COP8™
MICROCONTROLLER
DATABOOK

1994 Edition

COP8 Family 1

COP8 Applications 2

MICROWIRE/PLUS™ Peripherals 3

COP8 Development Support 4

Appendices/Physical Dimensions 5

TRADEMARKS

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

ABiC™
Abuseable™
AirShare™
Anadig™
APPS™
ARi^1™
ASPECT™
AT/LANTIC™
Auto-Chem Deflasher™
BCP™
BI-FET™
BI-FET II™
BI-LINE™
BIPLAN™
BLC™
BLX™
BMAC™
Brite-Lite™
BSI™
BSI-2™
CDD™
CDL™
CGS™
CIM™
CIMBUS™
CLASIC™
COMBO®
COMBO I™
COMBO II™
COPS™ microcontrollers
COP8™
CRD™
CROSSVOLT™
CSNI™
CTI™
CYCLONE™
DA4™
DENSPAK™
DIB™
DISCERN™
DISTILL™
DNR®
DPVM™
E^{2}CMOS™
ELSTAR™
Embedded System Processor™
EP™
E-Z-LINK™
FACT™
FACT Quiet Series™
FAIRCAD™
Fairtech™
FAST®
FASTr™
GENIX™
GNX™
GTO™
HEX 3000™
HPC™
HyBal™
I^{3}L®
ICM™
Integral ISE™
Intelisplay™
Inter-LERIC™
Inter-RIC™
ISE™
ISE/06™
ISE/08™
ISE/16™
ISE32™
ISOPLANAR™
ISOPLANAR-Z™
LERIC™
LMCMOS™
M^{2}CMOS™
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™
NSC800™
NSCISE™
NSX-16™
NS-XC-16™
NTERCOM™
NURAM™
OPAL™
Overture™
OXISS™
P^{2}CMOS™
Perfect Watch™
PLAN™
PLANAR™
PLAYER™
PLAYER+™
Plus-2™
Polycraft™
POP™
Power + Control™
POWERplanar™
QS™
QUAD3000™
Quiet Series™
QUIKLOOK™
RAT™
RIC™
RICKIT™
RTX16™
SCAN™
SCENIC™
SCX™
SERIES/800™
Series 32000®
SIMPLE SWITCHER®
SNI™
SNIC™
SofChek™
SONIC™
SPiKe™
SPIRE™
Staggered Refresh™
STAR™
Starlink™
STARPLEX™
ST-NIC™
SuperAT™
Super-Block™
SuperChip™
SuperI/O™
SuperScript™
SYS32™
TapePak®
TDS™
TeleGate™
The National Anthem®
TLC™
Trapezoidal™
TRI-CODE™
TRI-POLY™
TRI-SAFE™
TRI-STATE®
TROPIC™
Tropic Pele'™
Tropic Reef™
TURBOTRANSCEIVER™
VIP™
VR32™
WATCHDOG™
XMOS™
XPU™
Z STAR™
883B/RETS™
883S/RETS™

IBM®, PC®, PC-AT® and PC-XT® are registered trademarks of International Business Machines Corporation.
iceMASTER™ is a trademark of MetaLink Corporation.
Sun® is a registered trademark of Sun Microsystems.
SunOS™ is a trademark of Sun Microsystems.

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

Table of Contents (Continued)

Alpha-Numeric Index

Alpha-Numeric Index (Continued)

Section 1

COP8 Family

Section 1 Contents

National Semiconductor

The 8-Bit COP8™ Family: Optimized for Value

Key Features

- High-performance 8-bit microcontroller
- Full 8-bit architecture and implementation
- 1 μs instruction-cycle time
- High code efficiency with single-byte, multiple-function instructions
- UART
- A/D converter
- WATCHDOG™/clock monitor
- Brown Out Detect
- On-chip ROM from 768 bytes to 16k bytes
- On-chip RAM to 256 bytes
- EEPROM
- M²CMOS™ fabrication
- MICROWIRE/PLUS™ serial interface
- Wide operating voltage range: +2.3V to +6V
- Military temp range available: −55°C to +125°C
- MIL-STD-883C versions available
- 16- to 44-pin packages

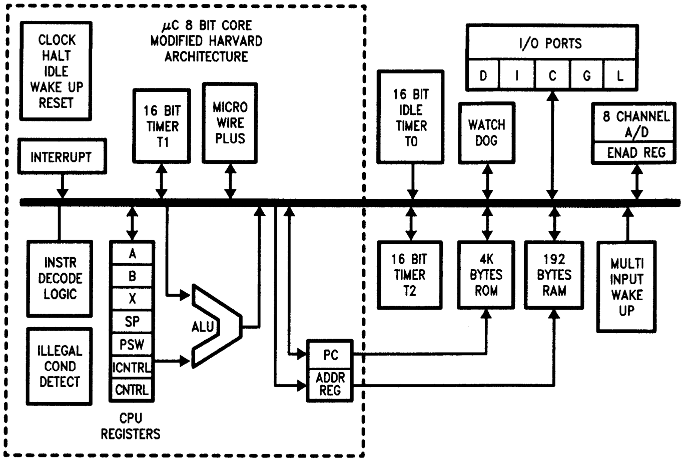
The COP8 combines a powerful single-byte, multiple-function instruction set with a memory-mapped core architecture.

Key Applications

- Automotive systems
- Process control
- Robotics
- Telecommunications
- AC-motor control
- DC-motor control
- Keyboard controllers
- Modems
- RS232C controllers
- Toys and games
- Industrial control
- Small appliances

The COP8 family offers high performance in a low-cost, easy-to-design-in package.

An Example of COP888 Block Diagram (COP888CF)



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 compute 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.** All 8-bit 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. And that's why National Semiconductor's COP8 family of 8-bit microcontrollers is enjoying widespread success in today's global embedded control marketplace.

One of the leaders in the design, manufacture and sale of 8-bit microcontrollers is National Semiconductor. Long a prominent player in the worldwide microcontroller market, National 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.

National's 8-bit COP8 microcontrollers enable the company to meet a wide range of embedded control application requirements. COP8 microcontrollers offer users cost-effective solutions at virtually every price/performance point in today's market for 8 bit applications.

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 on up 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, C compilers, and a "fuzzy logic" design environment 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.

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

A Growing Family

National's wide-ranging COP8 family is well-positioned to meet the expanding variety of consumer 8- bit microcontroller applications. Available in a wealth of different ROM (768 bytes to 16k bytes) and RAM (64 x 8, 128 x 8, and 512 x 8) configurations, COP8 microcontrollers provide designers with cost-effective solutions at every price/performance point in todays market. And the recent introduction of the new COP912C—National's first 8 bit microcontroller priced below 50¢ per unit when purchased in volume quantities—continues to drive prices down in the highly competitive 8-bit market.

A code-efficient instruction set. Low power operation. I/O pin efficiency. A "whole product" philosophy that includes superior development tools, documentation and support. These are the reasons that National's COP8 family is a key player in the worldwide 8-bit microcontroller market. As that market continues to expand. National continues its microcontroller technology research and development efforts—an ongoing commitment that began during the infancy of embedded control and continues in full force today.

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) • Comparators • 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 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 (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 COP840/COP880 COP888CL (Keyboards) COP8646 (Smart Cards)
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 (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 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: COP888CG/888EG/888EK Seat/Motional Control, Slave Controller: COP884BC Dashboard Control: COP888GW Mirror Control, etc.: COP8 Basic Family Climate Control: COP888CF

COP8 Family Selection Guide

Common Features:

- Multi-Source Interrupt
- Pinout
- Instruction Set
- MICROWIRE Serial Communication
- 1 μs Instruction Cycle Time
- Wide Power Supply—2.3V to 6.0V
- CMOS Process Technology
- Halt Mode
- Software Selectable I/O
- Wide Temperature Range
- Development Tools

Comm Temp 0°C to +70°C	Ind Temp −40°C to +85°C	Mil Temp −55°C to +125°C	Memory		I/O	Packages				Features								Single Chip Emulators		
			ROM (Bytes)	RAM (Bytes)	Pins	# of Pins	N	WM	V	Interrupt Sources	Timers PWM/ Capture	Compar-ators	UART	WATCH-DOG	Multi-Input Wakeup	Idle Timer	Additional Features	DIP	PLCC	SO
	COP823CJ		1.0k	64	11	16		x		3	1	1		x	x		Brown Out Detection			
	COP822CJ		1.0k	64	15	20	x	x		3	1	1		x	x		Modulator, Special PWM,	COP8722CJN[2]		COP8722CJWM[2]
	COP820CJ		1.0k	64	23	28	x	x		3	1	1		x	x		Timer, High Current Outputs	COP8720CJN[2]		COP8720CJWM[2]
COP912C			768	64	15	20	x	x		3	1							COP8782CJ		
COP922C	COP822C	COP622C	1.0k	64	15	20	x	x		3	1							COP8782CJ		
COP920C	COP820C	COP620C	1.0k	64	23	28	x	x		3	1							COP8781CJ		
COP942C	COP842C	COP642C	2.0k	128	15	20	x	x		3	1							COP8782CJ		
COP940C	COP840C	COP640C	2.0k	128	23	28	x	x		3	1							COP8781CJ		
COP981C	COP881C	COP681C (Note 1)	4.0k	128	23	28	x	x		3	1							COP8781CJ		
COP980C	COP880C	COP680C (Note 1)	4.0k	128	35	40/44	x		x	3	1							COP8780CJ	COP8780EL	
	COP8782C		4.0k	128	15	20	x	x		3	1						UV WINDOWED	COP8782CJ		
	COP8781C	COP6781C[1]	4.0k	128	23	28	x	x		3	1						&	COP8781CJ		
	COP8780C		4.0k	128	35	40/44	x		x	3	1						OTP	COP8780CJ	COP8780EL	
	COP8622C	COP6622C	1.0k	64	15	20	x	x		3	1							COP8642CMHD-X		
	COP86L22C	COP6622C	1.0k	64	15	20	x	x		3	1									
	COP8620C	COP6620C	1.0k	64	23	28	x	x		3	1						64 x 8	COP8640CMHD-X		
	COP86L20C	COP6620C	1.0k	64	23	28	x	x		3	1						EEPROM			
	COP8642C	COP6642C	2.0k	64	15	20	x	x		3	1						IN	COP8642CMHD-X		
	COP86L42C	COP6642C	2.0k	64	15	20	x	x		3	1						RAM			
	COP8640C	COP6640C	2.0k	64	23	28	x	x		3	1							COP8640CMHD-X		
	COP86L40C	COP6640C	2.0k	64	23	28	x	x		3	1									
COP984CL	COP884CL	COP684CL	4.0k	128	23	28	x	x		10	2			x	x	x	Clock	COP8784CLN		COP8784CLWM
COP988CL	COP888CL	COP688CL	4.0k	128	33/39	40/44	x		x	10	2			x	x	x	Monitor	COP8788CLN	COP8788CLV	
COP984CF	COP884CF		4.0k	128	23	28	x	x		10	2			x	x	x	8 Channel	COP8784CFN		COP8784CFWM
COP988CF	COP888CF		4.0k	128	33/37	40/44	x		x	10	2			x	x	x	(8-bit) A/D	COP8788CFN	COP8788CFV	

Note 1: MIL-STD-883 in J Pkg

Note 2: Contact sales office for availability.

N = Plastic DIP

V = Plastic Leaded Chip Carrier (PLCC)

WM = Small Outline Package—Wide Body

MHD = Ceramic DIP

MHEA = 28 Small-Outline Footprint

EL = Lead Chip Carrier

COP8 Family Selection Guide (Continued)

Common Features:

- **Multi-Source Interrupt**
- **Pinout**
- **Instruction Set**
- **MICROWIRE Serial Communication**
- **1 μs Instruction Cycle Time**
- **Wide Power Supply—2.3V to 6.0V**
- **CMOS Process Technology**
- **Halt Mode**
- **Software Selectable I/O**
- **Wide Temperature Range**
- **Development Tools**

Comm	Ind	Mil	Memory		I/O	Packages				Features								Single Chip Emulators		
Temp 0°C to +70°C	Temp −40°C to +85°C	Temp −55°C to +125°C	ROM (Bytes)	RAM (Bytes)	Pins	# of Pins	N	WM	V	Interrupt Sources	Timers PWM/ Capture	Compar- ators	UART	WATCH- DOG	Multi- Input Wakeup	Idle Timer	Additional Features	DIP	PLCC	SO
COP984CS	COP884CS	COP684CS	4.0k	192	23	28	x	x		12	1	1	x	x	x	x		COP8784EGN		COP8784EGWM
COP988CS	COP888CS	COP688CS	4.0k	192	35/39	40/44	x		x	12	1	1	x	x	x	x		COP8788EGN	COP8788EGV	
	COP884CG		4.0k	192	23	28	x	x		14	3	2	x	x	x	x	Reduced EMI	COP8784EGN		COP8784EGWM
	COP888CG		4.0k	192	35/39	40/44	x		x	14	3	2	x	x	x	x	Reduced EMI	COP8788EGN	COP8788EGV	
	COP884EK		8.0k	256	23	28	x	x		12	3	1		x	x	x	6 Analog Inputs, Constant Current Source, Reduced EMI			
	COP888EK		8.0k	256	35/39	40/44	x		x	12	3	1		x	x	x				
COP984EG	COP884EG	COP684EG	8.0k	256	23	28	x	x		14	3	2	x	x	x	x		COP8784EGN		COP884EGWM
COP988EG	COP888EG	COP688EG	8.0k	256	35/39	40/44	x		x	14	3	2	x	x	x	x		COP8788EGN	COP8788EGV	
	COP884BC		2.0k	64	18	28		x		12	1	2			x	x	CAN Interface, Motor Control Timer			
	COP888GW		16.0k	512	56	68			x	14	2		x		x	x	Hardware Multiply/ Divide Function, 4x Counter Block, Reduced EMI			

Note 1: MIL-STD-883 in J Pkg

Note 2: Contact sales office for availability.

N = Plastic DIP

V = Plastic Leaded Chip Carrier (PLCC)

WM = Small Outline Package—Wide Body

MHD = Ceramic DIP

MHEA = 28 Small-Outline Footprint

EL = Lead Chip Carrier

National Semiconductor

PRELIMINARY

COP912C/COP912CH Single-Chip microCMOS Microcontrollers

General Description

The COP912C/COP912CH are members of the COPS™ 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.

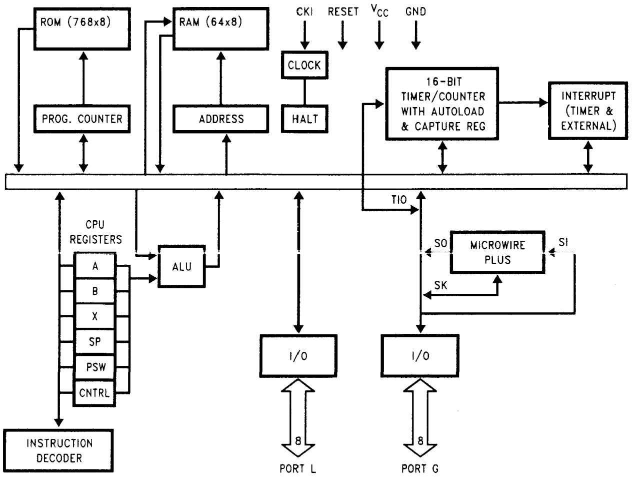
Features

- Low cost 8-bit MicroController
- Fully static CMOS
- Instruction Time
  - — 2 μs COP912CH
  - — 2.5 μs COP912C
- Low current drain
  Low current static HALT mode
- Single supply operation
- 768 x 8 on-chip ROM
- 64 Bytes on-chip RAM
- MICROWIRE/PLUS™ serial I/O
- 16-bit read/write timer operates in a variety of modes
  - — Timer with 16-bit auto reload register
  - — 16-bit external event counter
  - — Timer with 16-bit capture register (selectable edge)
- Multi-source interrupt
  - — External interrupt with selectable edge
  - — Timer interrupt or capture interrupt
  - — Software interrupt
- 8-bit stack pointer (stack in RAM)
- Powerful instruction set, most instructions single byte
- BCD arithmetic instructions
- 20-pin DIP/SO packages
- Software selectable I/O options (TRI-STATE®, push-pull, weak pull-up)
- Schmitt trigger inputs on Port G-Port
- Temperature range: COP912C/COP912CH from 0°C to 70°C
- Form Factor Emulator

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

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.***

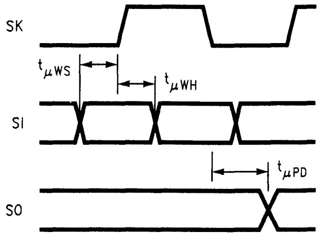
DC Electrical Characteristics COP912C/COP912CH; 0°C ≤ T_A ≤ +70°C unless other specified

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

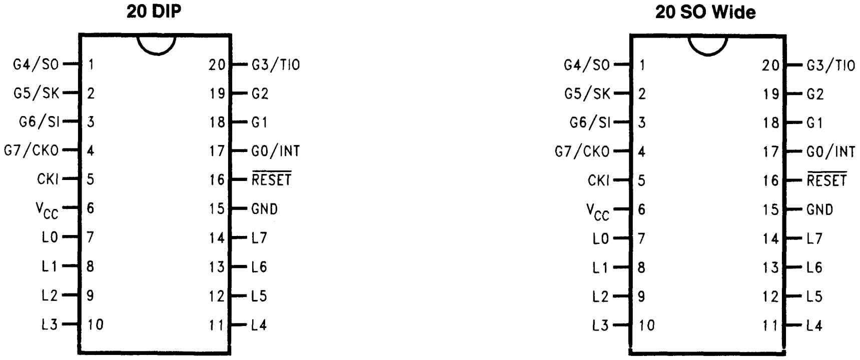
FIGURE 1. MICROWIRE/PLUS Timing

AC Electrical Characteristics COP912C/COP912CH; 0°C $\le T_A \le$ +70°C unless otherwise specified

Parameter	Conditions	Min	Typ	Max	Units
INSTRUCTION CYCLE TIME (tc)					
Crystal/Resonator	$4.0V \le V_{CC} \le 5.5V$	2		DC	µs
	$2.3V \le V_{CC} < 4.0V$	2.5		DC	µs
R/C Oscillator	$4.0V \le V_{CC} \le 5.5V$	3		DC	µs
	$2.3V \le V_{CC} < 4.0V$	7.5		DC	µs
Inputs					
t_{Setup}	$4.0V \le V_{CC} \le 5.5V$	200			ns
	$2.3V \le V_{CC} < 4.0V$	500			ns
t_{Hold}	$4.0V \le V_{CC} \le 5.5V$	60			ns
	$2.3V \le V_{CC} < 4.0V$	150			ns
Output Propagation Delay	$R_L = 2.2\ k\Omega, C_L = 100\ pF$				
t_{PD1}, t_{PD0}					
SO, SK	$4.0V \le V_{CC} \le 5.5V$			0.7	µs
	$2.3V \le V_{CC} < 4.0V$			1.75	µs
All Others	$4.0V \le V_{CC} \le 5.5V$			1	µs
	$2.3V \le V_{CC} < 4.0V$			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



TL/DD/12060-3

Order Number COP912C-XXX/N, COP912CH-XXX/N

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

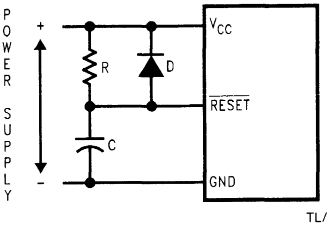
The data memory address space includes on chip RAM, I/O and registers. Data memory is addressed directly by the instruction or indirectly through B, X and SP registers. The device has 64 bytes of RAM. Sixteen bytes of RAM are mapped as "registers", these can be loaded immediately, decremented and tested.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

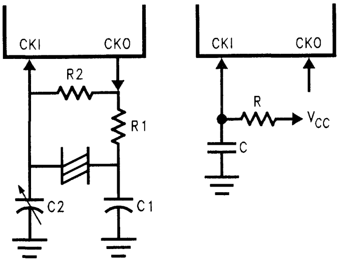
The device can be driven by a clock input which can be between DC and 5 MHz.

