/O stream utilizes 27 burst clock periods (equivalent to 13.5 μs), yielding an effective byte rate of 74.1 kHz.

***Setting the SPI system enable bit SPE to initialize the SPI must occur at the same time in each alternating 27 cycle loop for this program to work. Alternate cycle definitions are as follows:

```
                Even cycle:   New address input for table lookup offset value
                              Previous data result output
                Odd cycle:    New data input added to offset value from table lookup
                              New data input returned as output

        PORTD   EQU $03       ;       SPI Interface port
        SPCR    EQU $0A       ;       Serial Peripheral Control Register
        SPSR    EQU $0B       ;       Serial Peripheral Status Register
        SPDR    EQU $0C       ;       Serial Peripheral Data I/O Register

UPDATE: INCA                  ; 1/3  Increment byte count to compensate for one
        STA     A, SIZE       ; 2/4    byte throughput and store result in SIZE
        LDA     #$040         ; 2/2  Set up data for SPCR register
        STA     SPCR          ; 2/4  Enable SPI master with divide by 2 clock
        CLC                   ; 1/2  Initialize carry
LOOP:   BCC     BYP1          ; 2/3  2 3  Test and branch if no carry
        ADD     SPDR          ; 2/3  3 -  Add SPI input data to table offset value in A
        STA     SPDR          ; 2/4  4 -  Output result to SPDR for SPI output
        BSET    6, SPCR       ; 2/5  5 -  ***Turn on SPI system enable bit
        CLC                   ; 1/2  2 -  Clear carry
        BRA     BYP2          ; 2/3  3 -  Branch
BYP1:   LDA     X             ; 1/3  - 3  NOP for delay compensation for 27 cycle loop
        LDX     SPDR          ; 2/3  - 3  SPI input data address to index register
        BSET    6, SPCR       ; 2/5  - 5  ***Turn on SPI system enable bit
        LDA     X             ; 1/3  - 3  Load A with table lookup offset value
        SEC                   ; 1/2  - 2  Set carry
BYP2:   DEC     SIZE          ; 2/5  5 5  Decrement SIZE
        BNE     LOOP          ; 2/3  3 3  Test and branch if result equal to zero
        RTS                   ; 1/6       Return from subroutine

                              31 BYTES
                              27 CYCLES per Loop
```

80C51 BENCHMARK #6—SERIAL INPUT/OUTPUT WITH OFFSET TABLE

This benchmark inputs a sequence of byte couplets, with each couplet consisting of an 8-bit offset address followed by a data byte. The address is used to access a program memory data table to produce an offset value which is added to the second byte of the input couplet. The updated data byte is output while the address byte of the next couplet is being input. The benchmark is written as a subroutine, with the size of the input stream being an input parameter.

The program is written using the 8-bit Shift Register Mode 0, with the serial data being either transmitted or received (not both simultaneously) with the least significant bit first. The clock rate for this mode is 1/12 that of the on-chip oscillator frequency of 12 MHz, which equates to an I/O baud rate of 1 MHz. This is equivalent to the instruction cycle time of 1 μs. In Shift Register Mode 0, the RXD line is used for both data input and output, while the TXD line is used for the output clock. Consequently, the terms "RXD" and "TXD" are misleading in this mode of serial I/O.

The program takes 52 instruction cycles per couplet loop, which equates to 26 instruction cycles (26 μs) per byte. This yields an effective byte rate of 38.5 kHz.

```
            SIZE        EQU R2          ;

UPDATE:     MOV         SIZE, A         ; 1/1  Save and increment byte count to
            INC         SIZE            ; 1/1    compensate for one byte throughput
            MOV         DPTR, #TBLBASE  ; 3/2  Set up table starting address
LOOP:       CLR         C               ; 1/1  Clear carry
SINP:       MOV         SCON, #010H     ; 3/2  Initialize SCON for Mode 0 and enable receive
STALL1:     JNB         SCON.RI, STALL1 ; 3/2  Wait for receiving to finish
            JC          BYP             ; 2/2  Jump if carry
            MOV         A, SBUF         ; 2/1  Read SBUF to get data table address
            MOVC        A, @A + DPTR    ; 1/2  Offset value from program memory
                                                 data table lookup
            SETB        C               ; 1/1  Set carry
            AJMP        SINP            ; 2/2  Jump to SINP
BYP:        ADD         A, SBUF         ; 2/1  Add input data from SBUF to offset value
            MOV         SCON, #0        ; 3/2  Reinitialize SCON for Mode and no receive
SOUT:       MOV         SBUF, A         ; 2/1  Result to SBUF starts serial output
STALL2:     JNB         SCON.TI, STALL2 ; 3/2  Wait for transmitting to finish
            CLR         SCON.TI         ; 2/1  Clear TI (transmit interrupt) flag
            DJNZ        SIZE, LOOP      ; 2/2  Decrement & test SIZE and loop back
                                                 if not finished
            RET                         ; 1/2  Return from subroutine

                                        35 BYTES
                                        52 CYCLES per couplet Loop
```

PIC16C5X BENCHMARK #6—SERIAL INPUT/OUTPUT WITH OFFSET TABLE

This benchmark inputs a sequence of byte couplets, with each couplet consisting of an 8-bit offset address followed by a data byte. The address is used to access a program memory data table to produce an offset value which is added to the second byte of the input couplet. The updated data byte is output while the address byte of the next couplet is being input. The benchmark is written as a subroutine, with the size of the input stream being an input parameter.

The program is written using a transmit/receive subroutine and takes 214 instruction cycles per couplet loop. This equates to 107 instruction cycles (53.5 μs) per byte, yielding an effective byte rate of 18.7 kHz.

```
UPDATE:   MOVWF    SIZE          ; 1/1  Save and increment byte couplet count to
          INCF     SIZE          ; 1/1  compensate for one byte throughput
          CLRF     XRDATA        ; 1/1  Clear data register
          MOVLW    2             ; 1/1  Set up bit select for input
          TRIS     PORTB         ; 1/1  Tristate bit 2 of PORTB
LOOP:     CALL     XMITREC       ; 1/2  Call XMITREC transmit/receive subroutine
          MOVF     XRDATA, W     ; 1/1  Move offset table address to W
          CALL     TABLE         ; 1/2  Call TABLE (table lookup subroutine)
          CALL     XMITREC       ; 1/2  Call XMITREC subroutine
          ADDWF    XRDATA        ; 1/1  Add offset to received data
          DECFSZ   SIZE          ; 1/1  Decrement SIZE and test if zero
          GOTO     LOOP          ; 1/2  Branch back if not finished
          RETLW    0             ; 1/2  Return from subroutine

XMITREC:  MOVLW    8             ; 1/1  Set up bit count
          MOVWF    BITCNT        ; 1/1  Bit count to BITCNT
LUP:      BCF      PORTB, 0      ; 1/1  Reset clock output bit
          BCF      PORTB, 1      ; 1/1  Reset data output bit
          RRF      XRDATA        ; 1/1  Rotate right through carry
          BTFSC    F3, 0         ; 1/1  Test carry bit and skip if clear
          BSF      PORTB, 0      ; 1/1  Set data output bit
          BSF      PORTB, 1      ; 1/1  Set clock output bit
          BCF      XRDATA, 7     ; 1/1  Clear upper bit of XRDATA
          BTFSC    PORTB, 2      ; 1/1  Test receive bit
          BSF      XRDATA, 7     ; 1/1  Set upper bit of XRDATA if receive bit high
          DECFSZ   BITCNT        ; 1/1  Decrement count and skip if zero
          GOTO     LUP           ; 1/2  Branch back to XMITREC
          BCF      PORTB, 1      ; 1/1  Reset clock output bit
          RETLW    0             ; 1/2  Return from subroutine

TABLE:    ADDWF    PC            ; 1/1  Add offset to PC
          RETLW    Data 1        ; 1/2  1st entry of data table
          RETLW    Data 2        ; -/-  2nd entry of data table
          ----     ----          ; -/-  --------

                                 30 WORDS (Equivalent to 45 BYTES)
                                 214 CYCLES per couplet Loop
```

COP8 BENCHMARK #7—TIMEKEEPING

This timekeeping benchmark is interrupt driven, using 5 ms. timer cycle interrupts. The program emulates a real time clock, keeping track of hours, minutes, and seconds in packed BCD format.

```
TIMER SETUP ROUTINE:
          L0 = 088                  ;       5 ms low component for timer
          HI = 013                  ;       5 ms high component for timer
          TMRL0 = 0EA               ;       Timer low byte
          TMRHI = 0EB               ;       Timer high byte
          TAUL0 = 0EC               ;       Autoreload register low byte
          TAUHI = 0ED               ;       Autoreload register high byte
          CNTRL = 0EE               ;       CNTRL control register
          PSW = 0EF                 ;       PSW control register
          CNTR = 0F0                ;       5 ms counter
          HOUR = 0F1                ;       Hour register (packed BCD format)
          MIN = 0F2                 ;       Minute register (packed BCD format)
          SEC = 0F3                 ;       Second register (packed BCD format)
          TEMP = 0F4                ;       Temporary register

TSETUP:   LD        B, #TAUL0       ;  2/3  Load B pointer with address of low autoreload
          LD        [B+], #L0       ;  2/3  Load low autoreload reg with 5 ms component
          LD        [B+], #HI       ;  2/2  Load high autoreload reg with 5 ms component
          LD        [B+], #080      ;  2/2  Set up timer PWM mode in CNTRL register
          LD        [B+], #011      ;  2/2  Set up timer interrupt in PSW register
          LD        [B+], #200      ;  2/2  Initialize 5 ms counter to count one second
          LD        [B+], #1        ;  2/2  Initialize HOUR to 1
          LD        [B+], #0        ;  2/2  Initialize MIN to 0
          LD        [B], #0         ;  2/2  Initialize SEC to 0
          SBIT      4, CNTRL        ;  3/4  Start timer
STALL:    JP        STALL           ;  -/-  Loop back on self to simulate main program

                                        21 BYTES
```

COP8 BENCHMARK #7—TIMEKEEPING (Continued)

```
TIMER INTERRUPT SERVICE ROUTINE:

TIMEKEEP:  DRSZ        CNTR           ;  1/3  Decrement 5 ms counter
           RETI                       ;  1/5  Return from timer interrupt
           LD          CNTR, #200     ;  2/3  Reload 5 ms counter
           LD          B, #TEMP       ;  2/3  Load B pointer with address of TEMP
           X           A, [B-]        ;  1/2  Save A in TEMP
           JSR         INCBCD         ;  2/5  Call INCBCD subroutine to increment SEC
           RETI                       ;  1/5  Return from timer interrupt
           JSR         INCBCD         ;  2/5  Call INCBCD subroutine to increment MIN
           RETI                       ;  1/5  Return from timer interrupt
           SC                         ;  1/1  Set carry for BCD increment of HOUR
           LD          A, #066        ;  2/2  Set up BCD increment
           ADC         A, [B]         ;  1/1  Increment A in BCD
           DCOR                       ;  1/1  Decimal correct result of BCD increment
           X           A, [B]         ;  1/1  Store result in HOUR
           IFEQ        A, #012        ;  2/2  Test if result was BCD 13 (previous result 12)
           LD          [B], #1        ;  2/2  Reset HOUR from BCD 13 to BCD 1
           LD          A, TEMP        ;  2/3  Restore A from TEMP
           RETI                       ;  1/5  Return from timer interrupt

INCBCD:    SC                         ;  1/1  Set carry for BCD increment
           LD          A, #066        ;  2/2  Set up BCD increment
           ADC         A, [B]         ;  1/1  Increment A in BCD
           DCOR                       ;  1/1  Decimal correct result of BCD increment
           X           A, [B]         ;  1/1  Store result
           IFEQ        A, #059        ;  2/2  Test if result was BCD 60 (previous result 59)
           JP          RESTORE        ;  1/3  Jump to RESTORE if test successful
           RET                        ;  1/5  Return from subroutine
RESTORE:   LD          [B-], #0       ;  2/3  Reset to zero
           RETSK                      ;  1/5  Return from subroutine and skip

                                      39 BYTES

                       TOTAL:         60 BYTES
                                      80 CYCLES*

*Case where 12:59:59 is advancing to 1:00:00
```

7

M68HC05 BENCHMARK #7—TIMEKEEPING

This timekeeping benchmark is interrupt driven, using the timer output compare interrupt (in conjunction with the 131.072 ms timer cycle). The program emulates a real time clock, keeping track of hours, minutes, and seconds in packed BCD format.

The timer clock is equivalent to the instruction cycle clock (2 MHz) divided by 4, yielding a timer clock period of 2 μs. The 16-bit free running timer has a maximum of 65536 counts, which at 2 μs per count equates to a timer cycle of 131.072 ms. One second divided by the timer cycle (1,000,000 divided by 131,072) yields 7 cycles with a residual of 82.496 ms. This residual equates to 41248 timer counts, which represents the value (A120 in hex) added to the output compare register during each timer interrupt service.

```
TIMER SETUP ROUTINE:

         OCIE   EQU 6          ;       Output compare interrupt bit in TCR (timer control reg)
         CNTR   EQU 10         ;
         TEMP   EQU 11         ;
         ATEMP  EQU 12         ;
         SEC    EQU 20         ;
         MIN    EQU 21         ;
         HOUR   EQU 22         ;

TSETUP:  CLR    SEC            ; 2/5  Initialize SEC to 0
         CLR    MIN            ; 2/5  Initialize MIN to 0
         CLR    HOUR           ; 2/5  Clear HOUR to 1
         INC    HOUR           ; 2/5  Initialize HOUR to 1
         LDA    #7             ; 2/2  Load counter initialization value
         STA    CNTR           ; 2/4  Initialize timer loop counter
         BSET   OCIE, TCR      ; 2/5  Enable Output Compare Interrupt
STALL:   JMP    STALL          ; -/-  Loop back on self to simulate main program

                               14 BYTES
```

M68HC05 BENCHMARK #7—TIMEKEEPING (Continued)

```
TIMER INTERRUPT SERVICE ROUTINE:        ;       OCMP = OUTPUT COMPARE

TIMEKEEP:  DEC      CNTR        ;  2/5  Decrement 5 ms counter
           BNE      FINI        ;  2/3  Branch if result zero
           STA      TEMP        ;  2/4  Save A in TEMP
           LDA      #7          ;  2/2  Load A with counter reload value
           STA      CNTR        ;  2/4  Reload timer loop counter
           LDA      OCMPL0      ;  2/3  Read OCMPL0
           ADD      #$20        ;  2/2  ADD low byte of offset count
           STA      ATEMP       ;  2/4  Store in ATEMP until OCMPHI is updated
           LDA      OCMPHI      ;  2/3  Read OCMPHI
           ADC      #$A1        ;  2/2  Add high byte of offset count
           STA      OCMPHI      ;  2/4  Update OCMPHI with new offset
           LDA      TSR         ;  2/3  Read TSR to clear OCF flag
           LDA      ATEMP       ;  2/3  Retrieve update for OCMPL0 from ATEMP
           STA      OCMPL0      ;  2/4  Update OCMPL0 with new offset
           LDX      #20         ;  2/2  Load index register with address of SEC
           JSR      INCBCD      ;  2/5  Call INCBCD subroutine
           BCC      FIN         ;  2/3  Branch if carry clear
           JSR      INCBCD      ;  2/5  Call INCBCD subroutine
           BCC      FIN         ;  2/3  Branch if carry clear
           JSR      BCDFIX      ;  2/5  Call BCDFIX subroutine
           CMP      A, #$013    ;  2/2  Compare A with BCD 13
           BNE      FIN         ;  2/3  Branch if not equal
           CLR      X           ;  1/5  Clear result
           INC      X           ;  1/5  Set HOUR to 1
FIN:       LDA      TEMP        ;  2/3  Restore A from TEMP
FINI:      RTI                  ;  1/9  Return from interrupt

INCBCD:    JSR      BCDFIX      ;  2/5  Call BCDFIX subroutine
           CMP      A, #$060    ;  2/2  Compare A with BCD 60
           BNE      BYP         ;  2/3  Branch if not equal
           CLR      X           ;  1/5  Clear result (SEC or MIN) to zero
           SEC                  ;  1/2  Set carry
           INCX                 ;  1/3  Increment index
BYP:       RTS                  ;  1/5  Return from subroutine

BCDFIX:    LDA      X           ;  1/3  Load A with operand (SEC or MIN)
           ADD      #1          ;  2/2  Add 1 to A
           STA      X           ;  1/4  Store A in result
           AND      A, #$0F     ;  2/2  Extract low order BCD digit
           CMP      A, #10      ;  2/2  Compare result with 10
           BNE      BYPASS      ;  2/3  Branch if not equal
           LDA      #$010       ;  2/2  Load A with BCD 10
           STA      X           ;  1/4  Store BCD 10 in result
BYPASS:    RTS                  ;  1/5  Return from subroutine

*Case where 12:59:59 is advancing to 1:00:00  73 BYTES

                    TOTAL:      87 BYTES
                                218 CYCLES*
```

80C51 BENCHMARK #7—TIMEKEEPING

This timekeeping benchmark is interrupt driven, using 250 μs timer cycle interrupts. The program emulates a real time clock, keeping track of hours, minutes, and seconds in packed BCD format.

TIMER SETUP ROUTINE:

```
            CNTR1       EQU R2          ;
            CNTR2       EQU R3          ;
            TEMP        EQU R4          ;
            SEC         EQU 20          ;
            MIN         EQU 21          ;
            HOUR        EQU 22          ;

TSETUP:     MOV         CNTR1, #20      ; 2/1  Initialize 250 μs counter to 20
            MOV         CNTR2, #200     ; 2/1  Initialize 5 ms counter to 200
            MOV         SEC, #0         ; 2/1  Initialize SEC to 0
            MOV         MIN, #0         ; 2/1  Initialize MIN to 0
            MOV         HOUR, #1        ; 2/1  Initialize HOUR to 1
            MOV         TMOD, #02H      ; 3/2  Select Timer 0, Mode 2
            MOV         TH0, #-250      ; 3/2  Setup 250 μs delay for Timer 0
            SETB        TR0             ; 2/1  Start Timer 0
            MOV         IE, #82H        ; 3/2  Enable Timer 0 interrupt
LOOP:       SJMP        LOOP            ; -/-  Loop back on self to simulate main program

                                        21 BYTES
```

80C51 BENCHMARK #7—TIMEKEEPING (Continued)

```
TIMER INTERRUPT SERVICE ROUTINE:

TIMEKEEP:  DJNE      CNTR1, FIN       ;  2/2  Decrement 250 µs counter and jump if not zero
           MOV       CNTR1, #20       ;  2/1  Reload 250 µs counter
           DJNE      CNTR2, FIN       ;  2/2  Decrement 5 ms counter and jump if not zero
           MOV       CNTR2, #200      ;  2/1  Reload 5 ms counter
           MOV       TEMP, A          ;  1/1  Save A in TEMP
           MOV       R0, #20          ;  2/1  Load R0 with address of SEC
           ACALL     INCBCD           ;  2/2  Call INCBCD subroutine
           JNC       FIN              ;  2/2  Jump if no carry
           ACALL     INCBCD           ;  2/2  Call INCBCD subroutine
           JNC       FIN              ;  2/2  Jump if no carry
           MOV       A, @R0           ;  1/1  Move HOUR to A
           CLR       C                ;  1/1  Initialize carry
           ADD       A, #1            ;  2/1  Add 1 to A
           DA        A                ;  1/1  Decimal adjust A for BCD correction
           MOV       @R0, A           ;  1/1  Return A to HOUR
           CJNE      A, #013H, FIN    ;  3/2  Compare and jump if hour not equal to 13
           MOV       @R0, #1          ;  2/1  Correct HOUR to 1
FIN:       MOV       A, TEMP          ;  1/1  Restore A from TEMP
           RETI                       ;  1/2  Return from timer interrupt

INCBCD:    MOV       A, @R0           ;  1/1  Move operand to A (SEC or MIN)
           ADD       A, #1            ;  2/1  Add 1 to A
           DA        A                ;  1/1  Decimal adjust A for BCD correction
           MOV       @R0, A           ;  1/1  Return A to result register (SEC or MIN)
           CJNE      A, #060H, BYPASS;  3/2  Compare and jump if result not equal to 60
           MOV       @R0, #0          ;  2/1  Correct result to zero (SEC or MIN)
           SETB      C                ;  1/1  Set carry to indicate corrected result
BYPASS:    INC       R0               ;  1/1  Increment indirect address data pointer
           RET                        ;  1/2  Return from subroutine

                                      45 BYTES

                     TOTAL:           66 BYTES
                                      49 CYCLES*

*Case where 12:59:59 is advancing to 1:00:00
```

PIC16C5X BENCHMARK #7—TIMEKEEPING

This timekeeping benchmark uses a pseudo software 5 ms delay loop as the basic core timing for emulating a real time clock. Note that the PIC16C5X does not have any hardware interrupts. Consequently, the program must keep track of when the counter RTCC overflows for real time applications. The RTTC F1 file register is only 8 bits, but with an 8-bit prescaler selected (1 : 256 RTTC rate), the RTTC emulates a 16-bit timer. The program keeps track of hours, minutes, and seconds in packed BCD format.

5 ms timing analysis: 0.5 μs (instruction cycle time) × 256 (prescaler) = 128 μs RTTC increment rate

5 ms emulation: 5000/128 = 39 cycles + residue of 8 μs

```
            RTCC        EQU 1           ;       Register File F1
            STATUS      EQU 3           ;       Reg File F3
            TEMP        EQU 20          ;       Reg File F20
            SEC         EQU 21          ;       Reg File F21
            MIN         EQU 22          ;       Reg File F22
            HOUR        EQU 23          ;       Reg File F23
            CNTR        EQU 24          ;       Reg File F24

TIMEKEEP:   MOVLW       7               ; 1/1   Set up data for option register
            OPTION                      ; 1/1   Select a 1:256 prescaler with RTTC
            CLRF        SEC             ; 1/1   Initialize SEC to 0
            CLRF        MIN             ; 1/1   Initialize MIN to 0
            CLRF        HOUR            ; 1/1   Clear HOUR
            INCF        HOUR            ; 1/1   Initialize HOUR TO 1
            MOVLW       217             ; 1/1   256-39 = 217 (RTCC counter increments)
            MOVWF       RTCC            ; 1/1   Set up RTCC for 39 counts until overflow
STALL:      GOTO        STALL           ; -/-   Loop back on self to simulate main program

                                        8 WORDS
-------------------------------------------------------------------------
-------------------------------------------------------------------------

Main program must return within 5 ms to sense exactly when RTCC counter overflows ! ! !

-------------------------------------------------------------------------
-------------------------------------------------------------------------
```

PIC16C5X BENCHMARK #7—TIMEKEEPING (Continued)

```
TSTOVFLW:  BTFSC      RTCC, 7          ;  1/1  Test if RTCC has overflowed
           GOTO       TSTOVFLW         ;  1/2  Loop back if test fails
           MOVLW      217              ;  1/1  256-39 = 217
           MOVWF      RTCC             ;  1/1  Set up RTCC for 39 counts until overflow
           DECFSZ     CNTR             ;  1/1  Decrement CNTR and test for zero result
           GOTO       STALL            ;  1/2  Go to MAIN program
           MOVLW      200              ;  1/1  Set up 200 (200 × 5 ms = 1 sec)
           MOVWF      CNTR             ;  1/1  Initialize 5 ms counter
           MOVLW      SEC              ;  1/1  Address of SEC to W
           MOVWF      F4               ;  1/1  Address of SEC to F4
           CALL       INCBCD           ;  1/2  Call INCBCD subroutine to increment SEC
           BTFSS      F3, 2            ;  1/1  Test and skip if zero result (Z is F3 bit 2)
           GOTO       STALL            ;  1/2  Go to MAIN program
           CALL       INCBCD           ;  1/2  Call INCBCD subroutine to increment MIN
           BTFSS      F3, 2            ;  1/1  Test and skip if zero result
           GOTO       STALL            ;  1/2  Go to MAIN program
           INCF       HOUR             ;  1/1  Increment hour
           MOVLW      10               ;  1/1  Move 10 to W
           SUBWF      HOUR, W          ;  1/1  HOUR - 10 to W
           MOVLW      6                ;  1/1  Move 6 to W
           BTFSC      F3, 2            ;  1/1  Test if subtraction result not zero
           ADDWF      HOUR             ;  1/1  Add 6 to HOUR for BCD correction
           MOVLW      013H             ;  1/1  Move BCD 13 to W
           SUBWF      HOUR, W          ;  1/1  HOUR - BCD 13 to W
           MOVLW      1                ;  1/1  Move 1 to W
           BTFSC      F3, 2            ;  1/1  Test and skip if subtraction result not zero
           MOVWF      HOUR             ;  1/1  Move 1 to hour if subtraction result zero
           GOTO       STALL            ;  1/2  Go to MAIN program

INCBCD:    INCF       F0               ;  1/1  Increment BCD data operand
           MOVF       F0, W            ;  1/1  Move data to W
           MOVWF      TEMP             ;  1/1  Move data to TEMP
           MOVLW      0FH              ;  1/1  Move hex 0F to W
           ANDWF      TEMP, W          ;  1/1  Extract low order BCD digit
           MOVLW      10               ;  1/1  Move 10 to W
           BSF        F3, 0            ;  1/1  Set carry (reset borrow for subtraction)
           SUBWF      TEMP, W          ;  1/1  TEMP - 10 to TEMP
           MOVLW      6                ;  1/1  Move 6 to W
           BTFSC      F3, 2            ;  1/1  Test if subtraction result zero (Z is F3 bit 2)
           ADDWF      F0               ;  1/1  Add 6 to correct BCD operand if test successful
           MOVLW      060H             ;  1/1  Move BCD 60 to W
           BSF        F3, 0            ;  1/1  Set carry (reset borrow for subtraction)
           SUBWF      F0, W            ;  1/1  BCD operand - BCD 60 to W
           BTFSC      F3, 2            ;  1/1  Test if subtraction result zero
           CLRF       F0               ;  1/1  Reset BCD operand to zero if test successful
           INCF       F4               ;  1/1  Increment indirect address pointer
           RETLW      0                ;  1/2  Return from subroutine

*Case where 12:59:59 is advancing to 1:00:00 46 WORDS

                      TOTAL:           54 WORDS (Equivalent to 81 BYTES)
                                       69 CYCLES*
```

COP8 BENCHMARK #8—SWITCH ACTIVATED FIVE SECOND LED

This benchmark samples a switch input to activate a five second output for turning on an LED. The switch is debounced with a 50 ms delay both on opening and closure. Once activated, the switch will turn on an LED output for five seconds and then turn it off, regardless of whether or not the switch is still activated. Once the switch is turned off, the procedure is repeated. Both the switch input and the LED output are low true.

```
        CNTR1 = 0F0             ;       Three Counters in registers 0F0, 0F1, 0F2
        CNTR2 = 0F1             ;
        CNTR3 = 0F2             ;
        PORTLD = 0D1            ;       PORTL Data register
        PORTLC = 0D2            ;       PORTL Configuration register
        PORTLI = 0D3            ;       PORTL Input address

LED5:   LD      B, #PORTLD      ; 2/3   PORTL data register address to B pointer
        LD      [B+], #0F0      ; 2/2   Set upper 4 bits of data register for low true output
        LD      [B+], #0F0      ; 2/2   Configure L port for upper nibble output and
                                ;       lower nibble input (incr B ptr to port input addr)
WSWON:  IFBIT   0, [B]          ; 1/1   Sample input switch (low true)
        JP      WSWON           ; 1/3   Wait for Switch ON
SWON:   JSR     DLY50           ; 2/5   Debounce 50 ms
        RBIT    7, PORTLD       ; 3/4   Turn on LED (low true)
        LD      CNTR1, #100     ; 2/3   Call DLY50 50 ms subroutine 100 times
SEC5:   JSR     DLY50           ; 2/5     to get 5 second LED ON time
        DRSZ    CNTR1           ; 1/3   Decrement CNTR1 and test for zero result
        JP      SEC5            ; 1/3   Loop back until count finished
        SBIT    7, PORTLD       ; 3/4   Turn off LED (low true)
WSWOFF: IFBIT   0, [B]          ; 1/1   Sample input switch (low true)
        JP      SWOFF           ; 1/3   Jump to Switch OFF
        JP      WSWOFF          ; 1/3   Wait for Switch OFF
SOFF:   JSR     DLY50           ; 2/5   Debounce 50 ms
        JP      WSWON           ; 1/3   Repeat procedure (wait for Switch ON)

DLY50:  LD      CNTR2, #33      ; 2/3   Set up outer loop count
        LD      CNTR3, #118     ; 2/3   Set up initial inner loop count
LOOP:   DRSZ    CNTR3           ; 1/3   Decrement inner loop count and test for zero
        JP      LOOP            ; 1/3   Loop back until inner count finished
        DRSZ    CNTR2           ; 1/3   Decrement outer loop count and test for zero
        JP      LOOP            ; 1/3   Loop back until outer count finished
        RET                     ; 1/5   Return from subroutine

*Cycle times without wait loops37 BYTES
                                76 CYCLES*

DLY50 TIMING ANALYSIS:          CYCLES

        Initial (2 X 3)         6
        1 x (117 x 6 + 1 x 10)712
        32 x (255 x 6 + 1 x 104)9280
        Terminating (5 - 2)     3

        TOTAL                   50001 Equivalent to 50.001 ms @ 1 µs per cycle
```

M68HC05 BENCHMARK #8—SWITCH ACTIVATED FIVE SECOND LED

This benchmark samples a switch input to activate a five second output for turning on an LED. The switch is debounced with a 50 ms delay both on opening and closure. Once activated, the switch will turn on an LED output for five seconds and then turn it off, regardless of whether or not the switch is still activated. Once the switch is turned off, the procedure is repeated. Both the switch input and the LED output are low true.

```
            PORTB       EQU $01          ;
            DDRB        EQU $05          ;
            TEMP        EQU $9F          ;

LED5:       LDA         #$F0             ;  2/2  Output data and configuration to A
            STA         PORTB            ;  2/4  Set upper 4 bits of PORTB for low true output
            STA         DDRB             ;  2/4  Configure PORTB for upper nibble output
                                         ;       and lower nibble input
WSWON:      BRSET       0, PORTB, WSWON  ;  3/5  Sample input switch (low true)
SWON:       JSR         DLY50            ;  3/6  Debounce 50 ms
            BCLR        7, PORTB         ;  2/5  Turn on LED (low true)
            LDA         #100             ;  2/2  Call DLY50 50 ms subroutine 100 times
SEC5:       JSR         DLY50            ;  3/6    to get 5 second LED ON time
            DECA                         ;  1/3  Decrement count
            BNE         SEC5             ;  2/3  Loop back until count finished
            BSET        7, PORTB         ;  2/5  Turn off LED (low true)
WSWOFF:     BRCLR       0, PORTB, WSWOFF ;  3/5  Sample input switch (low true)
SWOFF:      JSR         DLY50            ;  3/6  Debounce 50 ms
            BRA         WSWON            ;  2/3  Repeat procedure (wait for Switch ON)

DLY50:      STA         TEMP             ;  2/4  Save A
            LDA         #65              ;  2/2  Set up outer loop count
            LDX         #226             ;  2/2  Set up inner loop count
LOOP:       DECX                         ;  1/3  Decrement inner loop count
            BNE         LOOP             ;  2/3  Loop back if non-zero
            DECA                         ;  1/3  Decrement outer loop count
            BNE         LOOP             ;  2/3  Loop back if non-zero
            LDA         TEMP             ;  2/3  Restore A
            RTS                          ;  1/6  Return from subroutine

                                        47 BYTES
                                        88 CYCLES*
*Cycle times without wait loops

DLY50 TIMING ANALYSIS:                  CYCLES

            Initial (4 + 2 + 2)         8
            1 x (225 x 6 + 1 x 11)      1361
            64 x (255 x 6 + 1 x 11)     98624
            Terminating (3 + 6 - 1)     8

            TOTAL                       100001 Equivalent to 50.000 ms @ 0.5 µs per cycle
```

80C51 BENCHMARK #8—SWITCH ACTIVATED FIVE SECOND LED

This benchmark samples a switch input to activate a five second output for turning on an LED. The switch is debounced with a 50 ms delay both on opening and closure. Once activated, the switch will turn on an LED output for five seconds and then turn it off, regardless of whether or not the switch is still activated. Once the switch is turned off, the procedure is repeated. Both the switch input and the LED output are low true.

```
LED5:    MOV     P1, #00F        ;  3/2  Configure lower 4 bits of port P1 for input
WSWON:   JB      P1.0, WSWON     ;  3/2  Sample input switch (low true)
SWON:    ACALL   DLY50           ;  2/2  Debounce 50 ms
         CLR     P1.7            ;  2/1  Turn on LED (low true)
         MOV     R2, #100        ;  2/1  Call DLY50 50 ms subroutine 100 times
SEC5:    ACALL   DLY50           ;  2/2  to get 5 second LED ON time
         DJNZ    R2, SEC5        ;  2/2  Decrement count and test for zero result
         SETB    P1.7            ;  2/1  Turn off LED (low true)
WSWOFF:  JNB     P1.0, WSWOFF    ;  3/2  Sample input switch (low true)
SWOFF:   ACALL   DLY50           ;  2/2  Debounce 50 ms
         AJMP    WSWON           ;  2/2  Repeat procedure (wait for Switch ON)

DLY50:   MOV     R3, #98         ;  2/1  Set up outer loop count
         MOV     R4, #68         ;  2/1  Set up inner loop count
LOOP:    DJNZ    R4, LOOP        ;  2/2  Decrement inner loop count and test for zero
         DJNZ    R3, LOOP        ;  2/2  Decrement outer loop count and test for zero
         RET                     ;  1/2  Return from subroutine

                                 34 BYTES
                                 27 CYCLES*
*Cycle times without wait loops

DLY50 TIMING ANALYSIS:           CYCLES

         INITIAL (2 x 1)         2
         1 x (67 x 2 + 1 x 4)    138
         97 x (255 x 2 + 1 x 4)  49858
         Terminating (2)         2

         TOTAL                   50000 Equivalent to 50.000 ms @ 1 μs per cycle
```

PIC16C5X BENCHMARK #8—SWITCH ACTIVATED FIVE SECOND LED

This benchmark samples a switch input to activate a five second output for turning on an LED. The switch is debounced with a 50 ms delay both on opening and closure. Once activated, the switch will turn on an LED output for five seconds and then turn it off, regardless of whether or not the switch is still activated. Once the switch is turned off, the procedure is repeated. Both the switch input and the LED output are low true.

```
            PORTB       EQU 6               ;       Register File F6
            CNTR        EQU 21              ;       Reg File F21
            CNTR2       EQU 22              ;       Reg File F22
            CNTR3       EQU 23              ;       Reg File F23

LED5:       MOVLW       FOH                 ; 1/1   Output data to W
            MOVWF       PORTB               ; 1/1   Set upper 4 bits of port for low true output
            MOVLW       OFH                 ; 1/1   Tri-state control to W
            TRIS        PORTB               ; 1/1   Tri-state lower 4 bits of port for input
WSWON:      BTFSC       PORTB, 0            ; 1/1   Sample input switch (low true)
            GOTO        WSWON               ; 1/2   Wait for Switch ON
SWON:       CALL        DLY50               ; 1/2   Debounce 50 ms
            BCF         PORTB, 7            ; 1/1   Turn on LED (low true)
            MOVLW       100                 ; 1/1   Five sec count to W
            MOVWF       CNTR                ; 1/1   Call DLY50 50 ms subroutine 100 times
SEC5:       CALL        DLY50               ; 1/2     to get 5 second LED ON time
            DECFSZ      CNTR                ; 1/1   Decrement CNTR1 and test for zero result
            GOTO        SEC5                ; 1/2   Loop back until count finished
            BSF         PORTB, 7            ; 1/1   Turn off LED (low true)
WSWOFF:     BTFSS       PORTB, 0            ; 1/1   Sample input switch (low true)
            GOTO        WSWOFF              ; 1/2   Wait for Switch OFF
SWOFF:      CALL        DLY50               ; 1/2   Debounce 50 ms
            GOTO        WSWON               ; 1/2   Repeat procedure (wait for Switch ON)

DLY50:      MOVLW       130                 ; 1/1   Outer loop count to W
            MOVWF       CNTR2               ; 1/1   Set up outer loop count
            MOVLW       221                 ; 1/1   Inner loop count to W
            MOVWF       CNTR3               ; 1/1   Set up inner loop count
LOOP:       DECFSZ      CNTR3               ; 1/1   Decrement inner loop count and test for zero
            GOTO        LOOP                ; 1/2   Loop back until inner count finished
            DECFSZ      CNTR2               ; 1/1   Decrement outer loop count and test for zero
            GOTO        LOOP                ; 1/2   Loop back until outer count finished
            RETLW       0                   ; 1/2   Return from subroutine

                                            27 WORDS (Equivalent to 40 1/2 BYTES)
*Cycle times without wait loops             37 CYCLES*

DLY50 TIMING ANALYSIS:                      CYCLES

            Initial (4 x 1)                 4
            1 x (220 x 3 + 1 x 5)            665
            129 x (255 x 3 + 1 x 5)         99330
            Terminating (2 - 1)             1

            TOTAL                           100000 Equivalent to 50.000 ms @ 0.5 µs/cycle
```

6.0 SUMMARY OF RESULTS (Both Positive (+) and Negative (−)

Benchmark #1 - BLOCK TRANSFER

COP8	Two indirect data pointers (+); IFBNE instruction (+)
M68HC05	Indexed addressing (+); Decrementing index to zero (+)
80C51	Two indirect data pointers (+)
PIC16C5X	Only one indirect data pointer (−)

Benchmark #2 - BINARY ADDITION

COP8	Two indirect data pointers (+); IFBNE instruction (+)
M68HC05	Indexed addressing with offset (+)
80C51	Two indirect data pointers (+)
PIC16C5X	Only one indirect data pointer (−)

Benchmark #3 - BCD SUBTRACTION

COP8	Two indirect data pointers (+); IFBNE instruction (+); DCOR (decimal correct) instruction (+)
M68HC05	Indexed addressing with offset (+); No BCD correction instruction (−)
80C51	Two indirect data pointers (+); DA instruction (BCD correction) only for add, not subtract (−)
PIC16C5X	Only one indirect data pointer (−); No BCD correction instruction (−)

Benchmark #4 - TABLE SEARCH

COP8	LAID instruction (+); Lack of IFNE comparison instruction (−); RETSK instruction (+)
M68HC05	Indexed addressing (+); Lack of loading immediate values directly to memory (−)
80C51	16-bit DPTR (data pointer (+); MOV A @A + DPTR instruction (+); CJNE instruction combine comparison and branch (+)
PIC16C5X	RETLW instruction (+)

Benchmark #5 - INPUT/OUTPUT MANIPULATION

COP8	I/O Port Configuration registers for individual bit control (+)
M68HC05	I/O Port Data Direction reg's for individual bit control (+); No SWAP to reverse nibbles (−)
80C51	Distinction between tristating for input versus output of 1 (?−); No configuration register (−)
PIC16C5X	TRIS (tristate instruction) (+); No configuration register (−)

Benchmark #6 - SERIAL INPUT/OUTPUT WITH OFFSET TABLE

COP8	Microwire/Plus for serial I/O (+); LAID instruction (+)
M68HC05	SPI (Serial Peripheral Interface) for serial I/O (+)
80C51	Serial I/O with 8-bit shift register mode cannot input and output simultaneously (−); MOV A, @A + DPTR instruction (+)
PIC16C5X	Lack of any dedicated serial I/O (−); RETLW instruction (+)

Benchmark #7 - TIMEKEEPING

COP8	Timer autoreload with interrupt very flexible (+)
M68HC05	Timer Output Compare interrupt structuring for real time cumbersome and confusing (−)
80C51	Lack of 16-bit timer autoreload (−)
PIC16C5X	OPTION instruction to select timer prescaler (+); Lack of timer interrupt (−); No timer autoreload (−)

Benchmark #8 - SWITCH ACTIVATED 5 SEC LED

COP8	Lack of bit testing for bit clear (−)
M68HC05	Bit testing for both set and clear (+); Versatility of index register (+)
80C51	DJNZ instruction combines testing and branch (+); JB and JNB instructions combine bit testing and branch (+)
PIC16C5X	Bit testing for both set and clear (+)

TABLE I. The Benchmark Results: Code Size Efficiency in Bytes

BENCHMARK BYTES	National COP8	Motorola 68HC05	Intel 80C51	Microchip PIC16C5X
Five Byte Block Move	7	9	11	21
Four Byte Binary Addition	10	13	14	28 ½
Four Byte BCD Subtraction	25	54	39	82 ½
Three Byte Table Search	42	40	29	51
Input/Output Manipulation	26	42	33	37 ½
Serial I/O with Offset Table	31	31	35	45
Timekeeping	60	87	66	81
Switch Activated 5s LED	37	47	34	40 ½
TOTAL	238	323	261	387
RATIO	1	1.36	1.10	1.63

TABLE II. The Benchmark Results: Code Execution Time Efficiency in Cycles and Microseconds

BENCHMARK CYCLE/μs	National COP8	Motorola 68HC05	Intel 80C51	Microchip PIC16C5X
Five Byte Block Move	47/47	76/38	32/32	62/31
Four Byte Binary Addition	48/48	85/42.5	33/33	62/31
Four Byte BCD Subtraction	97/97	567/283.5	91/91	274/137
Three Byte Table Search (Note 2)	77/77	80/40	30/30	52/26
Input/Output Manipulation (Note 3)	24/24	53/26.5	17/17	21/10.5
Serial I/O with Offset Table (Note 1)	36/36	54/27	52/52	214/107
Timekeeping (Note 5)	80/80	218/109	49/49	69/34.5
Switch Activated 5s LED (Note 4)	76/76	88/44	27/27	37/18.5
TOTAL	485/485	1221/610.5	331/331	791/395.5
RATIO	1/1	2.53/1.26	0.68/0.68	1.63/0.82

Note 1: Couplet (address, data) loop time: address and data input, updated data output.

Note 2: First search iteration falls with 1st byte mismatch, second search iteration successful.

Note 3: Case where P1 < P2.

Note 4: Cycle times without wait loops.

Note 5: Case where 12:59:59 is advancing to 1:00:00.

7.0 CONCLUSIONS

This report provides a detailed study of the instruction sets of popular 8-bit single chip microcontrollers. Eight benchmark programs, demonstrating data movement, arithmetic operations, I/O manipulation, and timekeeping, were coded for four current microcontrollers.

In terms of byte efficiency, the COP8 from National, uses 8% less program space than its nearest competitor, the Intel 80C51. The COP8 uses, 26% less program space than Motorola 68HC05, 39% less program space than the Microchip PIC16C5X. Code efficiency is important because it enables designers to pack more on-chip functionality into less program memory space. Selecting a microcontroller with smaller program memory size translates into lower system costs, and the added security of knowing that more code can be packed into the microcontroller.

In terms of code execution, Intel's 80C51 is the fastest, while COP8 executes these benchmark routines faster than Motorola 68HC05 and Microchip PIC16C5X.

Comparison of COP878x to the Enhanced COP8SAx7 Family—Hardware/ Software Considerations

National Semiconductor
Application Note 1043
Abdul Aleaf

INTRODUCTION

The COP8780, COP8781 and COP8782 (COP878x family) are members of the COP8 Basic Family that contain EPROM, RAM, a 16-bit multi-function timer with a single 16-bit autoreload/capture register, 3 interrupt sources with polling scheme, and the MICROWIRE/PLUS™ serial interface. These devices are offered in 20-pin SO/DIP, 28-pin SO/DIP, 40-pin DIP, and 44-pin PLCC. The operating voltage range is from 4.5V to 6.0V.

The COP8SAA7, COP8SAB7 and COP8SAC7 (COP8SAx7 family) devices, on the other hand, are members of the COP8 Feature Family that contain EPROM, RAM, a 16-bit multi-function timer with two autoreload/capture registers, eight interrupt sources supporting vectored interrupt scheme, and the enhanced MICROWIRE/PLUS serial interface. In addition, these devices contain features such as Multi-Input Wakeup, WATCHDOG™/Clock Monitor, Idle timer supporting Idle mode, internal Power-On Reset, on-chip RC oscillator, 8 bytes of user storage space in EPROM, and on-chip EMI reduction circuitry. The devices are offered in 16-pin SO/DIP, 20-pin SO/DIP, 28-pin SO/DIP, 40-pin DIP, and 44-pin PLCC. The operating voltage range is from 2.7V to 5.5V. See the datasheet for more details.

The purpose of this Application Note is to provide a detailed comparison and feature analysis of these two family of devices. Where applicable, this report can be used to assist in converting the code written for the COP878x device to operate on an equivalent COP8SAx7 device.

As suggested, the COP8SAx7 family offers a broad range of additional useful and enhanced features as compared to the COP878x family. With the additional features, the COP8SAx7 family may appear much different, but this family is designed to maintain downward compatibility with the COP878x family as much as possible. The intention has been to offer the COP878x user the capability to maintain the same PCB when converting to COP8SAx7 without any PCB changes.

HIGHLIGHTS OF COP8SAx7 ENHANCEMENTS OVER COP878x

1. The operating voltage range is wider (2.7V to 5.5V).
2. The dynamic supply current is lower.
3. Four additional high sink current outputs (L0–L3) are added (min sink current of 15 mA at V_{OL} = 1.7, V_{CC} = 4.5V).
4. The R/C oscillator option is expanded to include the choice of selecting on-chip R/C components. The use of on-chip R/C eliminates the cost associated with external R/C oscillator components.
5. The crystal oscillator option is expanded to include the choice of selecting on-chip biasing resistor.
6. COP878x's Port I is replaced with a bi-directional Port F.
7. Schmitt trigger inputs are added to Port L.
8. Pins 21–24 (Ports C4–C7), on the 44-pin PLCC package, are general purpose I/O ports. They are No Connection (NC) on the COP878x.
9. Stack Pointer is automatically initialized by hardware.
10. The user selectable Power-On reset feature is added, in addition to the ability to reset the device externally. The on-chip power-on reset eliminates the need for using external components associated with reset circuitry.
11. The HALT feature is made user selectable.
12. The unique power saving Multi-Input Wakeup feature is added.
13. The interrupt handling is enhanced to support a versatile vectored interrupt scheme. It has a total of eight interrupt sources with independent vectors. The COP878x has a polled interrupt scheme.
14. The Software Trap scheme is enhanced to include its own pending flag.
15. The MICROWIRE/PLUS serial interface is enhanced to include shifting on the alternate clock edge and programmable idle polarity for the shift clock. This allows compatibility with SPI peripherals.
16. The 16-bit multi-function timer/counter is improved to include two autoreload/capture registers. The timer block contains two separate interrupt vectors and two I/O pins. The COP878x timer has a single autoreload/capture register, a single interrupt source, and a single I/O pin.
17. The ECON register is expanded to select additional user selectable features such as WATCHDOG, Power-On Reset, on-chip R/C oscillator, on-chip crystal oscillator biasing resistor, and HALT mode.
18. Added a free running IDLE timer.
19. Added the power saving IDLE mode.
20. Added the user selectable WATCHDOG/Clock Monitor logic.
21. Nine instructions are added to the basic instruction set. The instructions are ANDSZ, IFNE, RLCA, VIS, POP A, PUSH A. RPND. LD B and IFEQ Mem.Imm.
22. The device is offered in 16-pin DIP/SO.
23. Eight bytes of user storage space in EPROM are added in addition to regular program memory space.
24. EMI reduction circuitry is added to achieve low radiated emissions.