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.

CURRENT DRAIN

The total current drain of the chip depends on:

1. Oscillator operating 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.

Thus the total current drain is given as

$$It = I1 + I2 + I3 + I4 + I5$$

To reduce the total current drain, each of the above components must be minimum. Operating with a crystal network will draw more current than an external square-wave. The R/C mode will draw the most. 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.

The following formula may be used to compute total current drain when operating the controller in different modes.

$$I2 = C \times V \times f$$

where C = equivalent capacitance of the chip

V = operating voltage

f = CKI 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 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.

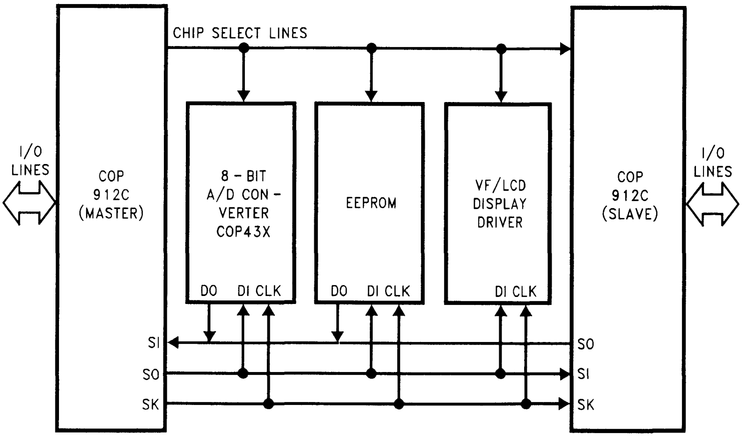
The following table details the different clock rates that may be selected.

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.

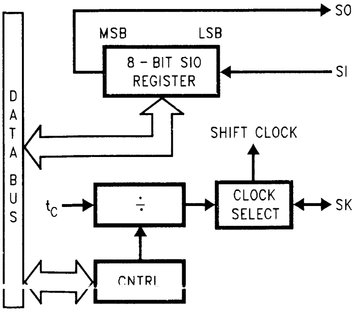
The table assumes that the control flag MSEL is set.

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.

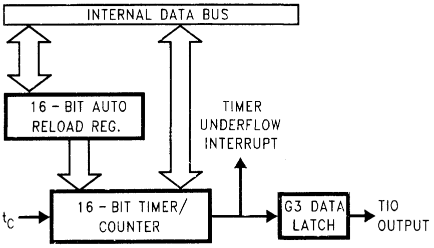
Timer/Counter

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*.



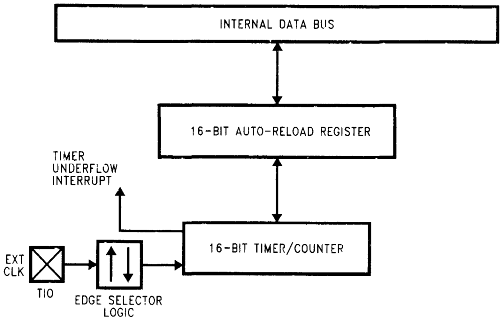
TL/DD/12060-10

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*.

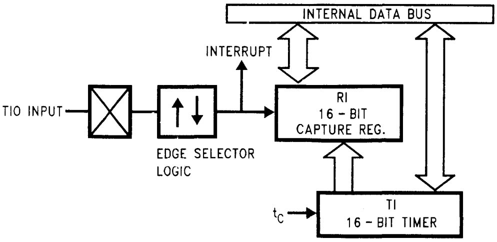


TL/DD/12060-11

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

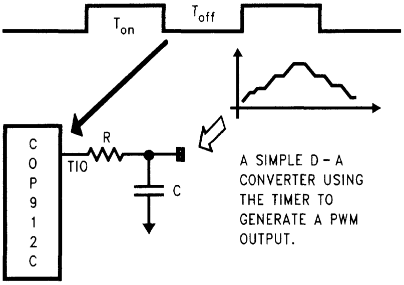
CNTRL Bits			Operation Mode	Timer Interrupt	Timer Counts On
7	6	5			
0	0	0	External Event Counter with Autoreload Register	Timer Underflow	TIO Positive Edge
0	0	1	External Event Counter with Autoreload Register	Timer Underflow	TIO Negative Edge
0	1	0	Not Allowed	Not Allowed	Not Allowed
0	1	1	Not Allowed	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.

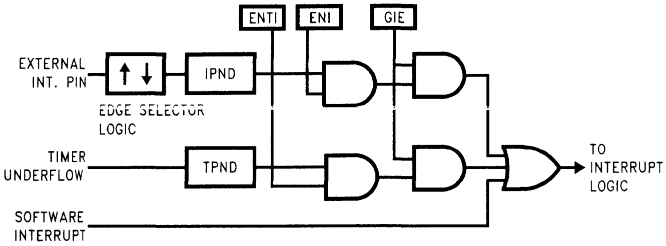
ENI and ENTI bits select external and timer interrupt respectively. Thus the user can select either or both sources to interrupt the microcontroller when GIE is enabled. IEDG selects the external interrupt edge (1 = rising edge, 0 = falling edge). The user can get an interrupt on both rising and falling 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.



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
MemI	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.

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

UPPER NIBBLE BITS 7-4

F	E	D	C	B	A	9	8	7	6	5	4	3	2	1	0	LOWER NIBBLE BITS 3-0
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

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

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

IN-CIRCUIT EMULATOR

The MetaLink iceMASTER™—COP8 Model 400 In-Circuit Emulator for the COP8 family of microcontrollers features high-performance operation, ease of use, and an extremely flexible user-interface for maximum productivity. Interchangeable probe cards, which connect to the standard common base, support the various configurations and packages of the COP8 family.

The iceMASTER provides real time, full speed emulation up to 10 MHz, 32 kBytes of emulation memory and 4k frames of trace buffer memory. The user may define as many as 32k trace and break triggers which can be enabled, disabled, set or cleared. They can be simple triggers based on code or address ranges or complex triggers based on code address, direct address, opcode value, opcode class or immediate operand. Complex breakpoints can be ANDed and ORed together. Trace information consists of address bus values, opcodes and user selectable probe clips status (external event lines). The trace buffer can be viewed as raw hex or as disassembled instructions. The probe clip bit values can be displayed in binary, hex or digital waveform formats.

During single-step operation the dynamically annotated code feature displays the contents of all accessed (read and write) memory locations and registers, as well as flow-of-control direction change markers next to each instruction executed.

The iceMASTER's performance analyzer offers a resolution of better than 6 μs. The user can easily monitor the time spent executing specific portions of code and find "hot spots" or "dead code". Up to 15 independent memory areas based on code address or label ranges can be defined. Analysis results can be viewed in bargraph format or as actual frequency count.

Emulator memory operations for program memory include single line assembler, disassembler, view, change and write to file. Data memory operations include fill, move, compare, dump to file, examine and modify. The contents of any memory space can be directly viewed and modified from the corresponding window.

The iceMASTER comes with an easy to use windowed interface. Each window can be sized, highlighted, color-controlled, added, or removed completely. Commands can be accessed via pull-down-menus and/or redefineable hot keys. A context sensitive hypertext/hyperlinked on-line help system explains clearly the options the user has from within any window.

The iceMASTER connects easily to a PC via the standard COMM port and its 115.2 kBaud serial link keeps typical program download time to under 3 seconds.

The following tables list the emulator and probe cards ordering information:

Emulator Ordering Information

Part Number	Description	Current Version
IM-COP8/400/1‡	MetaLink base unit in-circuit emulator for all COP8 devices, symbolic debugger software and RS-232 serial interface cable, with 110V @ 60 Hz Power Supply.	Host Software: Ver. 3.3 Rev. 5, Model File Rev 3.050
IM-COP8/400/2‡	MetaLink base unit in-circuit emulator for all COP8 devices, symbolic debugger software and RS-232 serial interface cable, with 220V @ 50 Hz Power Supply.	
DM-COP8/880/‡	MetaLink iceMASTER Debug Module. This is the low cost version of the MetaLink iceMASTER. Firmware: Ver. 6.07	

‡These parts include National's COP8 Assembler/Linker/Librarian Package (COP8-DEV-IBMA)

Development Support (Continued)

Probe Card Ordering Information

Part Number	Package	Voltage Range	Emulates
MHW-880C20D5PC	20 DIP	4.5V–5.5V	COP912C, COP12CH
MHW-880C20DWPC	20 DIP	2.5V–6.0V	COP912C, COP912CH
MHW-SOIC20 (20-pin SO Adapter)	20 SO	2.5V–6.0V	COP912C, COP912CH

MACRO CROSS ASSEMBLER

National Semiconductor offers a COP8 macro cross assembler. It runs on industry standard compatible PCs and supports all of the full-symbolic debugging features of the MetaLink iceMASTER emulators.

Assembler Ordering Information

Part Number	Description	Manual
COP8-DEV-IBMA	COP8 Assembler/Linker/Librarian for IBM®, PC-XT®, AT® or compatible	424410632-001

SINGLE CHIP EMULATOR DEVICE

The COP8 family is fully supported by single chip form, fit, and function emulators. For more detailed information refer to the emulation device specific data sheets and the emulator selection table below.

Single Chip Emulator Selection Table

Device Number	Package	Description	Emulates
COP8782CN	20 DIP	OTP	COP912C, COP912CH
COP8782CJ	20 DIP	UV Erasable	COP912C, COP912CH
COP8782CWM	20 SO	OTP	COP912C, COP912CH

Development Support (Continued)

PROGRAMMING SUPPORT

Programming of the single chip emulator devices is supported by different sources. The following programmers are certified for programming the One Time Programmable (OTP) devices:

EPROM Programmer Information

Manufacturer and Product	U.S. Phone Number	Europe Phone Number	Asia Phone Number
MetaLink –Debug Module	(602) 926-0797	Germany: (49-81-41) 1030	Hong Kong: (852) 737-1800
Xeltek –Superpro	(408) 745-7974	Germany: (49-20-41) 684758	Singapore: (65) 276-6433
BP Microsystems –EP-1140	(800) 225-2102	Germany: (49-89-85) 76667	Hong Kong: (852) 388-0629
Data I/O–Unisite; –System 29, –System 39	(800) 322-8246	Europe: (31-20) 622866 Germany: (49-89-85) 8020	Japan: (33) 432-6991
Abcom–COP8 Programmer		Europe: (89-80) 8707	
System General Turpro-1-FX; –APRO	(408) 263-6667	Switzerland: (31) 921-7844	Taiwan, Taipei: (2) 917-3005

INFORMATION SYSTEM

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 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. The minimum requirement for accessing the Dial-A-Helper is a Hayes compatible modem.

If the user has a PC with a communications package then files from the FILE SECTION can be down-loaded to disk for later use.

ORDER P/N: MOLE-DIAL-A-HLP

Information System Package contains:
- Dial-A-Helper Users Manual
- Public Domain Communications Software

FACTORY APPLICATIONS SUPPORT

Dial-A-Helper also provides immediate factory applications support. If a user has questions, he can leave messages on our electronic bulletin board, which we will respond to.

Voice: (800) 272-9959
Modem: CANADA/U.S.: (800) NSC-MICRO
(800) 672-6427
Baud: 14.4k
Setup: Length: 8-Bit
Parity: None
Stop Bit: 1
Operation: 24 Hrs. 7 Days

National Semiconductor

COP620C/COP622C/COP640C/COP642C/ COP820C/COP822C/COP840C/COP842C/ COP920C/COP922C/COP940C/COP942C Single-Chip microCMOS Microcontrollers

General Description

The COP820C and COP840C 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.

Features

- Low Cost 8-bit microcontroller
- Fully static CMOS
- 1 μs instruction time (10 MHz clock)
- Low current drain (2.2 mA at 3 μs instruction rate)
  Low current static HALT mode (Typically < 1 μA)
- Single supply operation: 2.5 to 6.0V
- 1024 bytes ROM/64 Bytes RAM—COP820C family
- 2048 bytes ROM/128 Bytes RAM—COP840C family
- 16-bit read/write timer operates in a variety of modes
  - Timer with 16-bit auto reload register
  - 16-bit external event counter
  - Timer with 16-bit capture register (selectable edge)
- Multi-source interrupt
  - Reset master clear
  - External interrupt with selectable edge
  - Timer interrupt or capture interrupt
  - Software interrupt
- 8-bit stack pointer (stack in RAM)
- Powerful instruction set, most instructions single byte
- BCD arithmetic instructions
- MICROWIRE/PLUS serial I/O
- 28 pin package (optionally 20 pin package)
- 24 input/output pins (28-pin package)
- Software selectable I/O options (TRI-STATE®, push-pull, weak pull-up)
- Schmitt trigger inputs on Port G
- Temperature ranges: 0°C to +70°C, −40°C to +85°C, −55°C to +125°C
- Form Factor emulation devices
- Fully supported by MetaLink's development systems

Block Diagram

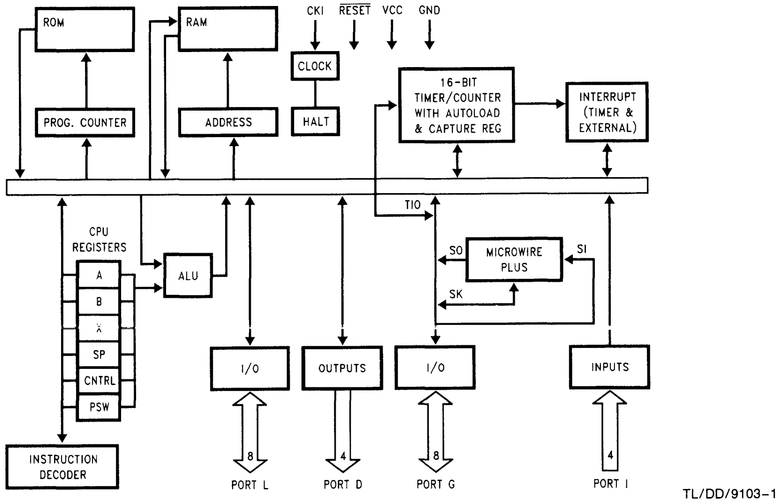


FIGURE 1

COP920C/COP922C/COP940C/COP942C

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; $0°C \leq T_A \leq +70°C$ 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

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 ≤ T_A ≤ +70°C unless otherwise specified

Parameter	Condition	Min	Typ	Max	Units
Instruction Cycle Time (tc)					
Ext., Crystal/Resonator	$V_{CC} \geq 4.0V$	1		DC	μs
(Div-by 10)	$2.3V \leq V_{CC} \leq 4.0V$	2.5		DC	μs
R/C Oscillator Mode	$V_{CC} \geq 4.0V$	3		DC	μs
(Div-by 10)	$2.3V \leq V_{CC} \leq 4.0V$	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}	$V_{CC} \geq 4.0V$	200			ns
	$2.3V \leq V_{CC} \leq 4.0V$	500			ns
t_{HOLD}	$V_{CC} \geq 4.0V$	60			ns
	$2.3V \leq V_{CC} \leq 4.0V$	150			ns
Output Propagation Delay	C_L = 100 pF, R_L = 2.2 kΩ				
t_{PD1}, t_{PD0}					
SO, SK	$V_{CC} \geq 4.0V$			0.7	μs
	$2.5V \leq V_{CC} \leq 4.0V$			1.75	μs
All Others	$V_{CC} \geq 4.0V$			1	μs
	$2.5V \leq V_{CC} \leq 4.0V$			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					
$\overline{\text{RESET}}$, 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	$V_{CC} = 6.0V$	−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 $\overline{\text{RESET}}$ 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.

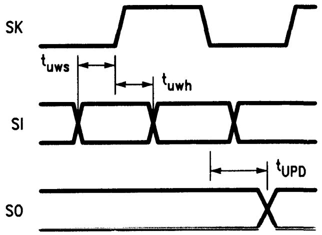
COP820C/COP822C/COP840C/COP842C

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

Parameter	Condition	Min	Typ	Max	Units
Instruction Cycle Time (tc)					
Ext. or Crystal/Resonator	$V_{CC} \geq 4.5V$	1		DC	μs
(Div-by 10)	$2.5V \leq V_{CC} < 4.5V$	2.5		DC	μs
R/C Oscillator Mode	$V_{CC} \geq 4.5V$	3		DC	μs
(Div-by 10)	$2.5V \leq V_{CC} < 4.5V$	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}	$V_{CC} \geq 4.5V$	200			ns
	$2.5V \leq V_{CC} < 4.5V$	500			ns
t_{HOLD}	$V_{CC} \geq 4.5V$	60			ns
	$2.5V \leq V_{CC} < 4.5V$	150			ns
Output Propagation Delay	$C_L = 100$ pF, $R_L = 2.2$ kΩ				
t_{PD1}, t_{PD0}					
SO, SK	$V_{CC} \geq 4.5V$			0.7	μs
	$2.5V \leq V_{CC} < 4.5V$			1.75	μs
All Others	$V_{CC} \geq 4.5V$			1	μs
	$2.5V \leq V_{CC} < 4.5V$			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: $-55°C \leq T_A \leq +125°C$ 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

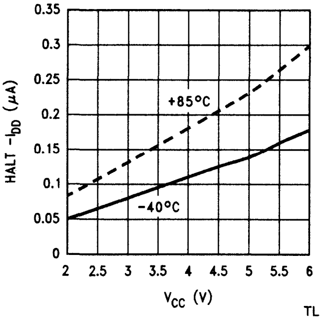
AC Electrical Characteristics $-55^\circ C \leq T_A \leq +125^\circ C$ unless otherwise specified

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).