SUMMARY OF DIFFERENCES BETWEEN COP878x AND COP8SAx7

The following table provides a detailed summary of differences between the COP878x and COP8SAx7.

TABLE I

Features	COP878x Family	COP8SAx7
Operating Voltage	4.5V to 6.0V	2.7V to 5.5V
Dynamic Supply Current CKI = 10 MHz	21 mA	7 mA
Typical HALT Current	$< 1\ \mu A$	$< 4\ \mu A$
Port L0–L7	Port L does not have Schmitt Trigger inputs.	Port L has Schmitt Trigger inputs.
Port L0–L3 Min Sink Current V_{OL} = 1.7V at V_{CC} = 4.5V	2.0 mA	15 mA
R/C Oscillator	External RC Components	On-Chip RC or On-Chip RC with External C.
R/C Oscillator Frequency Tolerance	R/C Oscillator tolerance is different than COP8SAx7. For example: for CKI = 1 MHz, R = ±1%, C = ±5%, frequency tolerance is approximately ±25% (see Datasheet for more points).	R/C Oscillator tolerance is different than COP878x. For example: for CKI = 1 MHz, using on-chip R/C or on-chip RC with external C, frequency tolerance is approximately ±35% (see Datasheet for more data points).
Crystal Oscillator	External crystal circuitry with external bias Resistor.	External crystal circuitry with the option of choosing external or internal bias resistor.
Input Only Port	Port I is an 8-bit input only port.	Port I is replaced with an 8-bit I/O (Port F).
Pin G1	Pin G1 is an I/O pin.	Pin G1 is a dedicated WATCHDOG output pin if WATCHDOG logic enabled. It is a general purpose I/O if WATCHDOG is not enabled.
Pin G2	Pin G2 is a general purpose I/O.	Pin G2 is a general purpose I/O or the timer second input capture.
NC Pins on the 44-Pin Package	Pins 21–24 are No Connection (NC).	Pins 21–24 are C4–C7 (I/O Pins).
Stack Pointer	The Stack Pointer must be initialized with software.	The Stack Pointer is initialized by hardware internally on reset.
RAM Map	1. Location 0FF Hex is a general purpose RAM register. 2. Location 0D7 hex is for Port I input pins.	1. Location 0FF Hex is reserved for future RAM expansion. If compatibility with future devices (with more RAM) is not desired, this location can be used as a general purpose RAM location. 2. Location 0D7 Hex is reserved. Port I is replaced by Port F. Port F is read into RAM location 96 Hex instead of location D7 Hex.
Reset	External reset only.	External reset or using on-chip user selectable power-on reset. The external RC delay must be greater than 5x Power Supply Rise time or 15 μs, whichever is greater.
HALT Mode	HALT mode is always enabled.	HALT can be enabled or disabled through ECON control bit.
Exit from HALT Mode	Exit from HALT through Reset or G7 pin (if not used as the oscillator output).	Exit from HALT through reset or G7 pin (if not used as the oscillator output) or Multi-Input Wakeup.
Interrupt Handling	Polled Interrupts	Vectored Interrupts
Interrupt Sources	Software Trap External Interrupt Timer Interrupt	Software Trap External Interrupt Timer Interrupt (2) Multi-Input Wakeup IDLE MICROWIRE/PLUS Default VIS

SUMMARY OF DIFFERENCES BETWEEN COP878x AND COP8SAx7 (Continued)

The following table provides a detailed summary of differences between the COP878x and COP8SAx7. (Continued)

TABLE I (Continued)

Features	COP878x Family	COP8SAx7
Software Trap	Software Trap does not set a pending flag.	Software Trap sets a pending flag.
MICROWIRE/PLUS	1. Does not allow shifting at alternate clock edge. 2. Does not support programmable idle polarity for the shift clock. 3. In Slave mode, the shift clock does not stop after 8 clock pulses.	1. Allows shifting at alternate clock edge. 2. Supports programmable idle polarity for the shift clock for SPI compatibility. 3. In Slave mode, the shift clock stop after 8 clock pulses.
Multifunction 16-Bit Timer	The timer has: 1. One 16-bit counter with a single 16-bit register. 2. One I/O pin. 3. One interrupt source. 4. No underflow interrupt pending flag in the capture mode.	The timer has: 1. One 16-bit counter with two 16-bit registers. 2. One output and two input pins. 3. Two interrupt sources. 4. Underflow interrupt pending flag in the capture mode.
ECON Register	ECON register selects the security, clock, and RAM size options.	ECON register selects the security, clock, WATCHDOG, power-on reset, and HALT options. Bit functions for the clock option and bit polarity for the security option are different. RAM size is selected by device.
Idle Timer/Idle Mode	Does not contain an Idle Timer. Does not support Idle Mode.	Contains an Idle Time. Supports Idle Mode.
WATCHDOG/Clock Monitor	Does not contain WATCHDOG/Clock Monitor.	Contains WATCHDOG/Clock Monitor.
Instruction Set	COP8 Basic family instruction set.	COP888 Feature family instruction set (9 additional instructions as compared to Basic family). The instructions are ANDSZ, IFNE, RLCA, VIS, POP A, PUSH A, RPND, LD B, and IFEQ Mem,Imm.
Packages	20-pin SO/DIP, 28-pin SO/DIP, 40-pin DIP, 44-pin PLCC.	16-pin SO/DIP, 20-pin SO/DIP, 28-pin SO/DIP, 40-pin DIP, 44-pin PLCC.
Reading EPROM with Security Enabled	Reads E0 Hex	Reads FF Hex
Contents of ECON Register Shipped from the Factory	Windowed: FF Hex OTP: 7F Hex	Windowed: 00 Hex OTP: 80 Hex
Contents of Program Memory in the Erased State	FF Hex	00 Hex **Note:** Erasure time higher (see Datasheet).
EMI	Does not include EMI reduction circuitry.	Contains EMI reduction circuitry.

COMPATIBILITY—CONVERTING FROM COP878x TO COP8SAx7

The review of differences outlined in Table I, leads to the following steps to be followed in converting from the COP878x to COP8SAx7:

Hardware/ECON Register Considerations

1. R/C Oscillator

Bits 3 and 4 of ECON register must be programmed with 1 and 0 respectively. This will select the on-chip R/C components. External R/C components are not needed unless different operating frequency is required. For a different operating frequency, only a small external capacitor needs to be added from the CKI to the GND pin. See the datasheet for the appropriate capacitor value. There are differences in R/C oscillator tolerances between the COP878x and COP8SAx7. For comparison, see the datasheets for these devices.

2. Crystal Oscillator

Bits 3 and 4 of ECON register must be programmed with 0 and 1 respectively. This will select the crystal oscillator with external biasing resistor. No change to the external crystal oscillator circuitry is required.

3. External Oscillator

Bits 3 and 4 of ECON register must be both programmed with 0.

4. Security Feature

Bit 5 of ECON register must be programmed with 1 to enable the security feature, otherwise must contain a 0 for the security feature to be disabled. This polarity is opposite of COP878x.

5. Reset

Bit 6 of ECON register must be programmed with 0 to disable the on-chip Power-On Reset. The RC delay must be greater than 5x Power Supply rise time or 15 μs, whichever is greater.

6. HALT Mode

Bit 0 of ECON register must be programmed with 0 to enable the HALT mode.

7. WATCHDOG

Bit 2 of ECON register must be programmed with 1 to disable the WATCHDOG logic. This will allow the G1 pin to serve as a general purpose I/O.

8. RAM Select

RAM size is selected by device. For example, COP8SAA has 64 bytes of RAM and COP8SAB/COP8SAC has 128 bytes of RAM.

9. Bit 7 of ECON Register

It is a factory test bit. The polarity is "Don't Care".

Software Considerations

1. Pins 21–24 (Ports C4–C7) on the 44-Pin Package

These pins are No Connection (NC) on the COP878x, but are general purpose I/O on the COP8SAx7.

2. Stack Pointer

If the Stack Pointer (SP) is initialized to a value different than 2F Hex (64 bytes of RAM selected) or to 6F Hex (128 bytes of RAM selected), in the code written for COP878x, the SP initialization instruction can be kept in the user program as is although the hardware initializes SP to 2F or 6F (depending on RAM size). If the COP878x code is initializing the SP to 2F Hex or 6F (depending on the RAM size), the initializing instruction can be deleted. If no code changed is desired, the initialization instruction can stay as is (although redundant).

3. Port I/Port F Configuration

Since Port I is replaced by Port F on the COP8SAx7, Port F is configured as Hi-Z inputs upon reset. This means Port F data register (RAM location 94 Hex) and Port F configuration register (RAM location 95 Hex) both contain values of 0 upon reset and user code must ensure they stay at 0. In addition, Port F is read into RAM location 96 Hex instead of RAM location D7 Hex. The COP878x code must be modified to reflect this change.

4. RAM Location FF Hex

RAM location FF Hex is a general purpose RAM location on the COP878x, but it is reserved on the COP8SAx7 for future RAM expansion. If compatibility with future devices (with more RAM) is not desired, this location can be used as a general purpose RAM location to maintain compatibility with COP878x.

5. Multi-Input Wakeup

The COP8SAx7 offers the power saving Multi-Input Wakeup feature. This feature is an alternate function of Port L. The COP878x does not offer this feature. Still there is no code change required because during reset initialization, the Wakeup Enable (RAM address location C9 Hex) register is cleared thus disabling the Multi-Input Wakeup feature.

6. Interrupt Handling

COP878x supports the interrupt polling scheme and it has only three interrupt sources. With the polling scheme, all interrupts cause immediate jump to the single fixed program memory location 0FF Hex. The user program must poll all interrupt pending bits to determine the cause of the interrupt. Once the cause is determined, the user program may jump to an appropriate interrupt handling routine.

The COP8SAx7, with up to eight interrupt sources, supports both polled and vectored interrupt schemes. With the vectored interrupt scheme, a vector table is used. The vector table placed in the program memory, contains the start addresses for each of the user's interrupt routines. This table is used by the device to determine where to jump when a particular interrupt occurs. When an interrupt occurs, the device initially jumps to the single fixed program memory location 0FF Hex. Then user program may call a special instruction (VIS) to cause a jump to the vector table followed by a jump to user specified interrupt service routine. The user may elect not to use the VIS instruction, not place the vector table in the program memory, and simply poll each interrupt pending bit as described for the COP878x.

In order to keep the COP878x interrupt handling code compatible with the COP8SAx7, all additional interrupt sources that are available on the COP8SAx7 must be disabled. These additional sources include interrupts associated with the Idle timer, MICROWIRE/PLUS, and Multi-Input Wakeup. Since the COP8SAx7 multi-function timer block contains two interrupt sources, to maintain compatibility with the COP878x, the second interrupt source (Timer T1B Interrupt Enable) must be disabled. All the Enable bits associated with these additional interrupt sources reside in the ICNTRL register (RAM location E8 Hex) and are cleared upon reset. Therefore, there is no code modification required as long as the user keeps the new additional sources disabled, and not use the additional COP8SAx7 interrupt feature (VIS instruction and vector table).

Software Interrupt

The COP8SAx7 software interrupt has a pending flag. This flag is not memory mapped and is cleared by the special RPND instruction. To keep the COP878x code compatible with that of the COP8SAx7 and vectored interrupt scheme is not used, upon entering the interrupt routine (program memory location 0FF Hex), the user program must execute the RPND instruction. This will reset the software interrupt pending flag. The next step is to check the timer and external interrupt pending bits. If none are set, it gives the indication that a software interrupt has occurred.

7. Timer

The timer block contains a second 16-bit register that can be used as an autoreload or capture register depending on the timer operating mode. The following steps need to be considered to run the code written for the COP878x timer on the COP8SAx7.

PWM Mode

a. The second interrupt source must remain disabled (T1ENB bit in the ICNTRL register must be 0). T1ENB bit is cleared upon reset.

b. Pin G3 (T1A) must be used as the timer PWM output pin. Pin G2 is available as a general purpose I/O.

c. The timer's second reload register (T1RB) must be loaded with the same value as the contents of T1RA. This means an instruction must be inserted into the COP8SAx7 code to initialize T1RB.

External Event Mode

a. The second interrupt source must remain disabled (T1ENB bit in the ICNTRL register must be 0). T1ENB bit is cleared upon reset.

b. Pin G3 (T1A) must be used as the timer external event input pin. Pin G2 is available as a general purpose I/O.

c. The timer's second reload register (T1RB) must be loaded with the same value as the contents of T1RA. This means an instruction must be inserted into the COP8SAx7 code to initialize T1RB.

Capture Mode

a. The second interrupt source must remain disabled (T1ENB bit in the ICNTRL register must be 0). T1ENB bit is cleared upon reset.

b. Pin G3 (T1A) must be used as the timer input capture pin. The alternate function of pin G2 (G2 is available as a general purpose I/O) and the capture register T1RB must be ignored. The user can simply read the capture register T1RA.

8. MICROWIRE/PLUS

The MICROWIRE/PLUS feature on the COP8SAx7, supports shifting serial data with the alternate edge of the shift clock and has the ability to select idle polarity for the shift clock. To ensure compatibility with COP878x, bit 6 of Port G configuration register and bit 5 of Port G data register must both contain 0. Upon reset, these bits are cleared and the user code must ensure they stay at 0. In the slave mode, the shift clock on the COP8SAx7 does not stop after 8 clock pulses. The shift clock on the COP8SAx7, on the other hand, stops after 8 clock pulses. No change is required as a result of this difference.

Comparison of COP82xCJ to the Enhanced COP8SAx7 Family—Hardware/ Software Considerations

National Semiconductor
Application Note 1044
Abdul Aleaf

INTRODUCTION

The COP820CJ, COP822CJ and COP823CJ are members of the COP8 Basic Family that contain ROM, RAM, a 16-bit multi-function timer with a single 16-bit autoreload/capture register, 3 interrupt sources with polling scheme, WATCHDOG™ Timer, Modulator/Timer, Brown Out detection, Multi-Input Wakeup, and the MICROWIRE/PLUS™ serial interface. These devices are offered in 28-pin SO/DIP, 20-pin SO/DIP, 16-pin SO packages. The operating voltage range is from 2.5V to 6.0V.

The COP8SAA7, COP8SAB7 and COP8SAC7 (COP8SAx7 family) devices, on the other hand, are members of the COP8 Feature Family that contain EPROM, RAM, a 16-bit multi-function timer with two autoreload/capture registers, eight interrupt sources supporting vectored interrupt scheme, and the enhanced MICROWIRE/PLUS serial interface. In addition, these devices contain features such as Multi-Input Wakeup, WATCHDOG/Clock Monitor, Idle timer supporting Idle mode, internal Power-On Reset, on-chip RC oscillator, 8 bytes of user storage space in EPROM, and on-chip EMI reduction circuitry. The devices are offered in 44-pin PLCC, 40-pin DIP, 28-pin SO/DIP, 20-pin SO/DIP, and 16-pin SO/DIP packages. The operating voltage range is from 2.7V to 5.5V. See the datasheet for more details.

The purpose of this Application Note is to provide a detailed comparison and feature analysis of these two family of devices. Where applicable, this report can be used to assist in converting the code written for the COP82xCJ device to operate on an equivalent COP8SAx7 device.

As suggested, the COP8SAx7 family offers additional useful and enhanced features as compared to the COP82xCJ family. With the additional features, the COP8SAx7 family may appear much different, but this family is designed to maintain downward compatibility with the COP82xCJ family as much as possible.

The COP82xCJ family offers some features that are not supported by the COP8SAx7 family. Examples of such features are Brown Out detection, comparator, Modulator, and a second 8-bit timer with 8-bit prescaler (if WATCHDOG function is not used). If these features are already used on the COP82xCJ design, the conversion to the COP8SAx7 family will be more involved and may require additional external components and code changes.

HIGHLIGHTS OF COP8SAx7 ENHANCEMENTS OVER COP82xCJ

1. The operating voltage range is wider (2.7V–5.5V).
2. The R/C oscillator option is expanded to include the choice of selecting on-chip R/C components. The use of on-chip R/C eliminates the cost associated with external R/C oscillator components.
3. The crystal oscillator option is expanded to include the choice of selecting on-chip biasing resistor.
4. COP82xCJ's Port I is replaced with a bi-directional Port F.
5. Stack Pointer is automatically initialized by hardware.
6. The HALT feature is made user selectable.
7. The interrupt handling is enhanced to support a versatile vectored interrupt scheme. It has a total of eight interrupt sources with independent vectors. The COP82xCJ has a polled interrupt scheme.
8. The Software Trap scheme is enhanced to include its own pending flag.
9. The MICROWIRE/PLUS serial interface is enhanced to include shifting on the alternate clock edge and programmable idle polarity for the shift clock. This allows compatibility with SPI peripherals.
10. The 16-bit multi-function timer/counter is improved to include two autoreload/capture registers. The timer block contains two separate interrupt vectors and two I/O pins. The COP82xCJ timer has a single autoreload/capture register, a single interrupt source, and a single I/O pin.
11. Added a free running IDLE timer.
12. Added the power saving IDLE mode.
13. Nine instructions are added to the basic instruction set. The instructions are ANDSZ, IFNE, RLCA, VIS, POP A, PUSH A, RPND, LD B and IFEQ Mem,Imm.
14. The device is offered in 44-pin PLCC, 40-pin DIP, and 16-pin DIP, in addition to 28-pin SO/DIP, 20-pin SO/DIP, and 16-pin SO packages.
15. EMI reduction circuitry is added to achieve low radiated emissions.

SUMMARY OF DIFFERENCES BETWEEN COP82xCJ AND COP8SAx7

The following table provides a detailed summary of differences between the COP82xCJ and COP8SAx7.

TABLE I

Features	COP82xCJ Family	COP8SAx7
Operating Voltage	2.5V–6.0V	2.7V–5.5V
Dynamic Supply Current CKI = 10 MHz	6 mA	7 mA
Typical HALT Current	< 1 μA (Brown Out Disabled)	< 4 μA
R/C Oscillator	External RC Components.	On-Chip RC or On-Chip RC with External C.
R/C Oscillator Frequency Tolerance	R/C Oscillator tolerance is different than COP8SAx7. For example: for CKI = 1 MHz, R = ±1%, C = ±5%, frequency tolerance is approximately ±25% (see Datasheet for more data points).	R/C Oscillator tolerance is different than COP82xCJ. For example: for CKI = 1 MHz, using on-chip R/C or on-chip RC with external C, frequency tolerance is approximately ±35% (see Datasheet for more data points).
Crystal Oscillator	External crystal circuitry with external bias Resistor.	External crystal circuitry with the option of choosing external or internal bias resistor.
Input Only Port	Port I is an 8-bit input only port.	Port I is replaced with an 8-bit I/O (Port F).
Pin G1	Pin G1 is an I/O pin.	Pin G1 is a dedicated WATCHDOG output pin if WATCHDOG logic enabled. It is a general purpose I/O if WATCHDOG is not enabled.
Pin G2	Pin G2 is a general purpose I/O.	Pin G2 is a general purpose I/O or the timer second input capture.
Stack Pointer	The Stack Pointer must be initialized with software.	The Stack Pointer is initialized by hardware internally on reset.
RAM Map	1. Location 0FF Hex is a general purpose RAM register. 2. Location 0D7 hex is for Port I input pins.	1. Location 0FF Hex is reserved for future RAM expansion. If compatibility with future devices (with more RAM) is not desired, this location can be used as a general purpose RAM location. 2. Location 0D7 Hex is reserved. Port I is replaced by Port F. Port F is read into RAM location 96 Hex instead of location D7 Hex.
Reset	External reset or Brown Out reset. Brown Out reset can be used as a Power-On Reset.	External reset or using on-chip user selectable Power-On Reset. The external RC delay must be greater than 5x Power Supply Rise time or 15 μs, whichever is greater.
HALT Mode	HALT mode is always enabled.	HALT can be enabled or disabled through ECON control bit.
Interrupt Handling	Polled Interrupts	Vectored Interrupts
Interrupt Sources	Software Trap External Interrupt Timer Interrupt	Software Trap External Interrupt Timer Interrupt (2) Multi-Input Wakeup IDLE MICROWIRE/PLUS Default VIS
Software Trap	Software Trap does not set a pending flag.	Software Trap sets a pending flag.

SUMMARY OF DIFFERENCES BETWEEN COP82xCJ AND COP8SAx7 (Continued)

The following table provides a detailed summary of differences between the COP82xCJ and COP8SAx7. (Continued)

TABLE I (Continued)

Features	COP82xCJ Family	COP8SAx7
MICROWIRE/PLUS	1. Does not allow shifting at alternate clock edge. 2. Does not support programmable idle polarity for the shift clock. 3. In Slave mode, the shift clock does not stop after 8 clock pulses.	1. Allows shifting at alternate clock edge. 2. Supports programmable idle polarity for the shift clock for SPI compatibility. 3. In Slave mode, the shift clock stop after 8 clock pulses.
Multifunction 16-Bit Timer	The timer has: 1. One 16-bit counter with a single 16-bit register. 2. One I/O pin. 3. One interrupt source. 4. No underflow interrupt pending flag in the capture mode.	The timer has: 1. One 16-bit counter with two 16-bit registers. 2. One output and two input pins. 3. Two interrupt sources. 4. Underflow interrupt pending flag in the capture mode.
Program Memory Security	COP82xCJ are mask ROMed devices. Program memory contents cannot be read in an EPROM programmer environment.	Contains EPROM security feature.
ECON Register	No ECON register. Clock configuration is selected by a mask option. RAM size is fixed. WATCHDOG is enabled by software. Brown Out detection (can be used as Power-On-Reset) is a mask option. No ROM security feature.	ECON register selects the security, clock, WATCHDOG, Power-On Reset, and HALT options. Bit functions for the clock option and bit polarity for the security option are different. RAM size is selected by device.
Idle Timer/Idle Mode	Does not contain an Idle Timer. Does not support Idle Mode.	Contains an Idle Time. Supports Idle Mode.
WATCHDOG/Clock Monitor	WATCHDOG is enabled by software. WATCHDOG window is programmable through an 8-bit timer preceeded by an 8-bit prescaler. WATCHDOG trigger generates internal reset. No Clock Monitor logic.	WATCHDOG is active upon Power-up. WATCHDOG has a programmable upper window (4 choices) and a fixed lower window. WATCHDOG pin goes active low upon WATCHDOG trigger condition. Contains Clock Monitor logic.
Brown Out Detection	Contains on-chip Brown Out detection circuit.	Does not contain on-chip Brown Out detection circuit.
Comparator	Contains an on-chip comparator.	No on-chip comparator.
Modulator Timer	Contains a Modulator Timer	No Modulator Timer.
WATCHDOG Timer	If WATCHDOG function is not used, the WATCHDOG timer can be used as a general purpose timer for internal time keeping.	The Idle timer provides timing for the WATCHDOG logic. Cannot use Idle timer as a general purpose internal timer.
Instruction Set	COP8 Basic family instruction set.	COP888 Feature family instruction set (9 additional instructions as compared to Basic family). The instructions are ANDSZ, IFNE, RLCA, VIS, POP A, PUSH A, RPND, LD B, and IFEQ Mem,Imm.
Packages	28-pin SO/DIP, 20-pin SO/DIP, 16-pin SO.	44-pin PLCC, 40-pin DIP, 28-pin SO/DIP, 20-pin SO/DIP, 16-pin SO/DIP.
EMI	Does not include EMI reduction circuitry.	Contains EMI reduction circuitry.

COMPATIBILITY—CONVERTING FROM COP82xCJ TO COP8SAx7

The review of differences outlined in Table I, leads to the following steps to be followed in converting from the COP82xCJ to COP8SAx7:

Hardware/ECON Register Considerations

1. R/C Oscillator

Bits 3 and 4 of ECON register must be programmed with 1 and 0 respectively. This will select the on-chip R/C components. External R/C components are not needed unless different operating frequency is required. For a different operating frequency, only a small external capacitor needs to be added from the CKI to the GND pin. See the datasheet for the appropriate capacitor value. There are differences in R/C oscillator tolerances between the COP82xCJ and COP8SAx7. For comparison, see the datasheets for these devices.

2. Crystal Oscillator

Bits 3 and 4 of ECON register must be programmed with 0 and 1 respectively. This will select the crystal oscillator with external biasing resistor. No change to the external crystal oscillator circuitry is required.

3. External Oscillator

Bits 3 and 4 of ECON register must be both programmed with 0.

4. Security Feature

Bit 5 of ECON register (COP8SAx7) must be programmed with 1 to enable the EPROM security feature, otherwise must contain a 0 for the security feature to be disabled. COP82xCJ devices are mask ROMed. Program memory (ROM) cannot be read in an EPROM programmer environment.

5. Reset

Bit 6 of the COP8SAx7 ECON register can be programmed with "0" to disable the on-chip Power-On-Reset. By doing this, external reset can be used and the device will perform the same as the COP82xCJ except that RC delay must be greater than 5x Power Supply rise time or 15 μs, whichever is greater.

If Brown Out detection feature is selected on the COP82xCJ (by mask option) as a Power-On-Reset, bit 6 of the COP8SAx7 ECON register must be programmed with "1" to enable the on-chip Power-On-Reset.

If Power-On-Reset is selected, the user needs to consider the V_{CC} rise time specification of 10 ns to 50 ms. There is no spec limitation on COP82xCJ Brown Out detection feature.

6. HALT Mode

Bit 0 of ECON register must be programmed with 0 to enable the HALT mode.

7. WATCHDOG

The COP82xCJ has an on-chip 8-bit WATCHDOG timer. The timer contains an 8-bit READ/WRITE down counter clocked by an 8-bit prescaler. The WATCHDOG can be enabled or disabled (only once) after the device is reset as a result of brown out reset or external reset. On power-up the WATCHDOG is disabled. The WATCHDOG is enabled by writing a "1" to a bit in WATCHDOG register (WDREG). Once enabled, the user program should write periodically into the 8-bit counter before the counter underflows. An internal reset is generated if the user programs fail to do so.

The COP8SAx7 family, on the other hand, contains a user selectable WATCHDOG and Clock Monitor. The WATCHDOG is enabled by bit 2 of the ECON register. When this bit is 0, the WATCHDOG is enabled and pin G1 becomes the WATCHDOG output pin with a weak pullup. The WATCHDOG logic contains two separate service windows. While the user selectable upper window selects the WATCHDOG service time, (four choices of 8k t_C, 32k t_C, 64k t_C, where t_C = instruction cycle time), the lower window (fixed time of 256 instruction cycles) provides protection against an infinite program loop that contains the WATCHDOG service instruction. The Clock Monitor is used to detect the absence of a clock or a very slow clock below a specified rate on the CKI pin. The WATCHDOG and Clock Monitor are disabled during reset. The devices comes out of reset with the WATCHDOG armed, and the Clock Monitor enabled. Servicing the WATCHDOG consists of writing a specific value to a WATCHDOG Service Register (WDSVR) which is memory mapped in the RAM. The service value is composed of three fields, consisting of a 2-bit Window Select, a 5-bit Key Data field, and the 1-bit Clock Monitor Select field. The WDSVR register can be written to only once after reset to select the proper upper window and the Clock Monitor feature. Upon triggering the WATCHDOG, to the logic will pull the WATCHDOG output pin (G1) low for 16 t_C–32 t_C. The WATCHDOG pin may be connected to the Reset pin externally to reset the device upon WATCHDOG trigger condition.

If the WATCHDOG feature is used on the COP82xCJ, the following steps need to be considered to convert to COP8SAx7.

a. Clear bit 2 of the ECON register to select WATCHDOG feature.

b. Select the proper values for bit 6 and bit 7 of the WDSVR (WATCHDOG Service Register) to obtain the appropriate upper window service time. Select one of the four choices that is as close to the COP82xCJ service time as possible. If there is mismatch between the service times, adjust the code to service the WATCHDOG within the new selected upper window.

c. Locate the instruction that enables the WATCHDOG and is placed in the beginning of the COP82xCJ code. Replace this instruction with a new one to perform a write to the COP8SAx7 WATCHDOG Service Register (WDSVR). This instruction should select the proper upper window, enable or disable Clock Monitor, and write a 5-bit data to match the Key Data.

d. The COP8SAx7 must be serviced at least once before the upper window expires and not more than once within a 256 instruction cycle window (lower window).

e. The WATCHDOG output (pin G1) must be connected externally to the Reset pin.

8. Bit 7 of ECON Register

It is a factory test bit. The polarity is "Don't Care".

Software Considerations

1. Stack Pointer

If the Stack Pointer (SP) is initialized to a value different than 2F Hex (64 bytes of RAM selected) or to 6F Hex (128 bytes of RAM selected), in the code written for COP82xCJ, the SP initialization instruction can be kept in the user program as is although the hardware initializes SP to 2F or 6F (depending on RAM size). If the COP82xCJ code is initializing the SP to 2F Hex or 6F (depending on the RAM size), the initializing instruction can be deleted. If no code changed is desired, the initialization instruction can stay as is (although redundant).

2. Port I/Port F Configuration

Since Port I is replaced by Port F on the COP8SAx7, Port F is configured as Hi-Z inputs upon reset. This means Port F data register (RAM location 94 Hex) and Port F configuration register (RAM location 95 Hex) both contain values of 0 upon reset and user code must ensure they stay at 0. In addition, Port F is read into RAM location 96 Hex instead of RAM location D7 Hex. The COP82xCJ code must be modified to reflect this change.

3. RAM Location FF Hex

RAM location FF Hex is a general purpose RAM location on the COP82xCJ, but it is reserved on the COP8SAx7 for future RAM expansion. If compatibility with future devices (with more RAM) is not desired, this location can be used as a general purpose RAM location to maintain compatibility with COP82xCJ.

4. Interrupt Handling

COP82xCJ supports the interrupt polling scheme and it has only three interrupt sources. With the polling scheme, all interrupts cause immediate jump to the single fixed program memory location 0FF Hex. The user program must poll all interrupt pending bits to determine the cause of the interrupt. Once the cause is determined, the user program may jump to an appropriate interrupt handling routine.

The COP8SAx7, with up to eight interrupt sources, supports both polled and vectored interrupt schemes. With the vectored interrupt scheme, a vector table is used. The vector table placed in the program memory, contains the start addresses for each of the user's interrupt routines. This table is used by the device to determine where to jump when a particular interrupt occurs. When an interrupt occurs, the device initially jumps to the single fixed program memory location 0FF Hex. Then user program may call a special instruction (VIS) to cause a jump to the vector table followed by a jump to user specified interrupt service routine. The user may elect not to use the VIS instruction, not place the vector table in the program memory, and simply poll each interrupt pending bit as described for the COP82xCJ.

In order to keep the COP82xCJ interrupt handling code compatible with the COP8SAx7, all additional interrupt sources that are available on the COP8SAx7 must be disabled. These additional sources include interrupts associated with the Idle timer, MICROWIRE/PLUS, and Multi-Input Wakeup. Since the COP8SAx7 multi-function timer block contains two interrupt sources, to maintain compatibility with the COP82xCJ, the second interrupt source (Timer T1B Interrupt Enable) must be disabled. All the Enable bits associated with these additional interrupt sources reside in the ICNTRL register (RAM location E8 Hex) and are cleared upon reset. Therefore, there is no code modification required as long as the user keeps the new additional sources disabled, and not use the additional COP8SAx7 interrupt feature (VIS instruction and vector table).

Software Interrupt

The COP8SAx7 software interrupt has a pending flag. This flag is not memory mapped and is cleared by the special RPND instruction. To keep the COP82xCJ code compatible with that of the COP8SAx7 and vectored interrupt scheme is not used, upon entering the interrupt routine (program memory location 0FF Hex), the user program must execute the RPND instruction. This will reset the software interrupt pending flag. The next step is to check the timer and external interrupt pending bits. If none are set, it gives the indication that a software interrupt has occurred.

5. Timer

The timer block contains a second 16-bit register that can be used as an autoreload or capture register depending on the timer operating mode. The following steps need to be considered to run the code written for the COP82xCJ timer on the COP8SAx7.

PWM Mode

a. The second interrupt source must remain disabled (T1ENB bit in the ICNTRL register must be 0). T1ENB bit is cleared upon reset.

b. Pin G3 (T1A) must be used as the timer PWM output pin. Pin G2 is available as a general purpose I/O.

c. The timer's second reload register (T1RB) must be loaded with the same value as the contents of T1RA. This means an instruction must be inserted into the COP8SAx7 code to initialize T1RB.

External Event Mode

a. The second interrupt source must remain disabled (T1ENB bit in the ICNTRL register must be 0). T1ENB bit is cleared upon reset.

b. Pin G3 (T1A) must be used as the timer external event input pin. Pin G2 is available as a general purpose I/O.

c. The timer's second reload register (T1RB) must be loaded with the same value as the contents of T1RA. This means an instruction must be inserted into the COP8SAx7 code to initialize T1RB.

Capture Mode

a. The second interrupt source must remain disabled (T1ENB bit in the ICNTRL register must be 0). T1ENB bit is cleared upon reset.

b. Pin G3 (T1A) must be used as the timer input capture pin. The alternate function of pin G2 (G2 is available as a general purpose I/O) and the capture register T1RB must be ignored. The user can simply read the capture register T1RA.

6. MICROWIRE/PLUS

The MICROWIRE/PLUS feature on the COP8SAx7, supports shifting serial data with the alternate edge of the shift clock and has the ability to select idle polarity for the shift clock. To ensure compatibility with COP82xCJ, bit 6 of Port G configuration register and bit 5 of Port G data register must both contain 0. Upon reset, these bits are cleared and the user code must ensure they stay at 0. In the slave mode, the shift clock on the COP8SAx7 does not stop after 8 clock pulses. The shift clock on the COP8SAx7, on the other hand, stops after 8 clock pulses. No change is required as a result of this difference.

Replacing Dedicated Protocol Controllers with Code Efficient and Configurable Microcontrollers—Low Speed CAN Network Applications

National Semiconductor
Application Note 1048
Martin Embacher

BACKGROUND

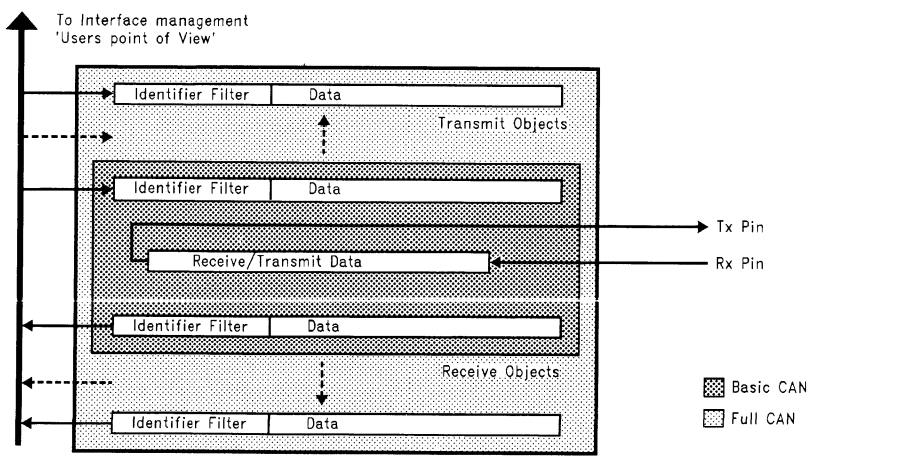
The CAN (Controller Area Network) is one of today's most widely accepted car networking systems. Various protocol implementations are available from different suppliers. Dedicated protocol controllers—Full-CAN controllers—are found as system bus interfaces connected to a main CPU or integrated into them. Yet in some applications, particularly in the low speed arena, these devices don't meet the price target or offer the flexibility required by the system designer. This Application Note outlines the application interfaces available for the CAN protocol, gives an overview to National Semiconductor's CAN devices and demonstrates in a practical example how these products can help to minimize the cost of Full-CAN controller applications while increasing the flexibility of such systems.

INTRODUCTION

CAN is designed to address the needs for a highly reliable protocol with maximum throughput for interconnecting multiple autonomous controller modules within harsh industrial or automotive applications. The need for such a system arose first when more and more electronic modules where introduced to the automobiles, resulting in huge amounts of wires being lain out within a car to perform interconnection between control modules and the sensors/actuators. The first objective of the CAN system was to reduce these kilometers of cabling and thereby reduce system cost by saving wiring effort. Additionally, the system had to have maximum reliability as basically all functions within a car could introduce a safety risk. Next to obviously important functions such as motor management and anti-blocking systems, comfort electronic functions can lead to unsafe operation of cars. Taking a faulty electronically controlled driver seat as an example this becomes more clear. Only assume due to a fault the seat is suddenly moving while the car runs at high speed.

BASIC AND Full-CAN IMPLEMENTATIONS

Many Semiconductor suppliers implemented various versions of a CAN user interface. Even the protocol remains the same. Basic-CAN implementations provide only the basic functionality of a CAN interface with the capability to buffer only one message with a limited acceptance filter. Though, an additional burden is placed on the CPU since it has to perform message filtering next to its regular task. Full-CAN controllers extend this basic features by not only implementing the protocol—moreover they implement a complete message "server" capable of automatically receiving and transmitting multiple messages on the CAN bus without interrupting the system's main CPU if it is not necessary. *Figure 1* shows the Basic-CAN interface with the extension for Full-CAN from the programmer's point of view.



TL/DD/12850–1

FIGURE 1. CAN Programming Model

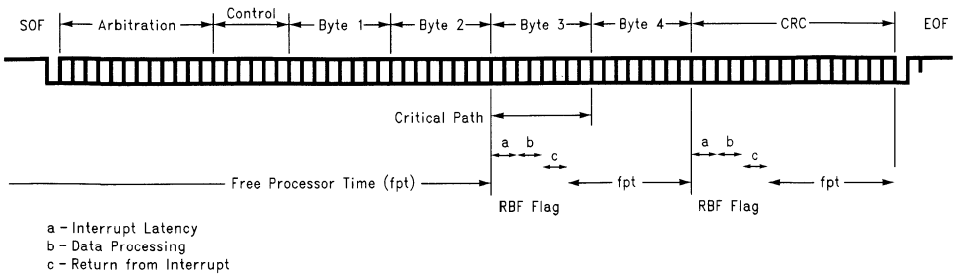
The hardware interface to the processor is provided with either serial links or a paralell interface with message data (identifier, control and data) being accesed on memory mapped address locations, so the user "sees" message data and control information.

Typical multiplex systems consist of one or more of each protocol implementation connected to each other over a common bus. The Full-CAN is used where the CPU has to perform a magnitute of other tasks and where communication needs to be highly independent from the rest of the software. A Basic-CAN controller is used in areas in which the CPU has some spare performance to assist the communication work. Full-CAN implementations with their higher level of functionality, require a larger silicon area than a Basic-CAN implementation, which translates directly into higher prices. Though, from a systems designer's point of view, it might be desirable to use as many Full-CAN controllers as possible to free up the CPU for applications tasks and provide free processing resources for future or different features. Common to both implementations is that they can process at least one receive and one transmit message object completely autonomously—which results in a specific number of registers being required on the CAN block to store the information. Lastly, fully autonomous CAN modules, commonly known as SLIO (Serial Linked I/O) devices are also available. These devices integrate a quasi Full-CAN controller with self-sufficient simple I/O capabilities, having no need for a main CPU within the module and therefore no dedicated programming requirements. With these devices a simple CAN module can be designed by only developing the input and output circurity required for the specific application. All I/O control is then provided via the CAN network. Examples are National's MM57C360 and MM57C362, which integrate a CAN cell, a message sequencer and I/O drivers. With this functionality the device is perfectly suited for simple modules like keypads or light clusters.

NATIONAL'S COPCAN INTERFACE

With the implementation of the COPCAN on the COP8™ microcontroller family, National has addressed especially the high implementation costs of previous CAN modules by reducing the amount of registers required to implement the CAN protocol. The driving factors were cost on the one side and the idea that not all applications require the high speed feature all CAN implementation known so far offered. To reduce cost the object buffer for both receive and transmit was reduced from 10 bytes (2 identifier + 8 data bytes) to 4 bytes (2 identifier + 2 data bytes). The "remainder" of the CAN interface, error management, BTL was not changed in order to achieve full compatibility. With the reduction of registers the interface is no longer capable to process messages with more than two bytes of data independently. Data needs to be provided by the main processor in time. This register reduction, however, has no influence on the interface performance in low speed (<125 kbit/s) applications, since the processor has enough time to store/provide the data when required. Interrupt flags indicate to the CPU when the processor needs to provide data to the COPCAN interface. This results in a ratio between the maximum possible bus speed and the time the processor needs to save and provide data which will now be explained in more detail. On the one side, the COP8 microcontroller core features an instruction cycle time of 1 μs = 1 t_C with an external clock of 10 MHz. Most instructions take one t_C to execute. On the other side, with a bus speed of 125 kbit/s, typical for low speed applications, one byte time on the CAN bus takes a minimum of: 8 (bit) * (1/125 kHz) (μs) = 64 μs, without the possible stuff bits. With a given interrupt latency time of 20 t_C maximum (including transfer of control instructions) this leaves 44 μs to store the receive data or write new data into the transmit register. *Figure 2* shows the timing of a message reception with four data bytes. It can be seen from the picture that adding data bytes to the frame would neither introduce a critical path nor decrease the processor's free time.

In this example, the CPU's usage to store the received data is given as approximately 20 t_C. The critical path is to read the first receive buffer byte after the receive buffer full flag (RBF) is set and before it gets overwritten by new incoming data. During the free processor time, other application tasks can be exceuted. Basically the same example is valid for a transmitter. The main difference is that transmitting data is mostly synchronus to the program's excecution where receiving is asynchronous. Thus, the transmission of data is not time-critical. Also, using a high speed link (>125 kbit/s) is possible for applications which don't need to receive more than two bytes of data.



TL/DD/12850-2

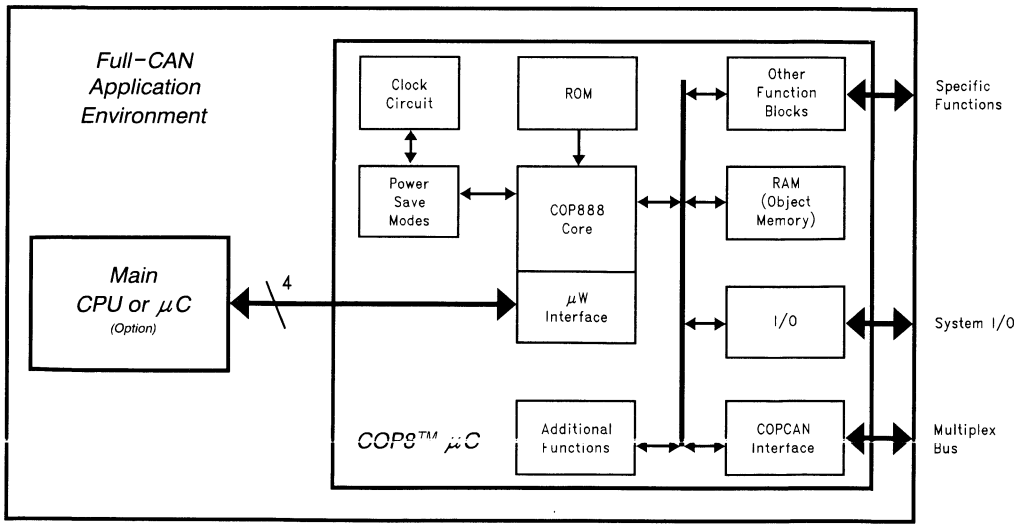
FIGURE 2. Message Processing with the COPCAN Interface

IMPLEMENTING A Full-CAN PROCESSOR WITH A COP8 MICROCONTROLLER

National's COP8 microcontroller core contains, beside the pure CPU, a serial synchronous MICROWIRE/PLUS™ interface and a processor independent Timer. Additional functional blocks like the COPCAN interface, Timers, USART, A/D convertors and various sizes of ROM and RAM can be added with the Configurable Controller Methodology (CCM). Today several standard parts are offered, of which two feature the COPCAN interface. The COP884BC and the COP888EB. All COP8 microcontroller family members are also available as one time programmable (OTP) devices. The following section shows how a protocol processor, providing a customizable Full-CAN interface, can be integrated with the COP8 family. A block diagram of the setup with the microcontroller and the main CPU is shown in *Figure 3*. Additional customized options, like time stamp for received messages or timed automatic transmission of data, can be integrated by simply altering the software. In addition, several data processing tasks, i.e., automatic keyboard scanning can be integrated—thus reducing the overhead on the main CPU and freeing up processing resources. Another advantage of having a second CPU in the system is automatic diagnostics, either with a specific protocol or with the WATCHDOG™ circuit integrated on the COP8. Finally, the power-save features of the COP8 microcontrollers help to minimize power consumption in the application by gradually switching off modules—including the main CPU. The multi-input wake-up feature allows multiple sources to return from the save mode to the active mode. The interface to the main CPU can be chosen to be provided with standard I/O ports of the microcontroller, the MICROWIRE/PLUS interface or, if very high speed communication is required, with a newly developed high-speed serial link.

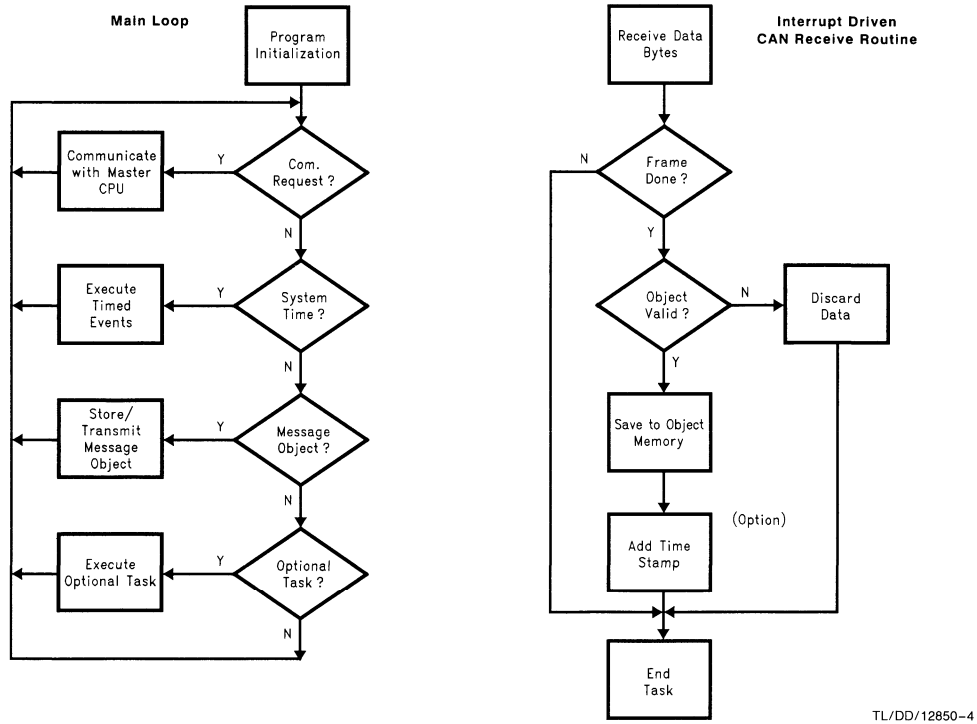
In this example, however, the MICROWIRE/PLUS interface is used. Communication is done with three wires and one handshake signal. The data from the main CPU to the microcontroller is transmitted in packets of eight bits with a customizable protocol. The MICROWIRE/PLUS interface can be programmed to generate an interrupt every eight clocks applied to the SK, thus indicating the master CPU wants to exchange some commands or data with the microcontroller. After the COP8 reads out the data from the MICROWIRE/PLUS register, it returns an acknowledge signal to the main processor, by toggling the handshake line. *Figure 3* shows a block diagram of the COP8 microcontroller linked with the main CPU. The instructions stored in the COP8 ROM first execute the protocol between the master and the COP8, then process CAN messages, and finally filter out unwanted data.

The software of the application is divided into several tasks which allow easy customization. A main loop continuously polls various flags. These are set by the microcontroller's hardware, like the system timer, the muti-sourced external interrupt/wake-up or by the interrupt handlers (e.g., of the MICROWIRE/PLUS interface or CAN interface). The MICROWIRE/PLUS interrupt indicates a main CPU communication request. The CAN interrupts are receive, transmit and error. All of them are leading to a separate interrupt vector within the COP8 memory. Upon detecting one flag to be set, the program branches to the certain subroutine. This program structure is chosen to ensure fast response times for the time-critical communication parts CAN and MICROWIRE/PLUS. A flowchart of the main routine is found in *Figure 4*, together with the CAN receive interrupt handler.



TL/DD/12850–3

FIGURE 3. COP8-Based Full-CAN Interface

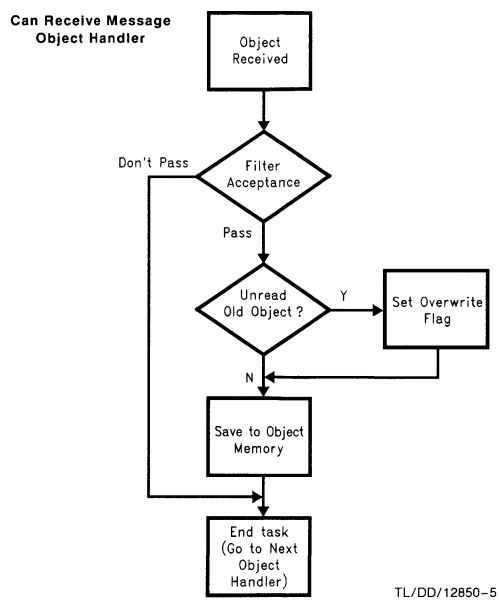


TL/DD/12850–4

FIGURE 4. Main Program and CAN Receive Interrupt Handler

The communication request flag is set as soon as the MICROWIRE/PLUS received the first byte, indicating the command for the COP8. This data was read from the μW shift register and the handshake signal set, to indicate the possibility to read or write the next data byte to the main CPU. Within the communication subroutine these data bytes are exchanged with the main processor. The system time can be generated by the idle timer's pending flag. This flag is set every 4096 t_C on the COP884BC and it can be programmed to be set every 4k, 8k, 16k or 32k t_C on the COP888EB. Secondly, the system time can be generated with a free programmable 16 bit auto-reload timer T1, for increased flexibility. Timed events, like automatic transmission of a CAN message or a software real time clock are then executed. CAN message objects are handled by a subroutine as described later. Finally, flags for optional tasks can be included and polled within the main loop in order to comprise additional features.

The CAN receive interrupt routine stores received data in receive buffers located within the RAM (*Figure 2*). For this data storage, special memory locations—base page RAM—are used as indirect addressing operations in this area and are executed faster than if used on the remaining RAM area. After a complete message object is received, and no errors occurred, a flag is set. Optionally, the system's time can be stored as well to allow verification of the creation time for a specific message in real time systems. Then the message object is filtered and stored into its final location within the message object handlers. One receive message object handler is shown in *Figure 5* and described below. CAN transmit interrupts work similarly to the receive routine on a different interrupt vector. Additions to the transmit



TL/DD/12850–5

FIGURE 5. CAN Receive Object Handler

schedule routine—which are not offered with standard Full-CAN chips—can be done as well. For example, a transmit object may be verified to be sent within a specific time or within a certain number of retries if they're likley to lose arbitration on a highly frequented bus. Another example is the automatic transmission of the system's (real) time in order to have a common time base over the network.

The CAN receive object handler is called after a CAN message was satisfactorily received. One object handler verifies the received message object with its specific acceptance filter and stores the data if the message's identifier matches. Otherwise, the next receive object handler is called until all possible receive objects are verified. Afterwards, a flag may be set to indicate the reception of a message by the main CPU. If new data for one object is received, a flag is set to indicate the data overwrite. The number of possible message objects to be stored is only limited by the processor's RAM.

SOFTWARE EXAMPLE

After theoretically outlining the implementation of the protocol processor's software, this section provides an example in COP8 assembly language to instanciate one receive object handler. The program is written in the form of a macro which allows multiple message handlers to be used within one program by simply calling the macro several times. The program uses 10 or 12 bytes of RAM for every message object and two global status bytes. One to indicate the reception of a message (rx__status) and one to indicate the overrun condition if new data is received before it was transmitted to the main controller (rx__overwrite).

The parameters to the macro are the number of the message object and a pointer to the message object memory. The program is executed as shown in *Figure 5*. After initializing some pointers (lines 004 and 005) the received message's identifier is compared with the identifier of the current object (lines 006 to 013). As the identifier now matches, it is checked if the object's data was read by the main processor. If the data was not read, the overrun flag is set (lines 014 and 015). Finally, the received data is copied into the object's memory (line 016 to 033).

This macro uses 41 bytes of ROM and takes a maximum of 19 t_C until the acceptance of one message is filtered.

CONCLUSION

This paper explained the different programming models of CAN chips. It showed how a Full-CAN controller to move and shape the information contained within the messages can be implemented at low cost. Finally, the flexibility of a microcontroller solution with customizable software compared to a fixed chip solution was outlined.

```
(001)    .macro rx_object, obj_number, msg_obj
(002)    .local                              ; local variables used
(003)    $message_filter:                    ;
(004)        ld      b, #msg_obj             ; point to msg_obj
(005)        ld      x, #rx_buffer           ; point to receive buffer
(006)        ld      a, [x+]                 ; get receive identifier
(007)        ifne    a, [b]                  ; compare with object id
(008)        jp      $end_msg                ; if fail - then end
(009)    $idlc_test:                         ;
(010)        ld      a, [b+]                 ; increment rx buff pointer
(011)        ld      a, [x+]                 ; get remaining receive id
(012)        ifne    a, [b]                  ; compare with object id
(013)        jp      $end_msg                ; if fail - then end
(014)        ifbit   obj_number, rx_status   ; data received before ?
(015)        sbit    obj_number, rx_overwite ; then indicate
(016)        andsz   a, #0x0f                ; mask data length code
(017)        jp      $copy_loop              ; and copy data
(018)        jp      $end_obj                ; if dlc == 0 then end
(019)        x       a, byte_count           ; save dlc to byte counter
(020)    $copy_loop:                         ;
(021)        ld      a, [x+]                 ; read bytes
(022)        x       a, [b+]                 ; and save to memory
(023)        drsz    byte_count              ; decrement counter
(024)        jp      $copy_loop              ; ..until done
(025)    $end_obj:                           ;
(026)        ld      b, #msg_time            ; point to time stamp
(027)        ld      a, system_time_high     ; get time high byte
(028)        x       a, [b+]                 ; and save
(029)        ld      a, system_time_low      ; get time low byte
(030)        x       a, [b]                  ; and save
(031)        sbit    obj_number, rx_status   ; indicate receive
(032)    $end_msg:                           ; end
(033)    .endm                               ;
```

Using CAN Networking for Cost Effective DC Motor Control in Vehicle Body Electronics

National Semiconductor
Application Note 1049
Martin Embacher

BACKGROUND

DC motor control is the most distributed application in automotive body systems. As the number of DC motors in cars increases, a multiplexed architecture becomes the only viable solution, reducing cost and weight, improving reliability and control efficiency, and significantly increasing passenger convenience. This application note demonstrates in a practical example how products from National Semiconductor help minimize the cost of these applications, using CAN as the state of the art, fault tolerant networking solution.

INTRODUCTION

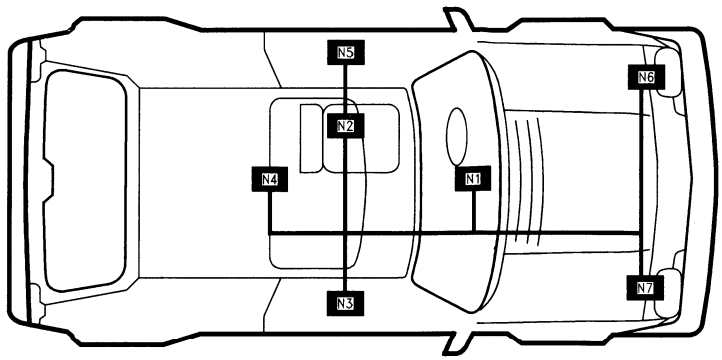
This application note decribes the issues associated with the implementation of DC motor control systems in automotive applications. It identifies the multiple locations of DC motors in automobiles. With reference to a basic DC motor control system structure it explains the approach National Semiconductor Corporation has taken to implement a system solution onto a single CAN microcontroller.

DC MOTORS IN AUTOMOTIVE APPLICATIONS

DC motors are used in automobiles to increase the convenience and safety of driving. These functions are distributed throughout the car in various applications. These include modules located in the door like window winders, mirror control or in the driver and passenger seat for seat positioning and functions such as trunk lock or sunroof control. All these systems have the characteristic that control is performed far away from the action. *Figure 1* shows various locations of DC-motors and their control systems in cars.

One example is a power seat where the driver wants to have the control buttons in the handrest and the motor is in the seat. Another example is the motors to control the mirror and window of the passenger door. Here again the driver needs to have the control buttons close to hand. Traditionally these systems require a large number of copper wires and a central control system to perform tasks like reading the switches and driving the motor to perform the desired action. The large number of wires used for control introduces a risk of malfunction. In a door module wires have to go through a small hole in the door and bend when the door opens and closes. If one wire breaks, the complete system may not operate any more. Also if the cabling to a motor in the power seat system breaks this could cause unwanted movement introducing a safety risk if this happens while the car is being driven.

The solution is to replace the wiring looms with a multiplexed wiring bus. This reduces the cost of wiring and improves system reliability, for example when a fault tolerant protocol, like CAN, is used for control. Moreover, multiplex wiring gives more "intelligence" to the motor helping to detect malfunctions directly. This can be done with the use of dedicated microcontrollers. This controller can directly take care of unwanted effects and can also perform a feasibility check of the command. Additionally a dedicated motor controller can react immediately when implementing safety critical functions. Take for example a window winder, which has to be switched off when something blocks the windows path, e.g. a child's head or the hand of an unconcentrating driver.



TL/DD/12852–1

Legend:

N1—Dashboard/steering wheel (control)
N2—Driver power seat
N3—Passenger door (mirror, window-winder)
N4—Sunroof
N5—Driver's door (mirror, window-winder, control)
N6—Headlight control left
N7—Headlight control right

FIGURE 1. DC-Motors in Automobiles

Also in industrial applications sensors and actuators are increasingly controlled by fault tolerant multiplex bus systems producing the need for integrated applications optimized controllers.

DC Motor Control System

A basic DC motor control system is shown in *Figure 2*. The regulator reads in the DEMAND value, sets the CONTROL value and compares the actual speed FEEDBACK value.

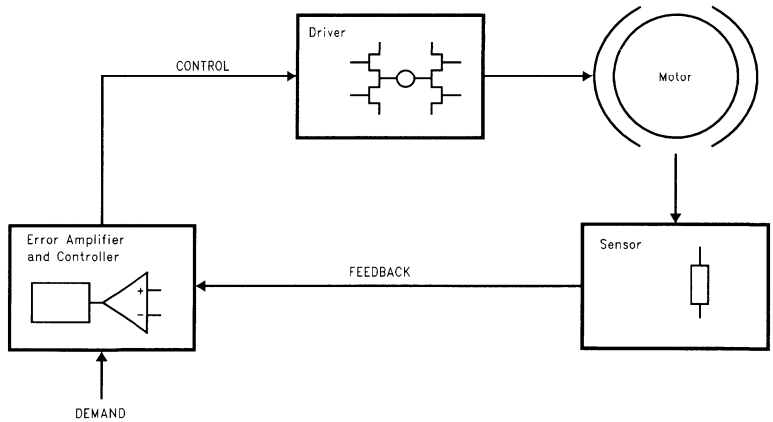
Currently there are two main implementations of a DC motor control system: an electromechanical solution where the motor is switched on and off by relays where only the speed of the motor is monitored, and secondly, a fully electrical solution with a discrete or integrated H-Bridge. In the latter the motor's speed can be adjusted much more precisely than in the former. The SENSOR block consists of either a slotted disk to scan the movement of the motor optoelectronically or of an converter measuring the current flowing through the motor.

INTEGRATING A DC MOTOR CONTROL SYSTEM USING NATIONAL'S COP884BC

National Semiconductor's COP884BC microcontroller addresses both implementations. The functional block found on this controller includes a high speed constant resolution PWM timer which can be used to control the H-Bridge with a minimum of external parts. However this PWM timer also has the option of monitoring the speed of the motor by high resolution frequency measurement if it runs in capture mode. Additionally two on-chip comparators allow sensing of the current through the motor. One of them has an input which can be switched to two pins. The integrated CAN interface addresses the need for highly reliable multiplexed data communication. *Figure 2* shows a block diagram of the implementation and the following paragraphs describe the functional blocks in more detail.

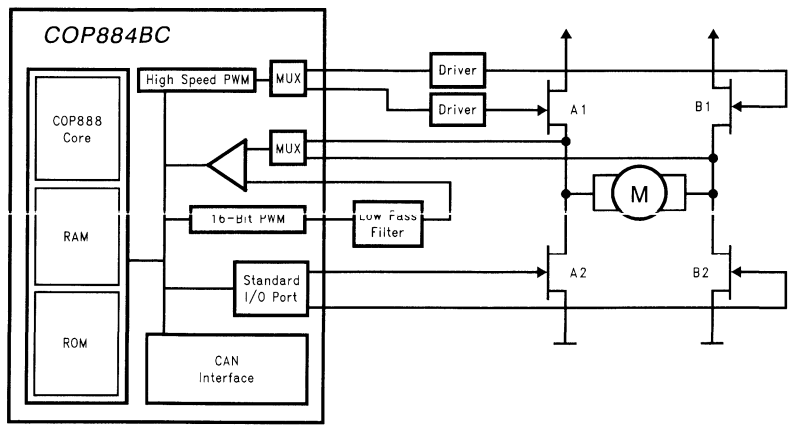
H-Bridge

The typical control means for DC motors is the H-bridge. It consists of four transistors A1, A2, B1 and B2 connected like an H with the motor in the middle. This scheme allows current to flow into the motor from either side. In *Figure 3* if



TL/DD/12852–2

FIGURE 2. Basic DC Motor Control System



TL/DD/12852–3

FIGURE 3. Motor Control with COP884BC Block Diagram

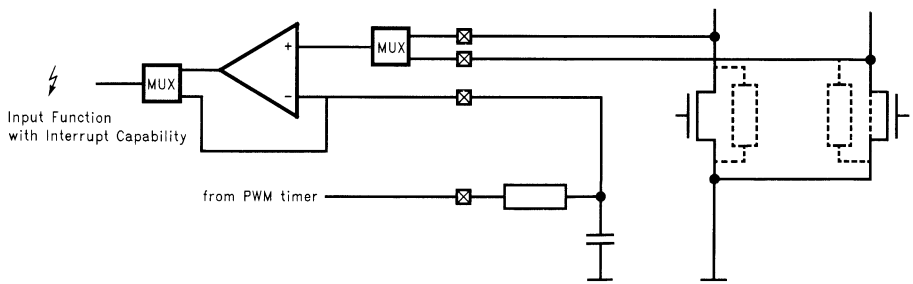
A1 and B2 conduct, current flows from left to right and vice versa if A2 and B1 are conducting. The other transistor pair is switched off. Care must be taken that both transistors of one side (A1 and B1 or A2 and B2) do not conduct at the same time, else a short circuit—called shoot-through—would result. The transistors are typically MOSFET type. To control the speed of the motor a PWM signal is applied to the gates A1 or B1. The voltage across the motor is dependent on the duty cycle of the PWM signal. If both transistors of one direction conduct, i.e. a duty cycle of 100% at the high side driver, the motor runs at full speed.

Two additional points have to be taken into consideration when operating a DC motor with a MOSFET H-bridge. Firstly, as the high side drivers typically are n-channel MOSFET devices, the voltage on the gate needs to be higher than the voltage on the drain. This can be done with a simple bootstrap circuit which charges up a capacitor during the PWM low cycle and generates a voltage of approximately Vbat*2 as soon as the MOSFET starts conducting. The higher the overall frequency of the PWM signal, the lower the required capacitor value. Secondly a motor consists of a coil and a magnetic core which resonates at the PWM frequency. The PWM signal permanently starts and stops the motor. If the PWM signal is below 20 kHz (audible range) someone next to the motor will hear the PWM induced resonance of the core. Therefore the PWM frequency should be chosen above the audible range.

National's COP884BC microcontroller offers a unique high speed constant resolution PWM timer which is capable of generating PWM frequencies up to 39 kHz completely processor independently and which covers the full PWM range from static high to static low with only one register. The PWM signal can be routed to two output pins of the microcontroller to directly connect to a bootstrap circuit for each of the two high side drivers saving external components. Both low side drivers can be directly connected to ports of the microcontroller.

Current Feedback

To measure the load of the motor and to be able to switch it off in the case that the rotor is jammed a comparator can be used with a current to voltage converter. Usually one wants to set a specific current limit which should not be exceeded to prevent motor failure. There are two solutions for voltage to current conversion: a resistor in the ground line of both low side drivers or to use the $r_{DS(ON)}$ resistance of each low side MOSFET. The latter has the advantage that no additional resistor is introduced to the system as an additional resistor would result in an additional voltage drop thus reducing the efficiency of the system. The COP884BC with its two comparators mapped to I/O ports supports both setups. Also the analog multiplexer on comparator 2 allows the measurement of two independent currents. A reference voltage, i.e. maximum voltage, is connected to the other terminal of the comparator. A simple way to generate this reference voltage is to use a PWM signal and a low pass RC-type filter. The PWM frequency is applied to the low pass filter. At the output of the filter a voltage directly proportional to the duty cycle will result. This circuit causes in the comparator output to toggle as soon as the voltage indicating the current through the motor exceeds the voltage set by the reference. *Figure 4* shows the block diagram of this comparator and the mapped I/O pins together with the PWM timer generating the reference value. The $r_{DS(ON)}$ resistances are also indicated in the figure.



TL/DD/12852–4

FIGURE 4. Comparator and PWM

Multiplex Interface

The highly reliable, state of the art CAN protocol was chosen. CAN operates in either single or dual wire configuration with minimal amount of external components (ISO low speed interface). The CAN interface implemented on the COP884BC addresses the need for a highly sophisticated—but low cost communication. The interface is fully compatible to the CAN specification 2.0 B (passive). To address the cost efficiency National reduced the number of registers down to four for each message resulting in fully automatic processing of two byte messages by the interface and the need for software interaction for longer messages. This scheme can be used as the part works as a motor controller and CAN is only used for set-up purposes. Also the acceptance filter is reduced to seven bits as fine filtering of the messages can be done by software as well. Typical bus speeds of a distributed controller are 125 kbit/s which gives 128 μs for two bit times which is sufficient for the controller to store away the data if the message is longer than two data bytes.

Glue Processor

The processor controlling the motor has to set the PWM frequency, read in the current and gets commands through the CAN interface. Additionally it may perform other tasks like doing diagnosis, reading in switches and setting status bits to the output world. For this functionality the 2 kbyte of ROM and 64 bytes of RAM are sufficient. The application optimized COP8 μC family provided by NSC offers a highly efficient code set and dedicated functional blocks for various applications. The feature family COP888 core supports a vectored interrupt scheme allowing the easy implementations of interrupt routines for the different on-chip peripherals. These include three interrupt sources for the CAN interface, one high prioritized external interrupt, one for the comparators and an additional multi-sourced external interrupt and a total of four for the various on-chip timers. To reduce the current consumption of the controller two power saving modes HALT and IDLE are implemented. The former completely freezes the processor and all on-chip peripheral blocks. However, the device will wake-up if a CAN message is received or by seven other input signals which are user programmable. IDLE mode freezes the core and all peripheral blocks except for the IDLE timer and the clock circuit alowing faster wake-up times or periodic wake-ups. All I/O ports of the chip have Schmitt trigger input and weak pull-up capability allowing the designer to reduce the external component count. Additionaly specific blocks to reduce electromagnetic emissions of the part have been implemented.

With the configurable controller methodology (CCM) and the newly installed European microcontroller design center, National Semiconductor addresses the need for system solutions and application optimized microcontrollers.

CONCLUSION

National's COP884BC, COPCAN-1, μC integrates most of the required blocks for highly sophisticated DC-motor control, allowing a designer to develop a cost effective network-controlled actuator with a few external components in a space saving 28-pin SO package. It integrates the PWM timer perfectly suited for H-bridge control and thereby removes an expensive integrated pure analog controller. Additionally COPCAN-1 shows that an application optimized CAN interface does not necesarily have to have the high cost associated with various CAN implementations.

Understanding and Eliminating EMI in Microcontroller Applications

National Semiconductor
Application Note 1050
Robin Getz
Bob Moeckel

1.0 ABSTRACT

In today's world, with increasing numbers of both fixed and mobile electronic devices, electromagnetic compatibility (EMC) is becoming a critical issue. Disastrous, if not annoying, results occur if a system, subsystem or component interferes with another through electromagnetic means. The EMC problem is first explained, and then the basic theory is presented with an explanation on how to use it to control an EMC problem during the initial design phase. The methods that some silicon manufacturers are using to control EMC and how these changes affect our systems will be examined. A total system (automobile radio) will be examined in detail to prove the effectiveness of a EMC reduced COP8™ (8-bit microcontroller). Overall design guidelines will also be given so that the designer can minimize EMI *before it becomes a problem.*

2.0 INTRODUCTION AND BACKGROUND

Electromagnetic Interference with Powered Wheelchairs

August 26, 1994

FDA is receiving inquiries about reports that electromagnetic interference (EMI) can cause some power-driven wheelchairs and scooters to move unexpectedly. The agency has investigated this matter and determined that EMI can cause unexpected movement in some power-driven wheelchairs when they are turned on. Not all brands and models of power wheelchairs and scooters have this problem. Some have greater immunity levels than others that may protect against EMI. Common sources of EMI include cellular phones, CB radios, TV and radio stations, amateur (HAM) radios and police, fire and ambulance radios.

After receiving reports of problems, FDA tested sample wheelchairs and scooters in its laboratories and obtained information from manufacturers and users about possible EMI-related incidents. As a result of its review, the agency last May asked wheelchair manufacturers to take steps to protect powered wheelchair users from the potential hazards of EMI. FDA required all firms to clearly label wheelchairs and scooters with the immunity level, or else state that the immunity level is not known. It also required them to label wheelchairs and scooters to warn users of the potential hazards of EMI. The agency also asked manufacturers to:

1. *Establish a minimum recommended immunity level of 20V per meter for all new motorized wheelchairs and scooters. This would provide a reasonable degree of protection against the more common sources of EMI.*
2. *Undertake an educational campaign to inform users and those who care for them about the risks of EMI and ways to avoid it; and*
3. *Solicit reports of possible EMI incidents and continue to monitor the full scope of the problem.*

FDA first learned of a possible problem with motorized wheelchairs from a complaint in June 1992. As a result, the agency began testing sample wheelchairs in its laboratories. Those tests, on several brands of power-driven wheelchairs and scooters, revealed that most—but not all—of those sampled were susceptible to interference at various radio frequencies. In some tests, the brakes released. In others, the wheels moved uncontrollably. Sometimes both occurred at once. The extent to which this happens in actual use is not known. In early 1993, FDA began inspecting facilities of all domestic and foreign motorized wheelchair and scooter manufacturers to gather information on the problem. The agency reviewed warranty and complaint files and looked at any manufacturing practices that could contribute to this type of problem. The inspections revealed numerous complaints about erratic, unintentional wheelchair movement, sometimes resulting in injuries. It is unclear, however, exactly what caused the chairs to move in those cases. In addition to EMI, unexpected movement can be caused by failure of an electronic component or by user error.

In May 1993, FDA sent letters to all U.S. manufacturers asking them to provide any information they had about possible problems with radio wave interference, including complaints, reports of injuries and studies on power-driven wheelchairs. In July 1993, the agency alerted consumer groups that represent wheelchair users about the possible problem and requested similar information. An estimated 10,000 motorized wheelchairs and 50,000 motorized scooters are sold annually in this country.

—FDA Report, August 1994

The control and minimization of EMC is a technology that is, out of necessity, growing rapidly. Manufacturers and regulatory agencies have made efforts to control this problem but the issue of meeting these ever tightening specifications is left to the system designer. EMC can no longer be a fix, designed in at the last moment, but must be a directed effort between the circuit designer, layout engineer, software engineer and purchaser. This paper will examine what can be done to reduce the chance of having a product fail EMC approval.

2.1 The EMC World Map

Before jumping into the midst of the EMC design methods used, it is best to gain an understanding of the problem situation and review or learn applicable electromagnetic (EM) theory. To begin, look at what I call the map of the EMC world, Table I. The EMC problem is between an emission culprit and susceptible victim which are coupled through a means, either radiation through space or conducted through wires *(Figure 1)*. We can map out the EMC problem cases with columns headed by radiation and conduction, and rows headed by emission and susceptibility, to get four cases abbreviated as RE, CE, RS, CS.

TABLE I. ElectroMagnetic Compatibility Map

	Radiation	Conduction
Emission	RE	CE
Susceptibility	RS	CS

2.2 What Designers Do Upon A Discovery of an EMC Problem

Typically, the first time system designers realize that they have a RE problem is after the prototype system has been built and evaluated. Instead of being close to market release they now find themselves applying costly and often ineffective patches and going back into redesign. The steps outlined in Table II are what a typical person does at this point. First reaction is to make the printed circuit board (PCB) trace length short or to upgrade to a higher cost PCB with a full ground plane. Although most system companies have experienced and good layout design rules for their PCBs, enforcing these rules is sometimes difficult to do as more and more components are being packed into smaller packages on a smaller board. Layout is often subcontracted and the designer may not have the necessary experience and motivation, cost reduction is the goal. Next, filters (RC, RL, or ferrite beads) are added in hopes that this may fix the problem. Also, the crystal frequency may be reduced by about 2x to run the system clock at a slower rate. Software may have to be rewritten to compensate for this new time base. Software can be redone to toggle the outputs less frequently. Shielding is added or the PCB is put into an electrically out of the way place (like hiding a monster in the basement). Having exhausted their design tricks, the designers now search other semiconductor suppliers to find a quieter part. Finally, when all the time and money runs out with EMC qualification failing, the decision is made not to market the product and everyone loses business.

It is from this examination that we can understand that EMC design issues have to be considered properly at the beginning of the design cycle, in order to reduce complex design changes which must be done at the completion of the project if EMC testing is failing.

TABLE II. List of Designer Efforts to Reduce EMI Emissions

Method	Complexity	Cost	Time
1. Revise PCB Layout	Medium	Low	High
2. Add Components (Filters)	Medium	Medium	Medium
3. Change Crystal Frequency	High	Low	High
4. Rewrite Software	High	Low	High
5. Add Shielding	Low	High	Low
6. Relocate PCB	Low	Low	Low
7. Change IC Supplier	High	Medium	Medium
8. Fail EMC Qualification—Don't Sell Product	High	High	High

3.0 DESCRIPTION OF NOISE

3.1 ElectroMagnetic Interference

EMI is a form of electrical-noise pollution. Consider the time when an electric drill or some other power tool jammed a nearby radio with buzzing or crackling noise. At times it got so bad that it prevented you from listening to the radio while the tool was in use. Or the ignition of an automobile idling outside your house caused interference to your TV picture making lines across the screen or even losing sync altogether making the picture flip. These examples are annoying but are not catastrophic.

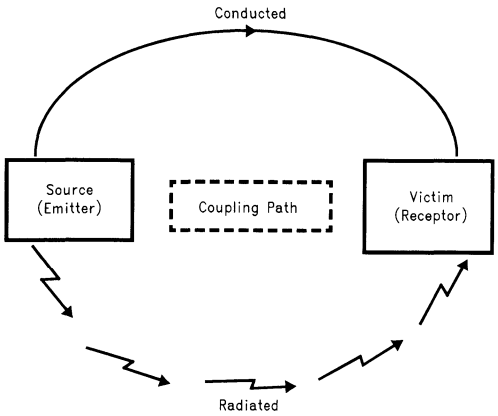
More serious, how about a sudden loss in telephone communication caused by electrical interference or noise while you are negotiating an important business deal? Now EMI can be economically damaging.

The results of EMI incidents can be even further reaching than these examples. Aircraft navigation errors resulting from EMI or interruption of air traffic controller service and may be even computer memory loss due to noise, could cause two aircraft to collide resulting in the loss of lives and property.

These were just a few instances to help identify the results of EMI in a familiar context. To help understand an EMI situation, the problem can be divided into three areas. They are the source, the victim, and the coupling path. Secondary categories involve the coupling path itself. If the source and victim are separated by space with no hard wire connection, then the coupling path may be a radiated path and we are dealing with radiated noise. If the source and victim are connected together through wires, cables, or connectors, then the coupling path may be a conducted path and we are dealing with conducted noise. It is possible for both types of noise to exist at the same time.

3.2 ElectroMagnetic Compatibility

EMI or electrical noise is of world wide concern. Governmental agencies and certain industry bodies have issued specifications with which all electrical, electromechanical, and electronic equipment must comply. These specifications and limitations are an attempt to ensure that proper EMC techniques are followed by manufacturers during the design and fabrication of their products. When these techniques are properly applied, the product can then operate and perform with other equipment in a common environment so that no degradation of performance exists due to internally or externally conducted or radiated electromagnetic emissions. This is defined as ElectroMagnetic Compatibility or EMC.



TL/DD/12857–1

FIGURE 1. EMI Situation

3.3 Inter-System EMI

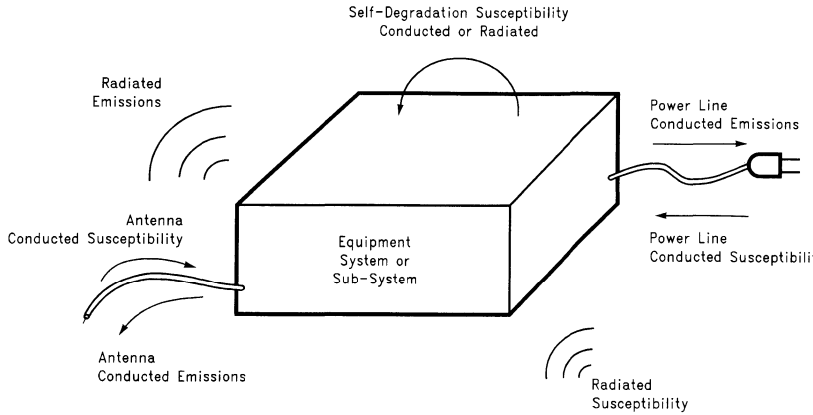
For the purpose of this paper, when the source of noise is a module, board, or system and the victim is a different and separate module, board, or system under the design or control of a different person, it will be considered an inter-system interference situation. Examples of inter-system interference situations could be a Personal Computer interfering with the operation of a TV or an anti-lock brake module in a car causing interference in the radio. This type of interference is more difficult to contain because, as mentioned earlier, the systems are generally not designed by a single person. However, design methods and control techniques used to contain the intra-system form of EMI, which are almost always under the control of a single user, will inherently help reduce the inter-system noise.

This paper will address problems and solutions in the area of intra-system noise. Intra-system interference situations occur when the sources, victims, and coupling paths are entirely within one system, module or PCB. Systems may provide emissions that are conducted out of power lines or be susceptible to emissions conducted through them. Systems may radiate emissions through space in addition to being susceptible to radiated noise. Noise conducted out antenna leads turns into radiated noise. By the same token, radiated noise picked up by the antenna is turned into conducted noise within the system. A perfect example is a PCB ground loop, which makes an excellent antenna. The system itself is capable of degrading its own performance due to its own internal generation of conducted and radiated noise and its susceptibility to it.

Some results of EMI within a system: Noise on power lines causing false triggering of logic circuits, rapidly changing signals causing "glitches" on adjacent steady state signal lines (crosstalk) causing erratic operation, multiple simultaneously switching logic outputs propagating ground bounce noise throughout system, etc.

3.4 Coupling Paths

The modes of coupling an emitter source to a receptor victim can become very complicated. Remember, each EMI situation can be classified into two categories of coupling, conducted and radiated. Coupling can also result from a combination of paths. Noise can be conducted from an emitter to a point of radiation at the source antenna, then picked up at the receptor antenna by induction, and re-conducted to the victim. A further complication that multiple coupling paths presents is that it makes it difficult to determine if eliminating a suspected path has actually done any good. If two or more paths contribute equally to the problem, eliminating only one path may provide little apparent improvement.



TL/DD/12857–2

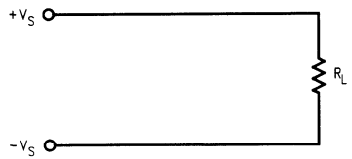
FIGURE 2. Intra-System EMI Manifestations

3.5 Conducted Interference

In order to discuss the various ways in which EMI can couple from one system to another, it is necessary to define a few terms. There are two varieties of conducted interference which concern us. The first variety is differential-mode interference. That is an interference signal that appears between the input terminals of a circuit. The other variety of conducted interference is called common-mode interference. A common-mode interference signal appears between each input terminal and a third point, the common-mode reference. That reference may be the equipment chassis, an earth ground, or some other point.

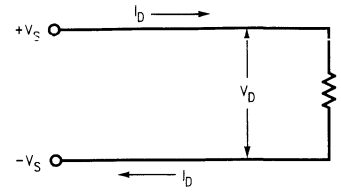
Let's look at each type of interference individually. In *Figure 3* we show a simple circuit consisting of a signal source, V_S, and a load, R_L. In *Figure 4* we show what happens when differential-mode interference is introduced into the circuit by an outside source. As is shown, an interference voltage, V_D, appears between the two input terminals, and an interference current, I_D, flows in the circuit. The result is noise at the load. If, for instance, the load is a logic gate in a computer, and the amplitude of V_D is sufficiently high, it is possible for the gate to incorrectly change states.

Figure 5 shows what happens when a ground loop is added to our circuit. Ground loops, which are undesirable current paths through a grounded body (such as a chassis), are usually caused by poor design or by the failure of some component. In the presence of an interference source, common-mode currents, I_C, and a common-mode voltage, V_C, can develop, with the ground loop acting as the common-mode reference. The common-mode current flows on both input lines, and has the same instantaneous polarity and direction (the current and voltage are in phase), and returns through the common-mode reference. The common-mode voltage between each input and the common-mode reference is identical.



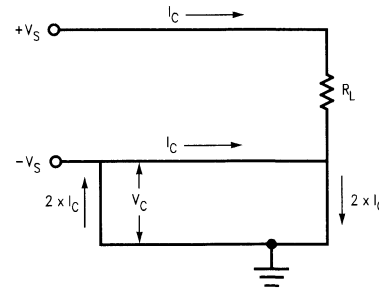
TL/DD/12857–3

FIGURE 3. Simple Circuit



TL/DD/12857–4

FIGURE 4. Differential-Mode Interference



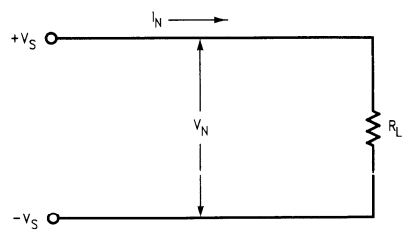
TL/DD/12857–5

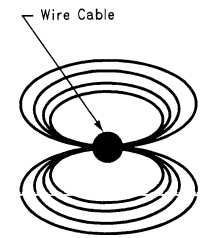
FIGURE 5. Common-Mode Interference

3.6 Radiated Interference

Radiated coupling itself can take place in one of several ways. Some of those include field-to-cable coupling, cable-to-cable coupling, and common-mode impedance coupling. Let's look at those types of coupling one at a time.

The principle behind field-to-cable coupling is the same as that behind the receiving antenna. That is, when a conductor is placed in a time-varying electromagnetic field, a current is induced in that conductor. That is shown in *Figure 6*. In this figure, we see a signal source, V_S, driving a load, R_L. Nearby there is a current carrying wire (or other conductor). Surrounding the wire is an electromagnetic field induced by the current flowing in the wire. The circuit acts like a loop antenna in the presence of this field. As such, an interference current, I_N, and an interference voltage, V_N, are induced in the circuit. The magnitude of the induced interference signal is roughly proportional to the magnitude of the incoming field, the frequency of the incoming field, the size of the loop, and the impedance of the loop.





TL/DD/12857–6

FIGURE 6. Field-to-Cable Coupling

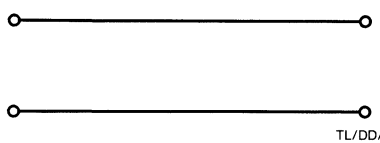
Cable-to-cable coupling occurs when two wires or cables are run close to each other. *Figure 7* shows how cable-to-cable coupling works. *Figure 7a* shows two lengths of cable (or other conductors) that are running side-by-side. Because any two conducting bodies have capacitance between them, called stray capacitance, a time-varying signal in one wire can couple via that capacitance into the other wire. That is referred to as capacitive coupling. This stray capacitance, as shown in *Figure 7c* makes the two cables behave as if there were a coupling capacitor between them.

Another mechanism of cable-to-cable coupling is mutual inductance. Any wire carrying a time-varying current will develop a magnetic field around it. If a second conductor is placed near enough to that wire, that magnetic field will induce a similar current in the second conductor. That type of coupling is called inductive coupling. Mutual inductance, as shown in *Figure 7b,* makes the cables behave as if a poorly wound transformer were connected between them.

In cable-to-cable coupling, either or both of those mechanisms may be responsible for the existence of an interference condition. Though there is no physical connection between the two cables, the properties we have just described make it possible for the signal on one cable to be coupled to the other. These effects, when combined are known as the transmission line effect and have been studied and described thoroughly in other texts.

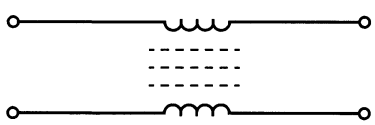
The basic transmission line equations are derived for distributed parameters R (series resistance), G (shunt conductance), L (series inductance), and C (shunt capacitance), all defined per unit length of line.

Either or both of the above-mentioned properties cause the cables to be electromagnetically coupled such that a time-varying signal present on one will cause a portion of that signal to appear on the other. The "efficiency" of the coupling increases with frequency and inversely with the distance between the two cables. One example of cable-to-cable coupling is telephone "crosstalk", in which several phone conversations can be overheard at once. The term crosstalk is now commonly used to describe all types of cable-to-cable coupling.



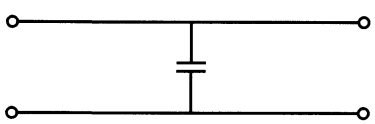
TL/DD/12857–7

FIGURE 7a. Parallel Conductors



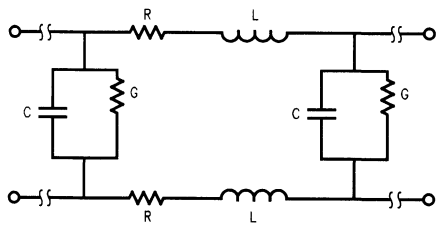
TL/DD/12857–9

FIGURE 7b. Inductive Coupling



TL/DD/12857–8

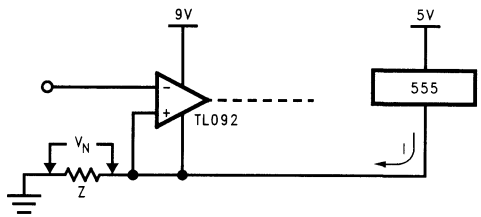
FIGURE 7c. Capacitive Coupling



TL/DD/12857–10

FIGURE 7d. Transmission Line Model

Common-mode impedance coupling occurs when two circuits share a common bus or wire. In *Figure 8* we show a circuit that is susceptible to that type of coupling. In that figure a TL092 op-amp and a 555 timer share a common return or ground. Since any conductor (including a PCB trace) is not ideal, that ground will have a non-zero impedance, Z. Because of that, the current, I, from pin 1 of the 555 will cause a noise voltage, V_N, to develop; that voltage is equal to I x Z. That noise voltage will appear in series with the input to the op-amp. If that voltage is of sufficient amplitude, a noise condition will result.



TL/DD/12857–11

FIGURE 8. Common-Mode Impedance Coupling

While not all inclusive, these coupling paths account for, perhaps, 98% of all intra-system EMI situations.

4.0 SOURCE OF NOISE

We will look at sources of EMI which involve components that may conduct or radiate electromagnetic energy. These sources, (component emitters), are different from the equipment and subsystems we have been talking about. Component emitters are sources of EMI which emanate from a single element rather than a combination of components such as was previously described. Actually, these component emitters require energy and connecting wires from other sources to function. Therefore, they are not true sources of EMI, but are EMI transducers. They convert electrical energy to electromagnetic noise.

4.1 Cables and Connectors

The three main concerns regarding the EMI role of cables are conceptualized in *Figure 9.* They act as (1) radiated emission antennae, (2) radiated susceptibility antennae, and (3) cable-to-cable or crosstalk couplers. Usually, whatever is done to harden a cable against radiated emission will also work in reverse for controlling EMI radiated susceptibility. The reason for this is usually, when differential-mode radiated emission or susceptibility is the failure mode, twisting leads and shielding cables reduces EMI. If the failure mechanism is due to common-mode currents circulating in the cable, twisting leads has essentially no effect on the rela-

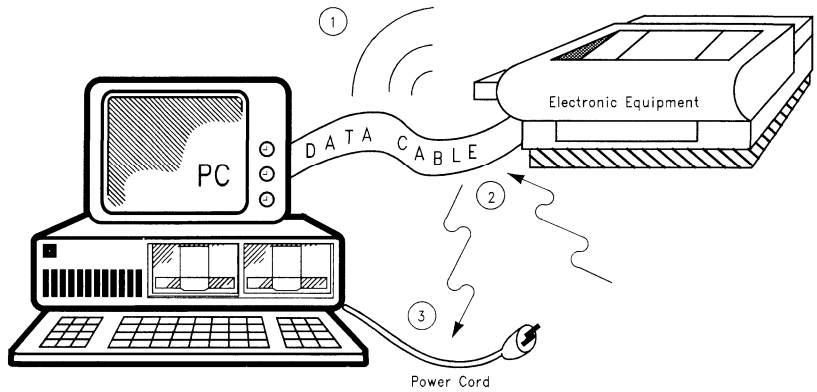
tionship between each conductor and the common-mode reference. Cable shields may help or aggravate EMI depending upon the value of the transfer impedance of the cable shield. Transfer impedance is a figure of merit of the quality of cable shield performance defined as the ratio of coupled voltage to surface current in Ω/meter. A good cable shield will have a low transfer impedance. The effectiveness of the shield also depends on whether or not the shield is terminated and, if so, how it is terminated.

Connectors usually are needed to terminate cables. When no cable shields or connector filters or absorbers are used, connectors play essentially no role in controlling EMI. The influence of connector types, however, can play a major role in the control of EMI above a few MHz. This applies especially when connectors must terminate a cable shield and/or contain lossy ferrites or filter-pins.

Connectors and cables should be viewed as a system to cost-effectively control EMI rather than to consider the role of each separately, even though each offers specific interference control opportunities.

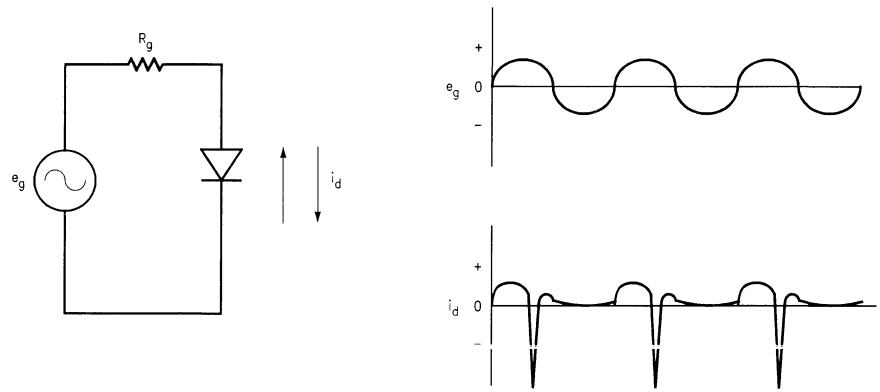
4.2 Components

Under conditions of forward bias, a semiconductor stores a certain amount of charge in the depletion region. If the diode is then reverse-biased, it conducts heavily in the reverse direction until all of the stored charge has been removed as shown in *Figure 10*. The duration, amplitude, and configuration of the recovery-time pulse (also called switching time or period) is a function of the diode characteristics and circuit parameters. These current spikes generate a broad spectrum of conducted transient emissions. Diodes with mechanical imperfections may generate noise when physically agitated. Such diodes may not cause trouble if used in a vibration-free environment.



TL/DD/12857-12

FIGURE 9. Cables and Connectors



TL/DD/12857-13

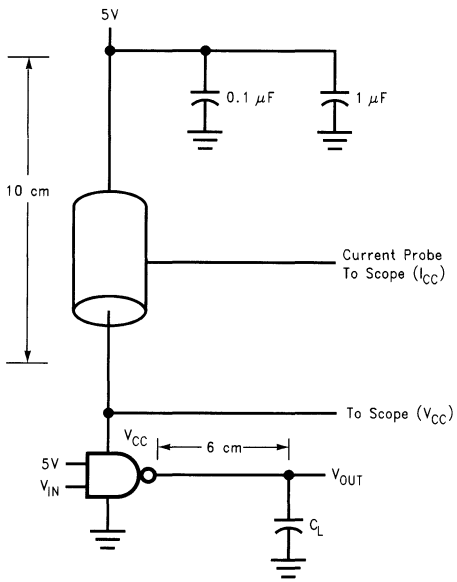
FIGURE 10. Diode Recovery Periods and Spikes

4.3 Power Supply Noise

Power-supply spiking is perhaps the most important contributor to system noise. When any element switches logic states, it generates a current spike that produces a voltage transient. If these transients become too large, they can cause logic errors because the supply voltage drop upsets internal logic, or because a supply spike on one circuit's output feeds an extraneous noise voltage into the next device's input.