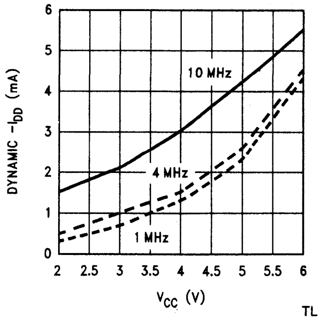
Typical Performance Characteristics (−40°C ≤ T_A ≤ +85°C)

Halt—I_{DD}



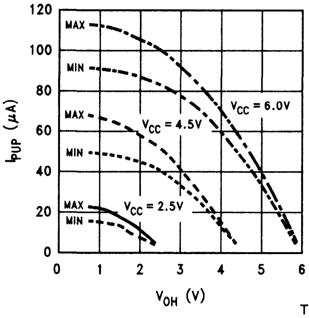
TL/DD/9103-20

Dynamic—I_{DD} (Crystal Clock Option)



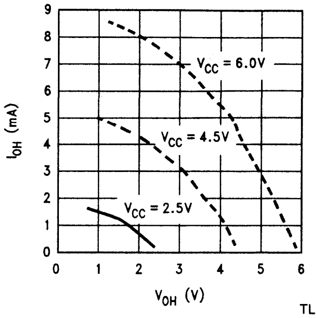
TL/DD/9103-21

Port L/G Weak Pull-Up Source Current



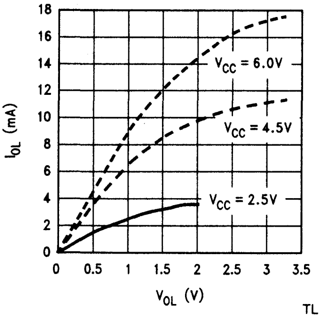
TL/DD/9103-22

Port L/G Push-Pull Source Current



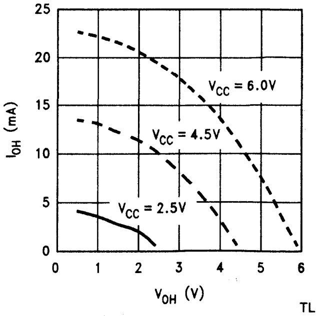
TL/DD/9103-23

Port L/G Push-Pull Sink Current



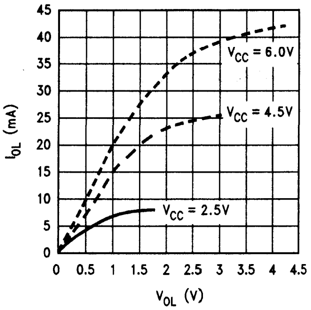
TL/DD/9103-24

Port D Source Current



TL/DD/9103-25

Port D Sink Current



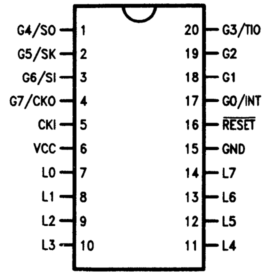
TL/DD/9103-26

Connection Diagrams

DUAL-IN-LINE PACKAGE

SURFACE MOUNT

20 DIP

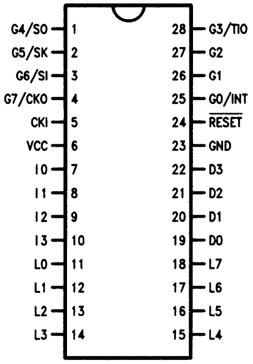


TL/DD/9103–3

Top View

Order Number COP622C-XXX/N, COP642C-XXX/N, COP822C-XXX/N, COP842C-XXX/N, COP922C-XXX/N or COP942C-XXX/N
See NS Package Number N20A

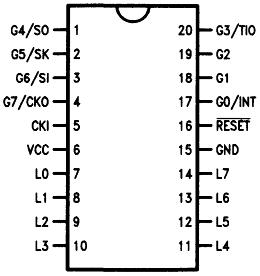
28 DIP



TL/DD/9103–5

Order Number COP620C-XXX/N, COP640C-XXX/N, COP820C-XXX/N, COP840C-XXX/D, COP920C-XXX/N or COP940C-XXX/N
See NS Package Number N28B

20 SO Wide

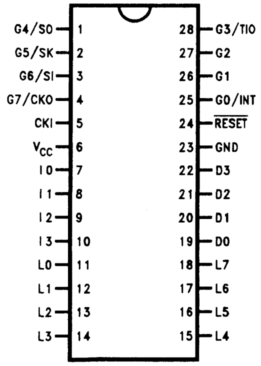


TL/DD/9103–3

Top View

Order Number COP822C-XXX/WM, COP842C-XXX/WM, COP922C-XXX/WM or COP942C-XXX/WM
See NS Package Number M20B

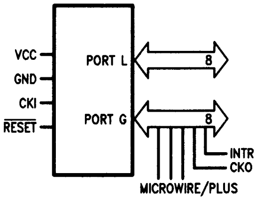
28-Lead SO



TL/DD/9103–18

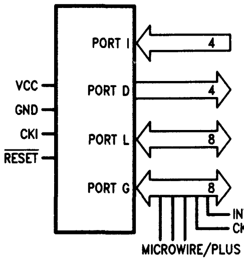
Order Number COP820C-XXX/WM, COP840C-XXX/WM, COP920C-XXX/WM or COP940C-XXX/WM
See NS Package Number M28A

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.

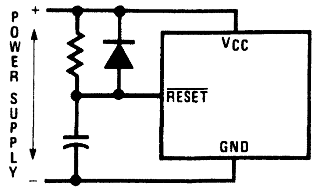
Note: RAM contents are undefined upon power-up.

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.

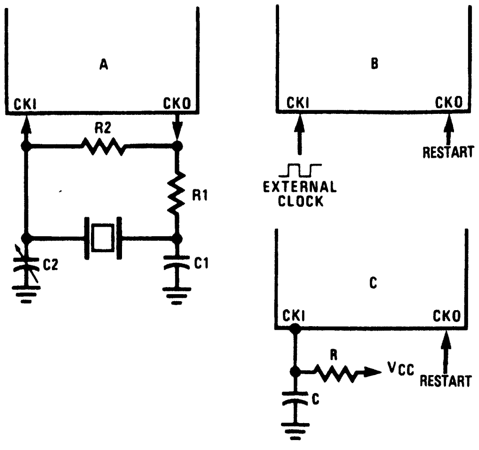
B. EXTERNAL OSCILLATOR

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.

CURRENT DRAIN

The total current drain of the chip depends on:

1) Oscillator operating 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

Thus the total current drain, It is given as

$$It = I1 + I2 + I3 + I4 + I5$$

To reduce the total current drain, each of the above components must be minimum.

Operating with a crystal network will draw more current than an external square-wave. The R/C mode will draw the most. 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.

$$I2 = C x V x f$$

Where

C = equivalent capacitance of the chip.

V = operating voltage

f = CKI frequency

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.

Functional Description (Continued)

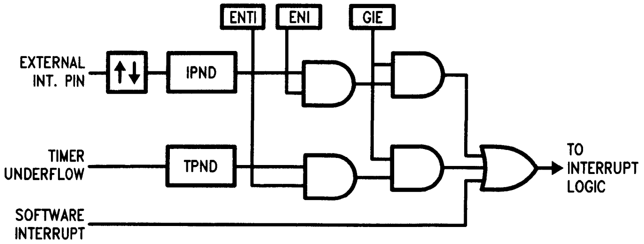


FIGURE 6. Interrupt Block Diagram

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*.)

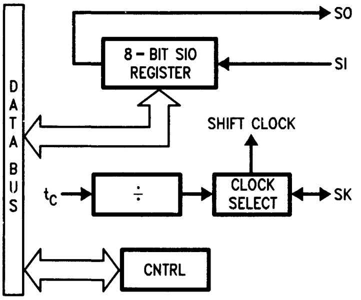
Functional Description (Continued)

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

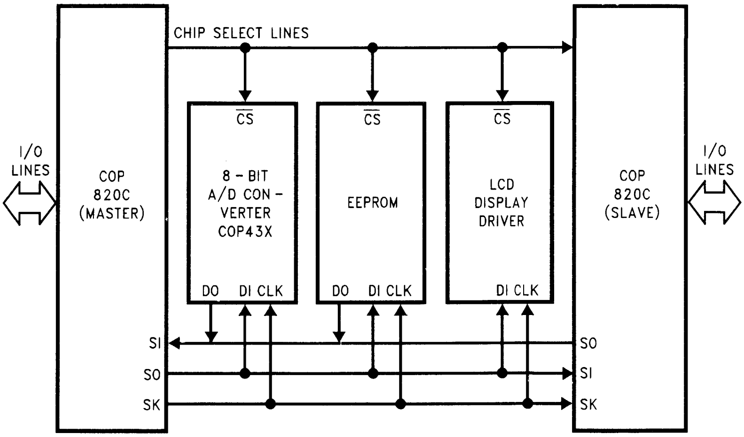
In this mode of operation, the timer T1 counts down at the instruction cycle rate. Upon underflow the value in the register R1 gets automatically reloaded into the timer which continues to count down. The timer underflow can be programmed to interrupt the microcontroller. A bit in the control register 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*.)



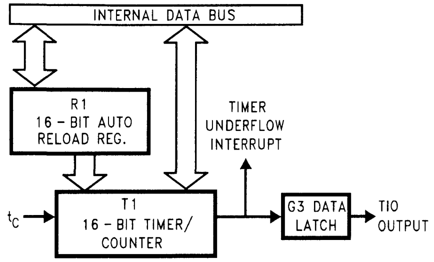
TL/DD/9103-13

FIGURE 8. MICROWIRE/PLUS Application

Functional Description (Continued)

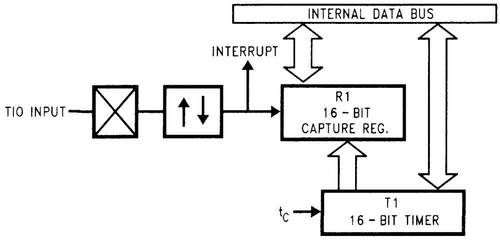
TABLE V. Timer Operating Modes

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



TL/DD/9103-15

FIGURE 9. Timer/Counter Auto Reload Mode Block Diagram

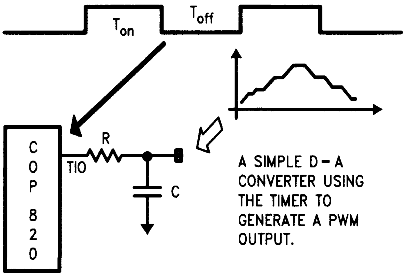


TL/DD/9103-14

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.



A SIMPLE D-A CONVERTER USING THE TIMER TO GENERATE A PWM OUTPUT.

TL/DD/9103-16

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.

1

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 / **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 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.

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 package
= 2 N.A.
= 3 20 pin 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

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 detail.

Development Support

IN-CIRCUIT EMULATOR

The MetaLink iceMASTER™—COP8 Model 400 In-Circuit Emulator for the COP8 family of microcontrollers features high-performance operation, ease of use, and an extremely flexible user-interface for maximum productivity. Interchangeable probe cards, which connect to the standard common base, support the various configurations and packages of the COP8 family.

The iceMASTER provides real time, full speed emulation up to 10 MHz, 32 kBytes of emulation memory and 4k frames of trace buffer memory. The user may define as many as 32k trace and break triggers which can be enabled, disabled, set or cleared. They can be simple triggers based on code or address ranges or complex triggers based on code address, direct address, opcode value, opcode class or immediate operand. Complex breakpoints can be ANDed and ORed together. Trace information consists of address bus values, opcodes and user selectable probe clips status (external event lines). The trace buffer can be viewed as raw hex or as disassembled instructions. The probe clip bit values can be displayed in binary, hex or digital waveform formats.

During single-step operation the dynamically annotated code feature displays the contents of all accessed (read and write) memory locations and registers, as well as flow-of-control direction change markers next to each instruction executed.

The iceMASTER's performance analyzer offers a resolution of better than 6 μs. The user can easily monitor the time spent executing specific portions of code and find "hot spots" or "dead code". Up to 15 independent memory areas based on code address or label ranges can be defined. Analysis results can be viewed in bargraph format or as actual frequency count.

Emulator memory operations for program memory include single line assembler, disassembler, view, change and write to file. Data memory operations include fill, move, compare, dump to file, examine and modify. The contents of any memory space can be directly viewed and modified from the corresponding window.

The iceMASTER comes with an easy to use windowed interface. Each window can be sized, highlighted, color-controlled, added, or removed completely. Commands can be accessed via pull-down-menus and/or redefineable hot keys. A context sensitive hypertext/hyperlinked on-line help system explains clearly the options the user has from within any window.

The iceMASTER connects easily to a PC via the standard COMM port and its 115.2 kBaud serial link keeps typical program download time to under 3 seconds.

The following tables list the emulator and probe cards ordering information:

Emulator Ordering Information

Part Number	Description	Current Version
IM-COP8/400/1‡	MetaLink base unit in-circuit emulator for all COP8 devices, symbolic debugger software and RS-232 serial interface cable, with 110V @ 60 Hz Power Supply.	HOST SOFTWARE: VER. 3.3 REV.5, Model File Rev 3.050.
IM-COP8/400/2‡	MetaLink base unit in-circuit emulator for all COP8 devices, symbolic debugger software and RS-232 serial interface cable, with 220V @ 50 Hz Power Supply.	
DM-COP8/880/‡	MetaLink iceMASTER Debug Module. This is the low cost version of the MetaLink iceMASTER. Firmware: Ver.6.07.	

‡These parts include National's COP8 Assembler/Linker/Librarian Package (COP8-DEV-IBMA)

Development Support (Continued)

Probe Card Ordering Information

Part Number	Package	Voltage Range	Emulates
MHW-880C20D5PC	20 DIP	4.5V–5.5V	COP822C, 842C, 8782C
MHW-880C20DWPC	20 DIP	2.5V–6.0V	COP822C, 842C, 8782C
MHW-880C28D5PC	28 DIP	4.5V–5.5V	COP820C, 840C, 881C, 8781C
MHW-880C28DWPC	28 DIP	2.5V–6.0V	COP820C, 840C, 881C, 8781C

MACRO CROSS ASSEMBLER

National Semiconductor offers a COP8 macro cross assembler. It runs on industry standard compatible PCs and supports all of the full-symbolic debugging features of the MetaLink iceMASTER emulators.

Assembler Ordering Information

Part Number	Description	Manual
COP8-DEV-IBMA	COP8 Assembler/ Linker/Librarian for IBM®, PC-XT®, AT® or compatible	424410632-001

SINGLE CHIP EMULATOR DEVICE

The COP8 family is fully supported by single chip form, fit, and function emulators. For more detailed information refer to the emulation device specific data sheets and the emulator selection table below.

Single Chip Emulator Selection Table

Device Number	Clock Option	Package	Description	Emulates
COP8781CN	Programmable	28 DIP	One Time Programmable (OTP)	COP840C, COP820C
COP8781CJ	Programmable	28 DIP	UV Erasable	COP840C, COP820C
COP8781CWM	Programmable	28 SO	OTP	COP840C, COP820C
COP8782CN	Programmable	20 DIP	OTP	COP842C, COP822C
COP8782CJ	Programmable	20 DIP	UV Erasable	COP842C, COP822C
COP8782CWM	Programmable	20 SO	OTP	COP842C, COP822C

Development Support (Continued)

PROGRAMMING SUPPORT

Programming of the single chip emulator devices is supported by different sources. The following programmers are certified for programming the One Time Programmable (OTP) devices:

EPROM Programmer Information

Manufacturer and Product	U.S. Phone Number	Europe Phone Number	Asia Phone Number
MetaLink-Debug Module	(602) 926-0797	Germany: +49-81-41-1030	Hong Kong: +852-737-1800
Xeltek-Superpro	(408) 745-7974	Germany: +49-20-41 684758	Singapore: +65 276 6433
BP Microsystems-EP-1140	(800) 225-2102	Germany: +49-89-857 66 67	Hong Kong: +852 388 0629
Data I/O- Unisite; -System 29, -System 39	(800) 322-8246	Europe: +31-20-622866 Germany: +49-89-85-8020	Japan: +33-432-6991
Abcom- COP8 Programmer		Europe: +89 80 8707	
System General Turpro-1-FX; -APRO	(408) 263-6667	Switzerland: +31-921-7844	Taiwan Taipei: +2-9173005

INFORMATION SYSTEM

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 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. The minimum requirement for accessing the Dial-A-Helper is a Hayes compatible modem.

If the user has a PC with a communications package then files from the FILE SECTION can be down-loaded to disk for later use.

ORDER P/N: MOLE-DIAL-A-HLP

Information System Package contains:

Dial-A-Helper Users Manual

Public Domain Communications Software

FACTORY APPLICATIONS SUPPORT

Dial-A-Helper also provides immediate factory applications support. If a user has questions, he can leave messages on our electronic bulletin board, which we will respond to.

Voice:	(800) 272-9959		
Modem:	CANADA/U.S.:	(800) NSC-MICRO	
		(800) 672-6427	
	Baud:	14.4k	
	Setup:	Length:	8-Bit
		Parity:	None
		Stop Bit:	1
	Operation:	24 Hrs. 7 Days	

National Semiconductor

PRELIMINARY

COP820CJ/COP822CJ/COP823CJ Single-Chip microCMOS Microcontroller

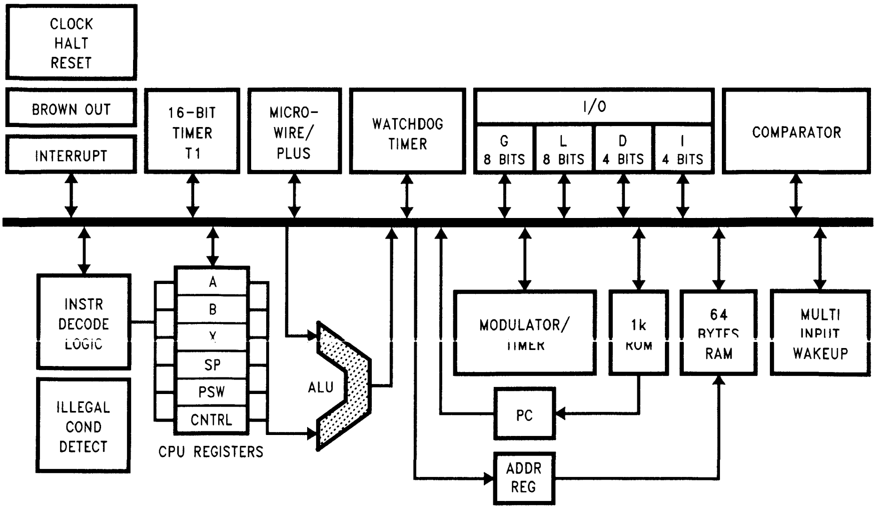
General Description

The COP820CJ is a member of the COPS™ 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.

Features

- Low cost 8-bit Microcontroller
- Fully static CMOS
- 1 μs instruction time
- Low current drain
  - Low current static HALT mode
- Single supply operation: 2.5V to 6.0V
- 1024 x 8 on-chip ROM
- 64 bytes on-chip RAM
- WATCHDOG Timer
- Comparator
- Modulator/Timer (High speed PWM Timer for IR Transmission)
- Multi-Input Wakeup (on the 8-bit Port L)
- Brown Out Protection
- 4 high current I/O pins with 15 mA sink capability
- MICROWIRE/PLUS™ serial I/O
- 16-bit read/write timer operates in a variety of modes
  - Timer with 16-bit auto reload register
  - 16-bit external event counter
  - Timer with 16-bit capture register (selectable edge)
- Multi-source interrupt
  - External interrupt with selectable edge
  - Timer interrupt or capture interrupt
  - Software interrupt
- 8-bit stack pointer (stack in RAM)
- Powerful instruction set, most instructions single byte
- BCD arithmetic instructions
- 28- and 20-pin DIP/SO package or 16-pin SO package
- Software selectable I/O options (TRI-STATE®, push-pull, weak pull-up)
- Schmitt trigger inputs on Port G and Port L
- Fully supported by MetaLink's development systems
- One-Time Programmable (OTP) emulator devices

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 $-40°C \leq T_A \leq +85°C$ 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.