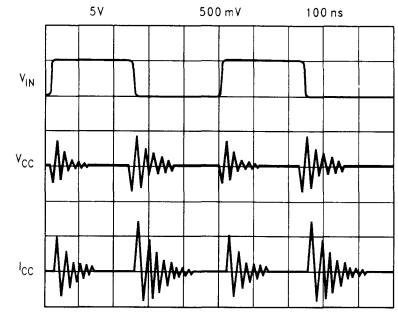
With CMOS logic in its quiescent state, essentially no current flows between V_{CC} and ground. But when an internal gate or an output buffer switches state, a momentary current flows from V_{CC} to ground. The switching transient caused by an unloaded output changing state typically equals 20 mA peak. Using the circuit shown in *Figure 11,* you can measure and display these switching transients under different load conditions.

Figure 12a shows the current and voltage spikes resulting from switching a single unloaded ($C_L = 0$ in *Figure 11*) NAND gate. These current spikes, seen at the switching edges of the signal on V_{IN}, increase when the output is loaded. *Figures 12b, 12c,* and *12d* show the switching transients when the load capacitance, C_L, is 15 pF, 50 pF, and 100 pF, respectively. The large amount of ringing results from the test circuit's transmission line effects. This ringing occurs partly because the CMOS gate switches from a very high impedance to a very low one and back again. Even for medium-size loads, load capacitance current becomes a major current contributor.



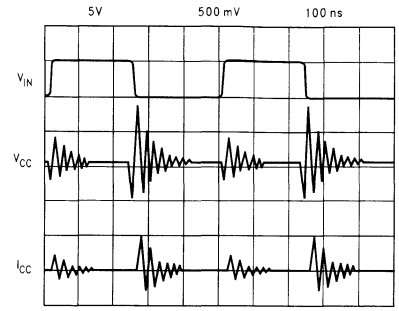
TL/DD/12857–14

FIGURE 11. Test Circuit



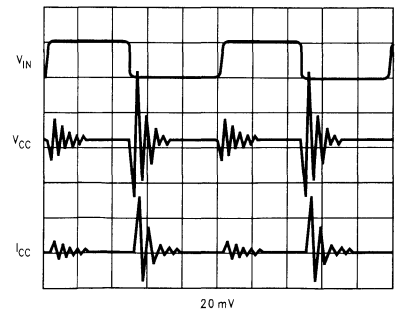
TL/DD/12857–15

FIGURE 12a. V_{IN}, V_{CC}, I_{CC} for C_L = 0 pF



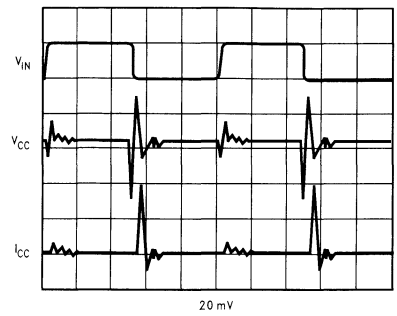
TL/DD/12857–16

FIGURE 12b. V_{IN}, V_{CC}, I_{CC} for C_L = 15 pF



TL/DD/12857–17

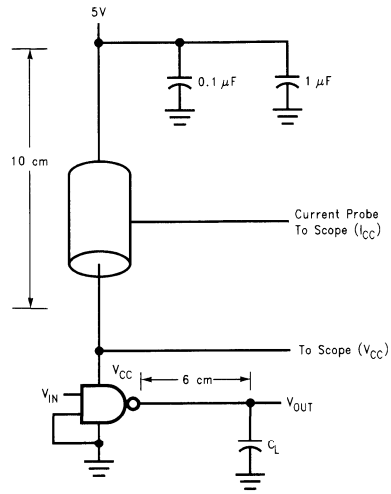
FIGURE 12c. V_{IN}, V_{CC}, I_{CC} for C_L = 50 pF



TL/DD/12857–18

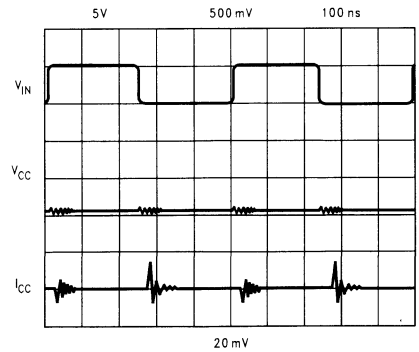
FIGURE 12d. V_{IN}, V_{CC}, I_{CC} for C_L = 100 pF

Although on standard CMOS logic internal gates generate current spikes when switching, the bulk of a spike's current comes from output circuit transitions. *Figure 13* shows the I_{CC} current for a NAND gate, as shown in the test circuit, with one input switching and the other at ground resulting in no output transitions. Note the very small power-supply glitches provoked by the input-circuit transitions.



TL/DD/12857–19

FIGURE 13a. Test Circuit



TL/DD/12857–20

FIGURE 13b. V_{IN}, V_{CC}, I_{CC}

4.4 High Speed CMOS Logic Switching

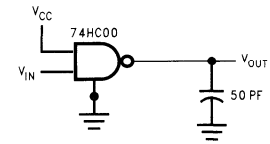
The magnitude of noise which can be tolerated in a system relates directly to the worst case noise immunity specified for the logic family. Noise immunity can be described as a device's ability to prevent noise on its input from being transferred to its output. It is the difference between the worst case output levels (V_{OH} and V_{OL}) of the driving circuit and the worst case input voltage requirements (V_{IH} and V_{IL}, respectively) of the receiving circuit.

Using Table III as a guide, it can be seen that for TTL (LS or ALS) devices the worst case noise immunity is typically 700 mV for the high logic level and 300 mV for the low logic level. For HCMOS devices the worst case noise immunity is typically 1.75V for high logic levels and 800 mV for low logic levels. AC high speed CMOS logic families have noise immunity of 1.75V for high logic levels and 1.25V for low logic levels. ACT CMOS logic families have noise immunity of 2.9V for high logic levels and 700 mV for low logic levels.

TABLE III. Logic Family Comparisons

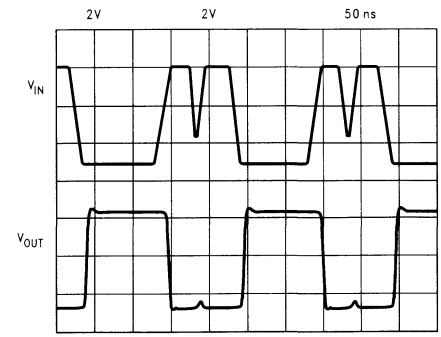
Characteristic	Symbol	LS/ALS/TTL	HCMOS	AC	ACT
Input Voltage (Limits)	V_{IH} (Min)	2.0V	3.15V	3.15V	2.0V
	V_{IL} (Max)	0.8V	0.9V	1.35V	0.8V
Output Voltage (Limits)	V_{OH} (Min)	2.7V	$V_{CC}-0.1V$	$V_{CC}-0.1V$	$V_{CC}-0.1V$
	V_{OL} (Max)	0.5V	0.1V	0.1V	0.1V

To illustrate noise margin and immunity, *Figures 14b* and *14c* show the output that results when you apply several types of simulated noise to a 74HC00's input, *Figure 14a*. Typically, even 2V or more input noise produces little change in the output. The top trace shows noise present on the input logic level signal and the bottom trace shows resultant noise induced on the output logic level signal.



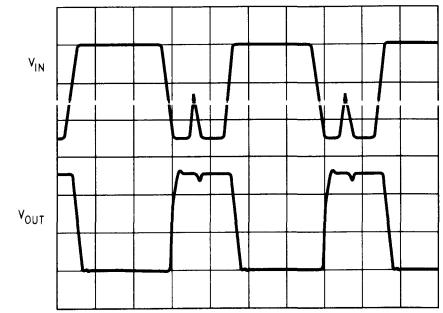
TL/DD/12857–23

FIGURE 14a. Test Circuit



TL/DD/12857–24

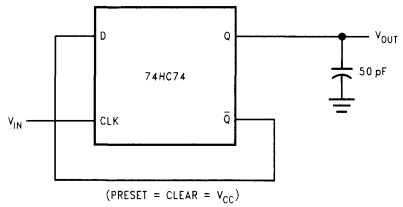
FIGURE 14b. V_{IN}, V_{OUT}



TL/DD/12857–25

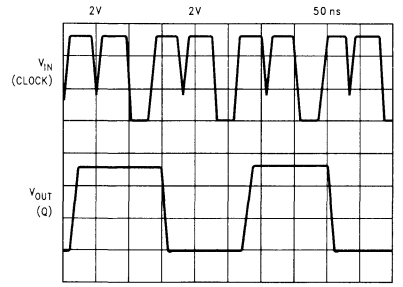
FIGURE 14c. V_{IN}, V_{OUT}

Figure 15b shows how noise affects 74HC74's clock input. Again, no logic errors occur with 2V or more of noise on the clock input.



TL/DD/12857-21

FIGURE 15a. Test Circuit



TL/DD/12857-22

FIGURE 15b. V_{IN}, V_{OUT}

When using high speed CMOS, even with its greater noise immunity, crosstalk, induced supply noise and noise transients become factors. Higher speeds allow the device to respond more quickly to externally induced noise transients and accentuate the parasitic interconnection inductances and capacitances that increase self-induced noise and crosstalk.

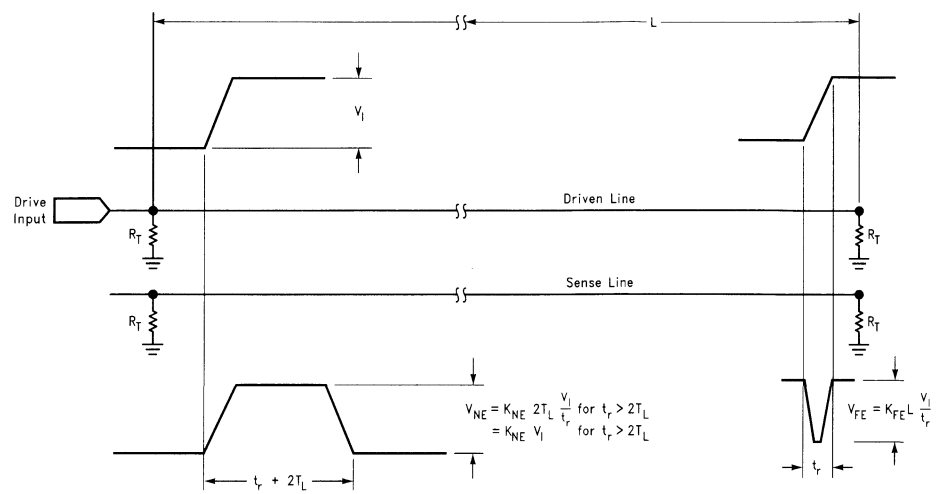
4.5 Signal Crosstalk

The problem of crosstalk, and how to deal with it, is becoming more important as system performance and board densities increase. Our discussion on cable-to-cable coupling described crosstalk as appearing due to the distributed capacitive coupling and the distributed inductive coupling between two signal lines. When crosstalk is measured on an undriven sense line next to a driven line (both terminated at their characteristic impedances), the near end crosstalk and the far end crosstalk have quite distinct features, as shown in *Figure 16*. It should be noted that the near end component reduces to zero at the far end and vice versa. At any point in between, the crosstalk is a fractional sum of the near and far end crosstalk waveforms as shown in the figure. It also can be noted that the far end crosstalk can have either polarity whereas the near end crosstalk always has the same polarity as the signal causing it.

The amplitude of the noise generated on the undriven sense line is directly related to the edge rates of the signal on the driven line. The amplitude is also directly related to the proximity of the two lines. This is factored into the coupling constants K_{NE} and K_{FE} by terms that include the distributed capacitance per unit length, the distributed inductance per unit length, and the length of the line. The lead-to-lead capacitance and mutual inductance thus created causes "noise" voltages to appear when adjacent signal paths switch.

Several useful observations that apply to a general case can then be made:

- The crosstalk always scales with the signal amplitude V_I.
- Absolute crosstalk amplitude is proportional to slew rate V_I/t_r not just $1/t_r$.
- Far end crosstalk width is always t_r.
- For $t_r << 2\ T_L$, where t_r is the transition time of the signal on the driven line and T_L is the propagation or bus delay down the line, the near end crosstalk amplitude, V_{NE}, expressed as a fraction of signal amplitude, V_I, is K_{NE} which is a function of physical layout only.
- The high the value of "t_r" (slower transition times) the lower the percentage of crosstalk (relative to signal amplitude).



TL/DD/12857-26

FIGURE 16. Crosstalk

Although all circuit conductors have transmission line properties, these characteristics become significant when the edge rates of the drivers are equal to or less than about three times the propagation delay of the line. Significant transmission line properties may be exhibited, for example, where devices having edge rates of 3 ns (or less) are used to drive traces of 8 inches or greater, assuming propagation delays of 1.7 ns/ft for an unloaded printed circuit trace.

4.6 Signal Interconnects

Of the many properties of transmission lines, two are of major interest to the system designer: Z_{OE}, the effective equivalent impedance of the line, and t_{pde}, the effective propagation delay down the line. It should be noted that the intrinsic values of line impedance and propagation delay, Z_O and t_{pd}, are geometry-dependent. Once the intrinsic values are known, the effects of gate loading can be calculated. The loaded values for Z_{OE} and t_{pde} can be calculated with:

$$Z_{OE} = \frac{Z_o}{\sqrt{1 + C_t/C_l}}$$

$$t_{pde} = t_{pd}\sqrt{1 + C_t/C_l}$$

where C_i = intrinsic line capacitance

C_t = additional capacitance due to gate loading.

These formulas indicate that the loading of lines decreases the effective impedance of the line and increases the propagation delay. As was mentioned earlier, lines that have a propagation delay greater than one third the rise time of the signal driver should be evaluated for transmission line effects. When performing transmission line analysis on a bus, only the longest, most heavily loaded and the shortest, least loaded lines need to be analyzed. All lines in a bus should be terminated equally; if one line requires termination, all lines in the bus should be terminated. This will ensure similar signal quality on all of the lines.

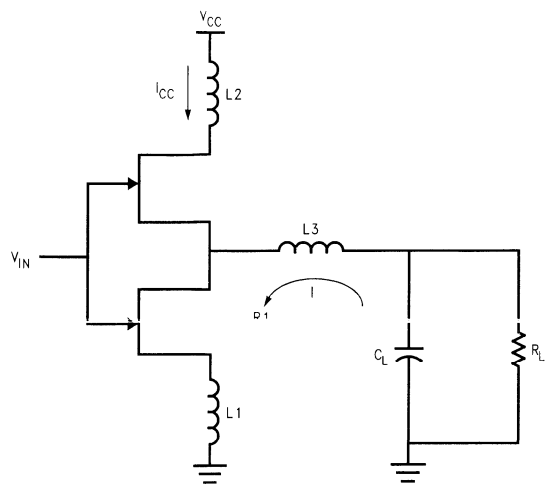
4.7 Ground Bounce

Ground bounce occurs as a result of the intrinsic characteristics of the leadframes and bond wires of the packages used to house CMOS devices. As edge rates and drive capability increase in advanced logic families, the effects of these intrinsic electrical characteristics become more pronounced. One of these parasitic electrical characteristics is the inductance found in all leadframe materials.

Figure 17 shows a simple circuit model for a CMOS device in a leadframe driving a standard test load. The inductor L1 represents the parasitic inductance in the ground lead of the package; inductor L2 represents the parasitic inductance in the power lead of the package; inductor L3 represents the parasitic inductance in the output lead of the package; the resistor R1 represents the output impedance of the device output, and the capacitor and resistor C_L and R_L represent the standard test load on the output of the device.

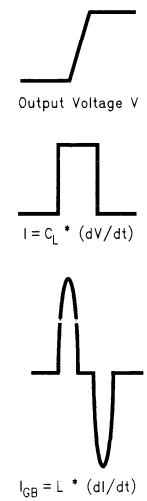
The three waveforms shown represent how ground bounce is generated. The top waveform shows the voltage (V) across the load as it is switched from a logic HIGH to a logic LOW. The output slew rate is dependent upon the characteristics of the output transistor, and the inductors L1 and L3, and C_L, the load capacitance. In order to change the output from a HIGH to a LOW, current must flow to discharge the load capacitance. The second waveform shows the current that is generated as the capacitor discharges [$I = -C_L{*}(dV/dt)$]. This current, as it changes, causes a voltage to be generated across the inductances in the circuit. The formula for the voltage across an inductor is $V = L(dI/dt)$. The third waveform shows the voltage that is induced across the inductance in the ground lead due to the changing currents [$V_{GB} = L1{*}(dI/dt)$]. This induced voltage creates what is known as ground bounce.

Because the inductor is between the external system ground and the internal device ground, the induced voltage causes the internal ground to be at a different potential than the external ground. This shift in potential causes the device inputs and outputs to behave differently than expected because they are referenced to the internal device ground, while the devices which are either driving into the inputs or being driven by the outputs are referenced to the their own device ground. External to the device, ground bounce causes input thresholds to shift and output levels to change.



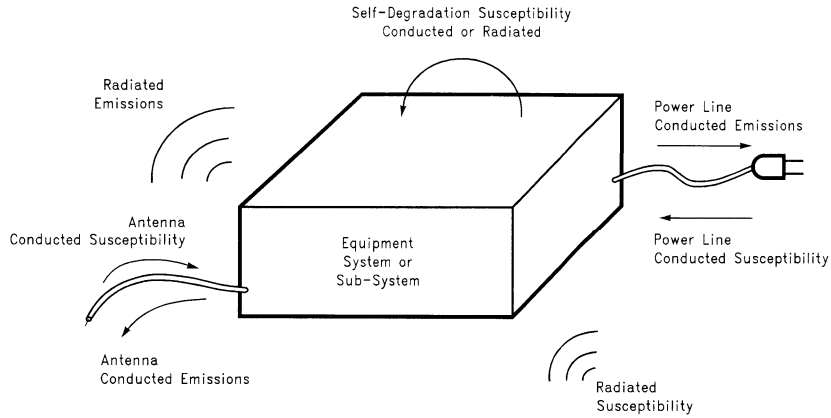
TL/DD/12857-27

FIGURE 17a. Test Circuit for Ground Bounce



TL/DD/12857-28

FIGURE 17b. V_{IN}, V_{CC}, I_{CC}



TL/DD/12857–29

FIGURE 18. Intra-System EMI Manifestations

Although this discussion is limited to ground bounce generated during HIGH-to-LOW transitions, it should be noted that the ground bounce is also generated during LOW-to-HIGH transitions. This ground bounce though, has a much smaller amplitude and therefore does not present the same concern.

There are many factors which affect the amplitude of the ground bounce. Included are:

- Number of outputs switching simultaneously: more outputs results in more ground bounce.
- Type of output load: capacitive loads generate two to three times more ground bounce than typical system traces. Increasing the capacitive load to approximately 60 pF–70 pF, increases ground bounce. Beyond 70 pF, ground bounce drops off due to the filtering effect of the load itself. Moving the load away from the output also reduces the ground bounce.
- Location of the output pin: outputs closer to the ground pin exhibit less ground bounce than those further away due to effectively lower L1 and L3.
- Voltage: lowering V_{CC} reduces ground bounce.

Ground bounce produces several symptoms:

- Altered device states.
- Propagation delay degradation.
- Undershoot on active outputs. The worst-case undershoot will be approximately equal to the worst-case quiet output noise.

5.0 NOISE SUPPRESSION TECHNIQUES

EMI control techniques involve both hardware implementations/methods and procedures. They may also be divided into intra-system and inter-system EMI control. Our major concern in this paper is intra-system EMI control, however, an overview of each may be appropriate at this time.

Figure 18 illustrates the basic elements of concern in an intra-system EMI problem. The test specimen may be a single box, an equipment, subsystem or system (an ensemble of boxes with interconnecting cables). From a strictly near-sighted or selfish point-of-view, the only EMI concern would appear to be degradation of performance due to self jamming such as suggested at the top of the figure. While this might be the primary emphasis, the potential problems associated with either (1) susceptibility to outside conducted and/or radiated emissions or (2) tendency to pollute the outside world from its own undesired emissions, come under the primary classification of intra-system EMI. Corresponding EMI-control techniques, however, address themselves to both self-jamming and emission/susceptibility in accordance with applicable EMI specifications. The techniques that will be discussed include filtering, shielding, wiring and grounding.

Inter-system EMI distinguishes itself by interference between two or more discrete and separate systems or platforms which are frequently under independent user control. Culprit emissions and/or susceptibility situations are divided into two classes: (1) antenna entry/exit and (2) back-door entry/exit. More than 95% of inter-system EMI problems involve the antenna entry/exit route of EMI. We can group inter-system EMI-control techniques by four fundamental categories: frequency management, time management, location management and direction management.

The first step in determining a solution is to identify the problem as either inter-system or intra-system EMI. Generally, if the specimen has an antenna and the problem develops from what exits or enters the antenna from another specimen or ambient, then the problem is identified as inter-system EMI. Otherwise, it is intra-system EMI which we will discuss now.

5.1 Intra-System EMI Control Techniques

5.1.1 Shielding

Shielding is used to reduce the amount of electromagnetic radiation reaching a sensitive victim circuit. Shields are made of metal and work on the principle that electromagnetic fields are reflected and/or attenuated by a metal surface. Different types of shielding are needed for different types of fields. Thus, the type of metal used in the shield and the shield's construction must be considered carefully if the shield is to function properly. The ideal shield has no holes or voids and, in order to accommodate cooling vents, buttons, lamps and access panels, special meshes and "EMI-hardened" components are needed.

TABLE IV. Effective Shielding for Different Materials

Shielding Material	Surface Resistance (Ω/square)	Shielding Effectiveness dB		
		At 10 MHz	At 100 MHz	At 1 GHz
Arc-Sprayed Zinc	0.002	106	92	98
Silver Acrylic Paint	0.004	67	93	97
Silver Deposition	0.05	57	82	89
Silver Epoxy Paint	0.1	59	81	87
Nickel Composite	3.0	35	47	57
Carbon Composite	10.0	27	35	41
Wire Screen (0.64 mm Grid)	N.A.	86	66	48

Effectiveness of shielding materials with 25 μm thickness and for frequencies for which the largest dimension of the shielding plate is less than a quarter of a wavelength.

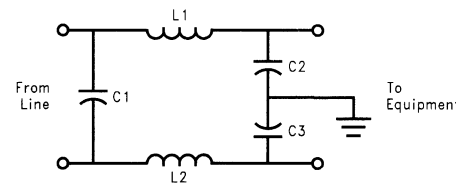
Once a printed-circuit board design has been optimized for minimal EMI, residual interference can be further reduced if the board is placed in a shielded enclosure. A box's shielding effectiveness depends on three main factors: its skin, the control of radiation leakage through the box's apertures or open areas (like cooling holes) and the use of filters or shields at interconnection points depending on shield grounding.

A box skin is typically fabricated from sheet metal or metallized plastic. Normally sheet metal skin that is 1 mm thick is more than adequate; it may have a shielding effectiveness of more than 100 dB throughout the high-frequency spectrum from 1 MHz to 20 GHz. Conductive coatings on plastic boxes are another matter. Table IV shows that at 10 MHz the shielding effectiveness can be as low as 27 dB if a carbon composite (cheap) is used, or it can run as high as 106 dB for zinc sprayed on plastic by an electric arc process (expensive). Plastic filled materials or composites having either conductive powder, flakes, or filament are also used in box shielding; they have an effectiveness similar to that of metallized plastics.

In many cases shielding effectiveness of at least 40 dB is required of plastic housings for microcontroller-based equipment to reduce printed-circuit board radiation to a level that meets FCC regulations in the United States or those of the VDE in Europe. Such skin shielding is easy to achieve. The problem is aperture leakage. The larger the aperture, the greater its radiation leakage because the shield's natural attenuation has been reduced. On the other hand, multiple small holes matching the same area as the single large aperture can attain the same amount of cooling with little or no loss of attenuation.

5.1.2 Filtering

Filters are used to eliminate conducted interference on cables and wires, and can be installed at either the source or the victim. *Figure 19* shows an AC power-line filter. The values of the components are not critical; as a guide, the capacitors can be between 0.01 and 0.001 μF, and the inductors are nominally 6.3 μH. Capacitor C1 is designed to shunt any high-frequency differential-mode currents before they can enter the equipment to be protected. Capacitors C2 and C3 are included to shunt any common-mode currents to ground. The inductors, L1 and L2, are called common-mode chokes, and are placed in the circuit to impede any common-mode currents.



TL/DD/12857–30

FIGURE 19. Filtering

5.1.3 Wiring

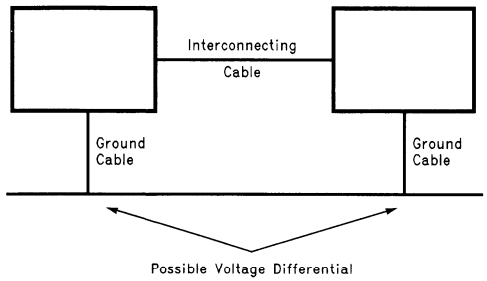
Now that the equipment in each box can be successfully designed to combat EMI emission and susceptibility separately, the boxes may be connected together to form a system. Here the input and output cables and, to a lesser extent, the power cable form an "antenna farm" that greatly threatens the overall electromagnetic compatibility of the system. Most field remedies for EMI problems focus on the coupling paths created by the wiring that interconnects systems. By this time most changes to the individual equipment circuits are out of the question.

Let us address five coupling paths that are encountered in typical systems comprised of two or more pieces of equipment connected by cables. These should adequately cover most EMI susceptibility problems. They are:

- common ground impedance coupling—a conducting path in which a common impedance is shared between an undesired emission source and the receptor.
- common-mode, radiated field-to-cable coupling—electromagnetic fields penetrate a loop formed by two pieces of equipment, a cable connecting them, and a ground plane.
- differential-mode, radiated field-to-cable coupling—the electromagnetic fields penetrate a loop formed by two pieces of equipment and an interconnecting transmission line or cable.
- crosstalk coupling—signals in one transmission line or cable are capacitively or inductively coupled into another transmission line.
- conductive path—through power lines feeding the equipment.

The first coupling path is formed when two pieces of equipment are connected to the same ground conductor at different points, an arrangement that normally produces a volt-

age difference between the two points. If possible, connecting both pieces of equipment to a single-point ground eliminates this voltage. Another remedy is to increase the high frequency impedance along a loop that includes the path between the ground connections of the two boxes. Examples include the isolation of printed-circuit boards from their cabinet or case, the use of a shielded isolation transformer in the signal path, or the insertion of an inductor between one or both boxes and the ground conductor. The use of balanced circuits, differential line drivers and receivers, and absorbing ferrite beads and rods on the interconnecting cable can further reduce currents produced by this undesirable coupling path.



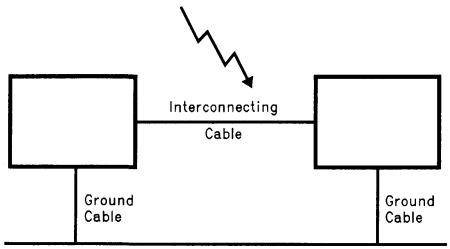
TL/DD/12857–31

FIGURE 20. Common Ground Impedance Coupling

A balanced circuit is configured so its two output signal leads are electrically symmetrical with respect to ground, as the signal increases on one output, the signal on the other decreases. Differential line drivers produce a signal that is electrically symmetrical with respect to ground from a single-ended circuit in which only one lead is changing with respect to ground. Ferrite beads, threaded over electrical conductors, substantially attenuate electromagnetic interference by turning radio-frequency energy into heat, which is dissipated in them.

In the second coupling path, a radiated electromagnetic field is converted into a common-mode voltage in the ground plane loop containing the interconnect cable and both boxes. This voltage may be reduced if the loop area is trimmed.

Ferrite beads and ferrite filtered connectors applied to the interconnecting cable can help reduce common mode currents.



TL/DD/12857–32

FIGURE 21. Common-Mode, Radiated Field-to-Cable Coupling

The third coupling path produces a differential-mode voltage that appears across the input terminals of the EMI receptor. One way of controlling this is to cancel or block the pickup of differential-mode radiation. In a balanced transmission line, this is done by use of twisted-wire pairs and a shielded cable.

For crosstalk, the fourth coupling path, the reduction of capacitive coupling, can be achieved by the implementation of at least one of these steps:

- Reducing the spacing between wire pairs in either or both of the transmission lines.
- Increasing the separation between the two transmission lines.
- Reducing the frequency of operation of the source, if possible.
- Adding a cable shield over either or both transmission lines.
- Twisting the source's or receptor's wire pairs.
- Twisting both wire pairs in opposite directions.

The fifth coupling path conductively produces both common-mode and differential-mode noise pollution on the power mains. Among several remedies that can suppress the EMI there are filters and isolation transformers.

There are only about 50 common practical remedies that can be used in most EMI situations. Of these, about 10 suffice in 80 percent of the situations. Most engineers are aware of at least some of these remedies—for example, twisting wires to reduce radiation pickup.

In order to attack the EMI problem, one can make use of the information contained in Tables V and VI. First, decide what coupling path has the worst EMI interference problem. From the 11 most common coupling paths listed at the top of the table, find the problem coupling path. Using the numbers found in that table entry, locate the recommended remedy or remedies from the 12 common EMI fixes identified at the bottom of the table. This procedure should be repeated until all significant coupling paths have been properly controlled and the design goal has been met.

TABLE V. Electromagnetic Interference Coupling Paths

Problem	Solution (See Table VI)
Radiated Field to Interconnecting Cable (Common-Mode)	2, 7, 8, 9, 11
Radiated Field to Interconnecting Cable (Differential-Mode)	2, 5, 6
Interconnecting Cable to Radiated Field (Common-Mode)	1, 3, 9, 11
Interconnecting Cable to Radiated Field (Differential-Mode)	1, 3, 5, 6, 7
Cable-to-Cable Crosstalk	1, 2, 3, 4, 5, 6, 10, 11
Radiated Field to Box	12, 13
Box to Radiated Field	12, 13
Box-to-Box Radiation	12, 13
Box-to-Box Conduction	1, 2, 7, 8, 9
Power Mains to Box Conduction	4, 11
Box to Power Mains Conduction	4, 11
Electromagnetic Interference Coupling Paths	4

TABLE VI. Electromagnetic Interference Fixes

Solution	Complexity	Cost
1. Insert Filter in Signal Source	Low	Medium
2. Insert Filter in Signal Receptor	Low	Medium
3. Insert Filter in Power Source	Low	Medium
4. Insert Filter in Power Receptor	Low	Medium
5. Twist Wire Pair	Low	Low
6. Shield Cable	Low	Medium
7. Use Balanced Circuits	Medium	Medium
8. Install Differential Line Drivers and Receivers	Medium	Medium
9. Float Printed Circuit Board(s)	Medium	Medium
10. Separate Wire Pair	Low	Medium
11. Use Ferrite Beads	Low	Medium
12. Use a Multilayer Instead of a Single Layer Printed Circuit Board(s)	Medium	High

5.2 Inter-System EMI-Control Techniques

There are many EMI controls that may be carried out to enhance the chances of inter-system EMC. They can be grouped into four categories which we will discuss briefly. The following discussion is not intended to be complete but merely provide an overview of some EMI control techniques available to the inter-system designer and user.

5.2.1 Frequency Management

Frequency management suggests both transmitter emission control and improvement of receptors against spurious responses. The object is to design and operationally maintain transmitters so that they occupy the least frequency spectrum possible in order to help control electromagnetic pollution. For example, this implies that long pulse rise and fall times should be used. Quite often one of the most convenient, economic and rapid solutions to an EMI problem in the field, is to change frequency of either the victim receiver or the culprit source.

5.2.2 Time Management

In those applications where information is passed between systems, a possible time management technique could be utilized where the amount of information transferred is kept to a minimum. This should reduce the amount of time that the receptor is susceptible to any EMI. In communication protocols, for example, essential data could be transmitted in short bursts or control information could be encoded into fewer bits.

5.2.3 Location Management

Location management refers to EMI control by the selection of location of the potential victim receptor with respect to all other emitters in the environment. In this regard, separation distance between transmitters and receivers is one of the most significant forms of control since interfering source emissions are reduced greatly with the distance between them. The relative position of potentially interfering transmitters to the victim receiver are also significant. If the emitting source and victim receiver are shielded by obstacles, the degree of interference would be substantially reduced.

5.2.4 Direction Management

Direction management refers to the technique of EMI control by gainfully using the direction and attitude of arrival of electromagnetic signals with respect to the potential victim's receiving antennae.

6.0 SILICON CHANGES

If we examine the modes of EMI emissions and the methods of suppression that were described, on standard CMOS silicon the techniques are exactly the same for what occurs on a PCB. What silicon design engineers at National Semiconductor's Embedded Technologies Division noticed, was that a majority of the EMI emitted from a controller or processor is caused by the transient I_{CC} current flowing in the loop formed from the bypass capacitor to the V_{CC} pin and out of the GND pin. When this current was measured by inserting a 1Ω resistor in this path, a 40 mA (10 ns) current spike was revealed at every clock edge (repetition rate = 2x clock frequency). This occurred even when outputs were toggling or tristated, and instructions that were being executed had very little impact on the magnitude or duration of the spike.

6.1 Chip Problems

Let's note the four items that produce higher amplitude and frequency fundamental and harmonics: large amplitude change, fast rise time or fall time, high repetition rate and finally large pulse width (a 50% duty cycle). Table VII lists these.

TABLE VII. IC Chip Problems

1. Large Amplitude Change
2. Fast Rise or Fall Time
3. High Repetition Rate
4. Large Pulse Width (i.e. 50% Duty Cycle).

As a rule, the waveforms di/dt (rate of change of current) and dv/dt (rate of change of voltage) must be kept to a minimum. If a silicon design engineer could shape both current and voltage waveforms around the device, these problems would be solved. If we look at the current and voltage problems separately, we can solve them both and decrease radiated emissions by more than 20 dB.

But, what about the antennae responsible for radiation? This is where packaging technology comes into the picture. The loop size of a typical system is approximately greater

than 90% PCB trace dimensions plus IC package dimensions and less than 10% IC chip dimensions, which both factor into the antenna gain. Therefore the IC package design should obey several rules, see Table VIII. First a ground pin (current return) adjacent to each power pin will minimize the loop sizes and simplify layout of decoupling circuits. Also an output pin which sources or sinks a large and frequent di/dt (for example crystal or clock pins) should also be adjacent to the power/ground pin pair to keep its current loops small. It helps to have small package dimensions. A last point is that sharing of a common power/ground should be limited to prevent their voltage from bouncing due to the total Ldi/dt + iR. This bouncing is conducted to inputs sharing the common power/ground which in several cases are susceptible since their V_{IH} and V_{IL} trip points are referenced to it. The bouncing is conducted through non-tristated outputs sharing this common power/ground and is passed on outside the IC as additional conducted and radiated emissions from bouncing V_{OH} or V_{OL} levels.

TABLE VIII. Package and PCB Traces form Loop or Whip Antennae which Radiate

1. Ground pin (current return) not adjacent to power pin). Pair should be together.
2. Power/Ground pair not adjacent to output pin (especially crystal of clock pins).
3. Too much sharing of common power/ground having voltage bounce due to its total Ldi/dt + iR. a. Conducted emissions to susceptible I/O b. Radiated emissions from susceptible I/O.
4. Large package dimensions.

6.2 Silicon Design Changes

National's design goal was to reshape the IC pin current waveforms from their original values as follows: I_{CC} pulses to 3X rise time, 0.2X amplitude and 5X width; I_{OL} and I_{OH} pulses to 10X rise time. A second goal is to provide separate power and ground bus systems on-chip for each of the following groups: Chip digital logic, Chip I/O buffers, and if present a chip A/D converter (or other analog intensive sections). This is an extension of proper IC packaging pinout covered earlier to prevent conducted emissions to susceptible circuitry.

Inspection of these goals leads us to divide the chip into separate areas which we named the nucleus, the perimeter and the A/D. Also the partitioning of the nucleus (containing the digital logic functions) and the perimeter (containing digital I/O, oscillator and comparator functions) allows use of an on-chip device to choke the nucleus' I_{CC} current, thereby wave shaping the I_{CC}. The I_{CC} choke, however, causes the nucleus V_{CC} voltage to dip by 0.5V to 1V at each clock edge. These V_{CC} voltage swings will be confined to within the chip's nucleus where the circuitry (digital logic) is tolerant and no significant radiation antennae exist.

On-chip level shifting circuitry is inserted as interface between the nucleus and the perimeter. It permits the nucleus to operate at a lower V_{CC} than the perimeter. This permits the nucleus V_{CC} to come through the on-chip I_{CC} choke emerging as a bouncing V_{CC}.

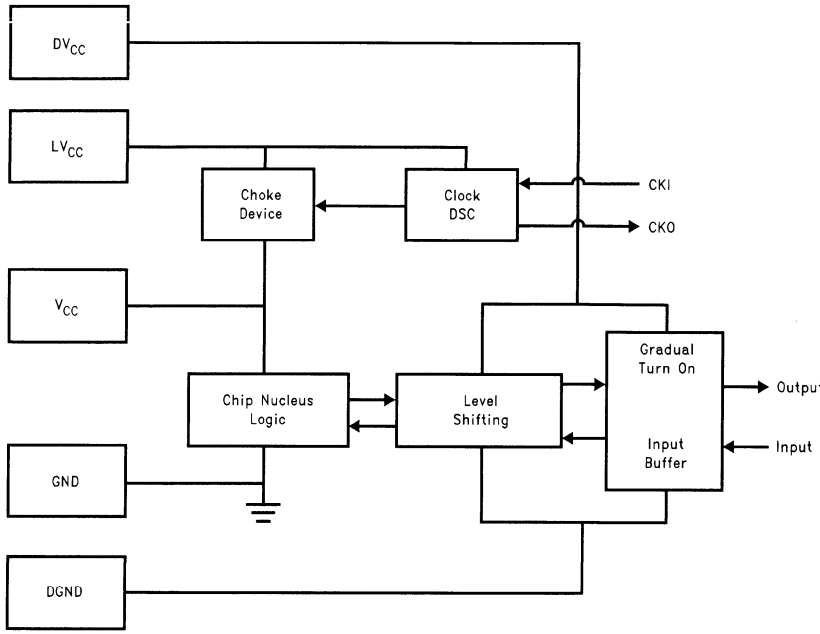
In general, outputs toggle much less frequently than the clock. This rate is limited by software execution rates. However, for the case of a clock or address/data bus outputs this may not be the case. Looking at time domain EMI instead of frequency domain EMI (which may have some time averaging) may motivate one to take action and reduce EMI from the chip's output drivers. CMOS output drivers turn on fast, in about 1 ns. This causes 2 problems. The first is called "shoot through" current which flows from DV_{CC} to DGND when both the pull up and the pull down drivers are on and the output load is small. The second is due to output current magnitude increasing to maximum in about 1 ns when the output changes state. Both problems are solved by using fast turn off, gradual turn on device drivers.

Figure 22 illustrates, in block diagram form, how the chip is partitioned for EMC. Note that the level shifter block is multichannel, and most channels need dedicated level shifting devices in one direction only as shown. The power/ground pads named DV_{CC} (Driver V_{CC}), LV_{CC} (Logic V_{CC}), V_{CC} (a global V_{CC} to most of the chip), GND (a global GND to most of the chip) and DGND (driver ground) are all available. To minimize package pin count, DV_{CC} and LV_{CC} can be bonded to the same pin, GND and DGND can be bonded to the same pin, and the V_{CC} pad is not bonded. A higher pin count package is used to bring out each pad to a separate pin to allow study of the several packaging options. The V_{CC} pad is not bonded when utilizing the on-chip I_{CC} choke to prevent RE (radiated emissions) from V_{CC} bounce.

Three operating modes can be evaluated with GND and DGND pads always connected together to 0V. The first mode applies 5V to DV_{CC}, LV_{CC} and V_{CC} to check the RE improvements due to gradual turn on outputs alone. The second mode applies 5V to DV_{CC} and LV_{CC} where V_{CC} floats as a test point. This checks the RE improvements due to gradual turn on and I_{CC} choking. The third mode applies 5V to DV_{CC} and 3V to V_{CC} and LV_{CC} to check the RE improvement in this mode which actually requires a V_{CC} additional package pin.

TABLE IX. Results of Various Pinout Configurations

Chip Pads	Configuration A	Configuration B	Configuration C
DV_{CC}	5V	5V	5V
LV_{CC}	5V	5V	3V
V_{CC}	5V	Test Point	3V
GND	0	0	0
DGND	0	0	0
EMI	Baseline	−20 dB	−6 dB
Clock Speed	10 MHz	>5 MHz	4 MHz

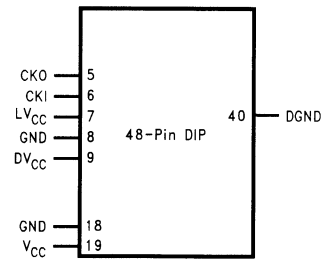


TL/DD/12857–33

FIGURE 22. Block Diagram of EMC Circuitry

6.3 Packaging Changes

Recalling our IC packaging rules for EMC, and upon reviewing the packaging pitfalls (Table VIII), one can now appreciate an IC package build sheet *(Figure 23)* that illustrates how to bond out and pin out the five power/ground pads on the die. As advised, power pins having current return ground pins adjacent are LV_{CC} and GND, V_{CC} and GND; but DV_{CC} and DGND are not adjacent. This is permitted since DV_{CC} current is returned via output high pins (not DGND), also DGND current is sourced via output low pins. Therefore placing DV_{CC} adjacent to DGND does not contribute to small area current loops. Note that CKI, CKO, LV_{CC}, and GND are all adjacent pins for the sake of clustering the oscillator external tank circuit into the smallest current loop area. Since the oscillator is supplied through LV_{CC} and GND, added benefit of running both oscillator and nucleus at 3V is possible if one were to choose defeating the on-chip choke by shorting LV_{CC} to V_{CC} in the choke block.



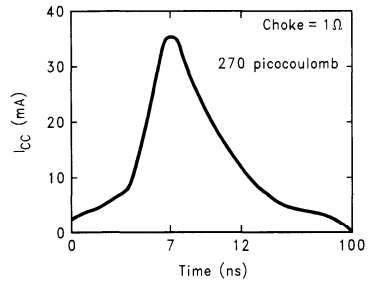
TL/DD/12857–34

FIGURE 23. EMC Pinout

6.4 Results of IC Circuit Changes

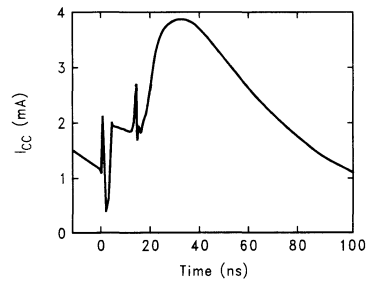
Simulations give an idea of the expected improvement from addition of the I_{CC} choke on the silicon. *Figure 24a* shows actual I_{CC} as measured by a series resistance during switching transients. Simulation of an equivalent circuit with the addition of the choke predicts a decrease of I_{CC} from about 37 mA to less than 4 mA.

This decrease of 10x would imply a 20 dB improvement in gross emissions.



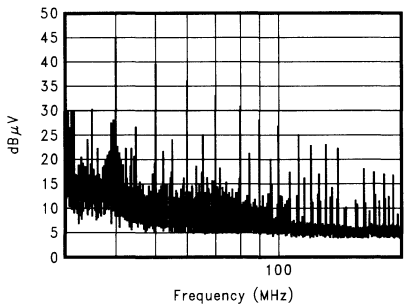
TL/DD/12857-35

FIGURE 24a. Original I_{CC}



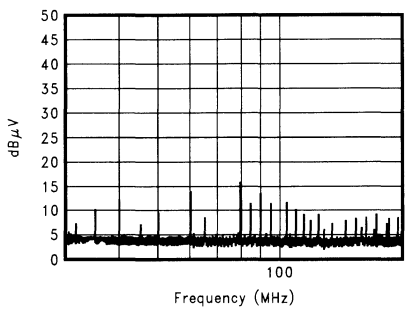
TL/DD/12857-36

FIGURE 24b. I_{CC} with Improvements



TL/DD/12857-37

FIGURE 25a. Emissions without V_{CC} Choke



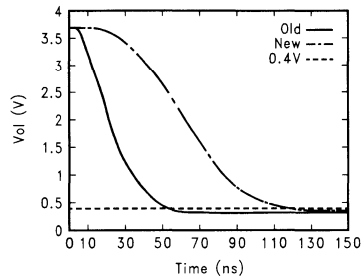
TL/DD/12857-38

FIGURE 25b. Emissions with Addition of V_{CC} Choke

Test results in *Figure 26* show emissions of 45 dBμV reduced to less than 20 dBμV on the same device. Unchoked tests were performed using the bonding options previously discussed.

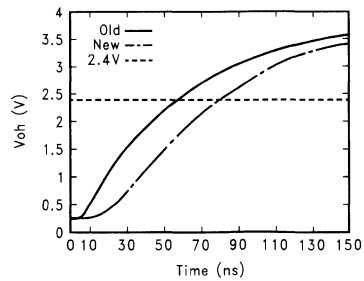
Figure 25 gives the predicted performance degradation from implementing gradual turn-on outputs. High to low transition time propagation delays are doubled and low to high transitions are slowed by about 50%. Remember, though, that propagation delays for an embedded microcontroller are usually of minimal concern since control is based on levels or edges of individual signals or on timing which is under software control. Software control is slow enough that the change in hardware timing is probably negligible.

An indication of the type of emission reduction resulting from the implementation of gradual turn-on can be seen if you look at the baselines of the plots of *Figure 26*. Changes of the outputs occur at low frequency with large amplitude, fast rise time current spikes. The low frequency of the signal keeps the harmonic close together and makes the emissions appear broadband. The fast rise time maintains a higher level of emissions to a higher frequency.



TL/DD/12857-39

FIGURE 26a. V_{OL}



TL/DD/12857-40

FIGURE 26b. V_{OH}

7.0 SYSTEM PERFORMANCE USING EMI REDUCED PARTS

We have discussed in previous sections that we can reduce EMI by proper board design and by the selection of low EMI devices. We will now investigate whether EMI reduced parts are worth it. The goal of this testing is to determine the relative electromagnetic emissions from various production options of the COP8 microcontroller. Since most people use OTPs to qualify their end product for emissions approval, both OTPs and ROMmed devices will be tested. The options chosen were:

1. COP8788EGMH (OTP) part used for development and Prototype approval
2. COP888EK (masked ROM) part used for final production (EMI reduced)
3. COP8788GG (OTP) part used for development and Prototype approval
4. COP888GG (masked ROM) part used for final production (EMI reduced)

7.1 Test Procedure

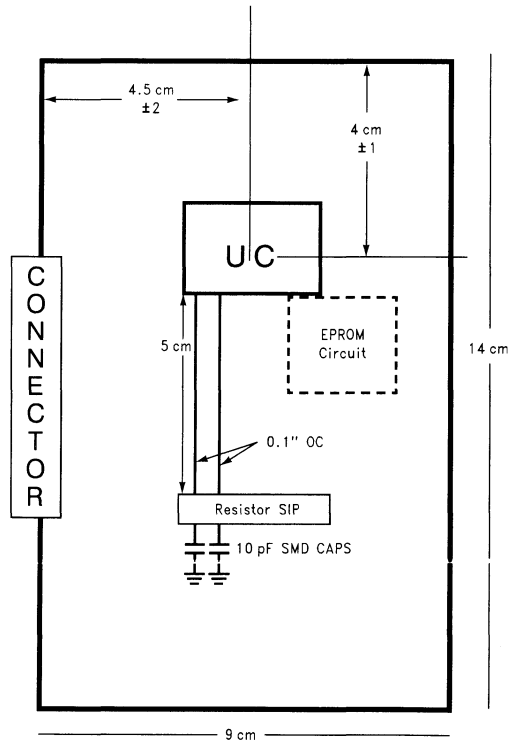
These tests were performed in two ways:

1. Two DIP parts, a COP888EK and a COP888EGMH OTP device, were programmed for use as the main processor in an automotive radio and the relative performance of the radio was measured for each device. The measure of performance chosen was Usable Sensitivity. This is defined as the minimum input required at the antenna in order to achieve a predetermined signal to noise ratio (SNR) at the speaker. In this case the input was chosen to be a 22 kHz deviation of a frequency modulated signal within the US commercial FM radio broadcast band. The signal to noise criterion used was 30 dB.
2. Two other devices, a COP888GG and a COP87M88GG OTP, were placed in a common two sided PCB in a TEM cell and were programmed to run a standard EMI test program included in National's first silicon engineering code. Load capacitors of 10 pF and 4.7 kΩ pull-up resistors used on the eight pins of Port L. The port was programmed to count in a binary manner with the least significant bit changing state every 100 μs. (*Figure 27* illustrates the board layout.) Four AA batteries were attached to the board and the supply voltage was regulated with a Zener diode.