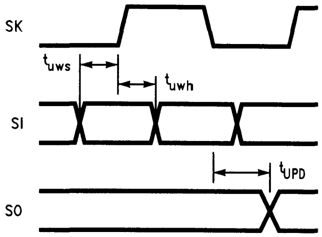
Note 4: Pins G6 and $\overline{\text{RESET}}$ are designed with a high voltage input network. These pins allow input voltages greater than V_{CC} and the pins will have sink current to V_{CC} when biased at voltages greater than V_{CC} (the pins do not have source current when biased at a voltage below 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

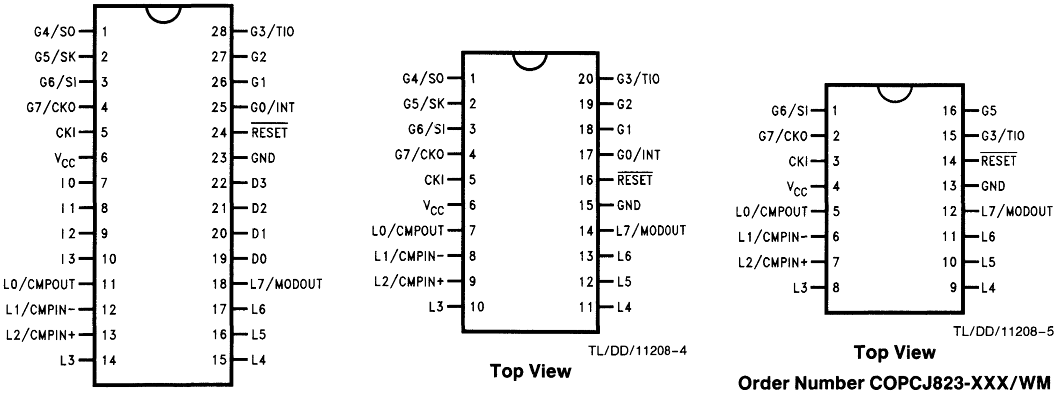
FIGURE 2. MICROWIRE/PLUS Timing

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

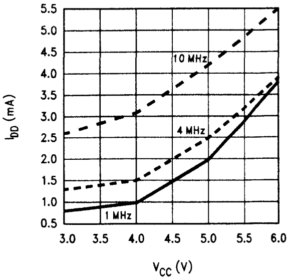
Top View

Order Number COPCJ823-XXX/WM

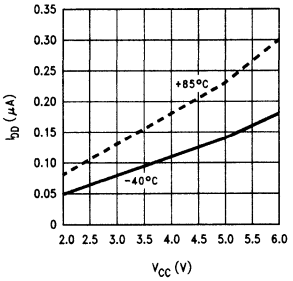
FIGURE 3. Connection Diagrams

Typical Performance Characteristics

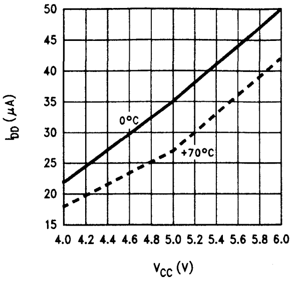
Dynamic—I_{DD} vs V_{CC} (Crystal Clock Option)



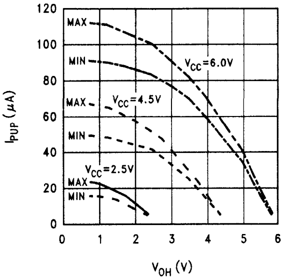
Halt—I_{DD} vs V_{CC} (Brown Out Disabled)



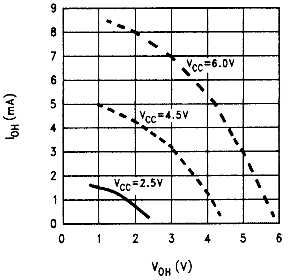
Halt—I_{DD} vs V_{CC} (Brown Out Enabled)



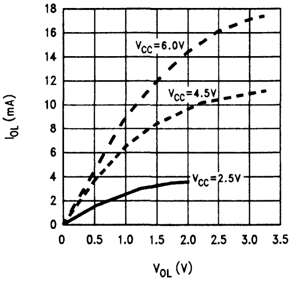
Ports L/G Weak Pull-Up Source Current



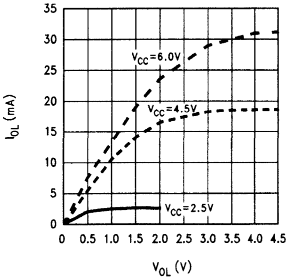
Ports L/G Push-Pull Source Current



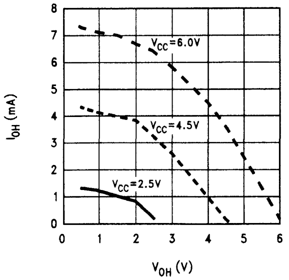
Ports L/G Push-Pull Sink Current



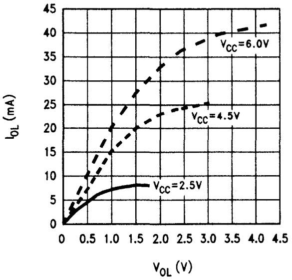
Ports L4–L7 Sink Current



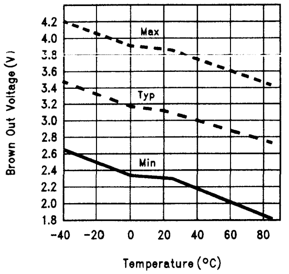
Port D Source Current



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

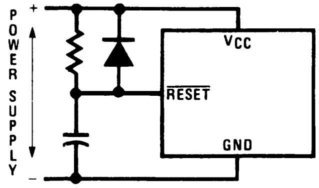
Functional Description (Continued)

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 $\overline{RESET}$ pin low. The Development System does not emulate the 256tc delay.

Brown Out Protection

An on-board protection 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

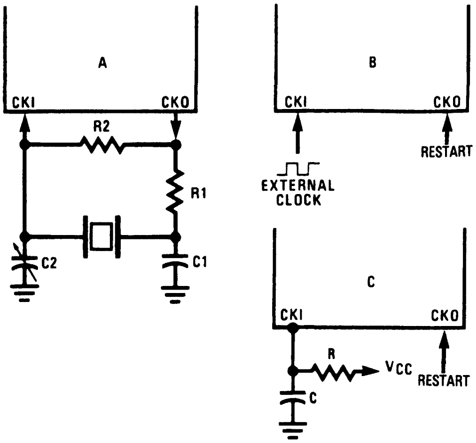
CKI can be driven by an external clock signal provided it meets the specified duty cycle, rise and fall times, and input levels. CKO is available as a general purpose input G7 and/or Halt control.

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)

Current Drain

The total current drain of the chip depends on:

1. Oscillator operating 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. DC current caused by the comparator (if comparator is enabled) - I6
7. DC current caused by the Brown Out - I7

Thus the total current drain 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. Operating with a crystal network will draw more current than an external square-wave. The R/C-mode will draw the most. 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.

The following formula may be used to compute total current drain when operating the controller in different modes.

$$I2 = C \times V \times f$$

where: C = equivalent capacitance of the chip
V = operating voltage
f = CKI 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 (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.

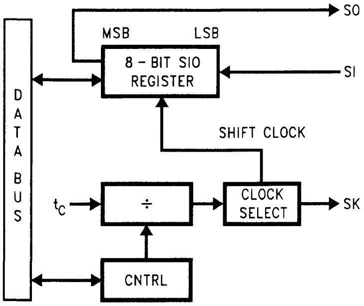
If the Brown Out Enable mask option is selected, the Brown Out circuit remains active during the HALT mode causing additional current to be drawn.

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

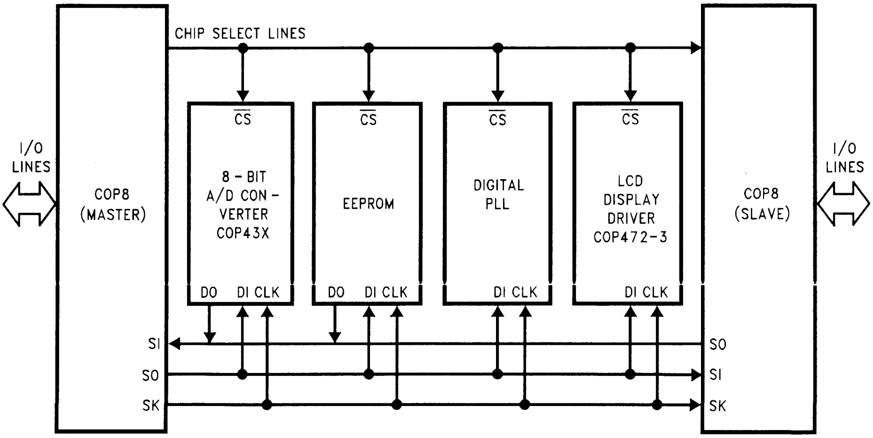
Setting the BUSY bit in the PSW register causes the MICROWIRE/PLUS arrangement to start shifting the data. It gets reset when eight data bits have been shifted. The user may reset the BUSY bit by software to allow less than 8 bits to shift. The device may enter the MICROWIRE/PLUS mode either as a Master 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

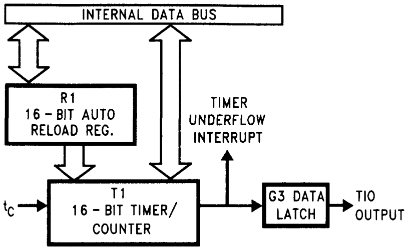
The device has a powerful 16-bit timer with an associated 16-bit register enabling it to perform extensive timer functions. The timer T1 and its register R1 are each organized as two 8-bit read/write registers. Control bits in the register CNTRL allow the timer to be started and stopped under software 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)*.

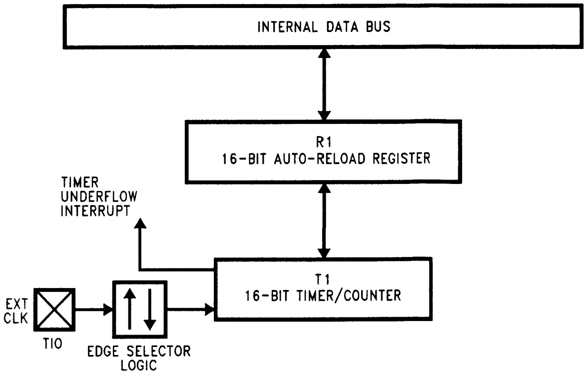


TL/DD/11208-24

FIGURE 8. Timer/Counter Auto Reload Mode Block Diagram

TABLE V. Timer Operating Modes

CNTRL Bits 765	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



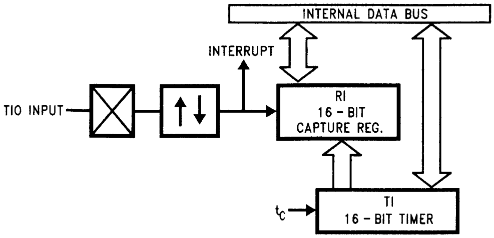
TL/DD/11208-29

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)*.

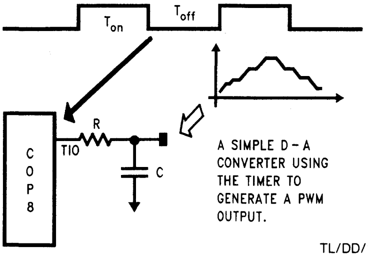


TL/DD/11208-25

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

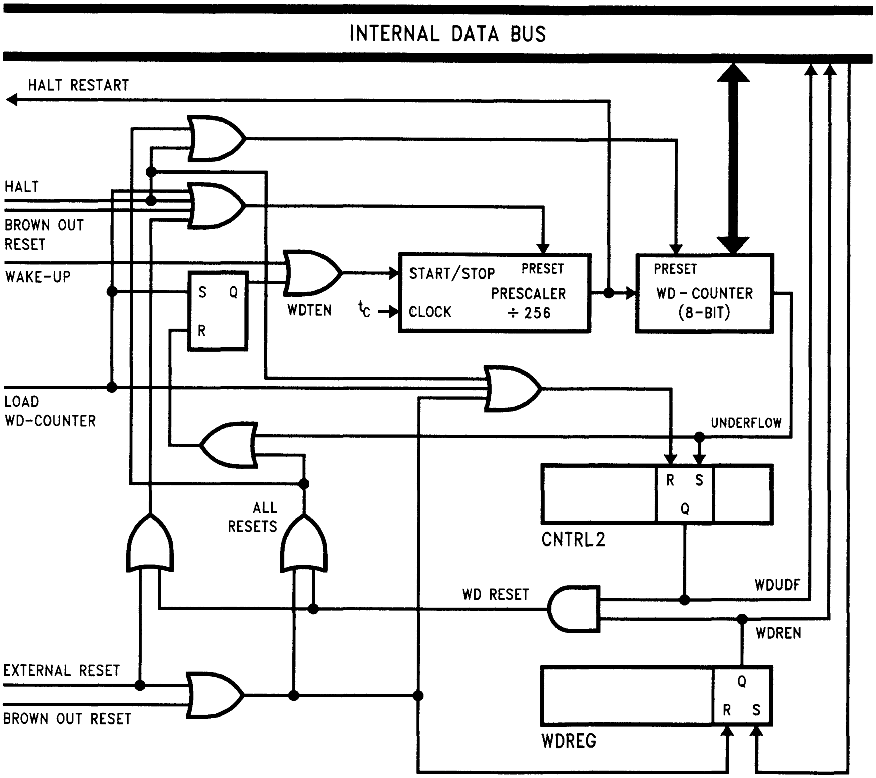
This bit resides in the CNTRL2 Register. The bit is set when the WATCHDOG timer underflows. The underflow resets the device if the WATCHDOG reset enable bit is set (WDREN = 1). Otherwise, WDUDF can be used as the timer underflow flag. The bit is cleared upon Brown-Out reset, external reset, load to the 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:

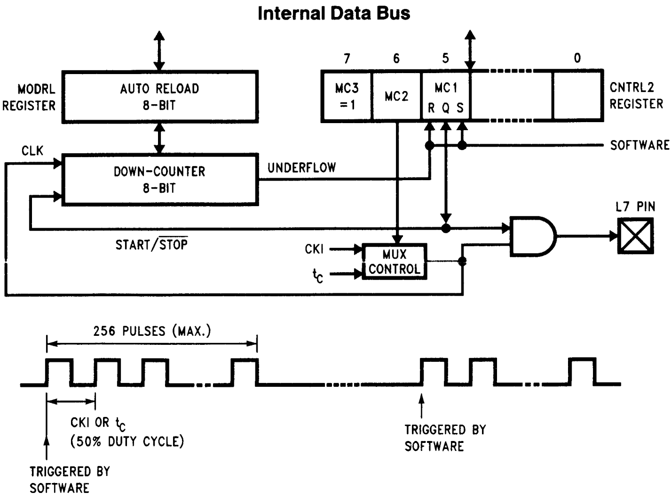
When MC3 = 0 and MC2 = 1, a variable duty cycle PWM signal is generated on the L7 output pin. The counter is clocked by tC. In this mode the 16-bit timer T1 along with the 8-bit down counter are used to generate a variable duty cycle PWM signal. The timer T1 underflow sets MC1 which starts the down counter 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)

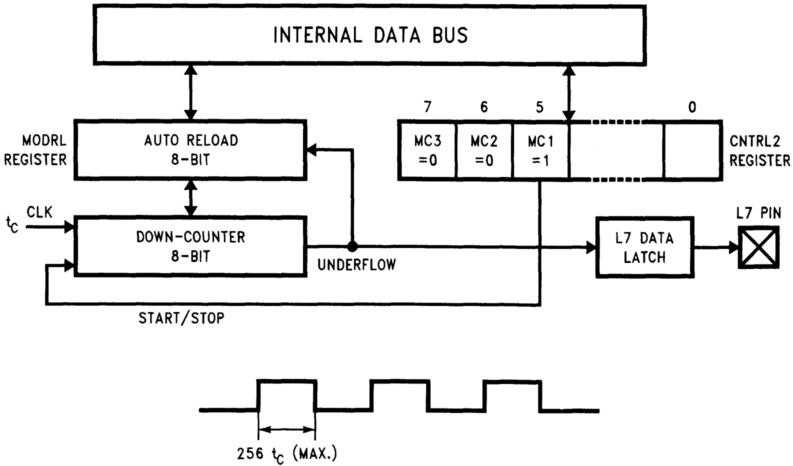
Note: MC1, MC2 and MC3 control bits are cleared upon reset.



TL/DD/11208-16

FIGURE 13. Mode 1: Modulator Block Diagram/Output Waveform

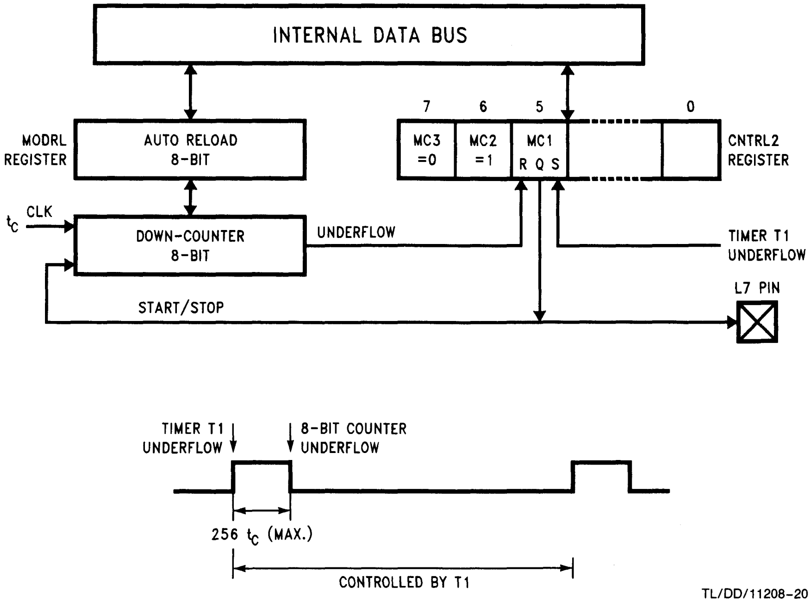
Modulator/Timer (Continued)



TL/DD/11208-17

TL/DD/11208-18

FIGURE 14. Mode 2a: 50% Duty Cycle Output



TL/DD/11208-19

TL/DD/11208-20

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)

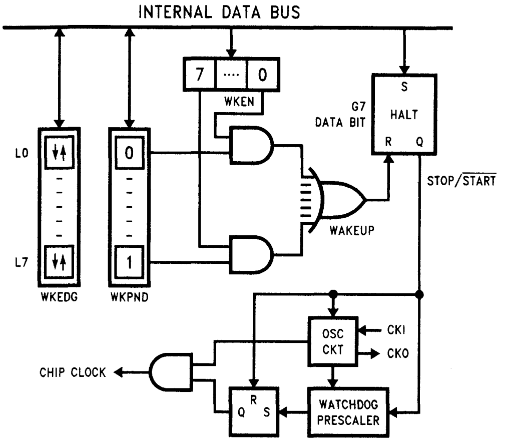


FIGURE 16. Multi-Input Wakeup Logic

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.