   Emissions were measured with the TEM cell on a Tektronix Model 2754P Spectrum Analyzer over the range of DC to 750 MHz.

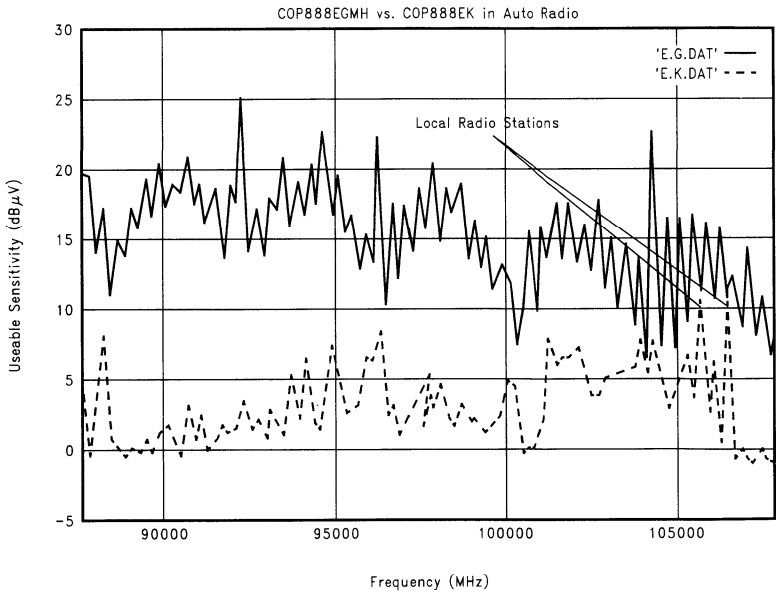
7.2 Radio Results

Worst case useable sensitivity is improved by about 17 dB (25 dBμV to 8 dBμV) by the change from the COP888EGMH OTP to the mask programmed COP888EK. Of course not all frequencies show this improvement, but there is some increased sensitivity throughout the spectrum. Note that two local radio stations (at 105.7 MHz and 106.5 MHz) were received even without an external antenna.



TL/DD/12857–42

FIGURE 27. PCB Layout for EMI Testing



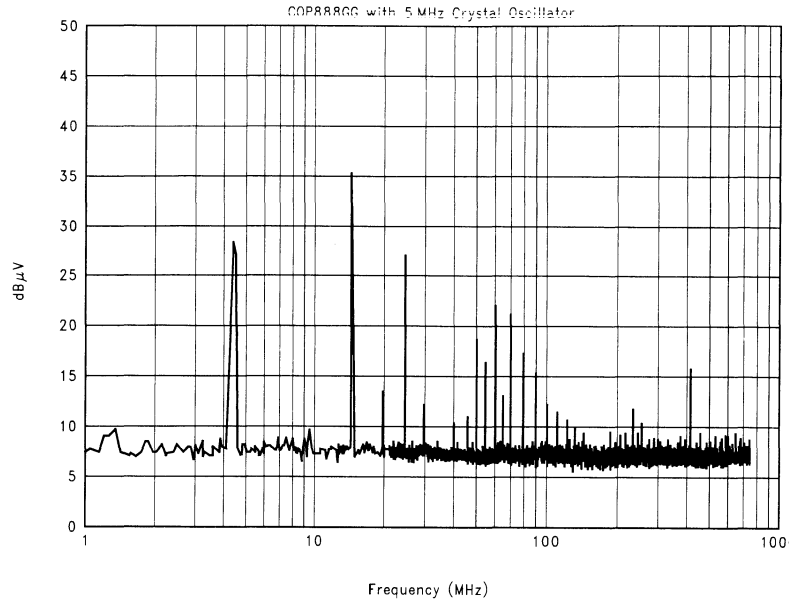
TL/DD/12857–41

FIGURE 28. Usable Sensitivity Comparison

7.3 EMI Board Results

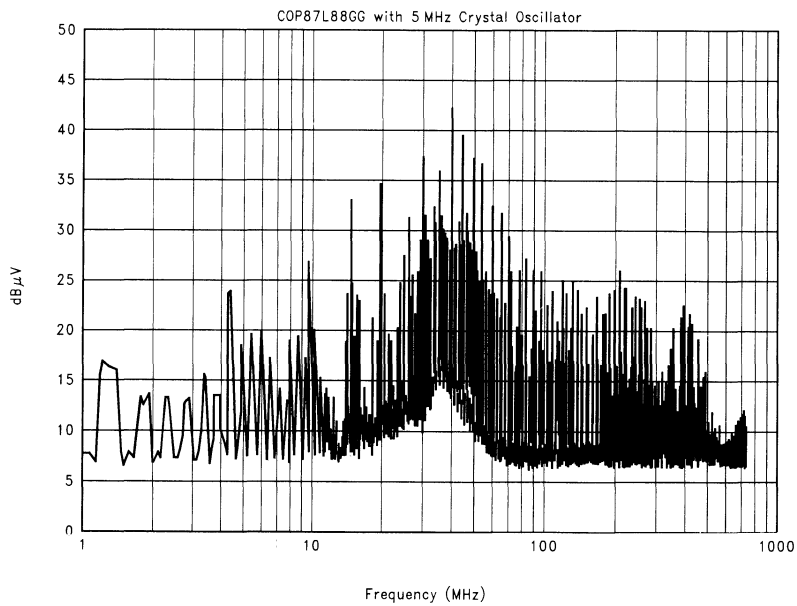
Except for the fundamental oscillator frequency of approximately 5 MHz, emissions are down at least 10 dB on the monolithic masked device over the multichip OTP device. Emissions beyond 100 MHz are practically eliminated.

Much more noticeable is the reduction of spectral density in the noise. Off harmonic noise is not noticeable.



TL/DD/12857–43

FIGURE 29. COP888GG with 5 MHz Crystal Oscillator



TL/DD/12857–44

FIGURE 30. COP87L88GG with 5 MHz Crystal Oscillator

7.4 Conclusions

Radiated Emissions do not come from a device, but from a board. Devices have some control over the emissions because they generate the noise which is radiated. The equation normally used to predict emission from a system is:

$$|E|_{Max} = \frac{1.32 \times 10^{-3} \times I \times A \times Freq^2}{D} \times \left[1 + \left(\frac{\lambda}{2\pi D}\right)^2\right]^{1/2}$$

where,

$|E|_{Max}$ is the maximum E-field in the plane of the loop in $\mu V/m$

I is the current amplitude, at the frequency of interest, in milliamps

A is the loop area is cm2

λ is the wavelength at the frequency of interest

D is the observation distance in meters

Freq is the frequency in MHz

and the perimeter of the loop $P << \lambda$

When applied to a chip, the area is very small and current is relatively low compared to current consumption in a typical system, thus the emissions are low.

8.0 DESIGN GUIDELINES

The growth of concern over electromagnetic compatibility (EMC) in electronic systems continues to rise in the years since the FCC proclaimed that there shall be no more pollution of the electromagnetic spectrum. Still, designers have not yet fully come to grips with a major source and victim of electromagnetic interference—the PCB. The most critical stage for addressing EMI is during the circuit board design. Numerous tales of woe can be recounted about the eleventh hour attempt at solving an EMI problem by retrofit because EMC was given no attention during design. This retrofit ultimately costs much more than design stage EMC, holds up production and generally makes managers unhappy. With these facts in mind, let's address electromagnetic compatibility considerations in PCB design.

8.1 Logic Selection

Logic selection can ultimately dictate how much attention must be given to EMC in the total circuit design. The first guideline should be: use the slowest speed logic that will do the job. Logic speed refers to transition times of output signals and gate responses to input signals. Many emissions and susceptibility problems can be minimized if a slow speed logic is used. For example, a square wave clock or signal pulse with a 3 ns rise time generates radio frequency (100 MHz and higher) energy that is gated about on the PCB. It also means that the logic can respond to comparable radio frequency energy if it gets onto the boards.

The type of logic to be used is normally an early design decision, so that control of edge speeds and, hence, emissions and susceptibiity is made early. Of course, other factors such as required system performance, speed, and timing considerations must enter into this decision. If possible, design the circuit with as slow speed logic as possible. The use of slow speed logic, however, does not guarantee that EMC will exist when the circuit is built; so proper EMC techniques should still be implemented consistently during the remainder of the circuit design.

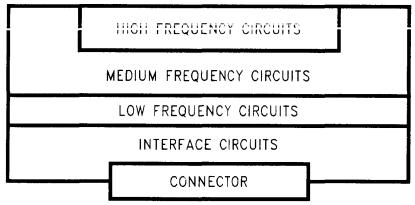
8.2 Component Layout

Component layout is the second stage in PCB design. Schematics tell little or nothing about how systems will perform once the board is etched, stuffed, and powered. A circuit schematic is useful to the design engineer, but an experienced EMC engineer refers to the PCB when trouble-shooting. By controlling the board layout in the design stage, the designer realizes two benefits: (1) a decrease in EMI problems when the circuit or system is sent for EMI or quality assurance testing; and (2) the number of EMI coupling paths is reduced, saving troubleshooting time and effort later on.

Some layout guidelines for arranging components according to logic speed, frequency, and function are shown in *Figure 31*. These guidelines are very general. A particular circuit is likely to require a combination and/or trade-offs of these arrangements. Isolation of the I/O from digital circuitry is important where emissions or susceptibility may be a problem. For the case of emissions, a frequently encountered coupling path involves digital energy coupling through I/O circuitry and signal traces onto I/O cables and wires, where the latter subsequently radiate. When susceptibility is a problem, it is common for the EMI energy to couple from I/O circuits onto sensitive digital lines, even though the I/O lines may be "opto-coupled" or otherwise supposedly isolated. In both situations, the solution often lies in the proper electrical and physical isolation of analog and low speed digital lines from high speed circuits. When high speed signals are designed to leave the board, the reduction of EMI is usually performed via shielding of I/O cables and is not considered here.

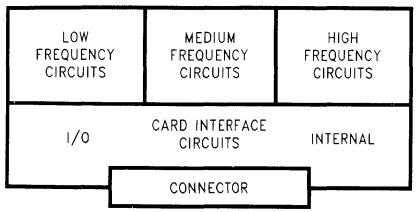
Therefore, a major guideline in layout out boards is to isolate the I/O circuitry from the high speed logic. This method applies even if the logic is being clocked at "only" a few MHz. Often, the fundamental frequency is of marginal interest, with the harmonics generated from switching edges of the clock being the biggest emission culprits. Internal system input/output PCB circuitry should be mounted as close to the edge connector as possible and capacitive filtering of these lines may be necessary to reduce EMI on the lines.

High speed logic components should be grouped together. Digital interface circuitry and I/O circuitry should be physically isolated from each other and routed on separate connectors, if possible as shown in *Figure 27d*.



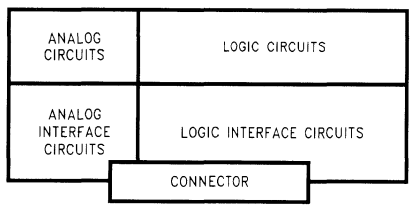
TL/DD/12857–45

FIGURE 31a. Board Layout



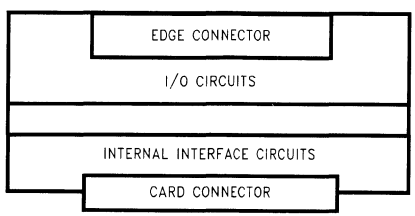
TL/DD/12857–46

FIGURE 31b. Board Layout



TL/DD/12857–47

FIGURE 31c. Board Layout



TL/DD/12857–48

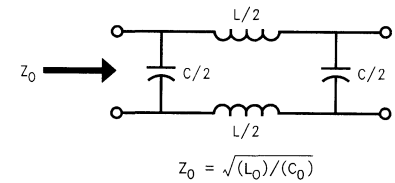
FIGURE 31d. Board Layout

8.3 Power Supply Bussing

Power supply bussing is the next major concern in the design phase. Isolated digital and analog power supplies must be used when mixing analog and digital circuitry on a board. The design preferably should provide for separate power supply distribution for both the analog and digital circuitry. Single point common grounding of analog and digital power supplies should be performed at one point and one point only—usually at the motherboard power supply input for multi-card designs, or at the power supply input edge connector on a single card system. The fundamental feature of good power supply bussing, however, is low impedance and good decoupling over a large range of frequencies. A low impedance distribution system requires two design features: (1) proper power supply and return trace layout and (2) proper use of decoupling capacitors.

An exception to the rule of analog and digital ground separation applies in the case of high speed interface between analog and digital portions. If possible, the ground plane should be continuous under these interface traces to improve the proximity of the return trace and thus minimize the size of the loop.

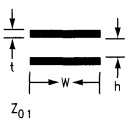
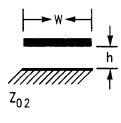
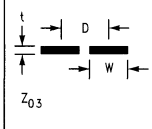
At high frequencies, PCB traces and the power supply busses ($+V_{CC}$ and 0V) are viewed as transmission lines with associated characteristic impedance, Z_O, as modeled in *Figure 28*. The goal of the designer is to maximize the capacitance between the lines and minimize the self-inductance, thus creating a low Z_O. Table X, shows the characteristic impedance of various two-trace configurations as a function of trace width, W, and trace separation, h.



TL/DD/12857–49

FIGURE 32. Transmission Line Modelling of PCB Traces

TABLE X. Characteristic Impedance of Different Conductor Pairs

			
W/h or D/W	**Parallel Strips**	**Strips Over Ground Plane**	**Strips Side by Side**
0.5	377	377	N/A
0.6	281	281	N/A
0.7	241	241	N/A
0.8	211	211	N/A
0.9	187	187	N/A
1.0	169	169	0
1.1	153	153	25
1.2	140	140	34
1.5	112	112	53
1.7	99	99	62
2.0	84	84	73
2.5	67	67	87
3.0	56	56	98
3.5	48	48	107
4.0	42	42	114
5.0	34	34	127
6.0	28	28	137
7.0	24	24	146
8.0	21	21	153
9.0	19	19	160
10.0	17	17	166
12.0	14	14	176
15.0	11.2	11.2	188
20.0	8.4	8.4	204
25.0	6.7	6.7	217
30.0	5.6	5.6	227
40.0	4.2	4.2	243
50.0	3.4	3.4	255

Any one of the three configurations may be viewed as a possible method of routing power supply (or signal) traces. The most important feature of this table is the noticeable difference in impedance between the parallel strips and strip over ground plane compared with the side-by-side configurations.

As an example of the amount of voltage that can be generated across the impedance of a power bus, consider TTL logic which pulls a current of approximately 16 mA from a supply that has a 25Ω bus impedance (this assumes no decoupling present). The transient voltage is approximately $dV = 0.016 \times 25\Omega = 400$ mV, which is equal to the noise immunity level of the TTL logic. A 25Ω (or higher) impedance is not uncommon in many designs where the supply and return traces are routed on the same side of the board in a side-by-side fashion. In fact, it is not uncommon to find situations where the power supply and return traces are routed quite a distance from each other, thereby increasing the overall impedance of the distribution system and the size of the loop antenna. This is obviously a poor layout.

Power and ground planes offer the lowest overall impedance. The use of these planes leads the designer closer to a multi-layer board. At the very least, it is recommended that all open areas on the PCB be "landfilled" with an interconnected 0V reference plane so that ground impedance is minimized. This landfill technique may be ineffective, however if minimum width traces connect the filled areas.

Multi-layer boards offer a considerable reduction in power supply impedance, as well as other benefits. Extending the table we just saw, the impedance of a multi-layer power/ground plane bus grows very small (on the order of an Ω or less), assuming a W/h ratio greater than 100. Multi-layer board designs also pay dividends in terms of greatly reduced EMI, and they provide close control of line impedances where impedance matching is important. In addition,

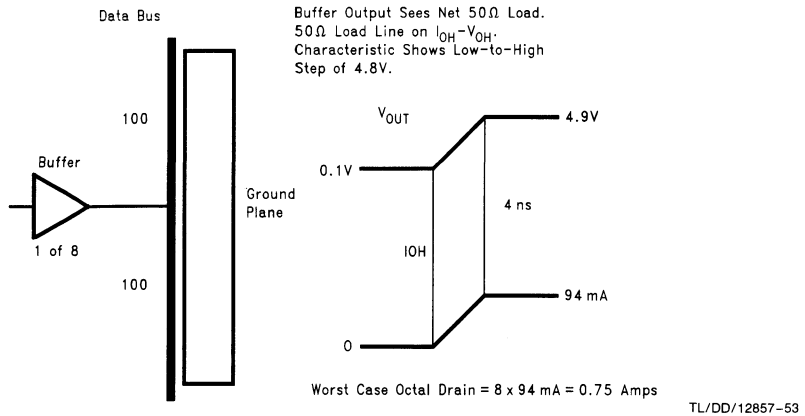


FIGURE 33. Decoupling Example

shielding benefits can be realized. For high-density, high-speed logic applications, the use of a multi-layer board is almost mandatory. The problem with multi-layer boards is the increased cost of design and fabrication and increased difficulty in board repair.

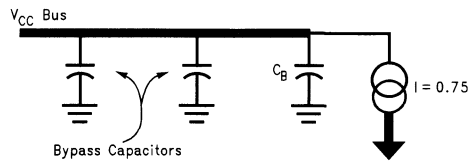
8.4 Decoupling

High-speed CMOS has special decoupling and PCB layout requirements. Adhering to these requirements will ensure that maximum advantage is gained with CMOS devices in system performance and EMC performance.

Local high frequency decoupling is required to supply power to the chip when it is transitioning from a LOW to a HIGH value. This power is necessary to charge the load capacitance or drive a line impedance.

For most power distribution networks, the typical impedance can be between 50Ω and 100Ω. This impedance appears in series with the load impedance and will cause a droop in the V_{CC} at the part. This droop limits the available voltage swing at the local node, unless some form of decoupling is used. This drooping of rails will cause the rise and fall times to become elongated. Consider the example presented in *Figure 29* used to help calculate the amount of decoupling necessary. This circuit utilizes an octal buffer driving a 100Ω bus from a point somewhere in the middle.

Being in the middle of the bus, the driver will see two 100Ω loads in parallel, or an effective impedance of 50Ω. To switch the line from rail to rail, a drive of 94 mA is needed (4.8V/50Ω) and more than 75 mA wil be required if all eight lines switch at once. This instantaneous current requirement will generate a voltage drop across the impedance of the power lines, causing the actual V_{CC} at the chip to droop. This droop limits the voltage swing available to the driver. The net effect of the voltage droop will be to lengthen device rise and fall times and slow system operation. A local decoupling capacitor is required to act as a low impedance supply for the driver chip during high current demands. It will maintain the voltage within acceptable limits and keep rise and fall times to a minimum. The necesssary values for decoupling capacitors can be calculated with the formula given in *Figure 30.*



Specify V_{CC} Droop = 0.1V max during switching time of 4 ns

Q = CV charge on capacitor

I = C dV/dt

C = I dt/dV = 750 mA × 4 ns/0.1V = 0.030 μF

Select C_B = 0.047 μF or greater

FIGURE 34. Decoupling Example

In this example, if the V_{CC} droop is to be kept below 0.1V and the edge rate equals 4 ns, we can calculate the value of the decoupling capacitor by use of the charge on a capacitor equation: Q = CV. The capacitor must supply the high demand current during the transition period and is represented by I = C(dV/dt). Rearranging this somewhat yields C = I(dt/dV).

Now, I = 750 mA assuming all 8 outputs switch simultaneously for worst case conditions, dt = switching period or 4 ns, and dV is the specified V_{CC} droop of 0.1V. This yields a calculated value of 0.030 μF for the decoupling capacitor. So, a selection of 0.047 μF or greater should be sufficient.

Higher gate count devices, e.g. microcontrollers, with high internal edge rates present additional decoupling problems. Capacitors often suffer from parasitic series inductive effects at high frequency. In fact, a poorly selected capacitor can become almost entirely inductive at high frequency. This can be counteracted by paralleling additional capacitors of lower value and corresponding higher self-resonant frequency. Of course the lower value capacitor should be placed closer to the chip to reduce the size of the resultant current loop.

Inductors or ferrite beads connected in series between the capacitors can help to minimize noise currents and voltages from reaching the rest of the circuit.

It is good practice to distribute decoupling capacitors evenly throughout the logic on the board, placing at least one capacitor for every package as close to the power and ground pins as possible. The parasitic inductance in the capacitor leads can be greatly reduced or eliminated by the use of surface mount chip capacitors soldered directly onto the board at the appropriate locations. Decoupling capacitors need to be of the high K ceramic type with low equivalent series resistance (ESR), consisting primarily of series inductance and series resistance. Capacitors using Z5U dielectric have suitable properties and make a good choice for decoupling capacitors; they offer minimum cost and effective performance.

8.5 Proper Signal Trace Layout

Although crosstalk cannot be totally eliminated, there are some design techniques that can reduce system problems resulting from crosstalk. In any design, the distance that lines run adjacent to each other should be kept as short as possible. The best situation is when the lines are perpendicular to each other. Crosstalk problems can also be reduced by moving lines further apart or by inserting ground lines or planes between them.

For those situations where lines must run parallel, as in address and data buses, the effects of crosstalk can be minimized by line termination. Terminating a line in its characteristic impedance reduces the amplitude of an initial crosstalk pulse by 50%. Terminating the line will also reduce the amount of ringing.

There are several termination schemes which may be used. They are series, parallel, AC parallel and Thevenin terminations. AC parallel and series terminations are the most useful for low power applications since they do not consume any DC power. Parallel and Thevenin terminations increase DC power consumption.

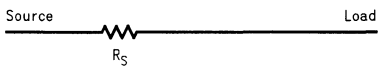
Series terminations are most useful in high-speed applications where most of the loads are at the far end of the line. Loads that are between the driver and the end of the line will receive a two-step waveform. The first wave will be the incident wave. The amplitude is dependent upon the output impedance of the driver, the value of the series resistor and the impedance of the line according to the formula:

$$V_W = \frac{V_{CC} \times Z_{OE}}{Z_{OE} + R_S + Z_S}$$

where R_S is the series Resistor

Z_S is the output impedance of the driver

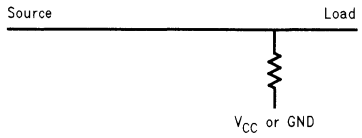
Z_{OE} is the equivalent line impedance



TL/DD/12857–55

FIGURE 35a. Series Termination

The amplitude will be one-half the voltage swing if R_S (the series resistor) plus the output impedance (Z_S) of the driver is equal to the line impedance (Z_{OE}). The second step of the waveform is the reflection from the end of the line and will have an amplitude equal to that of the first step. All devices on the line will receive a valid level only after the wave has propagated down the line and returned to the driver. Therefore, all inputs will see the full voltage swing within two times the delay of the line.



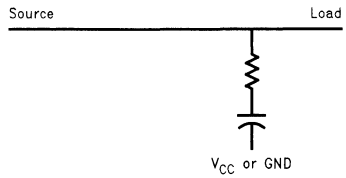
TL/DD/12857–56

FIGURE 35b. Parallel Termination

Parallel terminations are not generally recommended for CMOS circuits due to their power consumption, which can exceed the power consumption of the logic itself. The power consumption of parallel terminations is a function of the resistor value and the duty cycle of the signal. In addition, parallel termination tends to bias the output levels of the driver towards either V_{CC} or ground depending on which bus the resistor is connected to. While this feature is not desirable for driving CMOS inputs because the trip levels are typically $V_{CC}/2$, it can be useful for driving TTL inputs where level shifting is desirable in order to interface with CMOS devices.

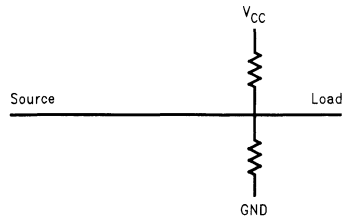
AC parallel terminations work well for applications where the increase in bus delays caused by series terminations are undesirable. The effects of AC parallel terminations are similar to the effects of standard parallel terminations. The major difference is that the capacitor blocks any DC current path and helps to reduce power consumption.

Thevenin terminations are not generally recommended due to their power consumption. Like parallel terminations, a DC path to ground is created by the terminating resistors. The power consumption of a Thevenin termination, though, will generally be independent of the signal duty cycle. Thevenin terminations are more applicable for driving CMOS inputs because they do not bias the output levels as paralleled terminations do. It should be noted that output lines with Thevenin terminations should not be left floating since this will cause the undriven input levels to float between V_{CC} and ground, increasing power consumption.



TL/DD/12857–57

FIGURE 36a. AC Parallel Termination



TL/DD/12857–58

FIGURE 36b. Thevenin Termination

8.6 Ground Bounce

Observing either one of the following rules is sufficient to avoid running into any of the problems associated with ground bounce:

- First, use caution when driving asynchronous TTL-level inputs from CMOS outputs. Ground bounce glitches may cause spurious inputs that will alter the state of non-clocked logic.
- Second, caution when running control lines (set, reset, load, clock, chip select) which are glitch-sensitive through the same devices that drive data or address lines.

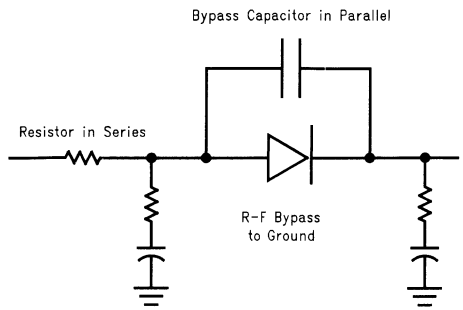
When it is not possible to avoid the above conditions, there are simple precautions available which can minimize ground bounce noise. These are:

- Choose package outputs that are as close to the ground pin as possible to drive asynchronous TTL-level inputs.
- Use the lowest V_{CC} possible or separate the power supplies.
- Use board design practices which reduce any additive noise sources, such as crosstalk, reflections, etc.

8.7 Components

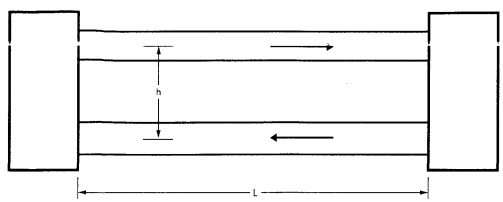
The interference effect by rectifier diodes, typically found in power supply sections of PCBs, can be minimized by one or more of the following measures:

- Placing a bypass capacitor in parallel with each rectifier diode
- Placing a resistor in series with each rectifier diode
- Placing an R-F bypass capacitor to ground from one or both sides of each rectifier diode
- Operating the rectifier diodes well below their rated current capability



TL/DD/12857–59

FIGURE 37a. RF Filtering of Diode



TL/DD/12857–60

FIGURE 37b. Connectors and Cables

8.8 Cables and Connectors

Several options are available to reduce EMI from a typical ribbon cable used to interconnect pieces of equipment. These include:

- Reduce spacing between conductors (h in *Figure 37*) by reducing the size of wires used and reducing the insulation thickness. (This could have a negative impact on crosstalk within the cable.)
- Join alternate signal returns together at the connectors at each end of the cable.
- Twist parallel wire pairs in ribbon cables.
- Shield ribbon cable with metal foil cover (superior to braid).
- Replace discrete ribbon cable with stripline flexprint cable.

In the case of joining alternate signal returns, wire N is carrying the signal current, I_N, whereas its mates, N − 1 and N + 1 wires are each carrying one half of the return currents, $I_N - 1$ and $I_N + 1$, respectively. Thus, radiation from pair N and N − 1 is out of phase with radiation from pair N and N + 1 and will tend to cancel. In practice, however, the net radiation is reduced by 20 dB–30 dB with 30 dB being a good default value.

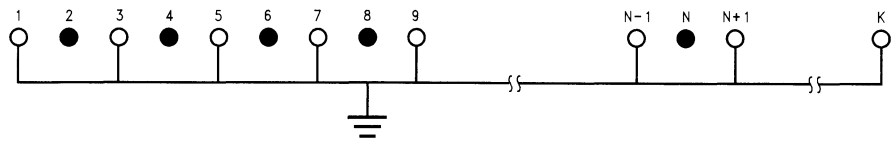
The opposite of this is to conserve signal returns by only using one, or two, wires to service N data lines in a ribbon cable. For data lines farther from the return line, the differential mode radiation becomes so great that this cable tends to maximize EMI radiation. Another disadvantage of this approach is poor impedance control in the resulting transmission line. This could result in distortion of pulses and causes reflections, especially for high-speed logic, and common return impedance noise in this single ground wire.

Ideally, connectors should have negligible resistance for obvious reasons other than EMI control. They should provide for foolproof alignment to minimize the possibility of contact damage over time and use which would increase the resistance and be prone to vibration and shock. The required force should be adequate to provide good mating between contacts which will insure low resistance, but should be minimized to limit likelihood of damage. Connectors should mate with little friction to minimize the effects of continual disconnections and connections increasing the contact resistance with use as the contacts wear out. A contamination free design should be used to avoid corrosion and oxidation increasing resistance and susceptibility to shock and vibration causing intermittent contact.

8.9 Special Considerations with Development Tools

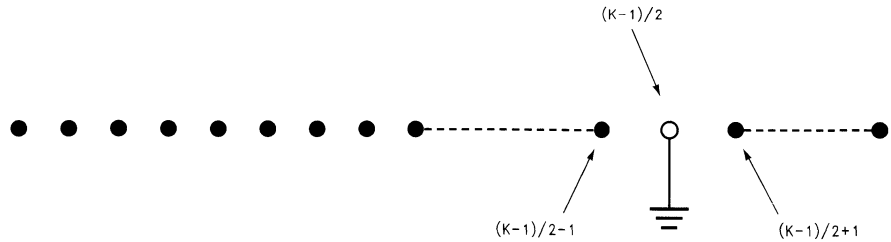
The following set of guidelines have been compiled from the experiences of the applications team at Embedded Technologies Division at National Semiconductor. They should be considered additional techniques and guidelines to be followed concurrently with the standard ones already presented. Some are general and some may be specific to development systems use.

Ground bounce prevention and minimization techniques presented in this paper should be strictly adhered to when using '373 type transparent latches on any controller's external address/data bus. Multiple simultaneously switching outputs could produce ground bounce significant enough to



TL/DD/12857-61

FIGURE 38a. Alternating Signal Return Minimizes Radiation



TL/DD/12857-62

FIGURE 38b. Single Signal Return Maximizes Radiation

cause false latching. Observe good EMI planning by locating the latches as close to the controller as possible. The use of multi-layer PCBs with good ground planes and following appropriate layout techniques is also essential, especially if emulation will be done at frequencies above 10 MHz. With the foregoing discussions about "antenna farms", radiated noise and ideal connector characteristics, it becomes obvious that wire-wrap boards and the use of IC sockets is absolutely out of the question. The concern here is not so much EMI affecting the outside world but EMI strangling the operation of the module itself.

The inputs to the buffers in a '244 type octal buffer package are placed adjacent or side-by-side outputs of other buffers in the package. This configuration would tend to maximize the crosstalk or noise coupling from the inputs to the outputs. On the other hand, the buffer inputs in a '544 type package are on one side of the package and the outputs are on the other. The use of these package types in high speed designs can facilitate board layout to help reduce the effects of crosstalk.

Use extra heavy ground wires between emulator and target board. Rely on the ground returns in the emulator cable for reduction of differential-mode noise radiated from the cable but heavy-duty help is required for reducing power line impedance in the integrated development system.

Unused controller inputs, most importantly any external interrupts and RDY, must be tied to V_{CC} (or GND) directly or through a pull-up (or pull-down) resistor. This not only tends to reduce power consumption, but will avoid noise problems triggering an unwanted action.

In order to reduce the effects of noise generated by high speed signal changes, a sort of Frequency Management technique might be applied. If possible, develop application hardware and software at a slower crystal operating frequency. If ringing, crosstalk, or other combination of radiated and conducted noise problems exists, the result may be to move the problem from one point in the affected signal waveform to a different point. Thus, apparent "noise glitches" that caused a latch to erroneously trigger when the input data was still changing, may now come at a time when they are non-destructive such as at a point when the input data is now stable.

Some applications require driving the controller clock input, CKI, with an external signal. Most emulator tools are all clocked using a crystal network with the controller so that the generation of the system timing is contained on the tool itself. Consequently, there is no connection between the emulator cable connector on the tool and the CKI pin at the controller. However, when the emulator cable is now inserted into the target board, the target board's clock signal traveling along the cable couples noise onto adjacent signal lines causing symptoms pointing to an apparent failure of the emulator tool. The recommendation is to disable the clock drive to the CKI pin at the controller pad on the target board whenever the emulator tool is connected. The emulator tools supply the system clock so there is no need for the clock on the target and signal crosstalk on the emulator cable can be greatly reduced with minimal implementation. If one insists that the emulator tool and the target be synchronous, then bring the clock signal from the target to the emulator tool external to the emulator cable via twisted wire pair or coax cable. Remove the clock drive connection to CKI at the target to prevent the signal from entering the cable. Finally, remove crystal components on the emulator tool to prevent problems with the signal.

Connecting boards and modules together to make a totally unique system in which EMC was practiced is necessary to ensure little problem with the environment. But, connecting an emulator tool makes it an entirely new and unique system, both in physical and electrical properties. Treat the emulator tool as part of the system during the design phase and development phase.

9.0 SUMMARY

The design and construction of an electromagnetically compatible PCB does not necessarily require a big change in current practices. On the contrary, the implementation of EMC principles during the design process can fit in with the ongoing design. When EMC is designed into the board, the requirements to shield circuitry, cables, and enclosures, as well as other costly eleventh hour surprises, will be drastically reduced or even eliminated. Without EMC in the design stage, production can be held up and the cost of the project increases.

REFERENCES

1. Alkinson, Kenn-Osburn, John-White, Donald, *Taming EMI in Microprocessor Systems*, **IEEE Spectrum**, December 1985, pp 30-37
2. Balakrishnan, R.V., *Reducing Noise on Microcomputer Buses*, **National Semiconductor Application Note 337**, May 1983
3. Brewer, Ron W., *Test Methodology to Determine Levels of Conducted and Radiated Emissions from Computer Systems*, **EMC Technology**, July 1982, p 10
4. Engineering Staff, Don White Consultants, Inc., *The role of Cables & Connectors in Control of EMI*, **EMC Technology**, July 1982, pp 16-26
5. Fairchild Semiconductor, *Design Considerations*, **Fairchild Advanced CMOS Technology Logic Data Book**, 1987, pp 4-3 to 4-12
6. Fairchild Semiconductor, *Understanding and Minimizing Ground Bounce*, **Application Note DU-6**, August 1987
7. Interference Control Technology, *Introduction to EMI/RFI/EMC*, **Course Notebook on Electromagnetic Compatibility**, August 1988
8. Violette, Michael F. and J.L., *EMI Control in the Design and Layout of Printed Circuit Boards*, **EMC Technology**, March-April 1986, pp 19-32
9. Violette, Michael F., *Curing Electromagnetic Interference*, **Radio-Electronics**, November 1985, pp 50-56
10. Wakeman, Larry, *High-speed-CMOS Designs Address Noise and I/O Levels*, **National Semiconductor Application Note 375**, September 1984
11. White, Donald R.J., *EMI Control Methods and Techniques*. **Electromagnetic Interference and Compatibility—Vol 3**, copyright 1988 by Interference Control Technologies

Section 8

Appendices/ Physical Dimensions

Section 8 Contents

Surface Mount 8-3
PLCC Packaging 8-23
Physical Dimensions 8-27
Bookshelf
Distributors
Worldwide Sales Offices

National Semiconductor

Surface Mount

Cost pressures today are forcing many electronics manufacturers to automate their production lines. Surface mount technology plays a key role in this cost-savings trend because:

1. The mounting of devices on the PC board surface eliminates the expense of drilling holes;
2. The use of pick-and-place machines to assemble the PC boards greatly reduces labor costs;
3. The lighter and more compact assembled products resulting from the smaller dimensions of surface mount packages mean lower material costs.

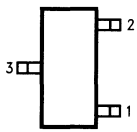
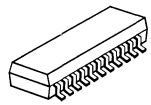
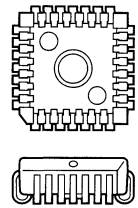
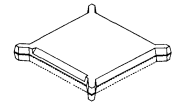
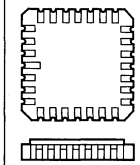
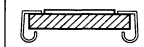
Production processes now permit both surface mount and insertion mount components to be assembled on the same PC board.

SURFACE MOUNT PACKAGING AT NATIONAL

To help our customers take advantage of this new technology, National has developed a line of surface mount packages. Ranging in lead counts from 3 to 360, the package offerings are summarized in Table I.

Lead center spacing keeps shrinking with each new generation of surface mount package. Traditional packages (e.g., DIPs) have a 100 mil lead center spacing. Surface mount packages currently in production (e.g., SOT, SOIC, PCC, LCC, LDCC) have a 50 mil lead center spacing. Surface mount packages in production release (e.g., PQFP) have a 25 mil lead center spacing. Surface mount packages in development (e.g., TAPEPAK®) will have a lead center spacing of only 12–20 mils.

TABLE I. Surface Mount Packages from National

Package Type	Small Outline Transistor (SOT)	Small Outline IC (SOIC)	Plastic Chip Carrier (PCC)	Plastic Quad Flat Pack (PQFP)	Leadless Chip Carrier (LCC) (LDCC)	Leaded Chip Carrier
						
Package Material	Plastic	Plastic	Plastic	Plastic	Ceramic	Ceramic
Lead Bend	Gull Wing	Gull Wing	J-Bend	Gull Wing	—	Gull Wing
Lead Center Spacing	50 Mils	50 Mils	50 Mils	25 Mils	50 Mils	50 Mils
Tape & Reel Option	Yes	Yes	Yes	tbd	No	No
Lead Counts	SOT-23 High Profile SOT-23 Low Profile	SO-8(*) SO-14(*) SO-14 Wide(*) SO-16(*) SO-16 Wide(*) SO-20(*) SO-24(*)	PCC-20(*) PCC-28(*) PCC-44(*) PCC-68 PCC-84 PCC-124	PQFP-84 PQFP-100 PQFP-132 PQFP-196(*) PQFP-244	LCC-18 LCC-20(*) LCC-28 LCC-32 LCC-44 (*) LCC-48 LCC-52 LCC-68 LCC-84 LCC-124	LDCC-44 LDCC-68 LDCC-84 LDCC-124

*In production (or planned) for linear products.

8

LINEAR PRODUCTS IN SURFACE MOUNT

Linear functions available in surface mount include:

- Op amps
- Comparators
- Regulators
- References
- Data conversion
- Industrial
- Consumer
- Automotive

A complete list of linear part numbers in surface mount is presented in Table III. Refer to the datasheet in the appropriate chapter of this databook for a complete description of the device. In addition, National is continually expanding the list of devices offered in surface mount. If the functions you need do not appear in Table III, contact the sales office or distributor branch nearest you for additional information.

Automated manufacturers can improve their cost savings by using Tape-and-Reel for surface mount devices. Simplified handling results because hundreds-to-thousands of semiconductors are carried on a single Tape-and-Reel pack (see ordering and shipping information—printed later in this section—for a comparison of devices/reel vs. devices/rail for those surface mount package types being used for linear products). With this higher device count per reel (when compared with less than a 100 devices per rail), pick-and-place machines have to be re-loaded less frequently and lower labor costs result.

With Tape-and-Reel, manufacturers save twice—once from using surface mount technology for automated PC board assembly and again from less device handling during shipment and machine set-up.

BOARD CONVERSION

Besides new designs, many manufacturers are converting existing printed circuit board designs to surface mount. The resulting PCB will be smaller, lighter and less expensive to manufacture; but there is one caveat—be careful about the thermal dissipation capability of the surface mount package.

Because the surface mount package is smaller than the traditional dual-in-line package, the surface mount package is not capable of conducting as much heat away as the DIP (i.e., the surface mount package has a higher thermal resistance—see Table II).

The silicon for most National devices can operate up to a 150°C junction temperature (check the datasheet for the rare exception). Like the DIP, the surface mount package can actually withstand an ambient temperature of up to 125°C (although a commercial temperature range device will only be specified for a max ambient temperature of 70°C and an industrial temperature range device will only be specified for a max ambient temperature of 85°C). See AN-336, "Understanding Integrated Circuit Package Power Capabilities", (reprinted in the appendix of each linear databook volume) for more information.

TABLE II: Surface Mount Package Thermal Resistance Range*

Package	Thermal Resistance** (θ_{jA}, °C/W)
SO-8	120–175
SO-14	100–140
SO-14 Wide	70–110
SO-16	90–130
SO-16 Wide	70–100
SO-20	60–90
SO-24	55–85
PCC-20	70–100
PCC-28	60–90
PCC-44	40–60

*Actual thermal resistance for a particular device depends on die size. Refer to the datasheet for the actual θ_{jA} value.

**Test conditions: PCB mount (FR4 material), still air (room temperature), copper traces (150 × 20 × 10 mils).

Given a max junction temperature of 150°C and a maximum allowed ambient temperature, the surface mount device will be able to dissipate less power than the DIP device. This factor must be taken into account for new designs.

For board conversion, the DIP and surface mount devices would have to dissipate the same power. This means the surface mount circuit would have a lower maximum allowable ambient temperature than the DIP circuit. For DIP circuits where the maximum ambient temperature required is substantially lower than the maximum ambient temperature allowed, there may be enough margin for safe operation of the surface mount circuit with its lower maximum allowable ambient temperature. But where the maximum ambient temperature required of the DIP current is close to the maximum allowable ambient temperature, the lower maximum ambient temperature allowed for the surface mount circuit may fall below the maximum ambient temperature required. The circuit designer must be aware of this potential pitfall so that an appropriate work-around can be found to keep the surface mount package from being thermally overstressed in the application.

SURFACE MOUNT LITERATURE

National has published extensive literature on the subject of surface mount packaging. Engineers from packaging, quality, reliability, and surface mount applications have pooled their experience to provide you with practical hands-on knowledge about the construction and use of surface mount packages.

The applications note AN-450 "Surface Mounting Methods and their Effect on Product Reliability" is referenced on each SMD datasheet. In addition, "Wave Soldering of Surface Mount Components" is reprinted in this section for your information.

TABLE III. Linear Surface Mount Current Device Listing

Amplifiers and Comparators

Part Number	Part Number
LF347WM LF351M LF451CM LF353M	LM392M LM393M LM741CM LM1458M LM2901M
LF355M LF356M LF357M LF444CWM	LM2902M LM2903M LM2904M LM2924M LM3403M
LM10CWM LM10CLWM LM308M LM308AM LM310M	LM4250M LM324M LM339M LM365WM LM607CM
LM311M LM318M LM319M LM324M	LMC669BCWM LMC669CCWM
LM339M LM346M LM348M LM358M LM359M	LF441CM

Regulators and References

Part Number	Part Number
LM317LM LF3334M	LM2931M-5.0 LM3524M LM78L05ACM LM78L12ACM LM78L15ACM
LM336M-2.5 LF336BM-2.5 LM336M-5.0 LM336BM-5.0 LM337LM	LM79L05ACM LM79L12ACM LM79L15ACM LP2951ACM LP2951CM
LM385M LM385M-1.2	
LM385BM-1.2 LM385M-2.5 LM385BM-2.5 LM723CM LM2931CM	

Data Acquisition Circuits

Part Number	Part Number
ADC0802LCV ADC0802LCWM ADC0804LCV ADC0804LCWM ADC0808CCV ADC0809CCV	ADC1025BCV ADC1025CCV DAC0800LCM DAC0801LCM DAC0802LCM DAC0806LCM DAC0807LCM
ADC0811BCV ADC0811CCV ADC0819BCV ADC0819CCV ADC0820BCV ADC0820CCV	DAC0808LCM DAC0830LCWM DAC0830LCV DAC0832LCWM DAC0832LCV
ADC0838BCV ADC0838CCV ADC0841BCV ADC0841CCV ADC0848BCV ADC0848CCV ADC1005BCV ADC1005CCV	

Industrial Functions

Part Number	Part Number
AH5012CM LF13331M LF13509M LF13333M LM555CM	LM13600M LM13700M LMC555CM LM567CM MF4CWM-50
LM556CM LM567CM LM1496M LM2917M	MF4CWM-100 MF6CWM-50 MF10CCWM MF6CWM-100 MF5CWM
LM3046M LM3086M LM3146M	

Commercial and Automotive

Part Number	Part Number
LM386M-1 LM592M LM831M LM832M LM833M	LM1837M LM1851M LM1863M LM1865M LM1870M
LM837M LM838M LM1131CM	LM1894M LM1964V
	LM2893M LM3361AM LM1881M

Hybrids

Part Number	Part Number
LH0002E	LH0032E
LH4002E	LH0033E

A FINAL WORD

National is a world leader in the design and manufacture of surface mount components.

Because of design innovations such as perforated copper leadframes, our small outline package is as reliable as our DIP—the laws of physics would have meant that a straight "junior copy" of the DIP would have resulted in an "S.O." package of lower reliability. You benefit from this equivalence of reliability. In addition, our ongoing vigilance at each step of the production process assures that the reliability we designed in stays in so that only devices of the highest quality and reliability are shipped to your factory.

Our surface mount applications lab at our headquarters site in Santa Clara, California continues to research (and publish) methods to make it even easier for you to use surface mount technology. Your problems are our problems.

When you think "Surface Mount"—think "National"!

Ordering and Shipping Information

When you order a surface mount semiconductor, it will be in one of the several available surface mount package types. Specifying the Tape-and-Reel method of shipment means that you will receive your devices in the following quantities per Tape-and-Reel pack: SMD devices can also be supplied in conventional conductive rails.

Package	Package Designator	Max/Rail	Per Reel*
SO-8	M	100	2500
SO-14	M	50	2500
SO-14 Wide	WM	50	1000
SO-16	M	50	2500
SO-16 Wide	WM	50	1000
SO-20	M	40	1000
SO-24	M	30	1000
PCL-20	V	50	1000
PCL-28	V	40	1000
PCL-44	V	25	500
PQFP-196	VF	TBD	—
TP-40	TP	100	TBD
LCC-20	E	50	—
LCC-44	E	25	—

*Incremental ordering quantities. (National Semiconductor reserves the right to provide a smaller quantity of devices per Tape-and-Reel pack to preserve lot or date code integrity. See example below.)

Example: You order 5,000 LM324M ICs shipped in Tape-and-Reel.

- Case 1: All 5,000 devices have the same date code
  - You receive 2 SO-14 (Narrow) Tape-and-Reel packs, each having 2500 LM324M ICs
- Case 2: 3,000 devices have date code A and 2,000 devices have date code B
  - You receive 3 SO-14 (Narrow) Tape-and-Reel packs as follows:

    Pack #1 has 2,500 LM324M ICs with date code A

    Pack #2 has 500 LM324M ICs with date code A

    Pack #3 has 2,000 LM324M ICs with date code B

Short-Form Procurement Specification

TAPE FORMAT

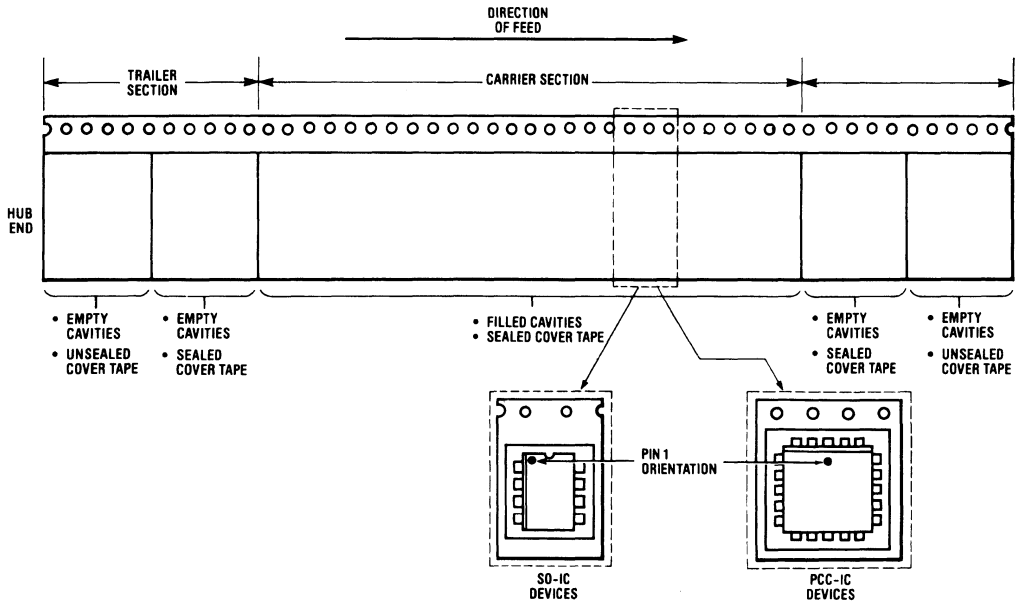
→ Direction of Feed

	Trailer (Hub End)*		Carrier*	Leader (Start End)*	
	Empty Cavities, min (Unsealed Cover Tape)	Empty Cavities, min (Sealed Cover Tape)	Filled Cavities (Sealed Cover Tape)	Empty Cavities, min (Sealed Cover Tape)	Empty Cavities, min (Unsealed Cover Tape)
Small Outline IC					
SO-8 (Narrow)	2	2	2500	5	5
SO-14 (Narrow)	2	2	2500	5	5
SO-14 (Wide)	2	2	1000	5	5
SO-16 (Narrow)	2	2	2500	5	5
SO-16 (Wide)	2	2	1000	5	5
SO-20 (Wide)	2	2	1000	5	5
SO-24 (Wide)	2	2	1000	5	5
Plastic Chip Carrier IC					
PCC-20	2	2	1000	5	5
PCC-28	2	2	750	5	5
PCC-44	2	2	500	5	5

*The following diagram identifies these sections of the tape and Pin #1 device orientation.

Short-Form Procurement Specification (Continued)

DEVICE ORIENTATION

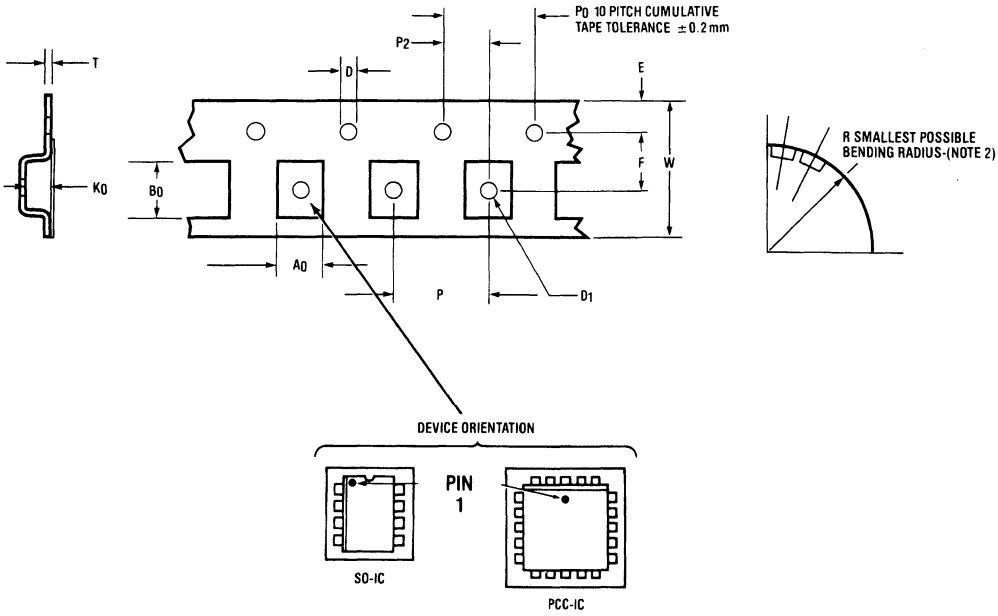


TL/DD/11325–7

MATERIALS

- Cavity Tape: Conductive PVC (less than 10^5 Ohms/Sq)
- Cover Tape: Polyester
  - (1) Conductive cover available
- Reel:
  - (1) Solid 80 pt fibreboard (standard)
  - (2) Conductive fibreboard available
  - (3) Conductive plastic (PVC) available

TAPE DIMENSIONS (24 Millimeter Tape or Less)



TL/DD/11325–8

Short-Form Procurement Specification (Continued)

	W	P	F	E	P_2	P_0	D	T	A_0	B_0	K_0	D_1	R
Small Outline IC													
SO-8 (Narrow)	12±.30	8.0±.10	5.5±.05	1.75±.10	2.0±.05	4.0±.10	1.55±.05	.30±.10	6.4±.10	5.2±.10	2.1±.10	1.55±.05	30
SO-14 (Narrow)	16±.30	8.0±.10	7.5±.10	1.75±.10	2.0±.05	4.0±.10	1.55±.05	.30±.10	6.5±.10	9.0±.10	2.1±.10	1.55±.05	40
SO-14 (Wide)	16±.30	12.0±.10	7.5±.10	1.75±.10	2.0±.05	4.0±.10	1.55±.05	.30±.10	10.9±.10	9.5±.10	3.0±.10	1.55±.05	40
SO-16 (Narrow)	16±.30	8.0±.10	7.5±.10	1.75±.10	2.0±.05	4.0±.10	1.55±.05	.30±.10	6.5±.10	10.3±.10	2.1±.10	1.55±.05	40
SO-16 (Wide)	16±.30	12.0±.10	7.5±.10	1.75±.10	2.0±.05	4.0±.10	1.55±.05	.30±.10	10.9±.10	10.76±.10	3.0±.10	1.55±.05	40
SO-20 (Wide)	24±.30	12.0±.10	11.5±.10	1.75±.10	2.0±.05	4.0±.10	1.55±.05	.30±.10	10.9±.10	13.3±.10	3.0±.10	2.05±.05	50
SO-24 (Wide)	24±.30	12.0±.10	11.5±.10	1.75±.10	2.0±.05	4.0±.10	1.55±.05	.30±.10	10.9±.10	15.85±.10	3.0±.10	2.05±.05	50
Plastic Chip Carrier IC													
PCC-20	16±.30	12.0±.10	7.5±.10	1.75±.10	2.0±.05	4.0±.10	1.55±.05	.30±.10	9.3±.10	9.3±.10	4.9±.10	1.55±.05	40
PCC-28	24±.30	16.0±.10	11.5±.10	1.75±.10	2.0±.05	4.0±.10	1.55±.05	.30±.10	13.0±.10	13.0±.10	4.9±.10	2.05±.05	50

Note 1: A_0, B_0 and K_0 dimensions are measured 0.3 mm above the inside wall of the cavity bottom.

Note 2: Tape with components shall pass around a mandril radius R without damage.

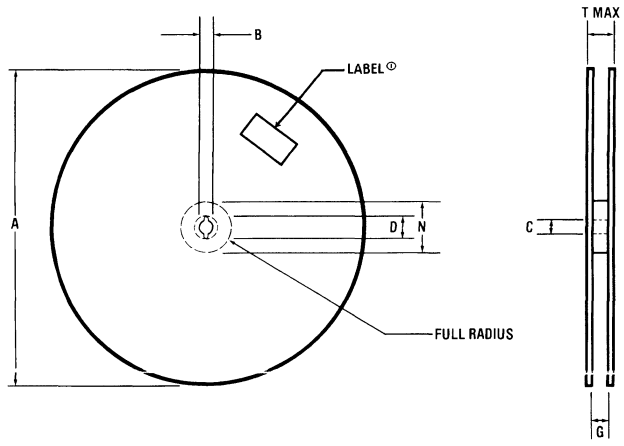
Note 3: Cavity tape material shall be PVC conductive (less than 10^5 Ohms/Sq).

Note 4: Cover tape material shall be polyester (30–65 grams peel-back force).

Note 5: D_1 Dimension is centered within cavity.

Note 6: All dimensions are in millimeters.

REEL DIMENSIONS



TL/DD/11325–9

START™* Surface Mount Tape and Reel

Short-Form Procurement Specifications (Continued)

		A (Max)	B (Min)	C	D (Min)	N (Min)	G	T (Max)
12 mm Tape	SO-8 (Narrow)	(13.00) (330)	.059 1.5	.512 ± .002 13 ± 0.05	.795 20.2	1.969 50	$0.488^{+.078}_{-.000}$ 12.4^{+2}_{-0}	.724 18.4
16 mm Tape	SO-14 (Narrow) SO-14 (Wide) SO-16 (Narrow) SO-16 (Wide) PCC-20	(13.00) (330)	.059 1.5	.512 ± .002 13 ± 0.05	.795 20.2	1.969 50	$0.646^{+.078}_{-.000}$ 16.4^{+2}_{-0}	.882 22.4
24 mm Tape	SO-20 (Wide) SO-24 (Wide) PCC-28	(13.00) (330)	.059 1.5	.512 ± .002 13 ± 0.05	.795 20.2	1.969 50	$0.960^{+.078}_{-.000}$ 24.4^{+2}_{-0}	1.197 30.4
32 mm Tape	PCC-44	(13.00) (330)	.059 1.5	.512 ± .002 13 ± 0.05	.795 20.2	1.969 50	$1.276^{+.078}_{-.000}$ 32.4^{+2}_{-0}	1.512 38.4

Units: Inches / Millimeters

Material: Paperboard (Non-Flaking)

LABEL

Human and Machine Readable Label is provided on reel. A variable (C.P.I) density code 39 is available. NSC STD label (7.6 C.P.I.)

FIELD

Lot Number

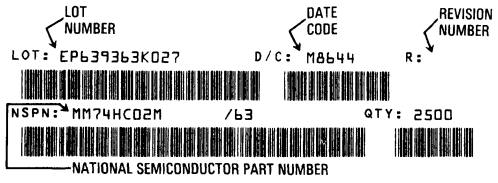
Date Code

Revision Level

National Part No. I.D.

Qty.

EXAMPLE



TL/DD/11325–10

Fields are separated by at least one blank space.

Future Tape-and-Reel packs will also include a smaller-size bar code label (high-density code 39) at the beginning of the tape. (This tape label is not available on current production.)

National Semiconductor will also offer additional labels containing information per your specific specification.

Wave Soldering of Surface Mount Components

ABSTRACT

In facing the upcoming surge of "surface mount technology", many manufacturers of printed circuit boards have taken steps to convert some portions of their boards to this new process. However, as the availability of surface mount components is still limited, may have taken to mixing the lead-inserted standard dual-in-line packages (DIPs) with the surface mounted devices (SMDs). Furthermore, to take advantage of using both sides of the board, surface-mounted components are generally adhered to the bottom side of the board while the top side is reserved for the conventional lead-inserted packages. If processed through a wave solder machine, the semiconductor components are now subjected to extra thermal stresses (now that the components are totally immersed into the molten solder).

A discussion of the effect of wave soldering on the reliability of plastic semiconductor packages follows. This is intended to highlight the limitations which should be understood in the use of wave soldering of surface mounted components.

ROLE OF WAVE-SOLDERING IN APPLICATION OF SMDs

The generally acceptable methods of soldering SMDs are vapor phase reflow soldering and IR reflow soldering, both requiring application of solder paste on PW boards prior to placement of the components. However, sentiment still exists for retaining the use of the old wave-soldering machine.

Wave Soldering of Surface Mount Components (Continued)

The reasons being:

1) Most PC Board Assembly houses already possess wave soldering equipment. Switching to another technology such as vapor phase soldering requires substantial investment in equipment and people.
2) Due to the limited number of devices that are surface mount components, it is necessary to mix both lead inserted components and surface mount components on the same board.
3) Some components such as relays and switches are made of materials which would not be able to survive the temperature exposure in a vapor phase or IR furnace.

PW BOARD ASSEMBLY PROCEDURES

There are two considerations in which through-hole ICs may be combined with surface mount components on the PW Board:

a) Whether to mount ICs on one or both sides of the board.
b) The sequence of soldering using Vapor Phase, IR or Wave Soldering singly or combination of two or more methods.

The various processes that may be employed are:

A) Wave Solder before Vapor/IR reflow solder.
   1. Components on the same side of PW Board.
      Lead insert standard DIPS onto PW Board Wave solder (conventional)
      Wash and lead trim
      Dispense solder paste on SMD pads
      Pick and place SMDs onto PW Board
      Bake
      Vapor phase/IR reflow
      Clean
   2. Components on opposite side of PW Board.
      Lead insert standard DIPs onto PW Board
      Wave Solder (conventional)
      Clean and lead trim
      Invert PW Board
      Dispense solder paste on SMD pads
      Dispense drop of adhesive on SMD sites (optional for smaller components)
      Pick and place SMDs onto board
      Bake/Cure
      Invert board to rest on raised fixture
      Vapor/IR reflow soldering
      Clean

B) Vapor/IR reflow solder then Wave Solder.
   1. Components on the same side of PW Board.
      Solder paste screened on SMD side of Printed Wire Board
      Pick and place SMDs
      Bake
      Vapor/IR reflow
      Lead insert on same side as SMDs
      Wave solder
      Clean and trim underside of PCB

C) Vapor/IR reflow only.
   1. Components on the same side of PW Board.
      Trim and form standard DIPs in "gull wing" configuration
      Solder paste screened on PW Board
      Pick and place SMDs and DIPs
      Bake
      Vapor/IR reflow
      Clean
   2. Components on opposite sides of PW Board.
      Solder paste screened on SMD-side of Printed Wire Board
      Adhesive dispensed at central location of each component
      Pick and place SMDs
      Bake
      Solder paste screened on all pads on DIP-side or alternatively apply solder rings (performs) on leads
      Lead insert DIPs
      Vapor/IR reflow
      Clean and lead trim

D) Wave Soldering Only
   1. Components on opposite sides of PW Board.
      Adhesive dispense on SMD side of PW Board
      Pick and place SMDs
      Cure adhesive
      Lead insert top side with DIPs
      Wave solder with SMDs down and into solder bath
      Clean and lead trim

All of the above assembly procedures can be divided into three categories for I.C. Reliability considerations:

1) Components are subjected to both a vapor phase/IR heat cycle then followed by a wave-solder heat cycle or vice versa.
2) Components are subjected to only a vapor phase/IR heat cycle.
3) Components are subjected to wave-soldering only and SMDs are subjected to heat by immersion into a solder pot.

Of these three categories, the last is the most severe regarding heat treatment to a semiconductor device. However, note that semiconductor molded packages generally possess a coating of solder on their leads as a final finish for solderability and protection of base leadframe material. Most semiconductor manufacturers solder-plate the component leads, while others perform hot solder dip. In the latter case the packages may be subjected to total immersion into a hot solder bath under controlled conditions (manual operation) or be partially immersed while in a 'pallet' where automatic wave or DIP soldering processes are used. It is, therefore, possible to subject SMDs to solder heat under certain conditions and not cause catastrophic failures.

Wave Soldering of Surface Mount Components (Continued)

THERMAL CHARACTERISTICS OF MOLDED INTEGRATED CIRCUITS

Since Plastic DIPs and SMDs are encapsulated with a thermoset epoxy, the thermal characteristics of the material generally correspond to a TMA (Thermo-Mechanical Analysis) graph. The critical parameters are (a) its Linear thermal expansion characteristics and (b) its glass transition temperature after the epoxy has been fully cured. A typical TMA graph is illustrated in *Figure 1*. Note that the epoxy changes to a higher thermal expansion once it is subjected to temperatures exceeding its glass transition temperature. Metals (as used on lead frames, for example) do not have this characteristic and generally will have a consistent Linear thermal expansion over the same temperature range.

In any good reliable plastic package, the choice of lead frame material should be such to match its thermal expansion properties to that of the encapsulating epoxy. In the event that there is a mismatch between the two, stresses can build up at the interface of the epoxy and metal. There now exists a tendency for the epoxy to separate from the metal lead frame in a manner similar to that observed on bi-metallic thermal range.

In most cases when the packages are kept at temperatures below their glass transition, there is a small possibility of separation at the expoxy-metal interface. Howerver, if the package is subjected to temprature above its glass-transition temperature, the epoxy will begin to expand much faster than the metal and the probability of separation is greatly increased.

CONVENTIONAL WAVE-SOLDERING

Most wave-soldering operations occur at temperatures between 240–260°C. Conventional epoxies for encapsulation have glass-transition temperature between 140–170°C. An I.C. directly exposed to these temperatures risks its long term functionality due to epoxy/metal separation.

Fortunately, there are factors that can reduce that element of risk:

1) The PW board has a certain amount of heat-sink effect and tends to shield the components from the temperature of the solder (if they were placed on the top side of the board). In actual measurements, DIPs achieve a temperature between 120–150°C in a 5-second pass over the solder. This accounts for the fact that DIPs mounted in the conventional manner are reliable.
2) In conventional soldering, only the tip of each lead in a DIP would experience the solder temperature because the epoxy and die are standing above the PW board and out of the solder bath.

EFFECT ON PACKAGE PERFORMANCE BY EPOXY-METAL SEPARATION

In wave soldering, it is necessary to use fluxes to assist the solderability of the components and PW boards. Some facilities may even process the boards and components through some form of acid cleaning prior to the soldering temperature. If separation occurs, the flux residues and acid residues (which may be present owing to inadequate cleaning) will be forced into the package mainly by capillary action as the residues move away from the solder heat source. Once the package is cooled, these contaminants are now trapped within the package and are available to diffuse with moisture from the epoxy over time. It should be noted that electrical tests performed immediately after soldering generally will give no indication of this potential problem. In any case, the end result will be corrosion of the chip metallization over time and premature failure of the device in the field.

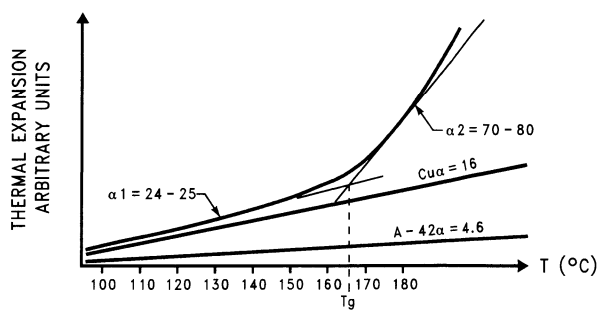
VAPOR PHASE/IR REFLOW SOLDERING

In both vapor phase and IR reflow soldering, the risk of separation between epoxy/metal can also be high. Operating temperatures are 215°C (vapor phase) or 240°C (IR) and duration may also be longer (30 sec–60 sec). On the same theoretical basis, there should also be separation. However, in both these methods, solder paste is applied to the pads of the boards; no fluxes are used. Also, the devices are not immersed into the hot solder. This reduces the possibility of solder forcing itself into the epoxy-lead frame interface. Furthermore, in the vapor phase system, the soldering environment is "oxygen-free" and considered "contaminant free". Being so, it could be visualized that as far as reliability with respect to corrosion, both of these methods are advantageous over wave soldering.

BIAS MOISTURE TEST

A bias moisture test was designed to determine the effect on package performance. In this test, the packages are pressured in a stream chamber to accelerate penetration of moisture into the package. An electrical bias is applied on the device. Should there be any contaminants trapped within the package, the moisture will quickly form an electrolyte and cause the electrodes (which are the lead fingers), the gold wire and the aluminum bond-pads of the silicon device to corrode. The aluminum bond-pads, being the weakest link of the system, will generally be the first to fail.

This proprietary accelerated bias/moisture pressure-test is significant in relation to the life test condition at 85°C and



TL/DD/11325–11

FIGURE 1. Thermal Expansion and Glass Transition Temperature

Wave Soldering of Surface Mount Components (Continued)

85% relative humidity. Once cycle of approximately 100 hours has been shown to be equivalent to 2000 hours in the 85/85 condition. Should the packages start to fail within the first cycle in the test, it is anticipated that the boards with these components in the harsh operating environment (85°C/85% RH) will experience corrosion and eventual electrical failures within its first 2000 hours of operation.

Whether this is significant to a circuit board manufacturer will obviously be dependent on the products being manufactured and the workmanship or reliability standards. Generally in systems with a long warranty and containing many components, it is advisable both on a reputation and cost basis to have the most reliable parts available.

TEST RESULTS

The comparison of vapor phase and wave-soldering upon the reliability of molded Small-Outline packages was performed using the bias moisture test (see Table IV). It is clearly seen that vapor phase reflow soldering gave more consistent results. Wave-soldering results were based on manual operation giving variations in soldering parameters such as temperature and duration.

TABLE IV. Vapor Phase vs. Wave Solder

1. Vapor phase (60 sec. exposure @ 215°C)
   = 9 failures/1723 samples
   = 0.5% (average over 32 sample lots)
2. Wave solder (2 sec total immersion @ 260°C)
   = 16 failures/1201 samples
   = 1.3% (average over 27 sample lots)

Package: SO-14 lead
Test: Bias moisture test 85% R.H., 85°C for 2000 hours
Device: LM324M

In Table V we examine the tolerance of the Small-Outlined (SOIC) package to varying immersion time in a hot solder pot. SO-14 lead molded packages were subjected to the bias moisture test after being treated to the various soldering conditions and repeated four (4) times. End point was an electrical test after an equivalent of 4000 hours 85/85 test. Results were compared for packages by itself against packages which were surface-mounted onto a FR-4 printed wire board.

TABLE V. Summary of Wave Solder Results (85% R.H./85°C Bias Moisture Test, 2000 hours) (# Failures/Total Tested)

	Unmounted	Mounted
Control/Vapor Phase 15 sec @ 215°C	0/114	0/84
Solder Dip 2 sec @ 260°C	2/144 (1.4%)	0/85
Solder Dip 4 sec @ 260°C	—	0/83
Solder Dip 6 sec @ 260°C	13/248 (5.2%)	1/76 (1.3%)
Solder Dip 10 sec @ 260°C	14/127 (11.0%)	3/79 (3.8%)
Package: SO-14 lead Device: LM324M		

Since the package is of very small mass and experiences a rather sharp thermal shock followed by stresses created by the mismatch in expansion, the results show the package being susceptible to failures after being immersed in excess of 6 seconds in a solder pot. In the second case where the packages were mounted, the effect of severe temperature excursion was reduced. In the second case where the packages were mounted, the effect of severe temperature excursion was reduced. In any case, because of the repeated treatment, the package had failures when subjected in excess of 6 seconds immersion in hot solder. The safety margin is therefore recommended as maximum 4 seconds immersion. If packages were immersed longer than 4 seconds, there is a probable chance of finding some long term reliability failures even though the immediate electrical test data could be acceptable.

Finally, Table VI examines the bias moisture test performed on surface mount (SOIC) components manufactured by various semiconductor houses. End point was an electrical test after an equivalent of 6000 hours in a 85/85 test. Failures were analyzed and corrosion was checked for in each case to detect flaws in package integrity.

TABLE VI. U.S. Manufacturers Integrated Circuits Reliability in Various Solder Environments (# Failure/Total Tested)

Package SO-8	Vapor Phase 30 sec	Wave Solder 2 sec	Wave Solder 4 sec	Wave Solder 6 sec	Wave Solder 10 sec
Manuf A	8/30*	1/30*	0.30	12/30*	16/30*
Manuf B	2/30*	8/30*	2/30*	22/30*	20/30*
Manuf C	0/30	0/29	0/29	0/30	0/30
Manuf D	1/30*	0/30	12/30*	14/30*	2/30*
Manuf E	1/30**	0/30	0/30	0/30	0/30
Manuf F	0/30	0/30	0/30	0/30	0/30
Manuf G	0/30	0/30	0/30	0/30	0/30

*Corrosion-failures

**No Visual Defects—Non-corrosion failures

Test: Accelerated Bias Moisture Test; 85% R.H./85°C, 6000 equivalent hours.

SUMMARY

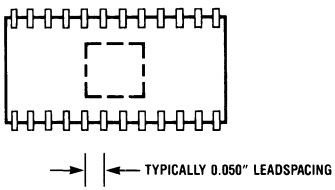
Based on the results presented, it is noted that surface-mounted components are as reliable as standard molded DIP packages. Whereas DIPs were never processed by being totally immersed in a hot solder wave during printed circuit board soldering, surface mounted components such as SOICs (Small Outline) are expected to survive a total immersion in the hot solder in order to capitalize on maximum population on boards. Being constructed from a thermoset plastic of relatively low Tg compared to the soldering temperature, the ability of the package to survive is dependent on the time of immersion and also the cleanliness of material. The results indicate that one should limit the immersion time of package in the solder wave to a maximum of 4 seconds in order to truly duplicate the reliability of a DIP. As the package size is reduced, as in a SO-8 lead, the requirement becomes even more critical. This is shown by the various manufacturers' performance. Results indicate there is room for improvement since not all survived the hot solder immersion without compromise to lower reliability.

Small Outline (SO) Package Surface Mounting Methods— Parameters and Their Effect on Product Reliability

The SO (small outline) package has been developed to meet customer demand for ever-increasing miniaturization and component density.

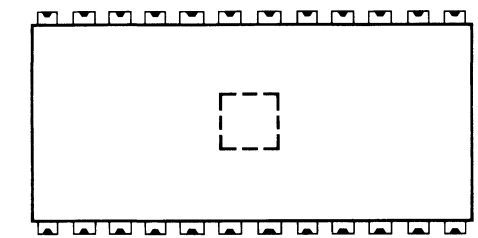
COMPONENT SIZE COMPARISON

S.O. Package



TL/DD/11325–12

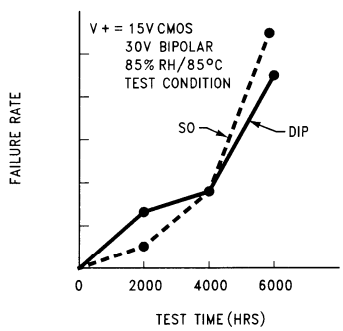
Standard DIP Package



TL/DD/11325–13

Because of its small size, reliability of the product assembled in SO packages needs to be carefully evaluated.

SO packages at National were internally qualified for production under the condition that they be of comparable reliability performance to a standard dual in line package under all accelerated environmental tests. *Figure A* is a summary of accelarated bias moisture test performance on 30V bipolar and 15V CMOS product assembled in SO and DIP (control) packages.



TL/DD/11325–14

FIGURE A

In order to achieve reliability performance comparable to DIPs—SO packages are designed and built with materials and processes that effectively compensate for their small size.

All SO packages tested on 85%RA, 85°C were assembled on PC conversion boards using vapor-phase reflow soldering. With this approach we are able to measure the effect of surface mounting methods on reliability of the process. As illustrated in *Figure A* no significant difference was detected between the long term reliability performance of surface mounted S.O. packages and the DIP control product for up to 6000 hours of accelerated 85%/85°C testing.

SURFACE-MOUNT PROCESS FLOW

The standard process flowcharts for basic surface-mount operation and mixed-lead insertion/surface-mount operations, are illustrated on the following pages.

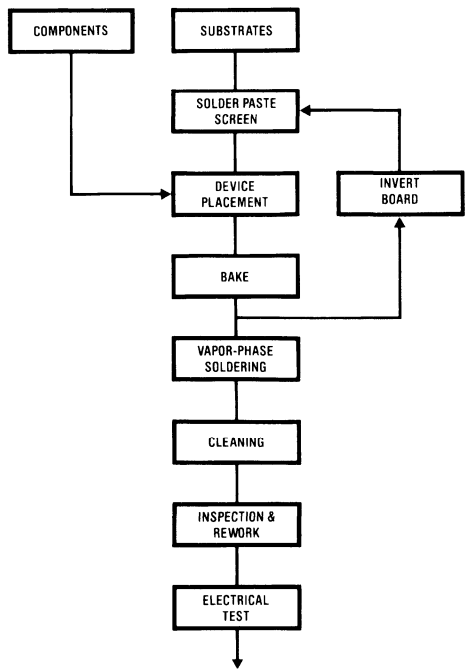
Usual variations encountered by users of SO packages are:

- Single-sided boards, surface-mounted components only.
- Single-sided boards, mixed-lead inserted and surface-mounted components.
- Double-sided boards, surface-mounted components only.
- Double-sided boards, mixed-lead inserted and surface-mounted components.

In consideration of these variations, it became necessary for users to utilize techniques involving wave soldering and adhesive applications, along with the commonly-used vapor-phase solder reflow soldering technique.

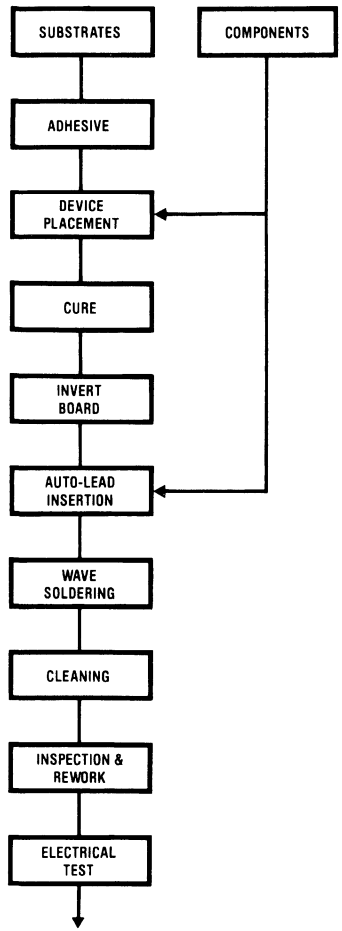
PRODUCTION FLOW

Basic Surface-Mount Production Flow



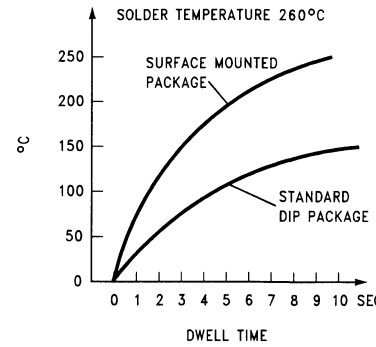
TL/DD/11325–15

Mixed Surface-Mount and Axial-Leaded Insertion Components Production Flow



TL/DD/11325–16

Thermal stress of the packages during surface-mounting processing is more severe than during standard DIP PC board mounting processes. *Figure B* illustrates package temperature versus wave soldering dwell time for surface mounted packages (components are immersed into the molten solder) and the standard DIP wave soldering process. (Only leads of the package are immersed into the molten solder).

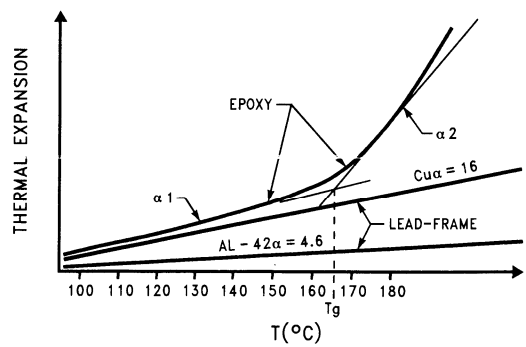


TL/DD/11325–17

FIGURE B

For an ideal package, the thermal expansion rate of the encapsulant should match that of the leadframe material in order for the package to maintain mechanical integrity during the soldering process. Unfortunately, a perfect matchup of thermal expansion rates with most presently used packaging materials is scarce. The problem lies primarily with the epoxy compound.

Normally, thermal expansion rates for epoxy encapsulant and metal lead frame materials are linear and remain fairly close at temperatures approaching 160°C, *Figure C*. At lower temperatures the difference in expansion rate of the two materials is not great enough to cause interface separation. However, when the package reaches the glass-transition temperature (T_g) of epoxy (typically 160–165°C), the thermal expansion rate of the encapsulant increases sharply, and the material undergoes a transition into a plastic state. The epoxy begins to expand at a rate three times or more greater than the metal leadframe, causing a separation at the interface.



TL/DD/11325–18

FIGURE C

When this happens during a conventional wave soldering process using flux and acid cleaners, process residues and even solder can enter the cavity created by the separation and become entrapped when the material cools. These contaminants can eventually diffuse into the interior of the package, especially in the presence of moisture. The result is die contamination, excessive leakage, and even catastrophic failure. Unfortunately, electrical tests performed immediately following soldering may not detect potential flaws.

Most soldering processes involve temperatures ranging up to 260°C, which far exceeds the glass-transition temperature of epoxy. Clearly, circuit boards containing SMD packages require tighter process controls than those used for boards populated solely by DIPs.

Figure D is a summary of accelerated bias moisture test performance on the 30V bipolar process.

Group 1 — Standard DIP package

Group 2 — SO packages vapor-phase reflow soldered on PC boards

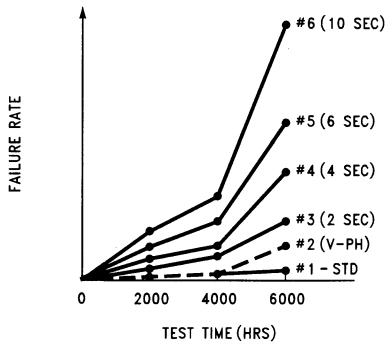
Group 3–6 SO packages wave soldered on PC boards

Group 3 — dwell time 2 seconds

4 — dwell time 4 seconds

5 — dwell time 6 seconds

6 — dwell time 10 seconds



TL/DD/11325–19

FIGURE D

It is clear based on the data presented that SO packages soldered onto PC boards with the vapor phase reflow process have the best long term bias moisture performance and this is comparable to the performance of standard DIP packages. The key advantage of reflow soldering methods is the clean environment that minimized the potential for contamination of surface mounted packages, and is preferred for the surface-mount process.

When wave soldering is used to surface mount components on the board, the dwell time of the component under molten solder should be no more than 4 seconds, preferrably under 2 seconds in order to prevent damage to the component. Non-Halide, or (organic acid) fluxes are highly recommended.

PICK AND PLACE

The choice of automatic (all generally programmable) pick-and-place machines to handle surface mounting has grown considerably, and their selection is based on individual needs and degree of sophistication.

The basic component-placement systems available are classified as:

(a) In-line placement

— Fixed placement stations

— Boards indexed under head and respective components placed

(b) Sequential placement

— Either a X-Y moving table system or a θ, X-Y moving pickup system used

—Individual components picked and placed onto boards

(c) Simultaneous placement

— Multiple pickup heads

— Whole array of components placed onto the PCB at the same time

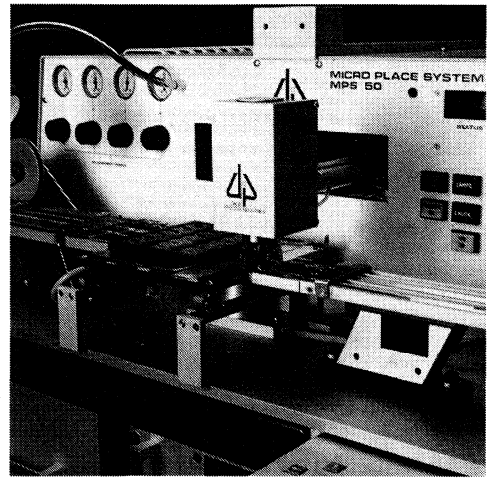
(d) Sequential/simultaneous placement

— X–Y moving table, multiple pickup heads system

— Components placed on PCB by successive or simultaneous actuation of pickup heads

The SO package is treated almost the same as surface-mount, passive components requiring correct orientation in placement on the board.

Pick and Place Action



TL/DD/11325–20

BAKE

This is recommended, despite claims made by some solder paste suppliers that this step be omitted.

The functions of this step are:

- Holds down the solder globules during subsequent reflow soldering process and prevents expulsion of small solder balls.
- Acts as an adhesive to hold the components in place during handling between placement to reflow soldering.
- Holds components in position when a double-sided surface-mounted board is held upside down going into a vapor-phase reflow soldering operation.
- Removes solvents which might otherwise contaminate other equipment.
- Initiates activator cleaning of surfaces to be soldered.
- Prevents moisture absorption.

The process is moreover very simple. The usual schedule is about 20 minutes in a 65°C–95°C (dependent on solvent system of solder paste) oven with adequate venting. Longer bake time is not recommended due to the following reasons:

- The flux will degrade and affect the characteristics of the paste.
- Solder globules will begin to oxidize and cause solderability problems.
- The paste will creep and after reflow, may leave behind residues between traces which are difficult to remove and vulnerable to electro-migration problems.

REFLOW SOLDERING

There are various methods for reflowing the solder paste, namely:

- Hot air reflow
- Infrared heating (furnaces)
- Convectional oven heating
- Vapor-phase reflow soldering
- Laser soldering

For SO applications, hot air reflow/infrared furnace may be used for low-volume production or prototype work, but vapor-phase soldering reflow is more efficient for consistency and speed. Oven heating is not recommended because of "hot spots" in the oven and uneven melting may result. Laser soldering is more for specialized applications and requires a great amount of investment.

HOT GAS REFLOW/INFRARED HEATING

A hand-held or table-mount air blower (with appropriate orifice mask) can be used.

The boards are preheated to about 100°C and then subjected to an air jet at about 260°C. This is a slow process and results may be inconsistent due to various heat-sink properties of passive components.

Use of an infrared furnace is the next step to automating the concept, except that the heating is promoted by use of IR lamps or panels. The main objection to this method is that certain materials may heat up at different rates under IR radiation and may result in damage to these components (usually sockets and connectors). This could be minimized by using far-infrared (non-focused) system.

VAPOR-PHASE REFLOW SOLDERING

Currently the most popular and consistent method, vapor-phase soldering utilizes a fluoroinert fluid with excellent heat-transfer properties to heat up components until the solder paste reflows. The maximum temperature is limited by the vapor temperature of the fluid.

The commonly used fluids (supplied by 3M Corp) are:

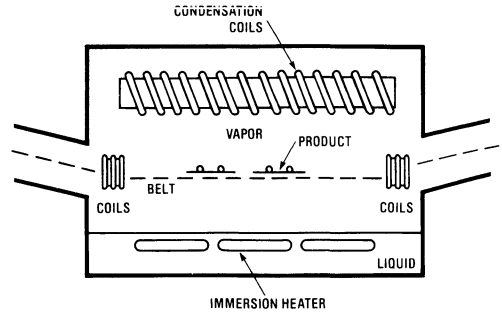
- FC-70, 215°C vapor (most applications) or FX-38
- FC-71, 253°C vapor (low-lead or tin-plate)

HTC, Concord, CA, manufactures equipment that utilizes this technique, with two options:

- Batch systems, where boards are lowered in a basket and subjected to the vapor from a tank of boiling fluid.
- In-line conveyorized systems, where boards are placed onto a continuous belt which transports them into a concealed tank where they are subjected to an environment of hot vapor.

Dwell time in the vapor is generally on the order of 15–30 seconds (depending on the mass of the boards and the loading density of boards on the belt).

In-Line Conveyorized Vapor-Phase Soldering



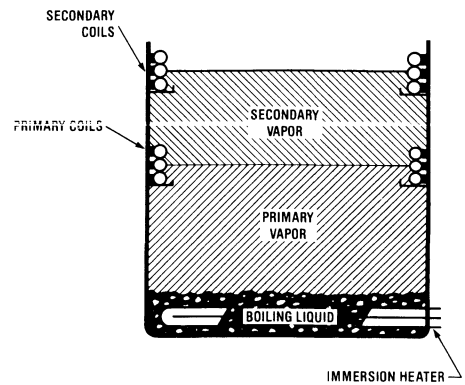
TL/DD/11325–21

The question of thermal shock is asked frequently because of the relatively sharp increase in component temperature from room temperature to 215°C. SO packages mounted on representative boards have been tested and have shown little effect on the integrity of the packages. Various packages, such as cerdips, metal cans and TO-5 cans with glass seals, have also been tested.

Vapor-Phase Furnace

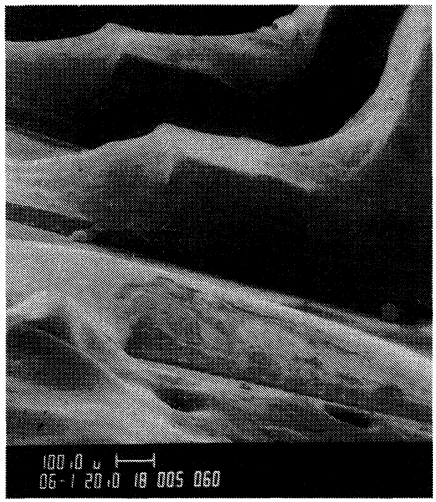
TL/DD/11325–22

Batch-Fed Production Vapor-Phase Soldering Unit



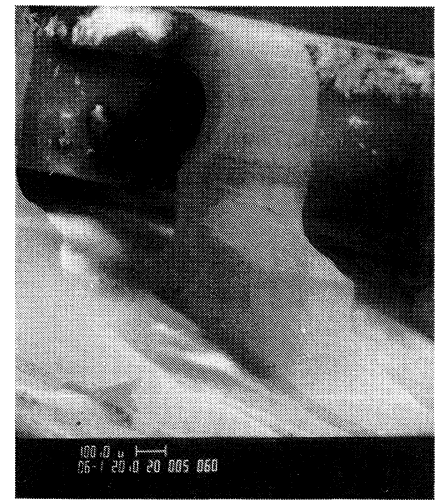
TL/DD/11325–23

Solder Joints on a SO-14 Package on PCB



TL/DD/11325–24

Solder Joints on a SO-14 Package on PCB



TL/DD/11325–25

PRINTED CIRCUIT BOARD

The SO package is molded out of clean, thermoset plastic compound and has no particular compatibility problems with most printed circuit board substrates.

The package can be reliably mounted onto substrates such as:

- G10 or FR4 glass/resin
- FR5 glass/resin systems for high-temperature applications
- Polymide boards, also high-temperature applications
- Ceramic substrates

General requirements for printed circuit boards are:

- Mounting pads should be solder-plated whenever applicable.
- Solder masks are commonly used to prevent solder bridging of fine lines during soldering.

  The mask also protects circuits from processing chemical contamination and corrosion.

  If coated over pre-tinned traces, residues may accumulate at the mask/trace interface during subsequent reflow, leading to possible reliability failures.

  Recommended application of solder resist on bare, clean traces prior to coating exposed areas with solder.

  General requirements for solder mask:

  — Good pattern resolution.

  — Complete coverage of circuit lines and resistance to flaking during soldering.

  — Adhesion should be excellent on substrate material to keep off moisture and chemicals.

  — Compatible with soldering and cleaning requirements.

SOLDER PASTE SCREEN PRINTING

With the initial choice of printed circuit lithographic design and substrate material, the first step in surface mounting is the application of solder paste.

The typical lithographic "footprints" for SO packages are illustrated below. Note that the 0.050″ lead center-center spacing is not easily managed by commercially-available air pressure, hand-held dispensers.

Using a stainless-steel, wire-mesh screen stencilled with an emulsion image of the substrate pads is by far the most common and well-tried method. The paste is forced through the screen by a V-shaped plastic squeegee in a sweeping manner onto the board placed beneath the screen.

The setup for SO packages has no special requirement from that required by other surface-mounted, passive components. Recommended working specifications are:

- Use stainless-steel, wire-mesh screens, #80 or #120, wire diameter 2.6 mils. Rule of thumb: mesh opening should be approximately 2.5–5 times larger than the average particle size of paste material.
- Use squeegee of Durometer 70.
- Experimentation with squeegee travel speed is recommended, if available on machine used.
- Use solder paste of mesh 200–325.
- Emulsion thickness of 0.005″ usually used to achieve a solder paste thickness (wet) of about 0.008″ typical.
- Mesh pattern should be 90 degrees, square grid.
- Snap-off height of screen should not exceed $\frac{1}{8}$″, to avoid damage to screens and minimize distortion.

SOLDER PASTE

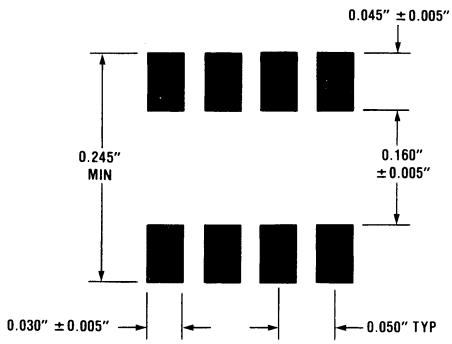
Selection of solder paste tends to be confusing, due to numerous formulations available from various manufacturers. In general, the following guidelines are sufficient to qualify a particular paste for production:

- Particle sizes (see photographs below). Mesh 325 (approximately 45 microns) should be used for general purposes, while larger (solder globules) particles are preferred for leadless components (LCC). The larger particles can easily be used for SO packages.

- Uniform particle distribution. Solder globules should be spherical in shape with uniform diameters and minimum amount of elongation (visual under 100/200 × magnification). Uneven distribution causes uneven melting and subsequent expulsion of smaller solder balls away from their proper sites.
- Composition, generally 60/40 or 63/37 Sn/Pb. Use 62/36 Sn/Pb with 2% Ag in the presence of Au on the soldering area. This formulation reduces problems of metal leaching from soldering pads.
- RMA flux system usually used.
- Use paste with aproximately 88–90% solids.