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.

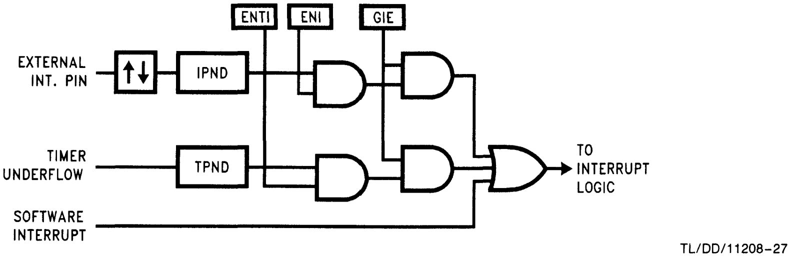


FIGURE 17. Interrupt Block Diagram

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

OPCODE LIST

Bits 7–4

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 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.

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

IN-CIRCUIT EMULATOR

The MetaLink iceMASTER™-COP8 Model 400 In-Circuit Emulator for the COP8 family of microcontrollers features high-performance operation, ease of use, and an extremely flexible user-interface for maximum productivity. Interchangeable probe cards, which connect to the standard common base, support the various configurations and packages of the COP8 family.

The iceMASTER provides real time, full speed emulation up to 10 MHz, 32 kBytes of emulation memory and 4k frames of trace buffer memory. The user may define as many as 32k trace and break triggers which can be enabled, disabled, set or cleared. They can be simple triggers based on code or address ranges or complex triggers based on code address, direct address, opcode value, opcode class or immediate operand. Complex breakpoints can be ANDed and ORed together. Trace information consists of address bus values, opcodes and user selectable probe clips status (external event lines). The trace buffer can be viewed as raw hex or as disassembled instructions. The probe clip bit values can be displayed in binary, hex or digital waveform formats.

During single-step operation the dynamically annotated code feature displays the contents of all accessed (read and write) memory locations and registers, as well as flow-of-control direction change markers next to each instruction executed.

The iceMASTER's performance analyzer offers a resolution of better than 6 μs. The user can easily monitor the time spent executing specific portions of code and find "hot spots" or "dead code". Up to 15 independent memory areas based on code address or label ranges can be defined. Analysis results can be viewed in bargraph format or as actual frequency count.

Emulator memory operations for program memory include single line assembler, disassembler, view, change and write to file. Data memory operations include fill, move, compare, dump to file, examine and modify. The contents of any memory space can be directly viewed and modified from the corresponding window.

The iceMASTER comes with an easy to use windowed interface. Each window can be sized, highlighted, color-controlled, added, or removed completely. Commands can be accessed via pull-down-menus and/or redefinable hot keys. A context sensitive hypertext/hyperlinked on-line help system explains clearly the options the user has from within any window.

The iceMASTER connects easily to a PC via the standard COMM port and its 115.2 kBaud serial link keeps typical program download time to under 3 seconds.

The following tables list the emulator and probe cards ordering information.

Development Support (Continued)

Emulator Ordering Information

Part Number	Description	Current Version
IM-COP8/400/1‡	MetaLink base unit in-circuit emulator for all COP8 devices, symbolic debugger software and RS 232 serial interface cable, with 110V @ 60 Hz Power Supply.	HOST SOFTWARE: VER. 3.3 REV.5, Model File Rev 3.050.
IM-COP8/400/2‡	MetaLink base unit in-circuit emulator for all COP8 devices, symbolic debugger software and RS 232 serial interface cable, with 220V @ 50 Hz Power Supply.	
DM-COP8/820CJ‡	MetaLink IceMaster Debug Module. This is the low cost version of MetaLinks IceMaster. Firmware: Ver. 6.07.	

‡These parts include National's COP8 Assembler/Linker/Librarian Package (COP8-DEV-IBMA).

Probe Card Ordering Information

Part Number	Package	Voltage Range	Emulates
MH-820CJ20D5PC	20 DIP	4.5V–5.5V	COP822CJ
MHW-820CJ20DWPC	20 DIP	2.3V–6.0V	COP822CJ
MHW-820CJ28D5PC	28 DIP	4.5V–5.5V	COP820CJ
MHW-820CJ28DWPC	28 DIP	2.3V–6.0V	COP820CJ

MACRO CROSS ASSEMBLER

National Semiconductor offers a COP8 macro cross assembler. It runs on industry standard compatible PCs and supports all of the full-symbolic debugging features of the MetaLink iceMASTER emulators.

Assembler Ordering Information

Part Number	Description	Manual
COP8-DEV-IBMA	COP8 Assembler/ Linker/Librarian for IBM® PC-XT®, AT® or compatible	424410632-001

SINGLE CHIP EMULATOR

The COP820CJ family is supported by One-Time Programmable (OTP) emulators. For more detailed information refer to the emulation device specific data sheets and the emulator selection table below.

PROGRAMMING SUPPORT

Programming of the single chip emulator devices is supported by different sources.

Development Support (Continued)

The following programmers are ceritfied for programming the One-Time Programmable (OTP) devices:

EPROM Programmer Information

Manufacturer and Product	U.S. Phone Number	Europe Phone Number	Asia Phone Number
MetaLink-Debug Module	(602) 926-0797	Germany: +49-8141-1030	Hong Kong: +852-737-1800
Xeltek-Superpro	(408) 745-7974	Germany: +49-2041-684758	Singapore: +65-276-6433
BP Microsystems-EP-1140	(800) 225-2102	Germany: +49-89-857-66-67	Hong Kong: +852-388-0629
Data I/O-Unisite; -System 29, -System 39	(800) 322-8246	Europe: +31-20-622866 Germany: +49-89-858020	Japan: +33-432-6991
Abcom-COP8 Programmer		Europe: +89-808707	
System General Turpro-1-FX; -APRO	(408) 263-6667	Switzerland: +31-921-7844	Taiwan Taipei: +2-9173005

One-Time Programmable (OTP) Selection Table

Device Number	Package	Emulates
COP8720CJN	28 DIP	COP820CJ
COP8720CJWM	28 SO	COP820CJ
COP8722CJWM	20 DIP	COP822CJ

DIAL-A-HELPER

Dial-A-Helper is a service provided by the Microcontroller Applications Group. The Dial-A-Helper is an Electronic Bulletin Board information system.

INFORMATION SYSTEM

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 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. The minimum requirement for accessing the Dial-A-Helper is a Hayes compatible modem.

If the user has a PC with a communications package then files from the FILE SECTION can be down-loaded to disk for later use.

FACTORY APPLICATIONS SUPPORT

Dial-A-Helper also provides immediate factory applications support. If a user has questions, he can leave messages on our electronic bulletin board.

Voice: (800) 272-9959
Modem: Canada/
U.S.: (800) NSC-MICRO
(800) 672-6427
Baud: 14.4k
Setup: Length: 8-Bit
Parity: None
Stop Bit: 1
Operation: 24 Hrs. 7 Days

National Semiconductor

COP8620C/COP8622C/COP8640C/COP8642C/ COP86L20C/COP86L22C/COP86L40C/COP86L42C Single-Chip microCMOS Microcontrollers

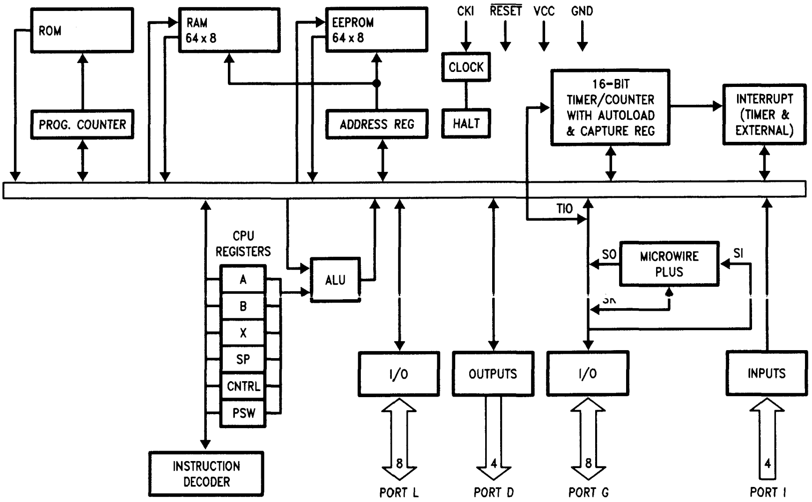
General Description

The COP8620C/COP8640C are members of the COPS™ microcontroller family. They are fully static parts, fabricated using double-metal silicon gate microCMOS technology. These low cost microcontrollers are complete microcomputers containing all system timing, interrupt logic, ROM, RAM, EEPROM, 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 4.5V to 6.0V. High throughput is achieved with an efficient, regular instruction set operating at a 1 microsecond per instruction rate.

Features

- Low Cost 8-bit microcontroller
- Fully static CMOS
- 1 μs instruction time
- Low current drain (2.2 mA at 3 μs instruction rate)
  Low current static HALT mode (Typically < 1 μA)
- Single supply operation: 4.5 to 6.0V
- 2048 Bytes ROM/64 Bytes RAM/64 Bytes EEPROM on COP8640C
- 1024 bytes ROM/64 bytes RAM/64 bytes EEPROM on COP8620C
- 16-bit read/write timer operates in a variety of modes
  - — Timer with 16-bit auto reload register
  - — 16-bit external event counter
  - — Timer with 16-bit capture register (selectable edge)
- Multi-source interrupt
  - — Reset master clear
  - — External interrupt with selectable edge
  - — Timer interrupt or capture interrupt
  - — Software interrupt
- 8-bit stack pointer (stack in RAM)
- Powerful instruction set, most instructions single byte
- BCD arithmetic instructions
- MICROWIRE PLUS™ serial I/O
- 28 pin package (optional 20 pin package)
- 24 input/output pins (28-pin package)
- Software selectable I/O options (TRI-STATE®, push-pull, weak pull-up)
- Schmitt trigger inputs on Port G
- Temperature range: −40°C to +85°C, −55°C to +125°C
- Hybrid emulator devices
- Fully supported by MetaLink's Development Systems

Block Diagram



TL/DD/10366-1

FIGURE 1

COP86L20C/COP86L22C/COP86L40C/COP86L42C

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 −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
Operating Voltage during EEPROM Write		4.5		6.0	V
Supply Current (Note 2)					
CKI = 10 MHz	V_{CC} = 6V, tc = 1 μs			9	mA
Supply Current during Write Operation (Note 2)					
CKI = 10 MHz	V_{CC} = 6.0V, tc = 1 μs			15	mA
HALT Current (Note 3)	V_{CC} = 6V, CKI = 0 MHz		<1	10	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	−2		+2	μA
Input Pullup Current	V_{CC} = 6.0V, V_{IN} = 0V	−40		−250	μA
G Port Input 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_{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			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 (Sink)				15	mA
All Others				3	mA
Maximum Input Current (Note 4) Without Latchup (Room Temp) (Note 5)	Room Temp			±100	mA
RAM Retention Voltage, Vr	500 ns Rise and Fall Time (Min)	2.0			V
Input Capacitance (Note 5)				7	pF
EEPROM Characteristics					
EEPROM Write Cycle Time				10	ms
EEPROM Number of Write Cycles				10,000	Cycle
EEPROM Data Retention		10			Years

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 are at TRI-STATE and tied to ground, all outputs low and tied to ground.

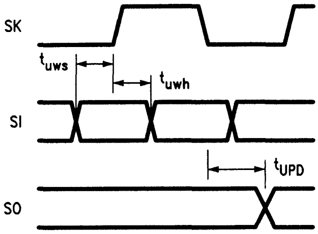
Note 4: Pins G6 and $\overline{\text{RESET}}$ are designed with a high voltage input network for factory testing. These pins allow input voltages greater than V_{CC} and the pins will have sink current to V_{CC} when biased at voltages greater than V_{CC} (the pins do not have source current when biased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

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, Crystal/Resonator	$V_{CC} \ge 4.5V$	1		DC	µs
(Div-by 10)	$2.5V \le V_{CC} \le 6.0V$	2.5		DC	µs
R/C Oscillator Mode	$V_{CC} \ge 4.5V$	3		DC	µs
(Div-by 10)	$2.5V \le V_{CC} \le 6.0V$	7.5		DC	µs
CKI Clock Duty Cycle (Note 5)		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}		200			ns
t_{HOLD}		60			ns
Output Propagation Delay	$C_L = 100$ pF, $R_L = 2.2$ kΩ				
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 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		1.0			µs

Note 5: Parameter sampled (not 100% tested).

Timing Diagram



TL/DD/10366-19

FIGURE 2. MICROWIRE/PLUS Timing

COP8620C/COP8622C/COP8640C/COP8642C

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 $-40°C \le T_A \le +85°C$ 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 (Note 2)					
CKI = 10 MHz	$V_{CC} = 6V$, tc = 1 μs			9	mA
Supply Current during Write Operation (Note 2)					
CKI = 10 MHz	$V_{CC} = 6.0V$, tc = 1 μs			15	mA
HALT Current (Note 3)	$V_{CC} = 6V$, CKI = 0 MHz		<1	10	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	$V_{CC} = 6.0V$	−2		+2	μA
Input Pullup Current	$V_{CC} = 6.0V$, $V_{IN} = 0V$	−40		−250	μA
G Port Input 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
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 (Note 4) Without Latchup (Room Temp) (Note 5)	Room Temp			±100	mA
RAM Retention Voltage, Vr	500 ns Rise and Fall Time (Min)	2.0			V
Input Capacitance (Note 5)				7	pF
EEPROM Characteristics					
EEPROM Write Cycle Time				10	ms
EEPROM Number of Write Cycles				10,000	Cycle
EEPROM Data Retention		10			Years

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 are at 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 (typical). These two pins will not latch up. The voltage at G6 and $\overline{\text{RESET}}$ pins must be limited to less than 14V.

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, Crystal/Resonator (Div-by 10)		1		DC	μs
R/C Oscillator Mode (Div-by 10)		3		DC	μs
CKI Clock Duty Cycle (Note 5)		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}		200			ns
t_{HOLD}		60			ns
Output Propagation Delay t_{PD1}, t_{PD0}	$C_L = 100$ pF, $R_L = 2.2$ kΩ				
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 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		1.0			μs

Note 5: Parameter sampled (not 100% tested).

COP6620C/COP6622C/COP6640C/COP6642C

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 $-55°C \leq T_A \leq +125°C$ 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			15	mA
Supply Current during Write Operation (Note 2)					
CKI = 10 MHz	V_{CC} = 5.5V, tc = 1 µs			21	mA
HALT Current (Note 3)	V_{CC} = 5.5V, CKI = 0 MHz		<10	40	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	−35		−300	µA
G Port Input Hysteresis (Note 5)				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 (Note 4) Without Latchup (Room Temp) (Note 5)	Room Temp			±100	mA
RAM Retention Voltage, Vr	500 ns Rise and Fall Time (Min)	2.5			V
Input Capacitance (Note 5)				7	pF
EEPROM Characteristics					
EEPROM Write Cycle Time				10	ms
EEPROM Number of Write Cycles				10,000	Cycle
EEPROM Data Retention		10			Years

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 are at 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 (typical). These two pins will not latch up. The voltage at G6 and $\overline{\text{RESET}}$ pins must be limited to less than 14V.

COP6620C/COP6622C/COP6640C/COP6642C (Continued)

AC Electrical Characteristics $-55^{\circ}C \leq T_A \leq +125^{\circ}C$ unless otherwise specified

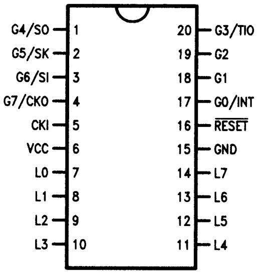
Parameter	Condition	Min	Typ	Max	Units
Instruction Cycle Time (tc) Ext, Crystal/Resonator (Div-by 10)		1		DC	μs
CKI Clock Duty Cycle (Note 5)		40		60	%
Rise Time (Note 5)	fr = 9 MHz Ext Clock			12	ns
Fall Time (Note 5)	fr = 9 MHz Ext Clock			8	ns
Inputs					
t_{SETUP}		220			ns
t_{HOLD}		66			ns
Output Propagation Delay t_{PD1}, t_{PD0}	$C_L = 100$ pF, $R_L = 2.2$ kΩ				
SO, SK				0.8	μs
All Others				1.1	μs
MICROWIRE™ Setup Time (t_{UWS})		20			ns
MICROWIRE Hold Time (t_{UWH})		56			ns
MICROWIRE Output Propagation Delay 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		1.0			μs

Note 5: Parameter sampled (not 100% tested).

Connection Diagrams

DUAL-IN-LINE PACKAGE

20 DIP

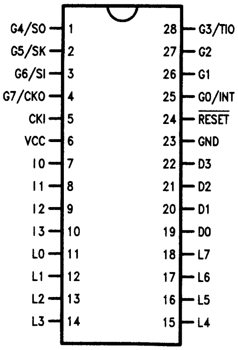


TL/DD/10366-3

Top View

Order Number
COP6622C-XXX/N, COP66L22C-XXX/N,
COP6642C-XXX/N, COP66L42C-XXX/N,
COP8622C-XXX/N, COP86L22C-XXX/N,
COP8642C-XXX/N, COP86L42C-XXX/N
See NS Package Number D20A or N20A
(D Package for Prototypes Only)

28 DIP

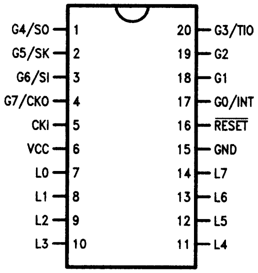


TL/DD/10366-5

Order Number
COP6620C-XXX/N, COP66L20C-XXX/N,
COP6640C-XXX/N, COP66L40C-XXX/N,
COP8620C-XXX/N, COP86L20C-XXX/N,
COP8640C-XXX/N, COP86L40C-XXX/N
See NS Package Number D28C or N28B
(D Package for Prototypes Only)

SURFACE MOUNT

20 SO Wide

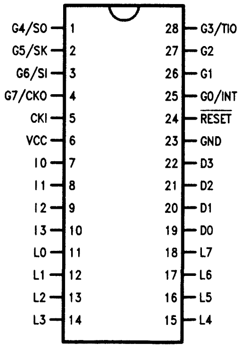


TL/DD/10366-3

Top View

Order Number
COP6622C-XXX/WM, COP66L22C-XXX/WM,
COP6642C-XXX/WM, COP66L42C-XXX/WM,
COP8622C-XXX/WM, COP86L22C-XXX/WM,
COP8642C-XXX/WM, COP86L42C-XXX/WM
See NS Package Number M20B

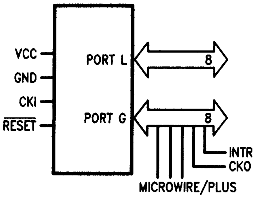
28 SO Wide



TL/DD/10366-5

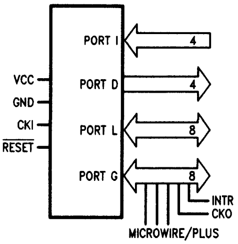
Order Number
COP6620C-XXX/WM, COP66L20C-XXX/WM,
COP6640C-XXX/WM, COP66L40C-XXX/WM,
COP8620C-XXX/WM, COP86L20C-XXX/WM,
COP8640C-XXX/WM, COP86L40C-XXX/WM
See NS Package Number M28B

COP8622C/COP8642C



TL/DD/10366-6

COP8620C/COP8640C



TL/DD/10366-8

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.