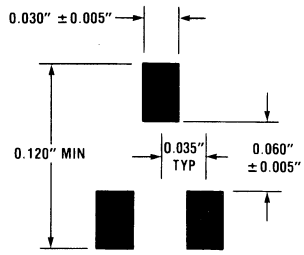
RECOMMENDED SOLDER PADS FOR SO PACKAGES

SO-8, SO-14, SO-16



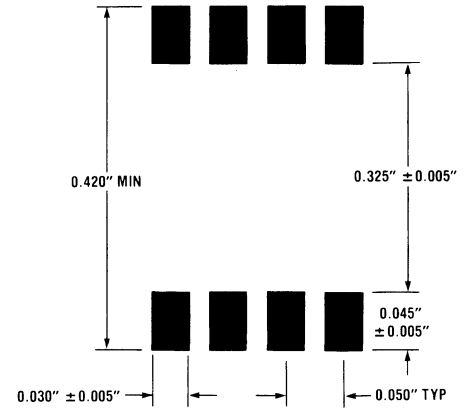
TL/DD/11325–26

SOT-23



TL/DD/11325–28

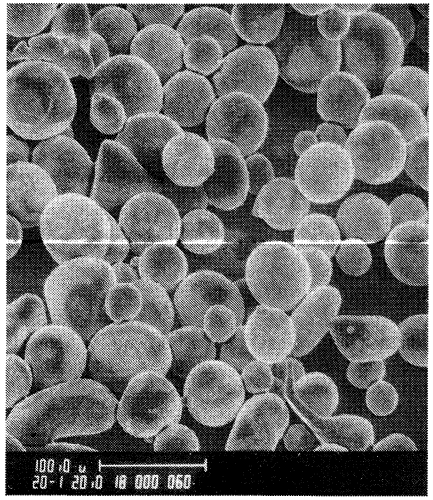
SO-16L, SO-20



TL/DD/11325–27

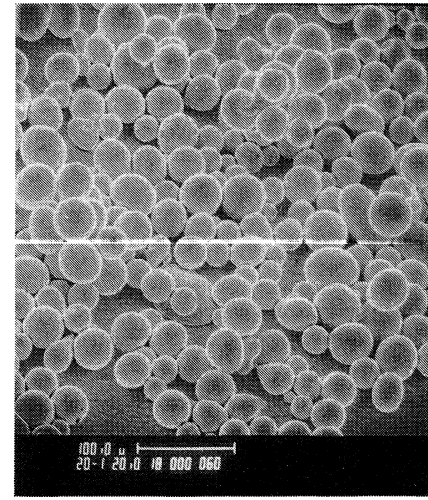
Comparison of Particle Size/Shape of Various Solder Pastes

200 × Alpha (62/36/2)



TL/DD/11325–29

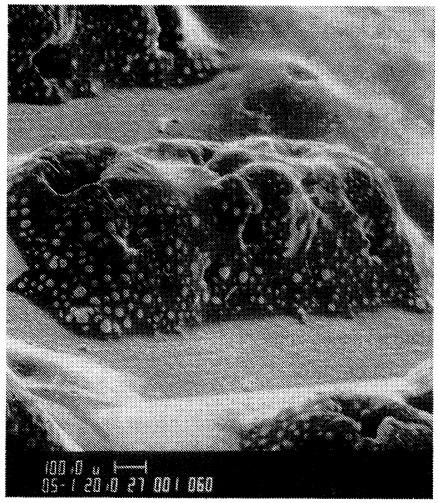
200 × Kester (63/37)



TL/DD/11325–30

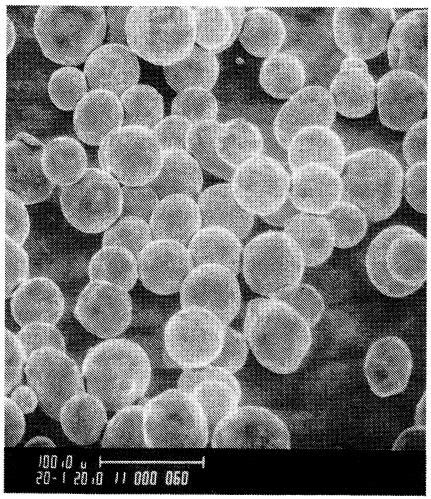
Comparison of Particle Size/Shape of Various Solder Pastes (Continued)

Solder Paste Screen on Pads



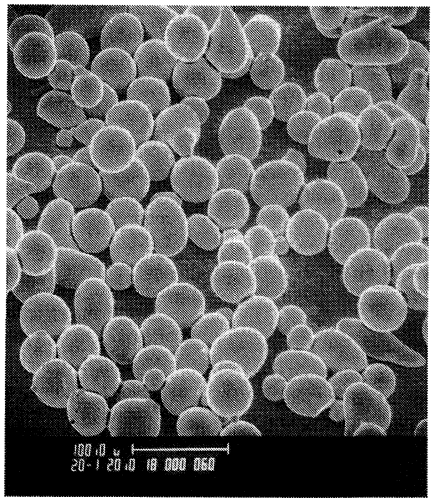
TL/DD/11325–31

200 × Fry Metal (63/37)



TL/DD/11325–32

200 ESL (63/37)



TL/DD/11325–33

CLEANING

The most critical process in surface mounting SO packages is in the cleaning cycle. The package is mounted very close to the surface of the substrate and has a tendency to collect residue left behind after reflow soldering.

Important considerations in cleaning are:

- Time between soldering and cleaning to be as short as possible. Residue should not be allowed to solidify on the substrate for long periods of time, making it difficult to dislodge.
- A low surface tension solvent (high penetration) should be employed. Solvents commercially available are:

  Freon TMS (general purpose)
  Freon TE35/TP35 (cold-dip cleaning)
  Freon TES (general purpose)

  It should also be noted that these solvents generally will leave the substrate surface hydrophobic (moisture repellent), which is desirable.

  Prelete or 1,1,1-Trichloroethane
  Kester 5120/5121
- A defluxer system which allows the workpiece to be subjected to a solvent vapor, followed by a rinse in pure solvent and a high-pressure spray lance are the basic requirments for low-volume production.
- For volume production, a conveyorized, multiple hot solvent spray/jet system is recommended.
- Rosin, being a natural occurring material, is not readily soluble in solvents, and has long been a stumbling block to the cleaning process. In recent developments, synthetic flux (SA flux), which is readily soluble in Freon TMS solvent, has been developed. This should be explored where permissible.

The dangers of an inadequate cleaning cycle are:

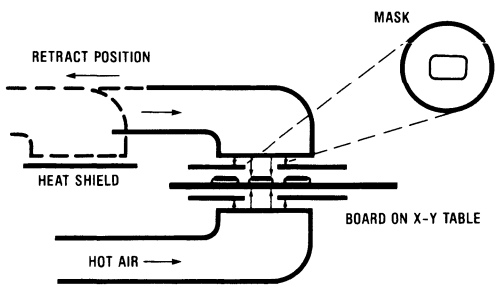
- Ion contamination, where ionic residue left on boards would cause corrosion to metallic components, affecting the performance of the board.
- Electro-migration, where ionic residue and moisture present on electrically-biased boards would cause dentritic growth between close spacing traces on the substrate, resulting in failures (shorts).

REWORK

Should there be a need to replace a component or re-align a previously disturbed component, a hot air system with appropriate orifice masking to protect surrounding components may be used.

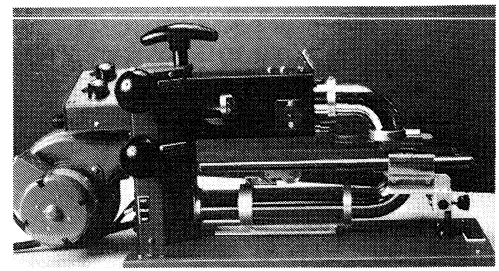
When rework is necessary in the field, specially-designed tweezers that thermally heat the component may be used to remove it from its site. The replacement can be fluxed at the lead tips or, if necessary, solder paste can be dispensed onto the pads using a varimeter. After being placed into position, the solder is reflowed by a hot-air jet or even a standard soldering iron.

Hot-Air Solder Rework Station



TL/DD/11325–34

Hot-Air Rework Machine



TL/DD/11325–35

WAVE SOLDERING

In a case where lead insertions are made on the same board as surface-mounted components, there is a need to include a wave-soldering operation in the process flow.

Two options are used:

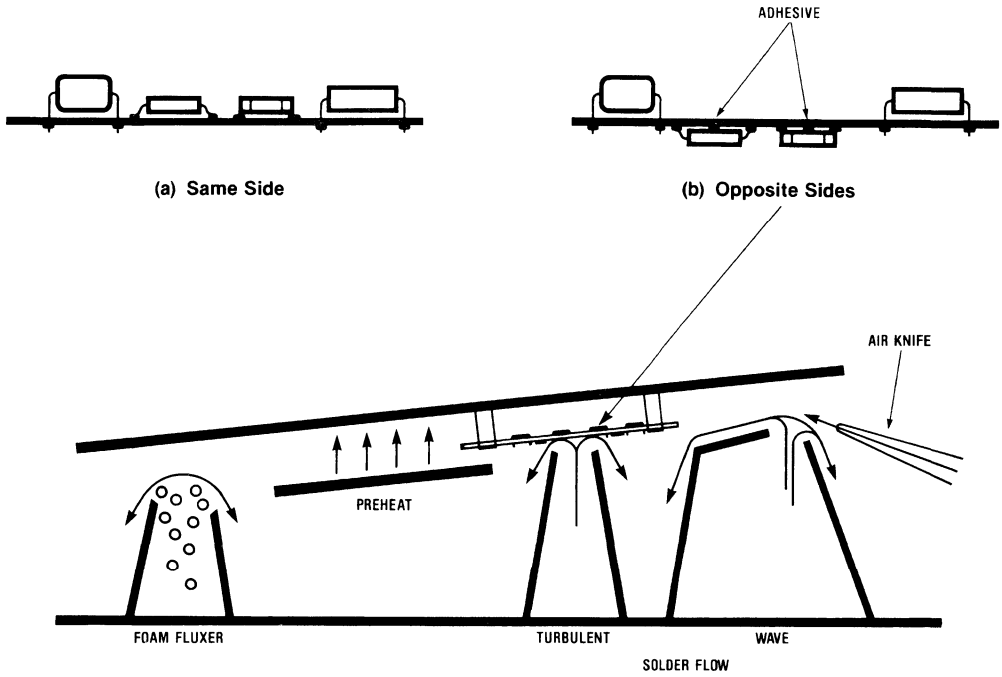
- Surface mounted components are placed and vapor phase reflowed before auto-insertion of remaining components. The board is carried over a standard wave-solder system and the underside of the board (only lead-inserted leads) soldered.
- Surface-mounted components are placed in position, but no solder paste is used. Instead, a drop of adhesive about 5 mils maximum in height with diameter not exceeding 25% width of the package is used to hold down the package. The adhesive is cured and then proceeded to auto-insertion on the reverse side of the board (surface-mounted side facing down). The assembly is then passed over a "dual wave" soldering system. Note that the surface-mounted components are immersed into the molten solder.

Lead trimming will pose a problem after soldering in the latter case, unless the leads of the insertion components are pre-trimmed or the board specially designed to localize certain areas for easy access to the trim blade.

The controls required for wave soldering are:

- Solder temperature to be 240–260°C. The dwell time of components under molten solder to be short (preferably kept under 2 seconds), to prevent damage to most components and semiconductor devices.
- RMA (Rosin Mildly Activated) flux or more aggressive OA (Organic Acid) flux are applied by either dipping or foam fluxing on boards prior to preheat and soldering. Cleaning procedures are also more difficult (aqueous, when OA flux is used), as the entire board has been treated by flux (unlike solder paste, which is more or less localized). Non-halide OA fluxes are highly recommended.
- Preheating of boards is essential to reduce thermal shock on components. Board should reach a temperature of about 100°C just before entering the solder wave.
- Due to the closer lead spacings (0.050" vs 0.100" for dual-in-line packages), bridging of traces by solder could occur. The reduced clearance between packages also causes "shadowing" of some areas, resulting in poor solder coverage. This is minimized by dual-wave solder systems.

Mixed Surface Mount and Lead Insertion



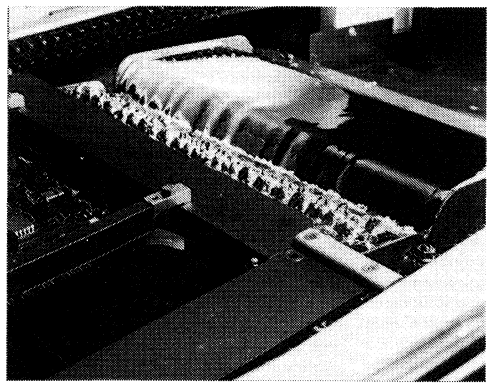
TL/DD/11325–36

A typical dual-wave system is illustrated below, showing the various stages employed. The first wave typically is in turbulence and given a transverse motion (across the motion of the board). This covers areas where "shadowing" occurs. A second wave (usually a broad wave) then proceeds to perform the standard soldering. The departing edge from the solder is such to reduce "icicles," and is still further reduced by an air knife placed close to the final soldering step. This air knife will blow off excess solder (still in the fluid stage) which would otherwise cause shorts (bridging) and solder bumps.

AQUEOUS CLEANING

- For volume production, a conveyorized system is often used with a heated recirculating spray wash (water temperature 130°C), a final spray rinse (water temperature 45–55°C), and a hot (120°C) air/air-knife drying section.
- For low-volume production, the above cleaning can be done manually, using several water rinses/tanks. Fast-drying solvents, like alcohols that are miscible with water, are sometimes used to help the drying process.
- Neutralizing agents which will react with the corrosive materials in the flux and produce material readily soluble in water may be used; the choice depends on the type of flux used.
- Final rinse water should be free from chemicals which are introduced to maintain the biological purity of the water. These materials, mostly chlorides, are detrimental to the assemblies cleaned because they introduce a fresh amount of ionizable material.

Dual Wave



TL/DD/11325–37

CONFORMAL COATING

Conformal coating is recommended for high-reliability PCBs to provide insulation resistance, as well as protection against contamination and degradation by moisture.

Requirements:

- Complete coating over components and solder joints.
- Thixotropic material which will not flow under the packages or fill voids, otherwise will introduce stress on solder joints on expansion.
- Compatibility and possess excellent adhesion with PCB material/components.
- Silicones are recommended where permissible in application.

SMD Lab Support

FUNCTIONS

Demonstration—Introduce first-time users to surface-mounting processes.

Service—Investigate problems experienced by users on surface mounting.

Reliability Builds—Assemble surface-mounted units for reliability data acquisition.

Techniques—Develop techniques for handling different materials and processes in surface mounting.

Equipment—In conjunction with equipment manufacturers, develop customized equipments to handle high density, new technology packages developed by National.

In-House Expertise—Availability of in-house expertise on semiconductor research/development to assist users on packaging queries.

National Semiconductor

Plastic Leaded Chip Carrier (PLCC) Packaging

General Description

The Plastic Leaded Chip Carrier (PLCC) is a miniaturized low cost semiconductor package designed to replace the Plastic Dual-In-Line Package (P-DIP) in high density applications. The PLCC utilizes a smaller lead-to-lead spacing—0.050" versus 0.100" - and leads on all four sides to achieve a significant footprint reduction over the P-DIP. The rolled under J-bend leadform separates this package style from other plastic quad packages with flat or gull wing lead forms. As with virtually all packages of 0.050" or less lead spacing, the PLCC requires surface mounting to printed circuit boards as opposed to the more conventional thru-hole mounting of the P-DIP.

History

The Plastic Leaded Chip Carrier with J-bend leadform was first introduced in 1976 as a premolded plastic package. The premolded version has yet to become popular but the quad format with J-Bend leads has been adapted to traditional post molded packaging technology (the same technology used to manufacture the P-DIP). In 1980 National Semiconductor developed a post molded version of the PLCC. The J-bend leadform allowed them to adopt the footprint connection pattern already registered with JEDEC for the leadless chip carrier (LCC). In 1981 a task force was organized within JEDEC to develop a PLCC registration for package I/O counts of 20, 28, 44, 52, 68, 84, 100, and 124. A registered outline was completed in 1984 (JEDEC Outline MO-047) after many changes and improvements over the original proposals. This first PLCC registration covers square packages with an equal number of leads on all sides. A second registration, MO-052, was completed in 1985 for rectangular packages with I/O counts of 18, 22, 28 and 32.

Since 1980 many additional semiconductor manufacturers and packaging subcontractors have developed PLCC capability. There are now well over 20 sources with the number growing steadily.

Surface Mounting

Surface mounting refers to component attachment whereby the component leads or pads rest on the surface of the PCB instead of the traditional approach of inserting the leads into through-holes which go through the board. With surface mounting there are solder pads on the PCB which align with the leads or pads on the component. The resulting solder joint forms both the mechanical and electrical connection.

ADVANTAGES

The primary reason for surface mounting is to allow leads to be placed closer together than the 0.100" standard for DIPs with through-hole mounting. Through-hole mounting on smaller than 0.100" spacing is difficult to achieve in production and generally avoided. The move to 0.050" lead spacing offered with the current generation of surface mounted components, along with a switch from a dual-in-line format to a quad format, has achieved a threefold increase in component mounting density. A need to achieve greater density is a major driving force in today's marketplace.

MANUFACTURING TECHNIQUES

Learning how to surface mount components to printed circuit boards requires the user to become educated in new assembly processes not typically associated with through-hole insertion/wave soldering assembly methods.

Surface mounting involves three basic process steps:

1) Application of solder or solder paste to the printed circuit board.
2) Positioning of the component onto the printed circuit board
3) Reflowing of the solder or solder paste.

As with any process, there are many details involved to achieve acceptable throughput and acceptable quality. National Semiconductor offers a surface mounting guide which deals with the specifics of successful surface mounting. We encourage the user to review this document and to contact us if further information on surface mounting is desired.

Benefits of the PLCC

There are four principle advantages offered the user by switching from P-DIP to PLCC. These four advantages are outlined below as follows:

1. Increased Density—
   - — Typically 3-to-1 size reduction of printed circuit boards. See *Figure 1* for a footprint comparison between PLCC and P-DIP. This can be as high as 6-to-1 in certain applications.
   - — Surface mounting allows components to be placed on both sides of the board.
   - — Surface mount and thru-hole mount components can be placed on the same board.
   - — The large diameter thru-holes can be reduced in number, entirely eliminated, or reduced in size (if needed for via connection).
2. Increased Performance—
   - — Shorter traces on printed circuit boards.
   - — Better high frequency operation.
   - — Shorter leads in package. *Figure 2* and Table I compare PLCC and P-DIP mechanical and electrical characteristics.
3. Increased Reliability—
   - — Leads are well protected.
   - — Fewer connectors.
   - — Simplified rework.
   - — Vibration and shock resistant.
4. Reduced Cost—
   - — Fewer or smaller printed circuit boards.
   - — Less hardware.
   - — Same low cost printed circuit board material.
   - — Plastic packaging material.
   - — Reduced number of costly plated-through-holes.
   - — Fewer circuit layers.

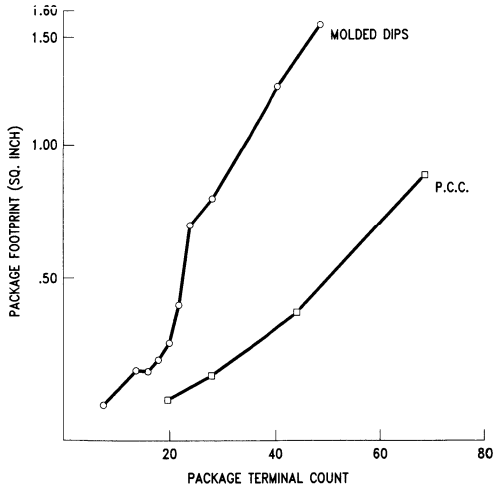


FIGURE 1. Footprint Area of PLCC vs. P-DIP

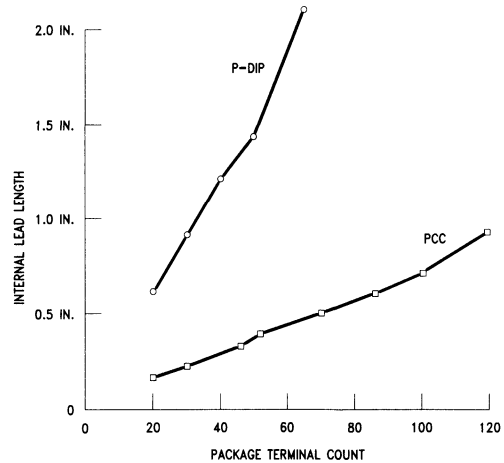


FIGURE 2. Longest Internal Lead PLCC vs. P-DIP

TABLE I. Electrical Performance of PLCC vs. P-DIP (44 I/O PLCC vs. 40 I/O P-DIP, both with Copper Leads)

Criteria	Shortest Lead		Longest Lead	
	PLCC	P-DIP	PLCC	P-DIP
Lead Resistance (Measured)	3Ω	4Ω	6Ω	7Ω
Lead-to-Lead Capacitance (Measured on Adjacent Leads)	0.1 pF	0.1 pF	0.3 pF	3.0 pF
Lead Self-Inductance (Calculated)	3.2 nH	1.4 nH	3.5 nH	19.1 nH

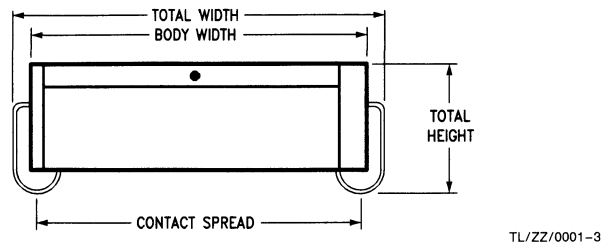


FIGURE 3. Package Outline

TABLE II. Principle Dimensions Inches/(Millimeters) (Refer to *Figure 3*)

Lead Count	Total Width		Total Height		Body Width		Contact Spread	
	Min	Max	Min	Max	Min	Max	Min	Max
20	0.385 sq. (9.779)	0.395 sq. (10.03)	0.165 sq. (4.191)	0.180 sq. (4.572)	0.345 sq. (8.763)	0.355 sq. (9.017)	0.310 sq. (7.874)	0.330 sq. (8.382)
28	0.485 sq. (12.32)	0.495 sq. (12.57)	0.165 sq. (4.191)	0.180 sq. (4.572)	0.445 sq. (11.30)	0.455 sq. (11.56)	0.410 sq. (10.41)	0.430 sq. (10.92)
44	0.685 sq. (17.40)	0.695 sq. (17.65)	0.165 sq. (4.191)	0.180 sq. (4.572)	0.645 sq. (16.38)	0.655 sq. (16.64)	0.610 sq. (15.49)	0.630 sq. (16.00)

TABLE II. Principle Dimensions Inches/(Millimeters) (Refer to *Figure 3*) (Continued)

Lead Count	Total Width		Total Height		Body Width		Contact Spread	
	Min	Max	Min	Max	Min	Max	Min	Max
68	0.985 sq. (25.02)	0.995 sq. (25.27)	0.165 sq. (4.191)	0.180 sq. (4.572)	0.945 sq. (24.00)	0.955 sq. (24.26)	0.910 sq. (23.11)	0.930 sq. (23.62)
84	1.185 sq. (30.10)	1.195 sq. (30.36)	0.165 sq. (4.191)	0.180 sq. (4.572)	1.150 sq. (29.21)	1.158 sq. (29.41)	1.110 sq. (28.20)	1.130 sq. (28.70)
124	1.685 sq. (49.13)	1.695 sq. (49.39)	0.180 sq. (4.572)	0.200 sq. (5.080)	1.650 sq. (41.91)	1.658 sq. (42.11)	1.610 sq. (40.90)	1.630 sq. (41.40)

TABLE III. Package Thermal Resistance (Deg. C/Watt, Junction-to-Ambient, Board Mount)

Lead Count	Device Size		
	1,000 Mil^2	10,000 Mil^2	100,000 Mil^2
20	102	85	67
28	95	73	55
44	54	47	40
68	44	40	38
84*	40	35	30
124*	40	35	30

*Estimated values

Package Design Criteria

Experience has taught us there are certain criteria to the PLCC design which must be followed to provide the user with the proper mechanical and thermal performance. These requirements should be carefully reviewed by the user when selecting suppliers for devices in PLCC. Some of these are covered by the JEDEC registration and some are not. These important requirements are listed in Table IV.

Reliability

National Semiconductor utilizes an assembly process for the PLCC which is similar to our P-DIP assembly process. We also utilize identical materials. This is a very important point when considering reliability. Many years of research and development have gone into steadily improving our P-DIP quality and maintaining a leadership position in plastic package reliability. All of this technology can be directly applied to the PLCC. Table V shows the results of applying this technology to the PLCC. As we make further advances in plastic package reliability, these will also be applied to the PLCC.

Sockets

There are several manufacturers currently offering sockets for the plastic chip carrier. Following is a listing of those manufacturers. The listing is divided into test/burn-in and production categories. There may be some individual sockets that will cover both requirements.

TABLE IV. Package Design Criteria

Criteria	Required to Comply with JEDEC Registration
Minimum Inside Bend Radius of Lead at Shoulder Equal or Greater than Lead Thickness—to Prevent Lead Cracking/Fatigue	Not Required
Minimum One Mil Clearance Between Lead and Plastic Body at all Points—to Provide Lead Compliancy and Prevent Shoulder Joint Cracking/Fatigue	Not Required
Copper Leads for Low Thermal Resistance	Not Required
Minimum 10 Mil Lead Thickness for Low Thermal Resistance and Good Handling Properties	Not Required
Minimum 26 Mil Lead Shoulder Width to Prevent Interlocking of Devices During Handling	Yes
Maximum 4 Mils coplanarity Across Seating Plane of all Leads	Yes

TABLE V. Reliability Test Data
(Expressed as Failures per Units Tested)

Device/Package	OPL	TMCL	TMSK	BHTL	ACLV
LM324/20 Lead	0/96	0/199	0/50	0/97	0/300
LF353/20 Lead	0/50	0/50	—	0/45	0/100
DS75451/20 Lead	0/47	—	0/50	0/93	0/179
DM875191/28 Lead	0/154	0/154	0/154	0/154	0/154
DM875181/28 Lead	0/77	0/77	0/77	0/77	0/77

OPL = Dynamic high temperature operating life at 125°C or 150°C, 1,000 hours.
TMCL = Temperature cycle, Air-to-Air, −40°C to +125°C or −65°C to +150°C, 2,000 cycles.
TMSK = Thermal shock, Liquid-to-Liquid, −65°C to +150°C, 100 cycles.
BHTL = Biased humidity temperature life, 85°C, 85% humidity, 1,000 hours.
ACLV = Autoclave, 15 psi, 121°C, 100% humidity, 1,000 hours.

Production Sockets

AMP
Harrisburg, PA
(715) 564-0100

Augat
Attleboro, MA
(617) 222-2202

Burndy
Norwalk, CT
(203) 838-4444

Methode
Rolling Meadows, IL
(312) 392-3500

Textool
Irving, TX
(214) 259-2676

Thomas & Betts
Raritan, NJ
(201) 469-4000

Test/Burn-In Sockets

Plastronics
Irving, TX
(214) 258-1906

Textool
Irving, TX
(214) 259-2676

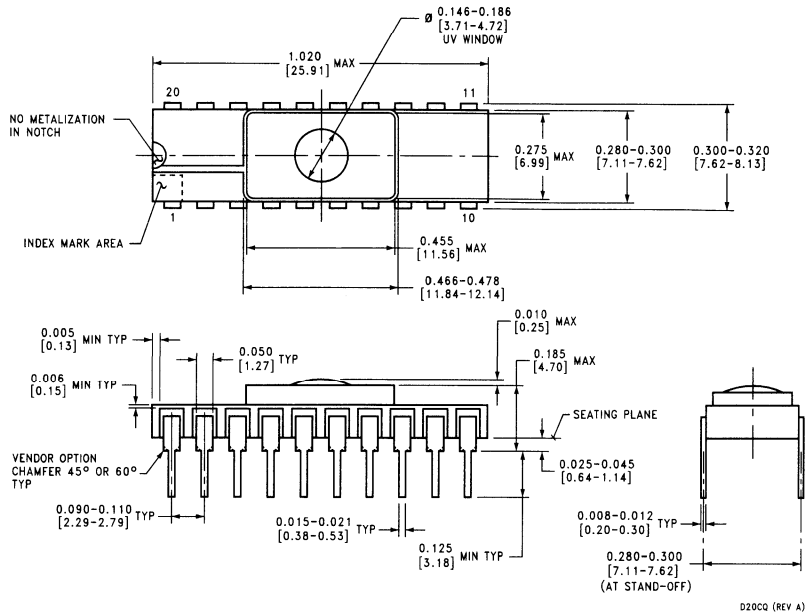
Yamaichi
c/o Nepenthe Dist.
(415) 856-9332

ADDITIONAL INFORMATION AND SERVICES

National Semiconductor offers additional Databooks which cover surface mount technology in much greater detail. We also have a surface mount laboratory to provide demonstrations and customer support, as well as technology development. Feel free to contact us about these additional resources.

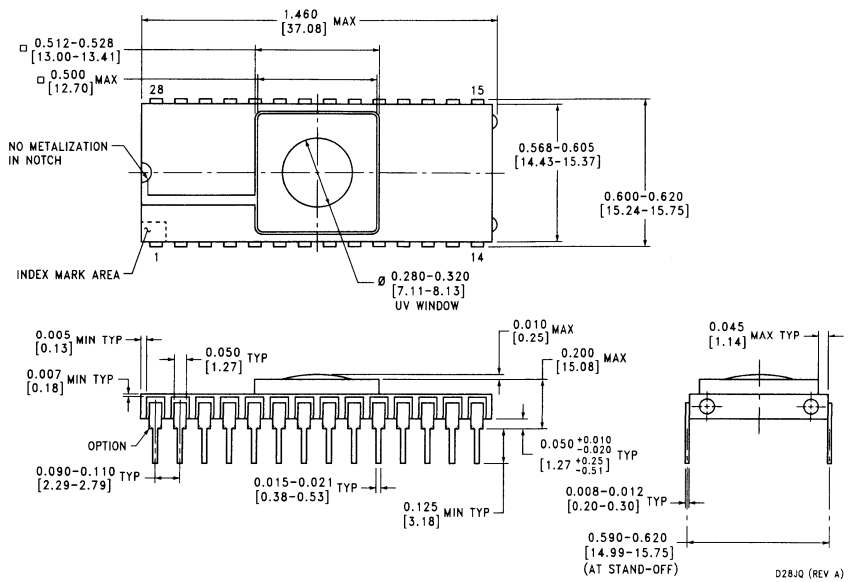
20 Lead Hermetic Dual-in-Line Package, EPROM
NS Package Number D20CQ

All dimensions are in inches [millimeters]



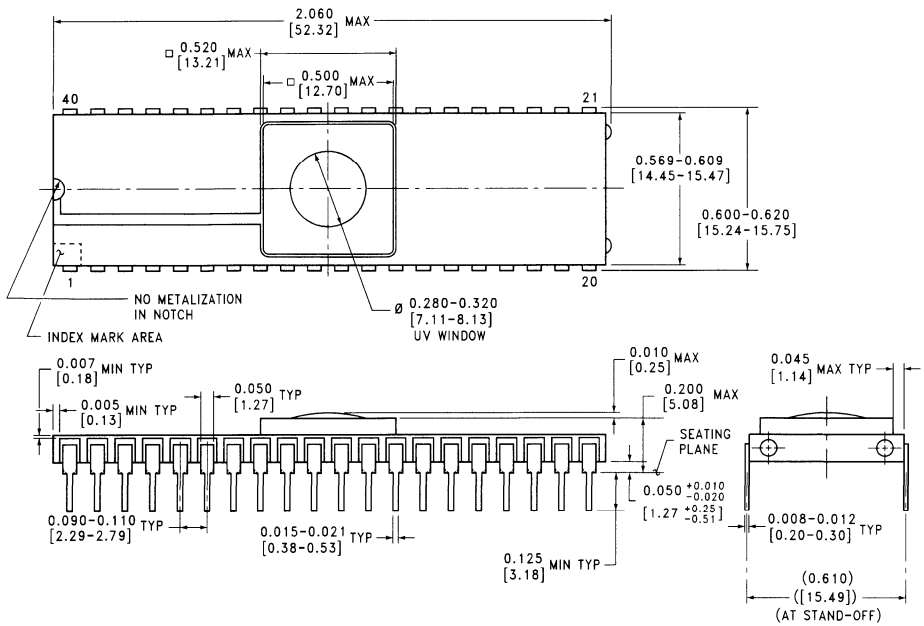
28 Lead (0.600" Centers) Hermetic Dual-in-Line Package, EPROM
NS Package Number D28JQ

All dimensions are in inches [millimeters]



40 Lead Hermetic Dual-in-Line Package, EPROM
NS Package Number D40KQ

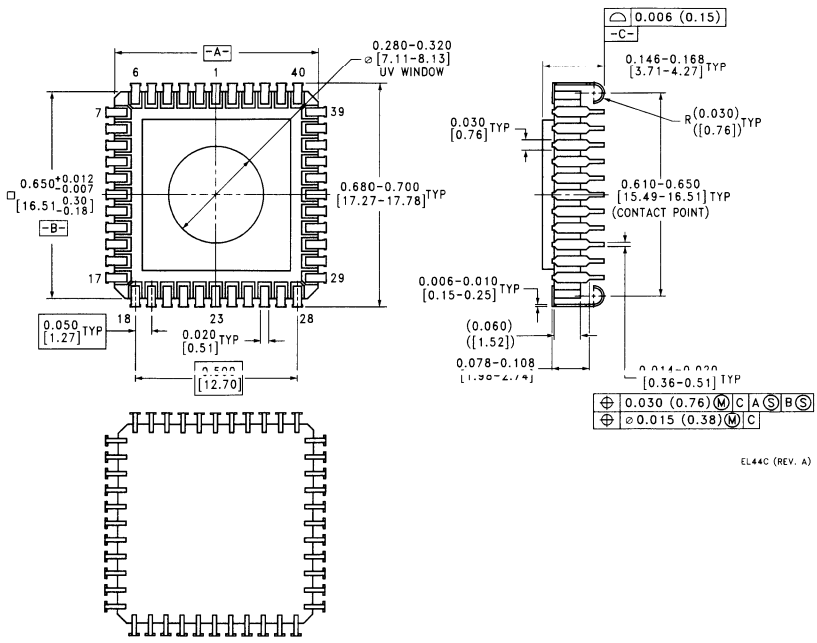
All dimensions are in inches [millimeters]



D40KQ (REV A)

44 Lead Ceramic Quad J-Bend, EPROM
NS Package Number EL44C

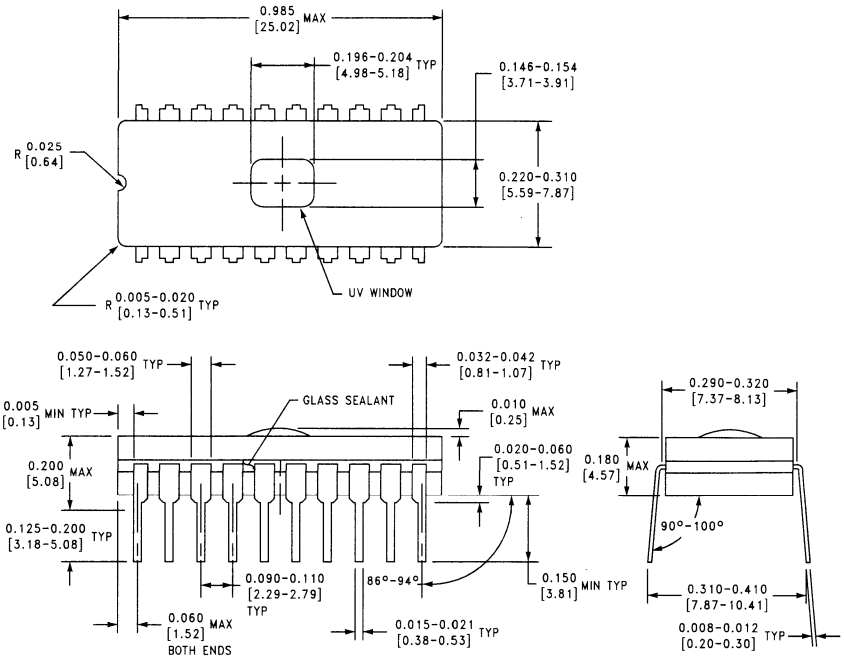
All dimensions are in inches [millimeters]



EL44C (REV. A)

20 Lead Ceramic Dual-in-Line Package, EPROM
NS Package Number J20AQ

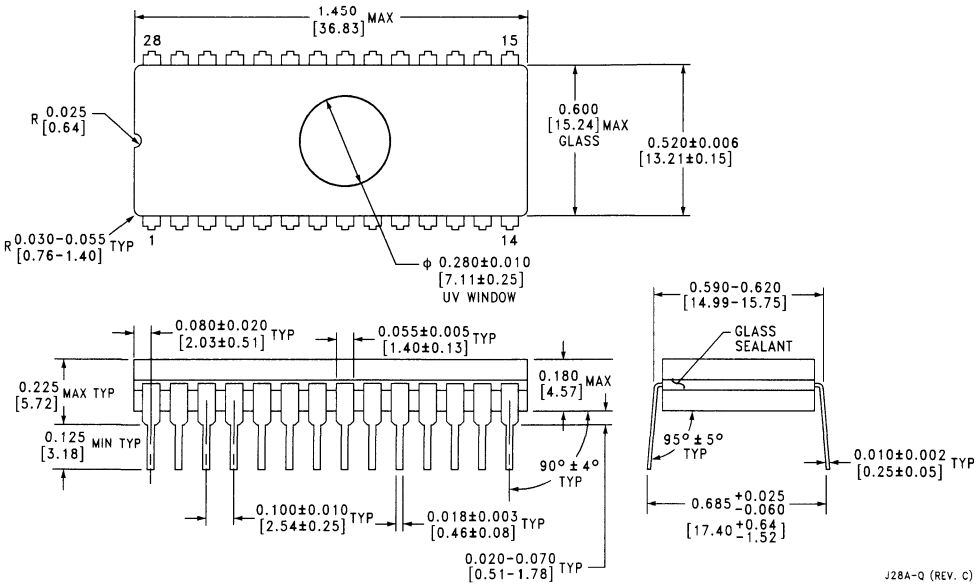
All dimensions are in inches [millimeters]



J20AQ (REV. C)

28 Lead Ceramic Dual-in-Line Package, EPROM
NS Package Number J28AQ

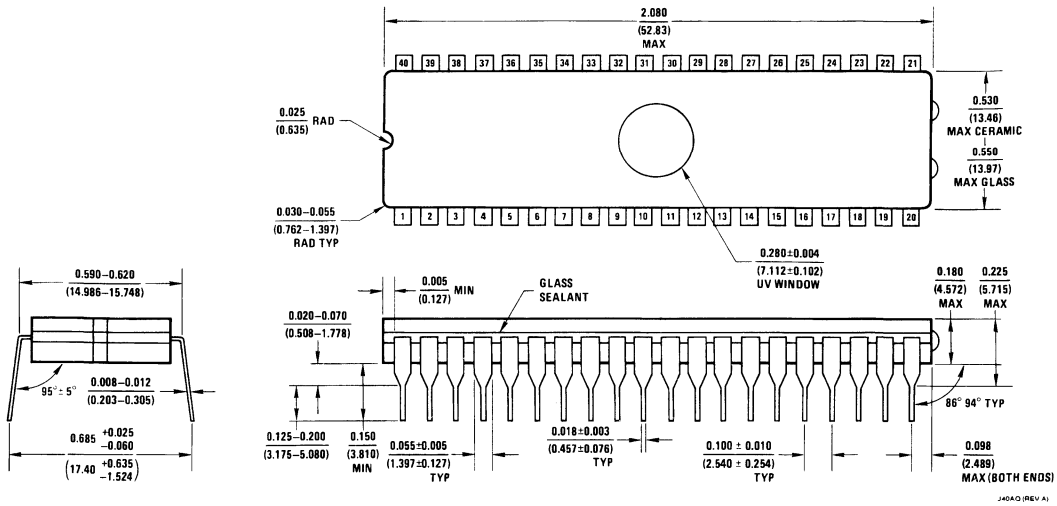
All dimensions are in inches [millimeters]



J28A-Q (REV. C)

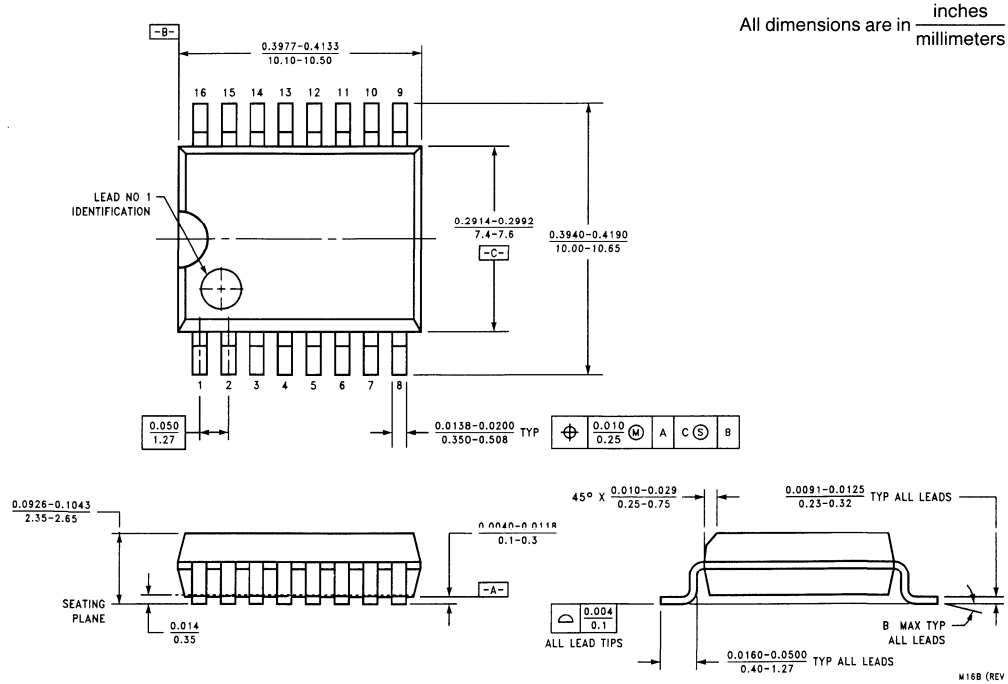
40 Lead Ceramic Dual-in-Line Package, EPROM
NS Package Number J40AQ

All dimensions are in inches (millimeters)



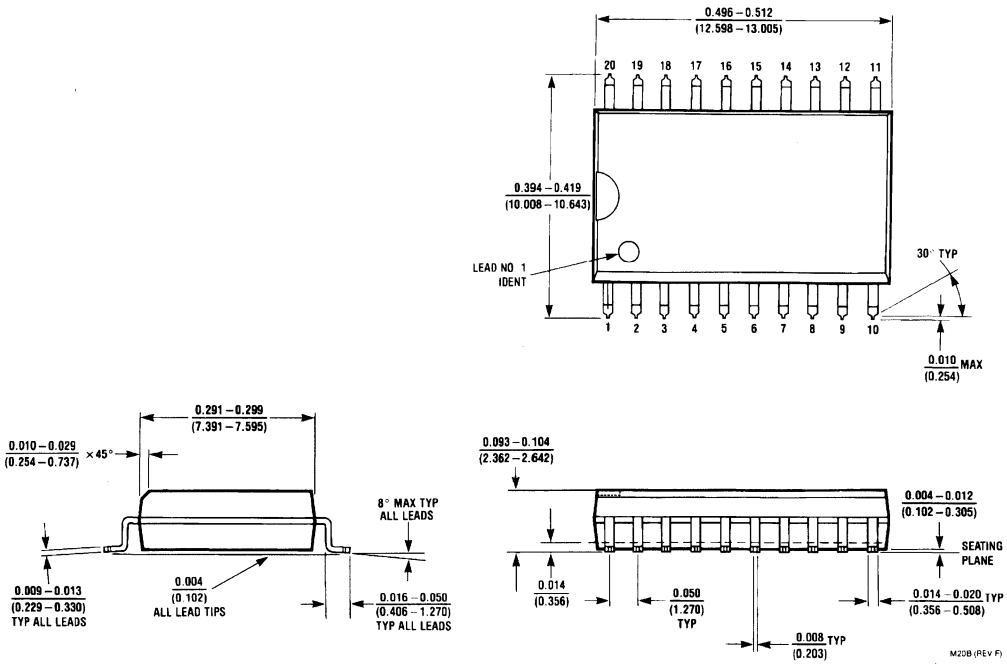
16 Lead (0.300" Wide) Molded Small Outline Package, JEDEC
NS Package Number M16B

All dimensions are in $\frac{\text{inches}}{\text{millimeters}}$



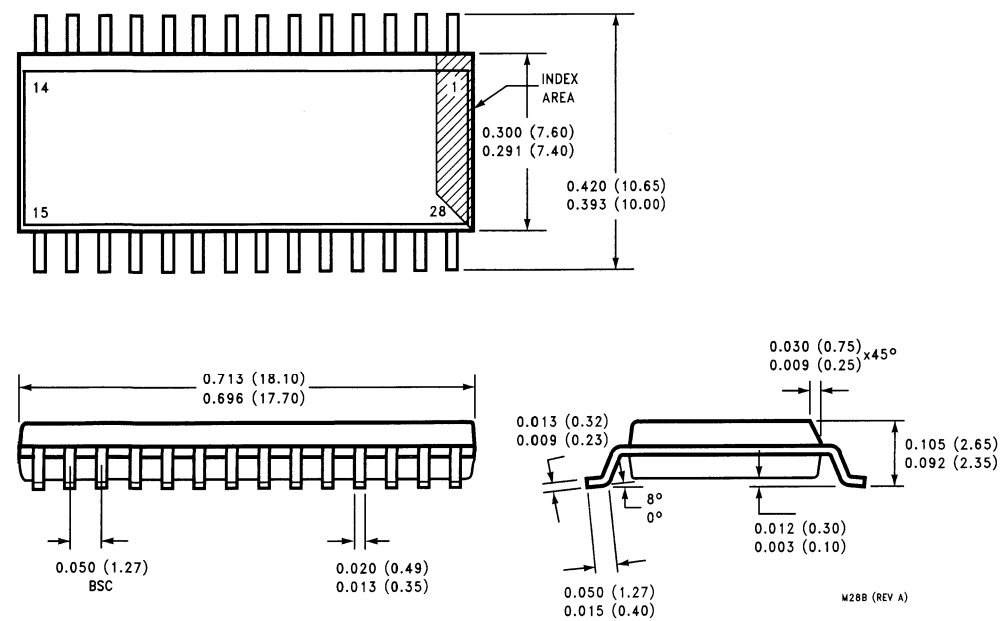
20 Lead (0.300" Wide) Molded Small Outline Package, JEDEC
NS Package Number M20B

All dimensions are in inches (millimeters)



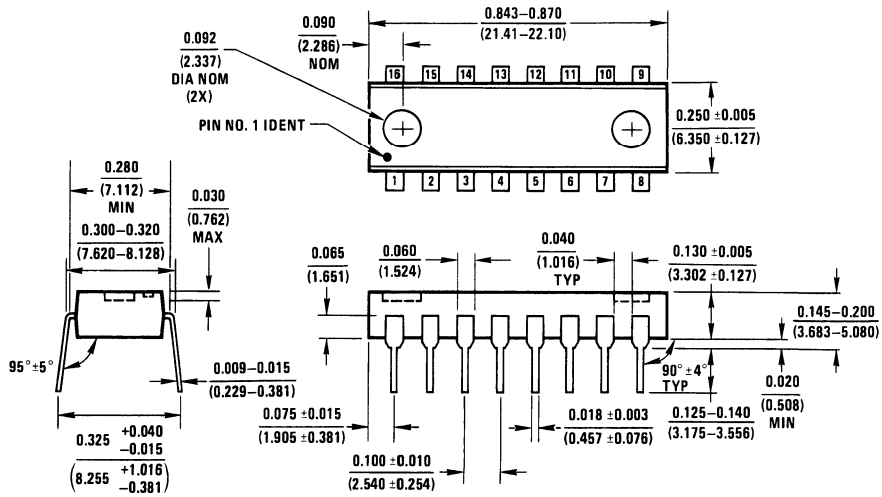
28 Lead (0.300" Wide) Molded Small Outline Package, JEDEC
NS Package Number M28B

All dimensions are in inches (millimeters)



16 Lead (0.300" Wide) Molded Dual-in-Line Package
NS Package Number N16A

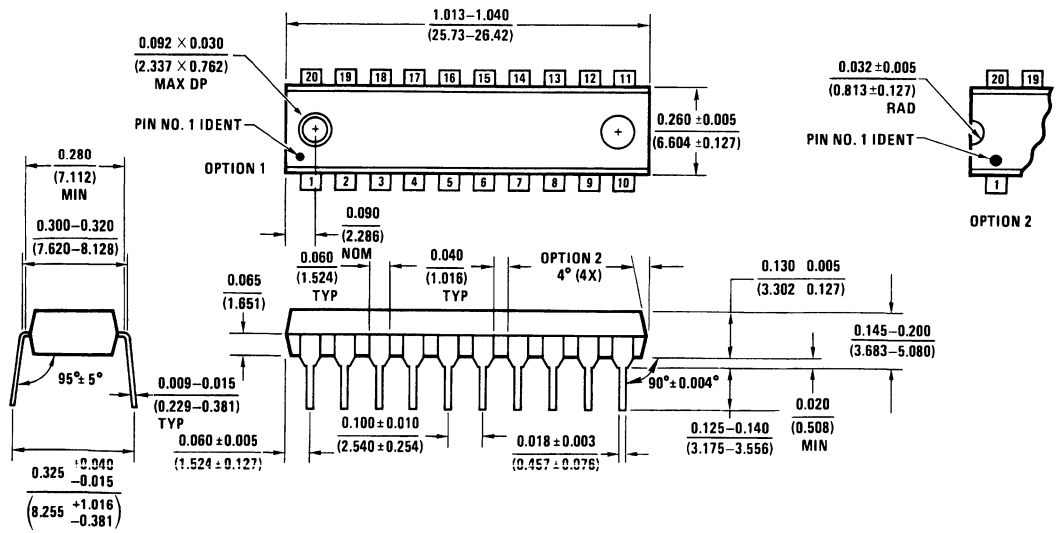
All dimensions are in inches (millimeters)



N16A (REV E)

20 Lead (0.300" Wide) Molded Dual-in-Line Package
NS Package Number N20A

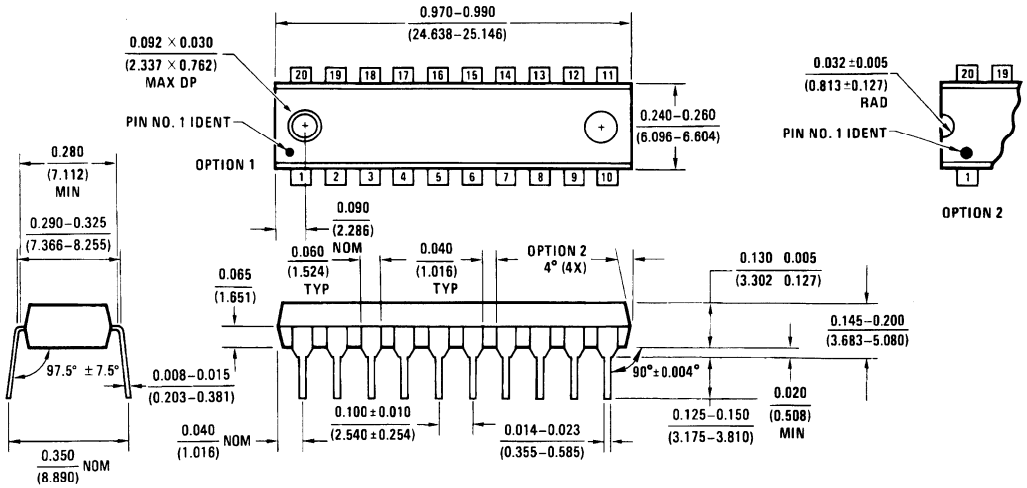
All dimensions are in inches (millimeters)



N20A (REV G)

20 Lead (0.300" Wide) Molded Dual-in-Line Package
NS Package Number N20B

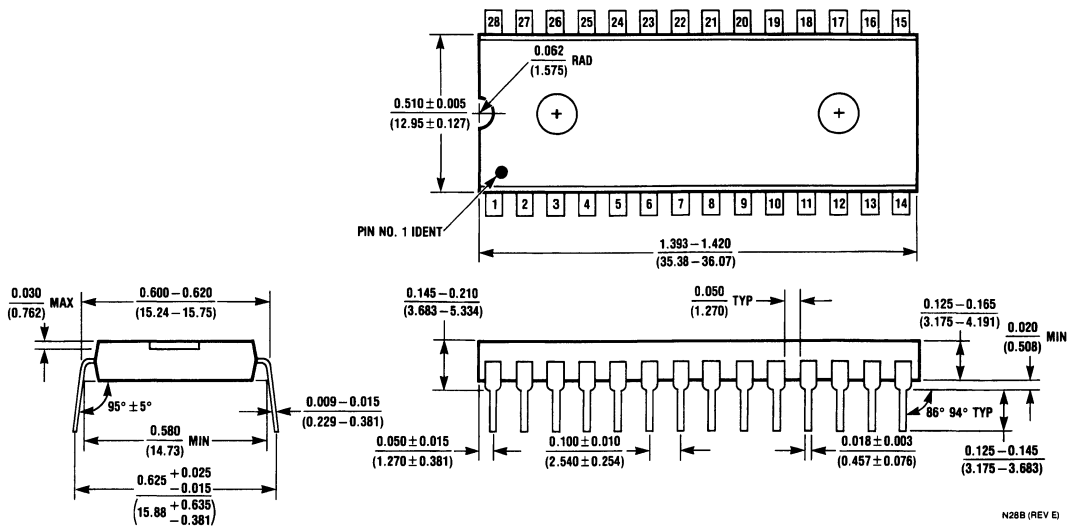
All dimensions are in inches (millimeters)



N20B (REV A)

28 Lead (0.600" Wide) Molded Dual-in-Line Package
NS Package Number N28B

All dimensions are in inches (millimeters)

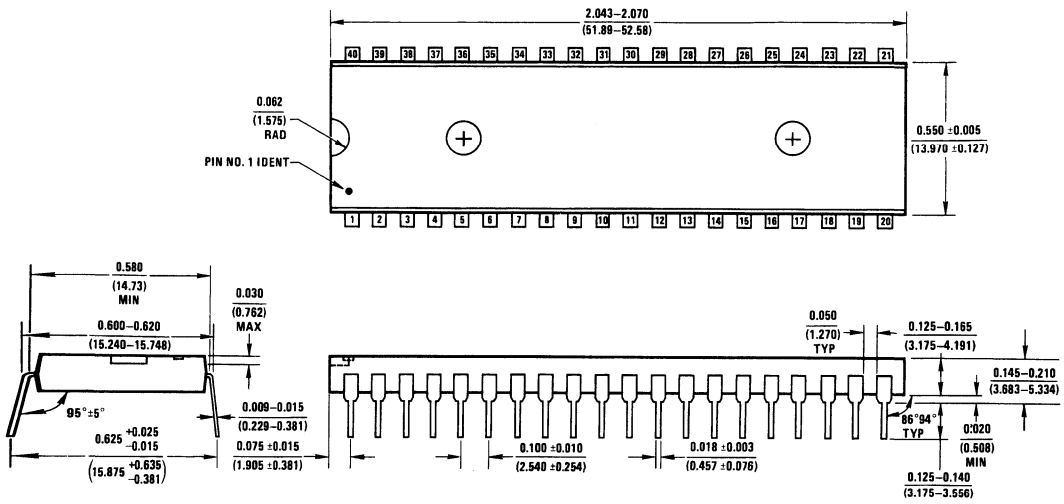


N28B (REV E)

8

40 Lead (0.600" Wide) Molded Dual-in-Line Package
NS Package Number N40A

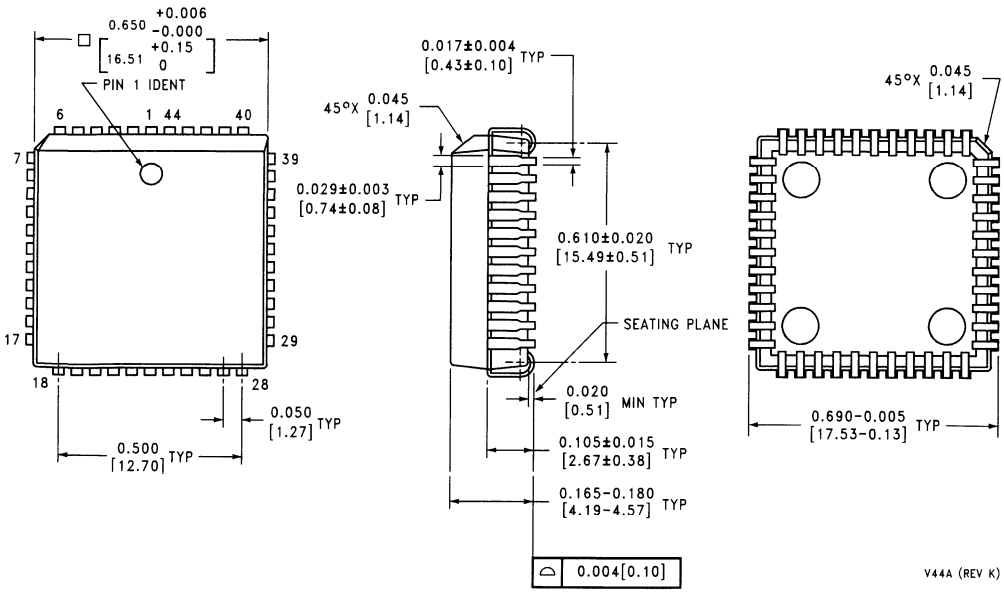
All dimensions are in inches (millimeters)



N40A (REV E)

44 Lead Molded Plastic Leaded Chip Carrier
NS Package Number V44A

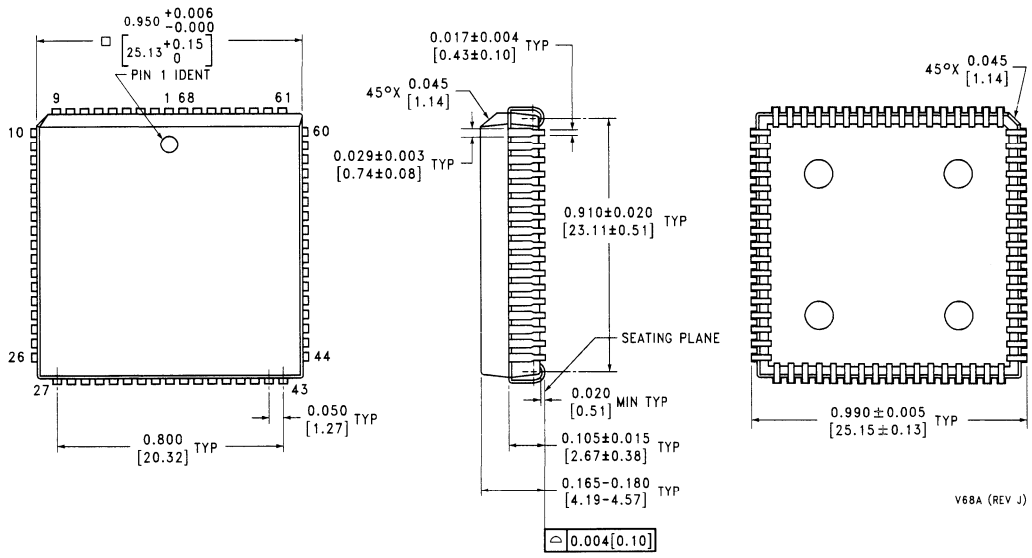
All dimensions are in inches [millimeters]



V44A (REV K)

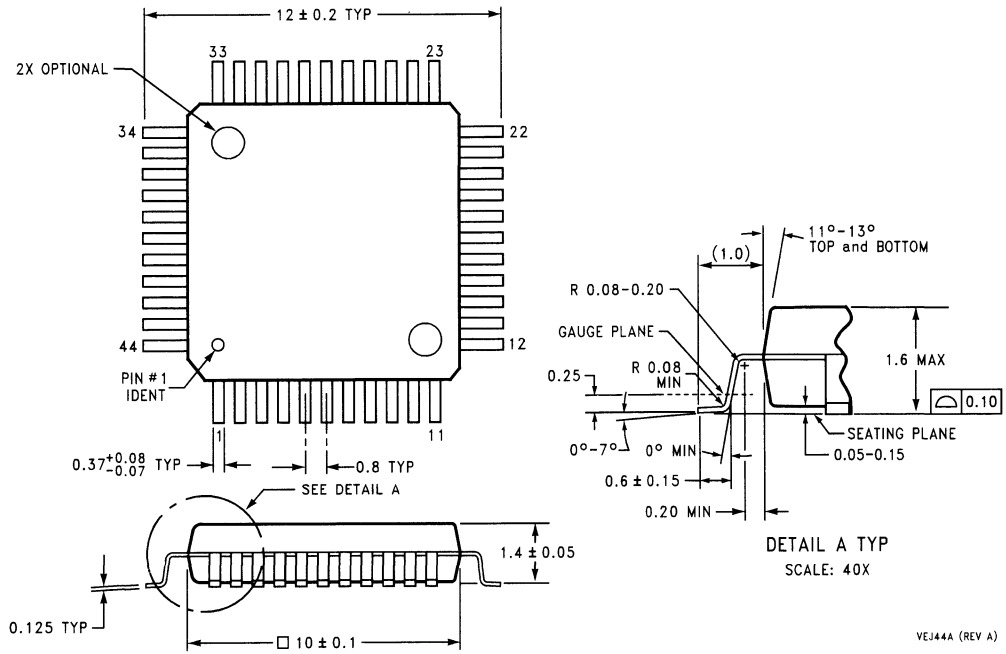
68 Lead Molded Plastic Leaded Chip Carrier
NS Package Number V68A

All dimensions are in inches [millimeters]



44 Lead (10mm x 10mm) Molded Plastic Quad Flat Package, JEDEC
NS Package Number VEJ44A

All dimensions are in millimeters



NOTES

National Semiconductor

Bookshelf of Technical Support Information

National Semiconductor Corporation recognizes the need to keep you informed about the availability of current technical literature.

This bookshelf is a compilation of books that are currently available. The listing that follows shows the publication year and section contents for each book.

For datasheets on new products and devices still in production but not found in a databook, please contact the National Semiconductor Customer Support Center at 1-800-272-9959.

We are interested in your comments on our technical literature and your suggestions for improvement.

Please send them to:

Technical Communications Dept. M/S 16-300
2900 Semiconductor Drive
P.O. Box 58090
Santa Clara, CA 95052-8090

ADVANCED BiCMOS LOGIC (ABT, BiCMOS SCAN, LOW VOLTAGE BiCMOS, EXTENDED TTL TECHNOLOGY) DATABOOK—1996

ABT Description and Family Characteristics • ABT Ratings, Specifications and Waveforms
ABT Applications and Design Considerations • Quality and Reliability
SCAN18xxxA BiCMOS 5V Logic with Boundary Scan • 74LVT Low Voltage BiCMOS Logic
VME Extended TTL Technology for Backplanes • Advanced BiCMOS Clock Generation and Support

ADVANCED BIPOLAR LOGIC FAST®, FASTr™, ALS, AS DATABOOK—1995

Introduction to Advanced Bipolar Logic Families • FAST/FASTr/ALS/AS • Family Characteristics
Ratings, Specifications and Waveforms • Design Considerations • Datasheets • Ordering and Packaging Information

APPLICATION SPECIFIC ANALOG PRODUCTS DATABOOK—1995

Audio Circuits • Video Circuits • Automotive • Special Functions • Surface Mount

ASIC DESIGN MANUAL/GATE ARRAYS & STANDARD CELLS—1987

SSI/MSI Functions • Peripheral Functions • LSI/VLSI Functions • Design Guidelines • Packaging

CLOCK GENERATION AND SUPPORT (CGS) DESIGN DATABOOK—1995

Low Skew Clock Buffers/Drivers • Video Clock Generators • Low Skew PLL Clock Generators
Crystal Clock Oscillators

COMLINEAR DATABOOK SUPPLEMENT—1995

Operational Amplifiers • Variable Gain Amplifiers • Buffer Amplifiers • Analog Multiplexers
Analog-to-Digital Converters • Serial Digital Drivers • Data Sheets • Application Notes • Evaluation Board Documentation

COP8™ DATABOOK—1996/1997

COP8 Family • COP8 OTP Products • COP8 32K OTP Products • COP8 Basic Family Products
COP8 Feature Family Products • MICROWIRE/PLUS Peripherals • COP8 Applications

CROSSVOLT™ LOW VOLTAGE LOGIC SERIES DATABOOK AND DESIGN GUIDE—1996

LCX Family • LVX Translator Family • LVX Bus Switch Family • LVX Family • LVQ Family • LVT Family
ALCX Family • GTL Family

DATA ACQUISITION DATABOOK—1995

Data Acquisition Systems • Analog-to-Digital Converters • Digital-to-Analog Converters • Voltage References
Temperature Sensors • Active Filters • Analog Switches/Multiplexers • Surface Mount

DATA ACQUISITION DATABOOK SUPPLEMENT—1992

New devices released since the printing of the 1989 Data Acquisition Linear Devices Databook.

DISCRETE SEMICONDUCTOR PRODUCTS DATABOOK—1989

Selection Guide and Cross Reference Guides • Diodes • Bipolar NPN Transistors
Bipolar PNP Transistors • JFET Transistors • Surface Mount Products • Pro-Electron Series
Consumer Series • Power Components • Transistor Datasheets • Process Characteristics

DRAM MANAGEMENT HANDBOOK—1993

Dynamic Memory Control • CPU Specific System Solutions • Error Detection and Correction
Microprocessor Applications

EEPROM MEMORY DATABOOK—1996

CMOS EEPROMs • Plug and Play Devices • Application Notes

EMBEDDED CONTROLLERS DATABOOK—1992

COP400 Family • COP800 Family • COPS Applications • HPC Family • HPC Applications
MICROWIRE and MICROWIRE/PLUS Peripherals • Microcontroller Development Tools

ETHERNET DATABOOK—1996

Integrated Network Interface Controller Products • 10 Mb/s Physical Layer Transceivers and ENDECs
10 Mb/s Repeater Interface Controller Products • 100 Mb/s Fast Ethernet Protocol Products
Glossary and Acronyms

FDDI DATABOOK—1994

Datasheets • Application Notes

F100K ECL LOGIC DATABOOK & DESIGN GUIDE—1992

Family Overview • 300 Series (Low-Power) Datasheets • 100 Series Datasheets • 11C Datasheets
Design Guide • Circuit Basics • Logic Design • Transmission Line Concepts • System Considerations
Power Distribution and Thermal Considerations • Testing Techniques • 300 Series Package Qualification
Quality Assurance and Reliability • Application Notes

FACT™ ADVANCED CMOS LOGIC DATABOOK—1993

Description and Family Characteristics • Ratings, Specifications and Waveforms
Design Considerations • 54AC/74ACXXX • 54ACT/74ACTXXX • Quiet Series: 54ACQ/74ACQXXX
Quiet Series: 54ACTQ/74ACTQXXX • 54FCT/74FCTXXX • FCTA: 54FCTXXXA/74FCTXXXA/B

FAST® APPLICATIONS HANDBOOK—1990

Reprint of 1987 Fairchild FAST Applications Handbook
Contains application information on the FAST family: Introduction • Multiplexers • Decoders • Encoders
Operators • FIFOs • Counters • TTL Small Scale Integration • Line Driving and System Design
FAST Characteristics and Testing • Packaging Characteristics

HIGH-PERFORMANCE BUS INTERFACE DATABOOK—1994

QuickRing • Futurebus+/BTL Devices • BTL Transceiver Application Notes • Futurebus+ Application Notes
High Performance TTL Bus Drivers • PI-Bus • Futurebus+/BTL Reference

IBM DATA COMMUNICATIONS HANDBOOK—1992

IBM Data Communications • Application Notes

INTERFACE DATABOOK—1996

LVDS Circuits, Bus Circuits, Data Transmission Circuits, System Design Guide

LINEAR APPLICATIONS HANDBOOK—1994

The purpose of this handbook is to provide a fully indexed and cross-referenced collection of linear integrated circuit applications using both monolithic and hybrid circuits from National Semiconductor.

Individual application notes are normally written to explain the operation and use of one particular device or to detail various methods of accomplishing a given function. The organization of this handbook takes advantage of this innate coherence by keeping each application note intact, arranging them in numerical order, and providing a detailed Subject Index.

LOW VOLTAGE DATABOOK—1992

This databook contains information on National's expanding portfolio of low and extended voltage products. Product datasheets included for: Low Voltage Logic (LVQ), Linear, EPROM, EEPROM, SRAM, Interface, ASIC, Embedded Controllers, Real Time Clocks, and Clock Generation and Support (CGS).

MASS STORAGE HANDBOOK—1989

Rigid Disk Pulse Detectors • Rigid Disk Data Separators/Synchronizers and ENDECs
Rigid Disk Data Controller • SCSI Bus Interface Circuits • Floppy Disk Controllers • Disk Drive Interface Circuits
Rigid Disk Preamplifiers and Servo Control Circuits • Rigid Disk Microcontroller Circuits • Disk Interface Design Guide

MEMORY APPLICATIONS HANDBOOK—1994

FLASH • EEPROMs • EPROMs • Application Notes

OPERATIONAL AMPLIFIERS DATABOOK—1995

Operational Amplifiers • Buffers • Voltage Comparators • Active Matrix/LCD Display Drivers
Special Functions • Surface Mount

PACKAGING DATABOOK—1993

Introduction to Packaging • Hermetic Packages • Plastic Packages • Advanced Packaging Technology
Package Reliability Considerations • Packing Considerations • Surface Mount Considerations

POWER IC's DATABOOK—1995

Linear Voltage Regulators • Low Dropout Voltage Regulators • Switching Voltage Regulators
Motion Control • Surface Mount

PRODUCTS FOR WIRELESS COMMUNICATIONS—1997

Radio Transceiver Components • Baseband Processing Components • Control and Signal Processing Components
Non-Volatile Memory • Audio Interface Components • Support Circuitry • Power Management
Complete Cordless Phone Solution

PROGRAMMABLE LOGIC DEVICE DATABOOK AND DESIGN GUIDE—1993

Product Line Overview • Datasheets • Design Guide: Designing with PLDs • PLD Design Methodology
PLD Design Development Tools • Fabrication of Programmable Logic • Application Examples

REAL TIME CLOCK HANDBOOK—1993

3-Volt Low Voltage Real Time Clocks • Real Time Clocks and Timer Clock Peripherals • Application Notes

RELIABILITY HANDBOOK—1987

Reliability and the Die • Internal Construction • Finished Package • MIL-STD-883 • MIL-M-38510
The Specification Development Process • Reliability and the Hybrid Device • VLSI/VHSIC Devices
Radiation Environment • Electrostatic Discharge • Discrete Device • Standardization
Quality Assurance and Reliability Engineering • Reliability and Documentation • Commercial Grade Device
European Reliability Programs • Reliability and the Cost of Semiconductor Ownership
Reliability Testing at National Semiconductor • The Total Military/Aerospace Standardization Program
883B/RETS™ Products • MILS/RETS™ Products • 883/RETS™ Hybrids • MIL-M-38510 Class B Products
Radiation Hardened Technology • Wafer Fabrication • Semiconductor Assembly and Packaging
Semiconductor Packages • Glossary of Terms • Key Government Agencies • AN/ Numbers and Acronyms
Bibliography • MIL-M-38510 and DESC Drawing Cross Listing

SCAN DATABOOK—1996

Design for Test Solutions • Description of Boundary SCAN • SCAN ABT Test Access Logic • SCAN CMOS Test Access Logic System Test Devices and Software • Application Notes

TELECOMMUNICATIONS—1994

COMBO and SLIC Devices • ISDN • Digital Loop Devices • Analog Telephone Components • Software • Application Notes

VHC ADVANCED CMOS LOGIC DATABOOK—1996

This databook introduces National's Very High Speed CMOS (VHC) and Very High Speed TTL Compatible CMOS (VHCT) designs. The databook includes Description and Family Characteristics • Ratings, Specifications and Waveforms Design Considerations • VHC Family Datasheets • VHC Specialty Function Datasheets and related Application Notes. The topics discussed are the advantages of VHC/VHCT AC Performance, Low Noise Characteristics and Improved Interface Capabilities.

NATIONAL SEMICONDUCTOR CORPORATION DISTRIBUTORS

ALABAMA
Huntsville
Anthem Electronics
(205) 890-0302
Future Electronics Corp.
(205) 830-2322
Hamilton/Hallmark
(205) 837-8700
Pioneer Technology
(205) 837-9300
Time Electronics
(205) 721-1134
Wyle Electronics
(205) 830-1119

ARIZONA
Phoenix
Future Electronics Corp.
(602) 968-7140
Hamilton/Hallmark
(602) 437-1200
Wyle Electronics
(602) 804-7000
Scottsdale
Alliance Electronics Inc.
(602) 483-9400
Tempe
Anthem Electronics
(602) 966-6600
Penstock Inc.
(602) 967-1620
Pioneer Standard
(602) 350-9335
Time Electronics
(602) 967-2000

CALIFORNIA
Agoura Hills
Future Electronics Corp.
(818) 865-0040
Pioneer Standard
(818) 865-5800
Time Electronics
(818) 707-2890
Calabasas
Wyle Electronics
(818) 880-9000
Chatsworth
Anthem Electronics
(818) 775-1333
Costa Mesa
Hamilton/Hallmark
(714) 641-4100
Irvine
Anthem Electronics
(714) 768-4444
Future Electronics Corp.
(714) 453-1515
Pioneer Standard
(714) 753-5090
Wyle Electronics
(714) 789-9953
Zeus Elect. an Arrow Co.
(714) 581-4622
Newbury Park
Penstock Inc.
(805) 375-6680
Rancho Cordova
Wyle Electronics
(916) 638-5282
Rocklin
Anthem Electronics
(916) 624-9744
Roseville
Future Electronics Corp.
(916) 783-7877
Hamilton/Hallmark
(916) 624-9781
San Diego
Anthem Electronics
(619) 453-9005
Future Electronics Corp.
(619) 625-2800
Hamilton/Hallmark
(619) 571-7540
Penstock Inc.
(619) 623-9100
Pioneer Standard
(619) 514-7700
Time Electronics
(619) 674-2800
Wyle Electronics
(619) 565-9171
San Jose
Anthem Electronics
(408) 453-1200
Future Electronics Corp.
(408) 434-1122
Hamilton/Hallmark
(408) 435-3500
Pioneer Technology
(408) 954-9100
Zeus Elect. an Arrow Co.
(408) 629-4789
Santa Clara
Wyle Electronics
(408) 727-2500
Sierra Madre
Penstock Inc.
(818) 355-6775
Sunnyvale
Penstock Inc.
(408) 745-8100
Time Electronics
(408) 734-9890
Tustin
Time Electronics
(714) 669-0216
Woodland Hills
Hamilton/Hallmark
(818) 594-0404
Time Electronics
(818) 593-8400

COLORADO
Englewood
Anthem Electronics
(303) 790-4500
Hamilton/Hallmark
(303) 790-1662
Penstock Inc.
(303) 799-7845
Pioneer Technology
(303) 773-8090
Time Electronics
(303) 799-5400
Lakewood
Future Electronics Corp.
(303) 232-2008
Thornton
Wyle Electronics
(303) 457-9953

CONNECTICUT
Cheshire
Future Electronics Corp.
(203) 250-0083
Hamilton/Hallmark
(203) 271-2844
Shelton
Pioneer Standard
(203) 929-5600
Wallingford
Advent Electronics
(800) 982-0014
Wyle Electronics
(203) 269-8077
Waterbury
Anthem Electronics
(203) 575-1575

FLORIDA
Altamonte Springs
Anthem Electronics
(407) 831-0007
Future Electronics Corp.
(407) 865-7900
Pioneer Technology
(407) 834-9090
Deerfield Beach
Future Electronics Corp.
(305) 426-4043
Pioneer Technology
(305) 428-8877
Wyle Electronics
(305) 420-0500
Fort Lauderdale
Hamilton/Hallmark
(305) 484-5482
Time Electronics
(305) 484-1864
Indialantic
Advent Electronics
(800) 975-8669
Lake Mary
Zeus Elect. an Arrow Co.
(407) 333-9300
Largo
Future Electronics Corp.
(813) 530-1222
Hamilton/Hallmark
(813) 541-7440
Maitland
Wyle Electronics
(407) 740-7450
Orlando
Chip Supply
"Die Distributor"
(407) 298-7100
Time Electronics
(407) 841-6566
St. Petersburg
Wyle Electronics
(813) 576-3004
Tampa
Penstock Inc.
(813) 247-7556
Winter Park
Hamilton/Hallmark
(407) 657-3300
Penstock Inc.
(407) 672-1113

GEORGIA
Duluth
Anthem Electronics
(404) 931-9300
Hamilton/Hallmark
(404) 623-4400
Pioneer Technology
(404) 623-1003
Time Electronics
(404) 623-5455
Norcross
Future Electronics Corp.
(404) 441-7676
Penstock Inc.
(770) 734-9990
Seymour Electronics
(770) 441-7878
Wyle Electronics
(770) 441-9045

ILLINOIS
Addison
Pioneer Standard
(630) 495-9680
Wyle Electronics
(630) 620-0969
Arlington Heights
Hamilton/Hallmark
(847) 797-7300
Des Plaines
Advent Electronics
(800) 323-1270
Hoffman Estates
Future Electronics Corp.
(847) 882-1255
Itasca
Zeus Elect. an Arrow Co.
(630) 595-9730
Palatine
Penstock Inc.
(847) 934-3700
Schaumburg
Anthem Electronics
(847) 884-0200
Time Electronics
(847) 303-3000

INDIANA
Carmel
Hamilton/Hallmark
(317) 575-3500
Fort Wayne
Penstock Inc.
(219) 432-1277
Indianapolis
Advent Electronics Inc.
(800) 732-1453
Future Electronics Corp.
(317) 469-0447
Pioneer Standard
(317) 573-0880
Wyle Electronics
(317) 581-6152

IOWA
Cedar Rapids
Advent Electronics
(800) 397-8407
Hamilton/Hallmark
(319) 393-0033

KANSAS
Olathe
Penstock Inc.
(913) 829-9330
Overland Park
Future Electronics Corp.
(913) 649-1531
Hamilton/Hallmark
(913) 663-7900

KENTUCKY
Lexington
Hamilton/Hallmark
(606) 288-4911

MARYLAND
Columbia
Anthem Electronics
(410) 995-6640
Future Electronics Corp.
(410) 290-0600
Hamilton/Hallmark
(410) 988-9800
Penstock Inc.
(410) 290-3746
Seymour Electronics
(410) 992-7474
Time Electronics
(410) 720-3600
Wyle Electronics
(410) 312-4844
Gaithersburg
Pioneer Technology
(301) 921-0660

MASSACHUSETTS
Bedford
Wyle Electronics
(617) 271-9953
Bolton
Future Electronics Corp.
(508) 779-3000

NATIONAL SEMICONDUCTOR CORPORATION DISTRIBUTORS (Continued)

MASSACHUSETTS (Continued)
Burlington
Penstock Inc.
(617) 229-9100
Lexington
Pioneer Standard
(617) 861-9200
Newburyport
Rochester Electronics
"Obsolete Products"
(508) 462-9332
Norwood
Gerber Electronics
(617) 769-6000
Peabody
Hamilton/Hallmark
(508) 532-3701
Time Electronics
(508) 532-9777
Wilmington
Anthem Electronics
(508) 657-5170
Zeus Elect. an Arrow Co.
(508) 658-0900

MICHIGAN
Farmington Hills
Advent Electronics
(800) 572-9329
Grand Rapids
Future Electronics Corp.
(616) 698-6800
Pioneer Standard
(616) 698-1800
Livonia
Future Electronics Corp.
(313) 261-5270
O'Fallon
Advent Electronics
(800) 888-9588
Plymouth
Hamilton/Hallmark
(313) 416-5800
Pioneer Standard
(313) 416-2157

MINNESOTA
Bloomington
Hamilton/Hallmark
(612) 881-2600
Wyle Electronics
(612) 853-2280
Burnsville
Penstock Inc.
(612) 882-7630
Eden Prairie
Anthem Electronics
(612) 944-5454
Future Electronics Corp.
(612) 944-2200
Pioneer Standard
(612) 944-3355
Minnetonka
Time Electronics
(612) 931-2131
Thief River Falls
Digi-Key Corp.
"Catalog Sales Only"
(800) 344-4539

MISSOURI
Earth City
Hamilton/Hallmark
(314) 291-5350
Manchester
Time Electronics
(314) 230-7500
St. Louis
Future Electronics Corp.
(314) 469-6805

NEW JERSEY
Bridgewater
Penstock Inc.
(908) 575-9490
Camden
Advent Electronics
(800) 255-4771
Cherry Hill
Hamilton/Hallmark
(609) 424-0110
Fairfield
Pioneer Standard
(201) 575-3510
Marlton
Future Electronics Corp.
(609) 596-4080
Time Electronics
(609) 596-1286
Mount Laurel
Seymour Electronics
(609) 235-7474
Wyle Electronics
(609) 439-9110
Parsippany
Future Electronics Corp.
(201) 299-0400
Hamilton/Hallmark
(201) 515-1641
Pine Brook
Anthem Electronics
(201) 227-7960
Wyle Electronics
(201) 882-8358
Wayne
Time Electronics
(201) 785-8250

NEW MEXICO
Albuquerque
Hamilton/Hallmark
(505) 828-1058

NEW YORK
Binghamton
Pioneer Standard
(607) 722-9300
Commack
Anthem Electronics
(516) 864-6600
Fairport
Pioneer Standard
(716) 381-7070
Hauppauge
Future Electronics Corp.
(516) 234-4000
Hamilton/Hallmark
(516) 434-7400
Penstock Inc.
(516) 724-9580
Time Electronics
(516) 273-0100
Wyle Electronics
(516) 231-7850
Henrietta
Wyle Electronics
(716) 334-5970
Port Chester
Zeus Elect. an Arrow Co.
(914) 937-7400
Rochester
Future Electronics Corp.
(716) 387-9550
Hamilton/Hallmark
(800) 475-9130
Syracuse
Future Electronics Corp.
(315) 451-2371
Time Electronics
(315) 434-9837
Woodbury
Pioneer Standard
(516) 921-9700
Seymour Electronics
(516) 496-7474

NORTH CAROLINA
Charlotte
Future Electronics Corp.
(704) 547-1107
Morrisville
Pioneer Technology
(919) 460-1530
Wyle Electronics
(919) 469-1502
Raleigh
Anthem Electronics
(919) 782-3550
Future Electronics Corp.
(919) 790-7111
Hamilton/Hallmark
(919) 872-0712

OHIO
Beavercreek
Future Electronics Corp.
(513) 426-0090
Cleveland
Pioneer Standard
(216) 587-3600
Columbus
Time Electronics
(614) 794-3301
Dayton
Hamilton/Hallmark
(513) 439-6735
Pioneer Standard
(513) 236-9900
Wyle Electronics
(513) 436-9953
Mayfield Heights
Future Electronics Corp.
(216) 449-6996
Solon
Hamilton/Hallmark
(216) 498-1100
Wyle Electronics
(216) 248-9996
Worthington
Hamilton/Hallmark
(614) 888-3313

OKLAHOMA
Tulsa
Hamilton/Hallmark
(918) 254-6110
Pioneer Standard
(918) 665-7840
Radio Inc.
(918) 587-9123

OREGON
Beaverton
Future Electronics Corp.
(503) 645-9454
Hamilton/Hallmark
(503) 526-6200
Pioneer Technology
(503) 626-7300
Wyle Electronics
(503) 643-7900
Portland
Anthem Electronics
(503) 598-9660
Penstock Inc.
(503) 646-1670
Time Electronics
(503) 684-3780

PENNSYLVANIA
Coatesville
Penstock Inc.
(610) 383-9536
Horsham
Anthem Electronics
(215) 443-5150
Pioneer Technology
(215) 674-4000
Pittsburgh
Pioneer Standard
(412) 782-2300

TEXAS
Austin
Anthem Electronics
(512) 388-0049
Future Electronics Corp.
(512) 502-0991
Hamilton/Hallmark
(512) 258-8848
Minco Technology Labs.
"Die Distributor"
(512) 834-2022
Penstock Inc.
(512) 346-9762
Pioneer Standard
(512) 835-4000
Time Electronics
(512) 219-3773
Wyle Electronics
(512) 833-9953
Carrollton
Zeus Elect. an Arrow Co.
(214) 380-4330
Dallas
Hamilton/Hallmark
(214) 553-4300
Pioneer Standard
(214) 386-7300
Houston
Future Electronics Corp.
(713) 785-1155
Hamilton/Hallmark
(713) 781-6100
Pioneer Standard
(713) 495-4700
Wyle Electronics
(713) 784-9953
Richardson
Anthem Electronics
(214) 238-7100
Future Electronics Corp.
(214) 437-2437
Penstock Inc.
(214) 479-9215
Time Electronics
(214) 480-5000
Wyle Electronics
(214) 235-9953

UTAH
Salt Lake City
Anthem Electronics
(801) 973-8555
Future Electronics Corp.
(801) 467-4448
Hamilton/Hallmark
(801) 266-2022
West Valley City
Time Electronics
(801) 973-0208
Wyle Electronics
(801) 974-9953

WASHINGTON
Bellevue
Penstock Inc.
(206) 643-6687
Pioneer Technology
(206) 644-7500
Bothell
Anthem Electronics
(206) 483-1700
Future Electronics Corp.
(206) 489-3400
Kirkland
Time Electronics
(206) 820-1525
Redmond
Hamilton/Hallmark
(206) 881-6697
Wyle Electronics
(206) 881-1150

NATIONAL SEMICONDUCTOR CORPORATION DISTRIBUTORS (Continued)

WISCONSIN
Brookfield
Future Electronics Corp.
(414) 879-0244
Pioneer Standard
(414) 784-3480
Wyle Electronics
(414) 879-0434
Mequon
Taylor Electric
(414) 241-4321
New Berlin
Hamilton/Hallmark
(414) 780-7200
West Allis
Advent Electronics
(800) 500-0441

CANADA
WESTERN PROVINCES
Burnaby
Hamilton/Hallmark
(604) 420-4101
Calgary
Electro Sonic Inc.
(403) 255-9550
Future Electronics Corp.
(403) 250-5550
Zentronics/Pioneer
(403) 295-8838
Edmonton
Future Electronics Corp.
(403) 438-2858
Zentronics/Pioneer
(403) 482-3038
Richmond
Electro Sonic Inc.
(604) 273-2911
Zentronics/Pioneer
(604) 273-5575
Vancouver
Future Electronics Corp.
(604) 294-1166

EASTERN PROVINCES
Kanata
Penstock Inc.
(613) 592-6088
Mississauga
Future Electronics Corp.
(905) 612-9200
Hamilton/Hallmark
(905) 564-6060
Time Electronics
(905) 712-3277
Zentronics/Pioneer
(905) 405-8300
Nepean
Hamilton/Hallmark
(613) 226-1700
Zentronics/Pioneer
(613) 226-8840
Ottawa
Electro Sonic Inc.
(613) 728-8333
Future Electronics Corp.
(613) 820-8313
Pointe Claire
Future Electronics Corp.
(514) 694-7710
Quebec
Future Electronics Corp.
(418) 877-6666
Ville St. Laurent
Hamilton/Hallmark
(514) 335-1000
Penstock Inc.
(514) 333-8837
Zentronics/Pioneer
(514) 737-9700
Willowdale
Electro Sonic Inc.
(416) 494-1666
Winnipeg
Electro Sonic Inc.
(204) 783-3105
Future Electronics Corp.
(204) 944-1446
Zentronics/Pioneer
(204) 694-1957

National Semiconductor

WORLDWIDE SALES OFFICES

AUSTRALIA

National Semiconductor (Australia) Pty. Ltd.
Bldg. 16 Business Park Dr.
Monash Business Park
Nottinghill Melbourne
Victoria 3168 Australia
Tel: (39) 558-9999
Fax: (39) 558-9998

BRAZIL

National Semiconductores Do Brazil Ltda.
Rue Deputado Lacorda
Franco 120-3A
Sao Paulo-SP Brazil 05418-000
Tel: (55-11) 212-5066
Fax: (55-11) 212-1181

CANADA

National Semiconductor (Canada)
5925 Airport Road, Suite 615
Mississauga, Ontario L4V 1W1
Tel: (416) 678-2920
Fax: (416) 678-2837

National Semiconductor (Canada)
39 Robertson Road, Suite 101
Nepean, Ontario K2H 8R2
Tel: (613) 596-0411
Fax: (613) 596-1613

National Semiconductor (Canada)
1870 Boul Des Sources,
Suite 101
Pointe Claire, Quebec H2R 5N4
Tel: (514) 426-2992
Fax: (514) 426-2710

CHINA

National Semiconductor Beijing China Liaison Office
Room 613 & 614
Sinochem Mansion
No. A2 Fuxingmenwai Avenue
Beijing 100046, PRC
China
Tel: 86-10-8568601
Fax: 86-10-8568606

National Semiconductor Shanghai China Liaison Office
R702 Universal Mansion
No. 172, Yuyuan Road
Shanghai 200040, PRC
China
Tel: 86-21-2496062
Fax: 86-21-2496063

FINLAND

National Semiconductor (U.K.) Ltd.
Mekaanikonkatu 13
SF-00810 Helsinki
Finland
Tel: (0) 759-1855
Fax: (0) 759-1393

FRANCE

National Semiconductor S.A.
Parc d'Affaires Technopolis
3, Avenue Du Canada
Bat. ZETA - L.P. 821 Les Ulis
F-91974 Courtaboeuf Cedex
France
Tel: (1) 69 18 37 00
Fax: (1) 69 18 37 69

GERMANY

National Semiconductor GmbH
Livry-Gargan-Strasse. 10
D-82256 Fürstenfeldbruck
Germany
Tel: (0-81-41) 35-0
Fax: (0-81-41) 35-15-06

HONG KONG

National Semiconductor Hong Kong Ltd.
13th Floor, Straight Block
Ocean Centre
5 Canton Road
Tsimshatsui, Kowloon
Hong Kong
Tel: (852) 2737-1600
Fax: (852) 2736-9960

INDIA

National Semiconductor India Liaison Office
1109, 11th Floor, West Wing
Raheja Towers, M.G. Road
Bangalore 560001 India
Tel: 91-80-559-9467
Fax: 91-80-559-9468

ISRAEL

National Semiconductor Ltd.
Maskit Street
PO Box 3007
Herzlia B. 46104
Israel
Tel: (09) 59 42 55
Fax: (09) 55 83 22

ITALY

National Semiconductor S.p.A.
Strada 7, Palazzo R/3
I-20089 Rozzano-Milanofiori
Italy
Tel: (02) 57 50 03 00
Fax: (02) 57 50 04 00

JAPAN

National Semiconductor Japan Ltd.
Sumitomo Chemical
Engineering Center Bldg. 8F
1-7-1, Nakase, Mihama-Ku
Chiba-City,
Chiba Prefecture 261
Japan
Tel: 81-043-299-2300
Fax: 81-043-299-2500

KOREA

National Semiconductor (Far East) Ltd.
13th Floor, Dai Han
Life Insurance 63 Building
60 Yoido-Dong
Youngdeungpo-KU
Seoul Korea 150-763
Tel: (02) 784-8051/3
(02) 785-0696/8
Fax: (02) 784-8054

MALAYSIA

National Semiconductor Sdn Bhd
Bayan Lepas Free Trade Zone
11900 Penang Malaysia
Tel: 4-644-9061
Fax: 4-644-9073

MEXICO

Electronica NSC de Mexico SA
Avenida de las Naciones
Col. Napoles
Mexico City, Mexico DF 03810
Tel: (525) 488-0135
Fax: (525) 488-0139

PUERTO RICO

National Semiconductor (Puerto Rico)
La Electronica Bldg.
Suite 512, R.D. #1 KM 14.5
Rio Piedias
Puerto Rico 00927
Tel: (809) 758-9211
Fax: (809) 763-6959

SINGAPORE

National Semiconductor Asia Pacific Pte. Ltd.
200 Cantonment Road #13-01
Southpoint Singapore 0208
Tel: (65) 225-2226
Fax: (65) 225-7080

SPAIN

National Semiconductor GmbH
Calle Almendralejos, 4
28140 Fuente el Saz del Jarama
Madrid, Spain
Tel: (01) 620 14 25
Fax: (01) 620 06 12

SWEDEN

National Semiconductor AB
P.O. Box 1009
Grosshandlarvägen 7
S-12123 Johanneshov,
Sweden
Tel: (08) 7 22 80 50
Fax: (08) 7 22 90 95

SWITZERLAND

National Semiconductor (U.K.) Ltd.
Alte Winterthurerstrasse 53
CH-8304 Wallisellen-Zürich
Switzerland
Tel: (01) 8-30-27-27
Fax: (01) 8-30-19-00

TAIWAN

National Semiconductor (Far East) Ltd.
9/F, No. 44 Section 2
Chungshan North Road
Taipei, Taiwan, R.O.C.
Tel: (02) 521-3288
Fax: (02) 561-3054

U.K. AND IRELAND

National Semiconductor (U.K.) Ltd.
1st Floor
Milford House
Milford Street
Swindon, Wiltshire SN1 1DW
United Kingdom
Tel: (07-93) 61 41 41
Fax: (07-93) 52 21 80
Telex: 444674

UNITED STATES

National Semiconductor Corporation
1111 West Bardin Road
Arlington, TX 76017
Tel: (800) 272-9959
Fax: (800) 737-7018