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 COP8620C/COP8622C consists of 1024 bytes of ROM and the COP8640C/COP8642C consists of 2048 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, EEPROM, I/O and registers. Data memory is addressed directly by the instruction or indirectly through B, X and SP registers.

The COP8620C/COP8640C has 64 bytes of RAM. Sixteen bytes of RAM are mapped as "registers", these can be loaded immediately and 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. RAM contents are undefined upon power-up.

The COP8620C/COP8640C provides 64 bytes of EEPROM for nonvolatile data memory. The data EEPROM can be read and written in exactly the same way as the RAM. All instructions that perform read and write operations on the RAM work similarly upon the data EEPROM. The data EEPROM contains all 00s when shipped by the factory.

A data EEPROM programming cycle is initiated by an instruction such as X, LD, SBIT and RBIT. The EE memory support circuitry sets the BsyERAM flag in the EECR register immediately upon beginning a data EEPROM write cycle. It will be automatically reset by the hardware at the end of the data EEPROM write cycle. The application program should test the BsyERAM flag before attempting a write operation to the data EEPROM. A second EEPROM write operation while a write operation is in progress will be ignored and the Werr flag in the EECR register will be set to indicate the error status. Once the write operation starts, nothing will stop the write operation, not by resetting the device, and not even turning off the V_{CC} will guarantee the write operation to stop.

Warning: The data memory pointer should not point to EEPROM unless the EEPROM is addressed. This will prevent inadvertent write to EEPROM.

Functional Description (Continued)

EECR AND EE SUPPORT CIRCUITRY

The EEPROM module contains EE support circuits to generate all necessary high voltage programming pulses. An EEPROM cell in the erase state is read out as a 0 and the written state as a 1. The EECR register provides control, status and test mode functions for the EE module. The EECR register bit assignments are shown below.

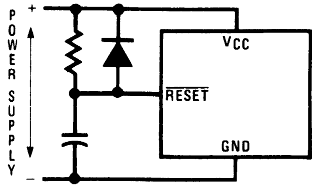
Werr	Write Error. Writing to EEPROM while a previous write cycle is still busy, that is BsyERAM is 1, causes Werr to be set to 1 indicating error status. Werr is a Read/Write bit and is cleared by writing a 0 into it.
BsyERAM	This bit is a read only bit and is set to 1 when EEPROM is being written. It is automatically reset by the hardware upon completion of the write operation. This bit is not cleared by reset. If the bit is set upon power up or reset, the application program should test the BsyERAM flag and wait for the flag to go low before attempting a write operation to the data EEPROM.

Bits 4 to 7 of the EECR register are used for encoding various EEPROM module test modes, most of which are for factory manufacturing tests. Except BsyERAM (bit 3) the EECR is cleared by reset. EECR is mapped into address location E0. Bit 2 can be used as flag. Bits 1 and 4 are always read as "0" and cannot be used as flags.

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. Except bit 3, the EECR register is 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/10366-9

RC ≥ 5X Power Supply Rise Time

FIGURE 4. Recommended Reset Circuit

Wr	**Test Mode Codes**					**Unused**	**Unused**	
Rd	**Test Mode Codes**				**BsyERAM**			**Werr**
Bit	7** R/W	6** R/W	5** R/W	4** R/O	3 R/O	2* R/W	1** R/O	0 R/W

*Can be used as flag bit

**Cannot be used as flag bit

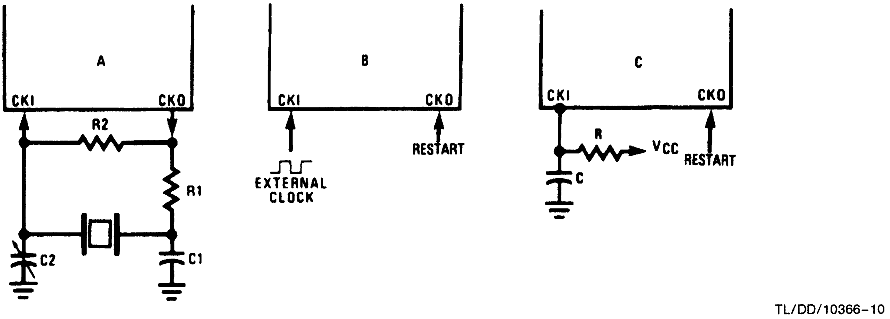


FIGURE 5. Crystal and R-C Connection Diagrams

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 (due to the part) as functions of the R/C component values (R/C tolerances not included).

TABLE I. Crystal Oscillator Configuration, $T_A = 25°C$, $V_{CC} = 5.0V$

R1 (kΩ)	R2 (MΩ)	C1 (pF)	C2 (pF)	CKI Freq (MHz)
0	1	30	30–36	10
0	1	30	30–36	4
5.5	1	100	100	0.455

TABLE II. RC Oscillator Configuration, $T_A = 25°C$, $V_{CC} = 5.0V$

R (kΩ)	C (pF)	CKI Freq. (MHz)	Instr. 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 9.0
6.8	100	0.9 to 1.1	8.8 to 10.8

Note: $3k \leq R \leq 200k$
$50\ pF \leq C \leq 200\ pF$

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/Resonator (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.

CURRENT DRAIN

The total current drain of the chip depends on:

1) Oscillator operating 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) EEPROM current during EE read operation. This current is active during 20% of the instruction cycle time—I6
7) EEPROM current during write operation—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.

Operating with a crystal network will draw more current than an external square-wave. The R/C mode will draw the most. 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

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 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.

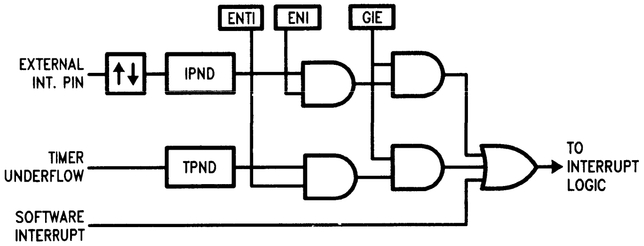
The software interrupt does not reset the GIE bit. This means that the controller can be interrupted by other interrupt sources while servicing the software interrupt.

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.

Functional Description (Continued)



TL/DD/10366-11

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 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 (02F), 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*.)

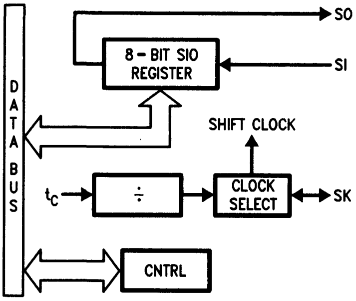
Functional Description (Continued)

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/10366-12

FIGURE 7. MICROWIRE/PLUS Block Diagram

MODE 1. TIMER WITH AUTO-LOAD REGISTER

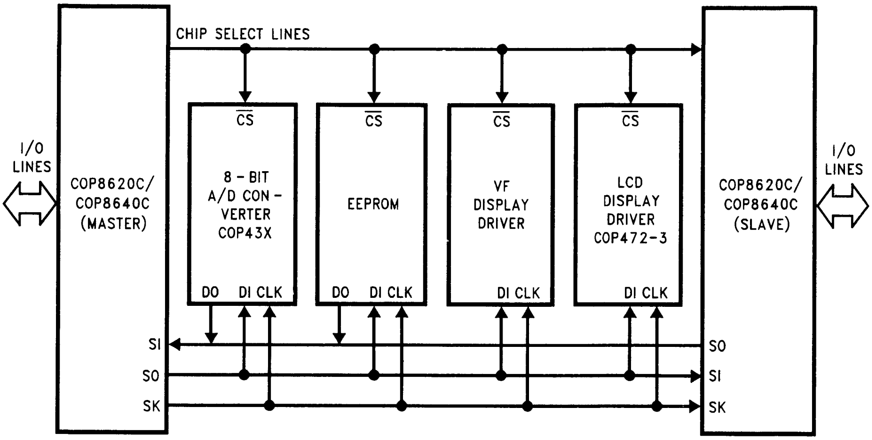
In this mode of operation, the timer T1 counts down at the instruction cycle rate. Upon underflow the value in the register R1 gets automatically reloaded into the timer which continues to count down. The timer underflow can be programmed to interrupt the microcontroller. A bit in the control register 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*.)



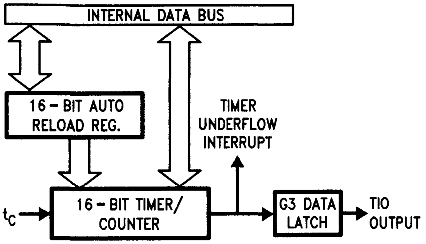
TL/DD/10366-13

FIGURE 8. MICROWIRE/PLUS Application

Functional Description (Continued)

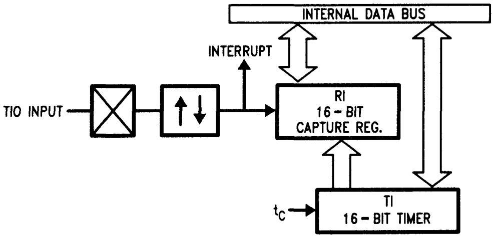
TABLE V. Timer Operating Modes

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



TL/DD/10366-15

FIGURE 9. Timer/Counter Auto Reload Mode Block Diagram

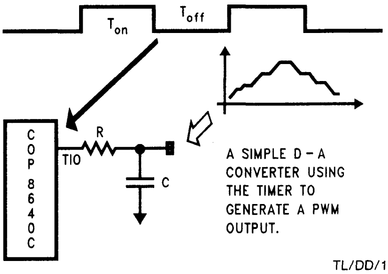


TL/DD/10366-14

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/10366-16

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

Memory Map

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

Address	Contents
COP8620C/COP8640C	
00 to 2F	On Chip RAM Bytes
30 to 7F	Unused RAM Address Space (Reads as all Ones)
80 to BF	On Chip EEPROM (64 bytes)
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	EECR
E1-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.

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.

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 (Continued)

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

OPCODE LIST

Bits 7–4

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 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.

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

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

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/Resonator	(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
= 2 N/A
= 3 20 pin DIP
= 4 20 SO
= 5 28 SO

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

IN-CIRCUIT EMULATOR

The MetaLink iceMASTER™-COP8 Model 400 In-Circuit Emulator for the COP8 family of microcontrollers features high-performance operation, ease of use, and an extremely flexible user-interface for maximum productivity. Interchangeable probe cards, which connect to the standard common base, support the various configurations and packages of the COP8 family.

The iceMASTER provides real time, full speed emulation up to 10 MHz, 32k bytes of emulation memory and 4k frames of trace buffer memory. The user may define as many as 32k trace and break triggers which can be enabled, disabled, set or cleared. They can be simple triggers based on code or address ranges or complex triggers based on code address, direct address, opcode value, opcode class or immediate operand. Complex breakpoints can be ANDed or ORed together. Trace information consists of address bus values, opcodes and user selectable probe clips status (external event lines). The trace buffer can be viewed as raw hex or as disassembled instructions. The probe clip bit values can be displayed in binary, hex or digital waveform formats.

During single-step operation the dynamically annotated code feature displays the contents of all accessed (read and write) memory locations and registers, as well as flow-of-control direction change markers next to each instruction executed.

The iceMASTER's performance analyzer offers a resolution of better than 6 μs. The user can easily monitor the time spent executing specific portions of code and find "hot spots" or "dead code". Up to 15 independent memory areas based on code address or label ranges can be defined. Analysis results can be viewed in bargraph format or as actual frequency count.

Emulator memory operations for program memory include single line assembler, disassembler, view, change and write to file. Data memory operations include fill, move, compare, dump to file, examine and modify. The contents of any memory space can be directly viewed and modified from the corresponding window.

The iceMASTER comes with an easy to use windowed interface. Each window can be sized, highlighted, color-controlled, added, or removed completely. Commands can be accessed via pull-down-menus and/or redefinable hot keys. A context sensitive hypertext/hyperlinked on-line help system explains clearly the options the user has from within any window.

The iceMASTER connects easily to a PC via the standard COMM port and its 115.2k baud serial link keeps typical program download time to under 3 seconds.

The following tables list the emulator and probe cards ordering information.

Emulator Ordering Information

Part Number	Description	Current Version
IM-COP8/400/1‡	MetaLink base unit in-circuit emulator for all COP8 devices, symbolic debugger software and RS-232 serial interface cable, with 110V @ 60 Hz Power Supply.	Host Software: Ver. 3.3 Rev. 5, Model File Rev 3.050.
IM-COP8/400/2‡	MetaLink base unit in-circuit emulator for all COP8 devices, symbolic debugger software and RS-232 serial interface cable, with 220V @ 50 Hz Power Supply.	

‡ These parts include National's COP8 Assembler/Linker/Librarian Package (COP8-DEV-IBMA)

Development Support (Continued)

Probe Card Ordering Informaton

Part Number	Pkg.	Voltage Range	Emulates
MHW-8640C20D5PC	20 DIP	4.5–5.5V	COP8642C, 8622C
MHW-8640C20DWPC	20 DIP	2.5–6.0V	COP8642C, 8622C
MHW-8640C28D5PC	28 DIP	4.5–5.5V	COP8640C, 8620C
MHW-8640C28DWPC	28 DIP	2.5–6.0V	COP8640C, 8620C

MACRO CROSS ASSEMBLER

National Semiconductor offers a COP8 macro cross assembler. It runs on industry standard compatible PCs and supports all of the full-symbolic debugging features of the MetaLink iceMASTER emulators.

Assembler Ordering Information

Part Number	Description	Manual
COP8-DEV-IBMA	COP8 Assembler/ Linker/Librarian for IBM®, PC/XT®, AT® or compatible.	424410632-001

SINGLE CHIP EMULATOR DEVICE

The COP8 family is fully supported by single chip hybrid emulators. For more detailed information refer to the emulation device specific data sheets and the emulator selection table below.

PROGRAMMING SUPPORT

Programming of the single chip emulator devices is supported by different sources. The table below shows the programmers certified for programming the hybrid emulator versions.

DIAL-A-HELPER

Dial-A-Helper is a service provided by the Microcontroller Applications Group. The Dial-A-Helper is an Electronic Bulletin Board information system.

INFORMATION SYSTEM

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 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. The minimum requirement for accessing the Dial-A-Helper is a Hayes compatible modem.

EPROM Programmer Information

Manufacturer and Product	U.S. Phone Number	Europe Phone Number	Asia Phone Number
MetaLink-Debug Module	(602) 926-0797	Germany: +49-8141-1030	Hong Kong: +852-737-1800
Xeltek-Superpro	(408) 745-7974	Germany: +49 2041 684758	Singapore: +65 276 6433
BP Microsystems-EP-1140	(800) 224-2102	Germany +49 89 857 66 67	Hong Kong: +852 388 0629
Data I/O - Unisite; - System 29, - System 39	(800) 322-8246	Europe: +31-20-622866 Germany: +49-89-85-8020	Japan: +33-432-6991
Abcom- COP8 Programmer		Europe: +89 808707	
System General Turpro-1-FX; -APRO	(408) 263-6667	Switzerland: +31-921-7844	Taiwan: +2-9173005

Single Chip Emulator Selection Table

Device Number	Clock Option	Package	Description	Emulates
COP8640CMHD-X	X=1: Crystal X=2: External X=3: R/C	28 DIP	Hybrid, UV Erasable	COP8640C, 8620C
COP8640CMHEA-X	X=1: Crystal X=2: External X=3: R/C	28 SO	Hybrid, UV Erasable	COP8640C, 8620C
COP8642CMHD-X	X=1: Crystal X=2: External X=3: R/C	20 DIP	Hybrid, UV Erasable	COP8642C, 8622C

Development Support (Continued)

If the user has a PC with a communications package then files from the FILE SECTION can be down-loaded to disk for later use.

ORDER P/N: MOLE-DIAL-A-HLP

Information System Package contains:

- Dial-A-Helper Users Manual
- Public Domain Communications Software

FACTORY APPLICATIONS SUPPORT

Dial-A-Helper also provides immediate factory applications support. If a user has questions, he can leave messages on our electronic bulletin board, which we will respond to.

Voice:	(800) 272-9959		
Modem:	Canada/U.S.:	(800) NSC-MICRO	
	Baud:	14.4k	
	Setup:	Length:	8-Bit
		Parity:	None
		Stop Bit:	1
	Operation:	24 Hrs. 7 Days	

National Semiconductor

COP680C/COP681C/COP880C/COP881C/COP980C/COP981C Microcontrollers

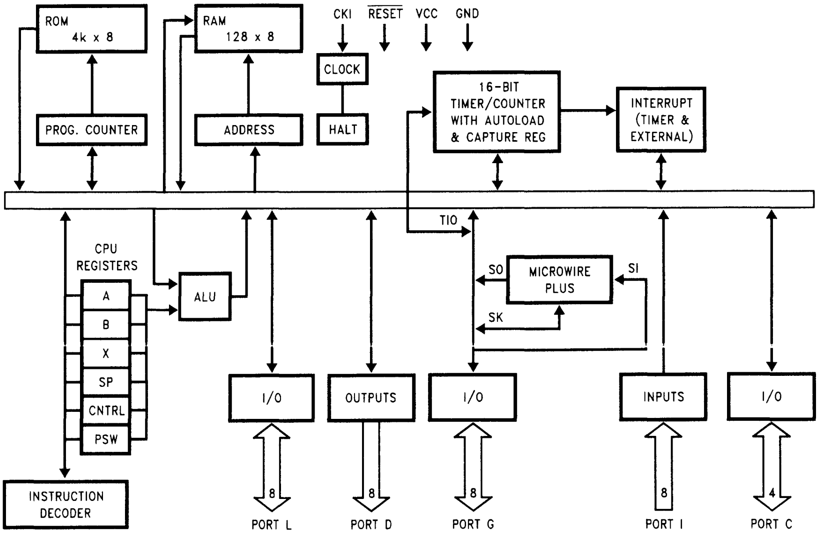
General Description

The COP680C/COP681C/COP880C/COP881C/COP980C, and COP981C 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.

Features

- Low cost 8-bit microcontroller
- Fully static CMOS
- 1 µs instruction time
- Low current drain
  Low current static HALT mode (Typically < 1 µA)
- Single supply operation: 2.5 to 6.0V
- 4096 bytes ROM/128 Bytes RAM
- 16-bit read/write timer operates in a variety of modes
  - Timer with 16-bit auto reload register
  - 16-bit external event counter
  - Timer with 16-bit capture register (selectable edge)
- Multi-source interrupt
  - Reset master clear
  - External interrupt with selectable edge
  - Timer interrupt or capture interrupt
  - Software interrupt
- 8-bit stack pointer (stack in RAM)
- Powerful instruction set, most instructions single byte
- BCD arithmetic instructions
- MICROWIRE PLUS™ serial I/O
- 44 PLCC, 36 I/O pins
- 40 DIP, 36 I/O pins
- 28 DIP and SO, 24 I/O pins
- Software selectable I/O options (TRI-STATE®, push-pull, weak pull-up)
- Schmitt trigger inputs on Port G
- Temperature ranges: COP98XC/COP98XCH 0°C to 70°C, COP88XC −40°C to +85°C, COP68XC −55°C to +125°C.
- Form factor emulation devices
- Fully supported by Metalink's development systems

Block Diagram



TL/DD/10802-1

FIGURE 1

COP980C/COP981C

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 COP980XC; $0°C \leq T_A \leq +70°C$ 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					
$\overline{\text{RESET}}$, 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	$V_{CC} = 6.0V$	−1.0		+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 $\overline{RESET}$ 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 $0°C \leq T_A \leq +70°C$ unless otherwise specified

Parameter	Condition	Min	Typ	Max	Units
Instruction Cycle Time (tc)					
Crystal/Resonator or External	$V_{CC} \geq 4.0V$	1		DC	μs
(Div-by 10)	$2.3V \leq V_{CC} \leq 4.0V$	2.5		DC	μs
R/C Oscillator Mode	$V_{CC} \geq 4.0V$	3		DC	μs
(Div-by 10)	$2.3V \leq V_{CC} \leq 4.0V$	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}	$V_{CC} \geq 4.0V$	200			ns
	$2.3V \leq V_{CC} \leq 4.0V$	500			ns
t_{HOLD}	$V_{CC} \geq 4.0V$	60			ns
	$2.3V \leq V_{CC} \leq 4.0V$	150			ns
Output Propagation Delay	$C_L = 100$ pF, $R_L = 2.2$ kΩ				
t_{PD1}, t_{PD0}					
SO, SK	$V_{CC} \geq 4.0V$			0.7	μs
	$2.3V \leq V_{CC} \leq 4.0V$			1.75	μs
All Others	$V_{CC} \geq 4.0V$			1	μs
	$2.3V \leq V_{CC} \leq 4.0V$			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.

COP880C/COP881C

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; $-40°C \le T_A \le +85°C$ 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					
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					
$\overline{\text{RESET}}$, 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	−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

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 $\overline{RESET}$ 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.

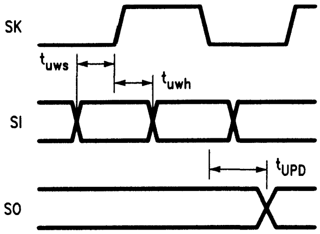
Note 5: To maintain RAM integrity, the voltage must not be dropped or raised instantaneously.

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

Parameter	Condition	Min	Typ	Max	Units
Instruction Cycle Time (tc)					
Crystal/Resonator or External	$V_{CC} \geq 4.5V$	1		DC	μs
(Div-by 10)	$2.5V \leq V_{CC} < 4.5V$	2.5		DC	μs
R/C Oscillator Mode	$V_{CC} \geq 4.5V$	3		DC	μs
(Div-by 10)	$2.5V \leq V_{CC} < 4.5V$	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}	$V_{CC} \geq 4.5V$	200			ns
	$2.5V \leq V_{CC} < 4.5V$	500			ns
t_{HOLD}	$V_{CC} \geq 4.5V$	60			ns
	$2.5V \leq V_{CC} < 4.5V$	150			ns
Output Propagation Delay	$C_L = 100$ pF, $R_L = 2.2$ kΩ				
t_{PD1}, t_{PD0}					
SO, SK	$V_{CC} \geq 4.5V$			0.7	μs
	$2.5V \leq V_{CC} < 4.5V$			1.75	μs
All Others	$V_{CC} \geq 4.5V$			1	μs
	$2.5V \leq V_{CC} < 4.5V$			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

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: $-55°C \leq T_A \leq +125°C$ 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					
$\overline{\text{RESET}}$, 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	$V_{CC} = 5.5V$, $V_{IN} = 0V$	−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 $\overline{\text{RESET}}$ 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: To maintain RAM integrity, the voltage must not be dropped or raised instantaneously.

COP680C/COP681C

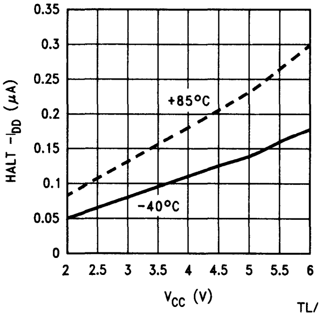
AC Electrical Characteristics $-55^{\circ}C \le T_A \le +125^{\circ}C$ unless otherwise specified

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.

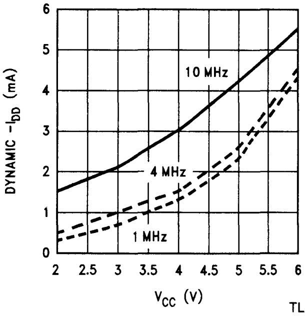
Typical Performance Characteristics $(-40°C \leq T_A \leq +85°C)$

Hall—I_{DD}



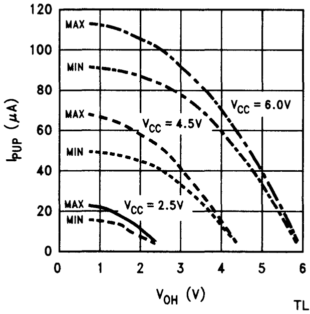
TL/DD/10802–16

Dynamic—I_{DD} (Crystal Clock Option)



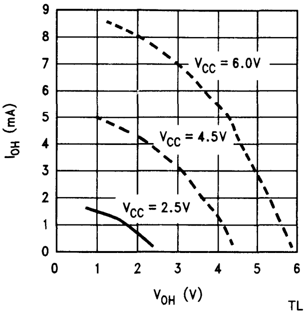
TL/DD/10802–17

Port L/C/G Weak Pull-Up Source Current



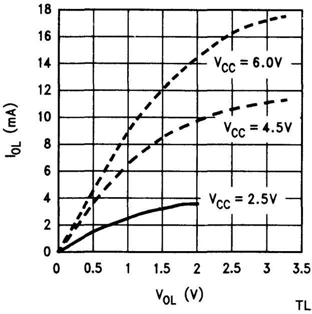
TL/DD/10802–18

Port L/C/G Push-Pull Source Current



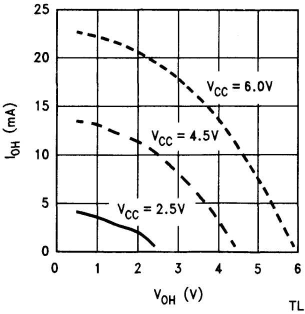
TL/DD/10802–19

Port L/C/G Push-Pull Sink Current



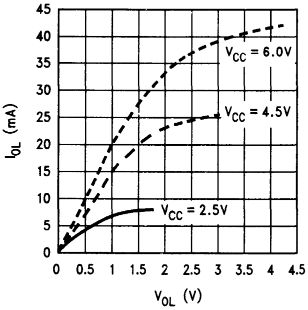
TL/DD/10802–20

Port D Source Current



TL/DD/10802–21

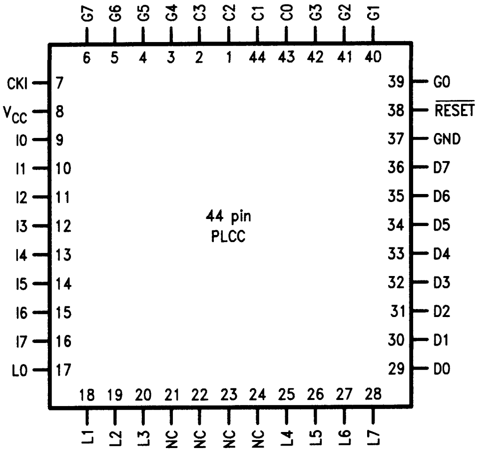
Port D Sink Current



TL/DD/10802–22

Connection Diagrams

Plastic Chip Carrier

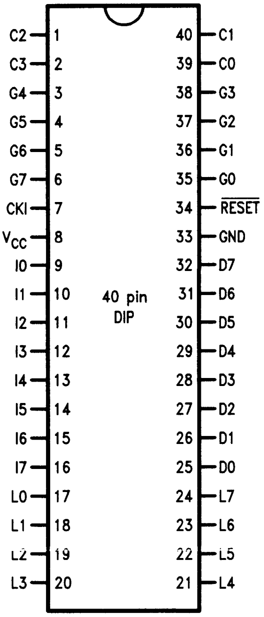


TL/DD/10802-3

Top View

Order Number COP680C-XXX/V, COP880C-XXX/V, COP980C-XXX/V or COP980CH-XXX/V

Dual-In-Line Package

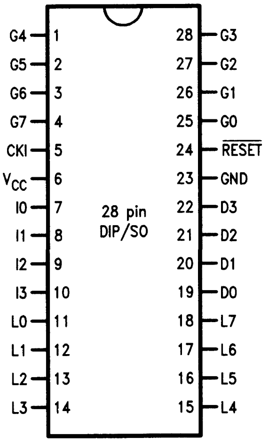


TL/DD/10802-4

Top View

Order Number COP680C-XXX/N, COP880C-XXX/N, COP980C-XXX/N or COP980CH-XXX/N

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

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 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 $\overline{\text{RESET}}$ 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.

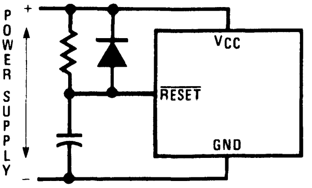
Note: RAM contents are undefined upon power-up.

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.

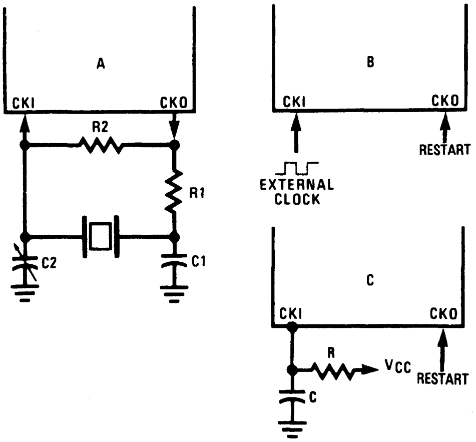
B. EXTERNAL OSCILLATOR

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.

CURRENT DRAIN

The total current drain of the chip depends on:

1) Oscillator operating 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

Thus the total current drain, It is given as

$$It = I1 + I2 + I3 + I4 + I5$$

To reduce the total current drain, each of the above components must be minimum.

Operating with a crystal network will draw more current than an external square-wave. The R/C mode will draw the most. 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

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.

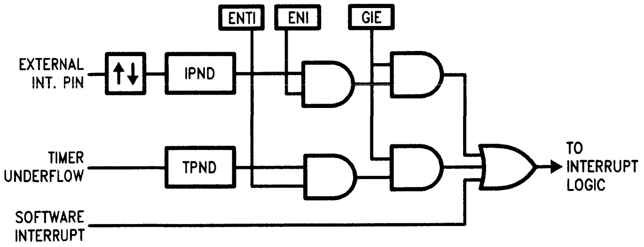
IPND and TPND bits signal which interrupt is pending. After interrupt is acknowledged, the user can check these two bits to determine which interrupt is pending. This permits the interrupts to be prioritized under software. The pending flags have to be cleared by the user. Setting the GIE bit high inside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



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*.)

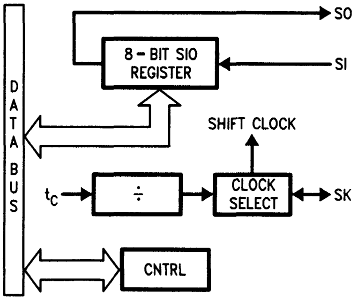
Functional Description (Continued)

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

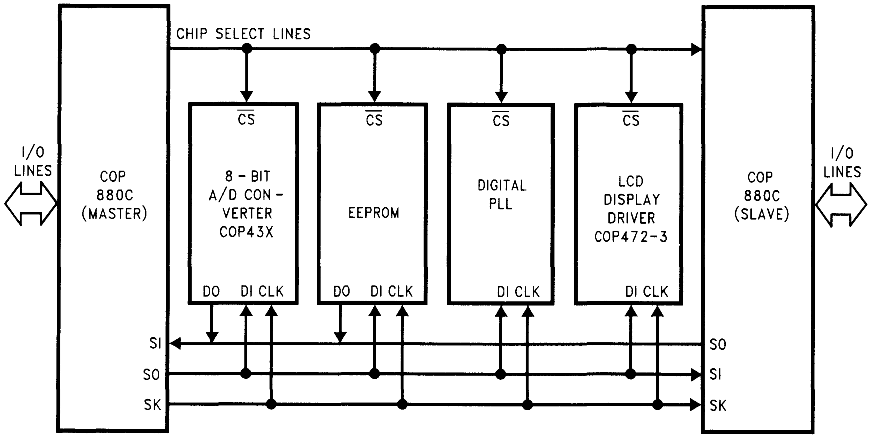
In this mode of operation, the timer T1 counts down at the instruction cycle rate. Upon underflow the value in the register R1 gets automatically reloaded into the timer which continues to count down. The timer underflow can be programmed to interrupt the microcontroller. A bit in the control register 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.*)

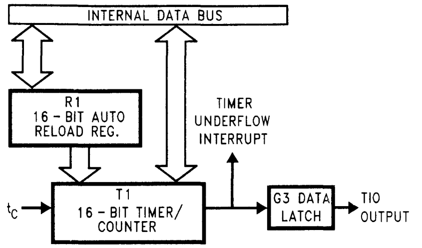


TL/DD/10802-10

FIGURE 8. MICROWIRE/PLUS Application

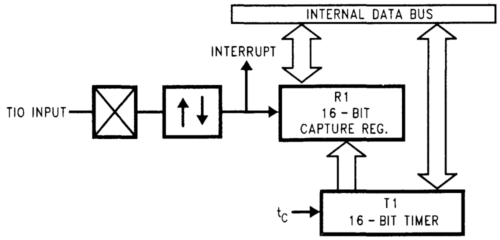
TABLE V. Timer Operating Modes

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

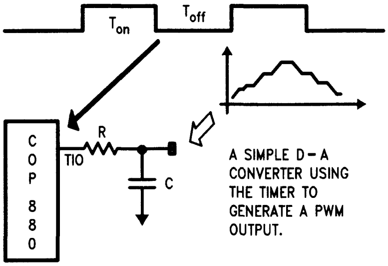


TL/DD/10802-12

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

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 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.

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		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			

* => 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

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

IN-CIRCUIT EMULATOR

The MetaLink iceMASTER™-COP8 Model 400 In-Circuit Emulator for the COP8 family of microcontrollers features high-performance operation, ease of use, and an extremely flexible user-interface for maximum productivity. Interchangeable probe cards, which connect to the standard common base, support the various configurations and packages of the COP8 family.

The iceMASTER provides real-time, full-speed emulation up to 10 MHz, 32 kbytes of emulation memory and 4k frames of trace buffer memory. The user may define as many as 32k trace and break triggers which can be enabled, disabled, set or cleared. They can be simple triggers based on code or address ranges or complex triggers based on code address, direct address, opcode value, opcode class or immediate operand. Complex breakpoints can be ANDed and ORed together. Trace information consists of address bus values, opcodes and user selectable probe clips status (external event lines). The trace buffer can be viewed as raw hex or as disassembled instructions. The probe clip bit values can be displayed in binary, hex or digital waveform formats.

During single-step operation the dynamically annotated code feature displays the contents of all accessed (read and write) memory locations and registers, as well as flow-of-control direction change markers next to each instruction executed.

The iceMASTER's performance analyzer offers a resolution of better than 6 μs. The user can easily monitor the time spent executing specific portions of code and find "hot spots" or "dead code". Up to 15 independent memory areas based on code address or label ranges can be defined. Analysis results can be viewed in bargraph format or as actual frequency count.

Emulator memory operations for program memory include single line assembler, disassembler, view, change and write to file. Data memory operations include fill, move, compare, dump to file, examine and modify. The contents of any memory space can be directly viewed and modified from the corresponding window.

The iceMASTER comes with an easy-to-use windowed interface. Each window can be sized, highlighted, color-controlled, added, or removed completely. Commands can be accessed via pull-down menus and/or redefinable hot keys. A context sensitive hypertext/hyperlinked on-line help system explains clearly the options the user has from within any window.

The iceMASTER connects easily to a PC via the standard COMM port and its 115.2 kBaud serial link keeps typical program download time to under 3 seconds.

The following tables list the emulator and probe cards ordering information.

Emulator Ordering Information

Part Number	Description	Current Version
IM-COP8/400/1‡	MetaLink base unit in-circuit emulator for all COP8 devices, symbolic debugger software and RS232 serial interface cable, with 110V @ 60 Hz Power Supply	HOST SOFTWARE: VER. 3.3 REV. 5, Model File Rev 3.050.
IM-COP8/400/2‡	Metalink base unit in-current emulator for all COP8 devices, symbolic debugger software and RS232 serial interface cable, with 220V @ 50 Hz Power Supply.	
DM-COP8/880C‡	Metalink IceMASTER Debug Module. This is the low cost version of Metalink's IceMASTER. Firmware: Ver. 6.07.	

‡ These parts include National's COP8 Assembler/Linker/Librarian Package (COP8/DEV-IBMA)

MACRO CROSS ASSEMBLER

National Semiconductor offers a COP8 macro cross assembler. It runs on industry standard compatible PCs and supports all of the full-symbolic debugging features of the MetaLink iceMASTER emulators.

Assembler Ordering Information

Part Number	Description	Manual
COP8-DEV-IBMA	COP8 Assembler/ Linker/Librarian for IBM®, PC/XT®, AT® or compatible.	424410632-001

Probe Card Ordering Information

Part Number	Package	Voltage Range	Emulates
MHW-880C28D5PC	28 DIP	4.5V–5.5V	COP820C, 840C, 881C, 8781C
MHW-880C28DWPC	28 DIP	2.5V–6.0V	COP820C, 840C, 881C, 8781C
MHW-880C40D5PC	40 DIP	4.5V–5.5V	COP880C, 8780C
MHW-880C40DWPC	40 DIP	2.5V–6.0V	COP880C, 8780C
MHW-880C44D5PC	44 PLCC	4.5V–5.5V	COP880C, 8780C
MHW-880C44DWPC	44 PLCC	2.5V–6.0V	COP880C, 8780C

Development Support (Continued)

SINGLE-CHIP EMULATOR DEVICE

The COP8 family is fully supported by single chip form, fit and function emulators. The emulators are available as UV erasable or one Time Programmable (OTP).

For more detailed information, refer to the emulation device specific data sheets and the emulator selection table below.

Single-Chip Emulator Selection Table

Device Number	Clock Option	Package	Description	Emulates
COP8780CV	Programmable	44 PLCC	One-Time Programmable (OTP)	COP880C
COP8780CEL	Programmable	44 LDCC	UV Erasable	COP880C
COP8780CN	Programmable	40 DIP	OTP	COP880C
COP8780CJ	Programmable	40 DIP	UV Erasable	COP880C
COP8781CN	Programmable	28 DIP	OTP	COP881C
COP8781CJ	Programmable	28 DIP	UV Erasable	COP881C
COP8781CWM	Programmable	28 SO	OTP	COP881C

PROGRAMMING SUPPORT

Programming of the single-chip emulator devices is supported by different sources. The following programmers are certified for programming the One Time Programmable (OTP) devices:

EPROM Programmer Information

Manufacturer and Product	U.S. Phone Number	Europe Phone Number	Asia Phone Number
Metalink-Debug Module	(602) 926-0797	Germany: +49-8141-1030	Hong Kong: +852-737-1800
Xeltek-Superpro	(408) 745-7974	Germany: +49 2041 684758	Singapore: +65 276 6433
BP Microsystems-EP-1140	(800) 225-2102	Germany: +49 89 857 66 67	Hong Kong: +852 388 0629
Data I/O-Unisite; -System 29, -System 39	(800) 322-8246	Europe: +31-20-622866 Germany: +49-89-85-8020	Japan: +33-432-6991
Abcom-COP8 Programmer		Europe: + 89 808707	
System General Turpro-1-FX; -APRO	(408) 263-6667	Switzerland: +31-921-7844	Taiwan Taipei: +2-9173005

Development Support (Continued)

DIAL-A-HELPER

Dial-A-Helper is a service provided by the Microcontroller Applications Group. The Dial-A-Helper is an Electronic Bulletin Board information system.

INFORMATION SYSTEM

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 communications to and from the Microcontroller Applications Group and a FILE SECTION which consists of several file areas where valuable application software and utilities can be found. The minimum requirement for accessing the Dial-A-Helper is a Hayes compatible modem.

If the user has a PC with a communications package then files from the FILE SECTION can be down-loaded to disk for later use.

FACTORY APPLICATIONS SUPPORT

Dial-A-Helper also provides immediate factory applications support. If a user has questions, he can leave messages on our electronic bulletin board, which we will respond to.

Voice:	(800) 272-9959	
Modem:	CANADA/U.S.:	(800) NSC-MICRO (800) 672-6427
	Baud:	14.4k
	Setup:	Length: 8-Bit Parity: None Stop Bit: 1
	Operation:	24 Hrs., 7 Days

National Semiconductor

COP684BC/COP884BC Single-Chip microCMOS Microcontroller

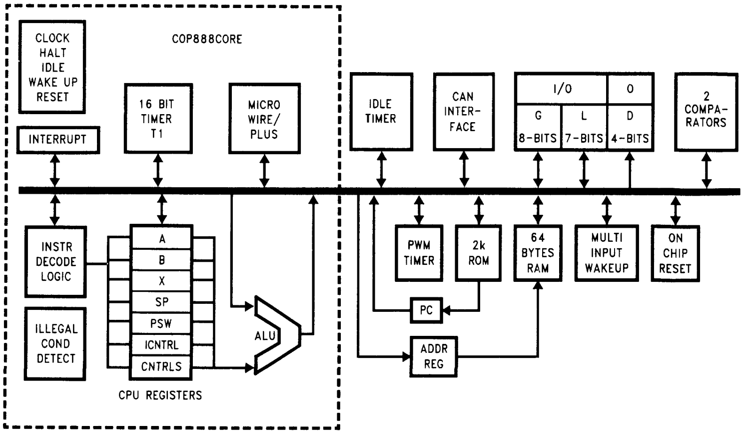
General Description

The COP684BC and COP884BC are members of the COP888BC 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)

Features

- Low cost 8-bit microcontroller
- Fully static CMOS, with low current drain
- Two power saving modes: HALT and IDLE
- 1 µs instruction cycle time
- 2048 bytes on-board ROM
- 64 bytes on-board RAM
- Single supply operation: 4.5V–5.5V
- MICROWIRE/PLUS™ serial I/O
- Idle Timer
- Multi-Input Wake Up (MIWU) with optional interrupts (7)
- On chip reset
- CAN Interface
- 2 comparators
- High speed, constant resolution 8-bit PWM/frequency monitor timer with 2 output pins
- One 16-bit timer, with two 16-bit registers supporting:
  - — Processor Independent PWM mode
  - — External Event counter mode
  - — Input Capture mode
- 8-bit Stack Pointer SP (stack in RAM)
- Two 8-bit Register Indirect Data Memory Pointers (B and X)
- Versatile instruction set
- True bit manipulation
- Memory mapped I/O
- BCD arithmetic instructions
- Package:
  - — 28 SO with 18 general I/O pins
- 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
- Temperature ranges:
  - — COP88xBC −40°C to +85°C,
  - — COP68xBC −55°C to +125°C
- Single chip hybrid emulation device—COP884BCMH
- Real time emulation and full program debug offered by MetaLink's Development Systems
- 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)

Block Diagram



TL/DD/12067-1

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. 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.

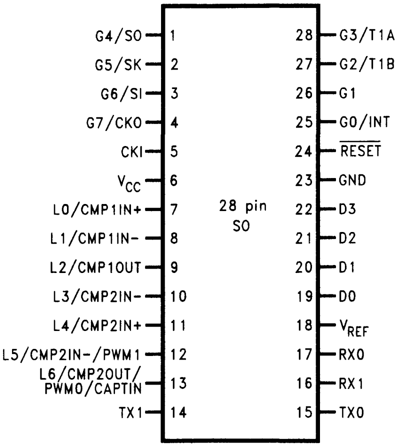
Connection Diagram

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 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)

Dual-In-Line Package



TL/DD/12067-2

Top View

28-Lead (0.300″ Wide) Molded Small Outline Package, JEDEC Order Number COP884BC-xxx/WM or COP684BC-xxx/WM See NS Package Number M28B

FIGURE 2

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 COP88xBC: $-40°C \leq T_A \leq +85°C$

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)	$V_{CC} = 5.5V$, $t_c = 1 \mu s$			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	$V_{CC} = 5.5V$, $t_c = 1 \mu s$			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 6)		$0.05 V_{CC}$		V
Output Current Levels D Outputs					
Source	$V_{CC} = 4.5V$, $V_{OH} = 3.3V$	−0.4			mA
Sink	$V_{CC} = 4.5V$, $V_{OL} = 1.0V$	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
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
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 COP68xBC: $-55°C \le T_A \le +125°C$

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)	$V_{CC} = 5.5V$, $t_c = 1\ \mu s$			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	$V_{CC} = 5.5V$, $t_c = 1\ \mu s$			5.5	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	$V_{CC} = 5.5V$			±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
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
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: COP68xBC and COP88xBC: $-55°C \le T_A \le +125°C$

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 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: $-55°C \le T_A \le +125°C$

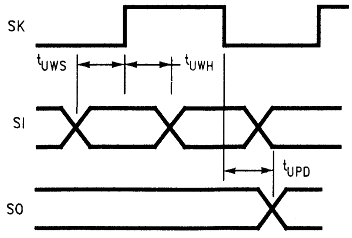
Parameter	Conditions	Min	Max	Units
Reference Voltage V_{REF}	$I_{OUT} < 80$ µA, $V_{CC} = 5V$	0.5 V_{CC} −0.12	0.5 V_{CC} +0.12	V
Reference Supply Current, I_{DD}	$I_{OUT} = 0A$, (No Load) $V_{CC} = 5V$ (Note A)		120	µA

Note A: Reference supply I_{DD} is supplied for information purposes only, it is not tested.

Comparator DC/AC Characteristics: $4.5V \le V_{CC} \le 5.5V$, $-55°C \le T_A \le +125°C$

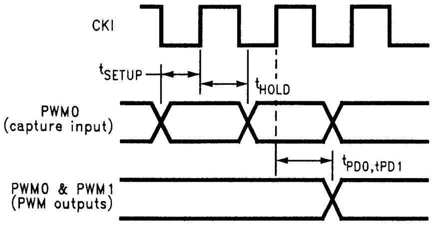
Parameter	Conditions	Min	Typ	Max	Units
Input Offset Voltage	$0.4V < V_{IN} < V_{CC} - 1.5V$		±10	±25	mV
Input Common Mode Voltage Range		0.4		$V_{CC} - 1.5$	V
Voltage Gain			300k		V/V
Outputs Sink/Source	See I/O-Port DC Specifications				
DC Supply Current (when enabled)	$V_{CC} = 6.0V$			250	µA
Response Time	TBD mV Step, TBD mV Overdrive, 100 pF Load		1		µs

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)

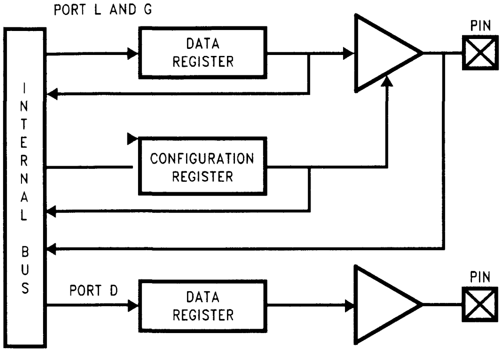
Port G has the following dedicated function:

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)

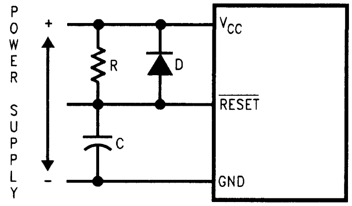
ON-CHIP POWER-ON RESET

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

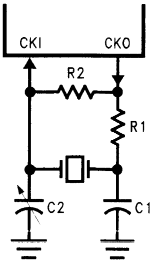
RC > 5 x Power Supply Rise Time

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	$V_{CC} = 5V$

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. V_{REF} of CAN—I7
8. Comparator of CAN block—I8
9. On-chip Reset—I9

Thus the total current drain, It, is given as

$$It = I1 + I2 + I3 + I4 + I5 + I6 + I7 + I8 + I9$$

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. Switching current, governed by the equation, can be reduced by lowering voltage and frequency. Leakage current can be reduced by lowering voltage and temperature. The other items can be reduced by carefully designing the end-user's system.

$$I2 = C * V * f$$

where C = equivalent capacitance of the chip

V = operating voltage

f = CKI frequency

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
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

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 ($t_C = 1\ \mu s$). 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.

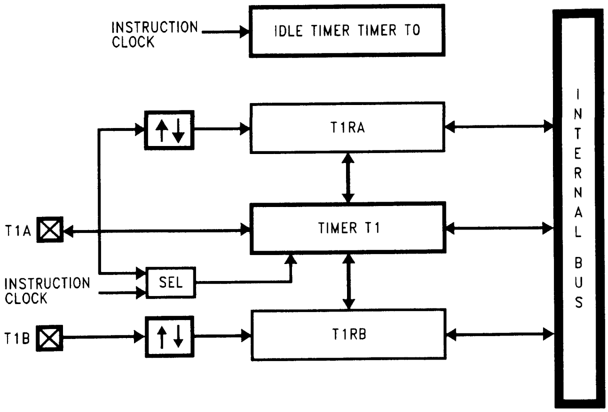
The T1 Timer control bits, T1C3, T1C2 and T1C1 set up the timer for PWM mode operation.

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.

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.

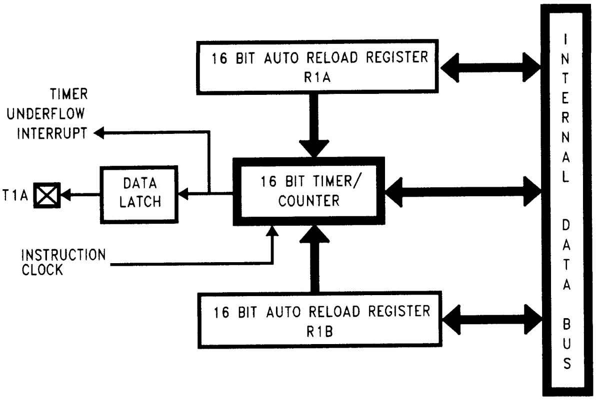
Either or both of the timer underflow interrupts may be enabled. This gives the user the flexibility of interrupting once per PWM period on either the rising or falling edge of the PWM output. Alternatively, the user may choose to interrupt on both edges of the PWM output.

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.

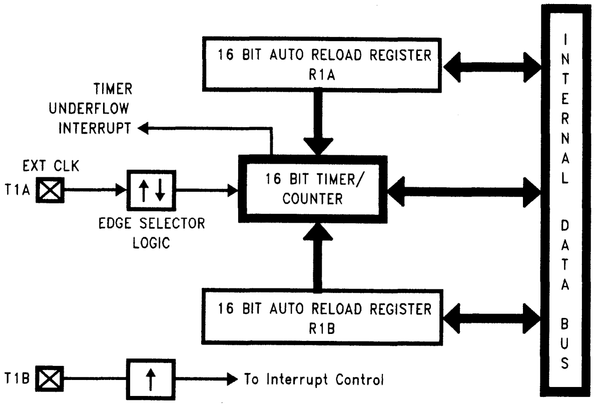
The timer value gets copied over into the register when a trigger event occurs on its corresponding pin. Control bits, T1C3, T1C2 and T1C1, allow the trigger events to be specified either as a positive or a negative edge. The trigger condition for each input pin can be specified independently.

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.

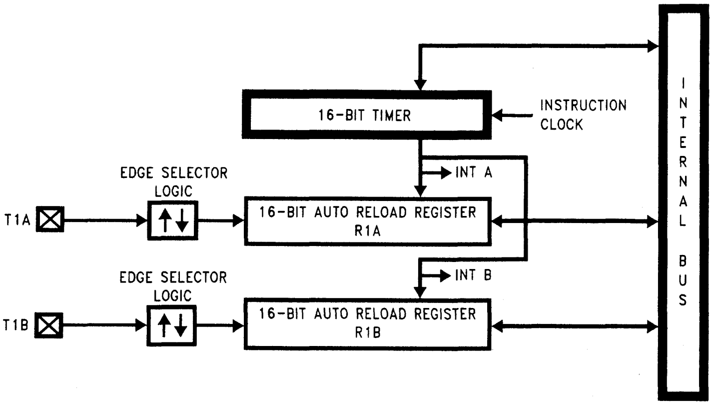
Underflows from the timer can also be programmed to generate interrupts. Underflows are latched into the timer T1C0 pending flag (the T1C0 control bit serves as the timer underflow interrupt pending flag in the Input Capture mode). Consequently, the T1C0 control bit should be reset when entering the 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

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 T1B 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).

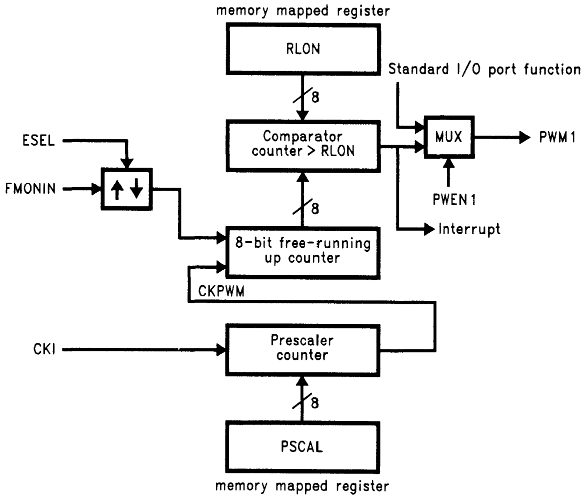
PWM On-time Register (RLON) (Address X'00A1)

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.

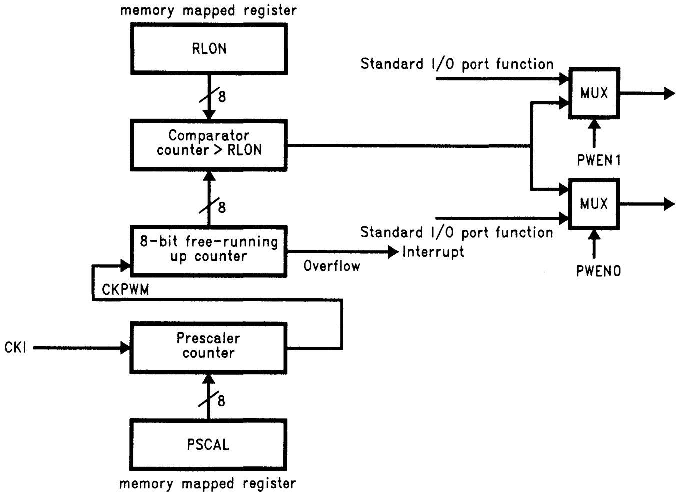
Note: The effect of changing the RLON register during active PWM mode operation is delayed until the boundary of a PWM cycle. In capture mode the effect takes place immediately.

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

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 (@ $t_C = 1\ \mu s$) 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:

$$\text{fpwm} = \frac{\text{CKI}}{(1 + (\text{PSCAL-contents})) \times 255}$$

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)

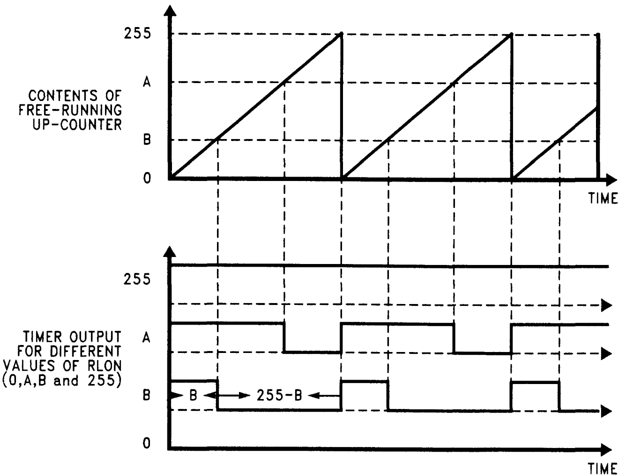


FIGURE 14. PWM Mode Operation

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.)

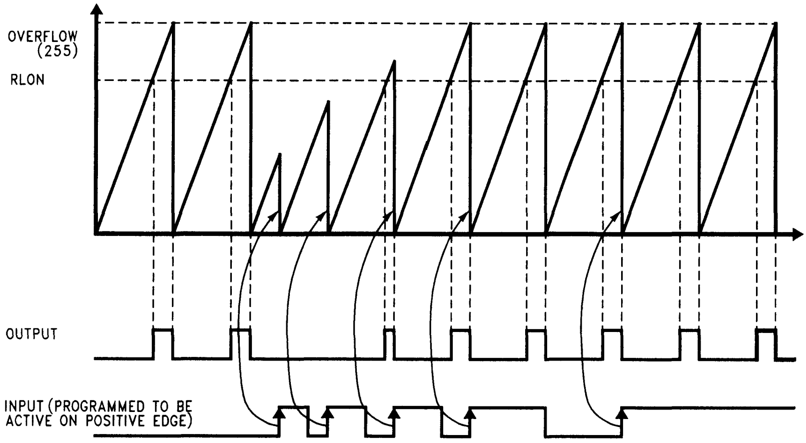


FIGURE 15. Frequency Monitor Mode Operation

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:

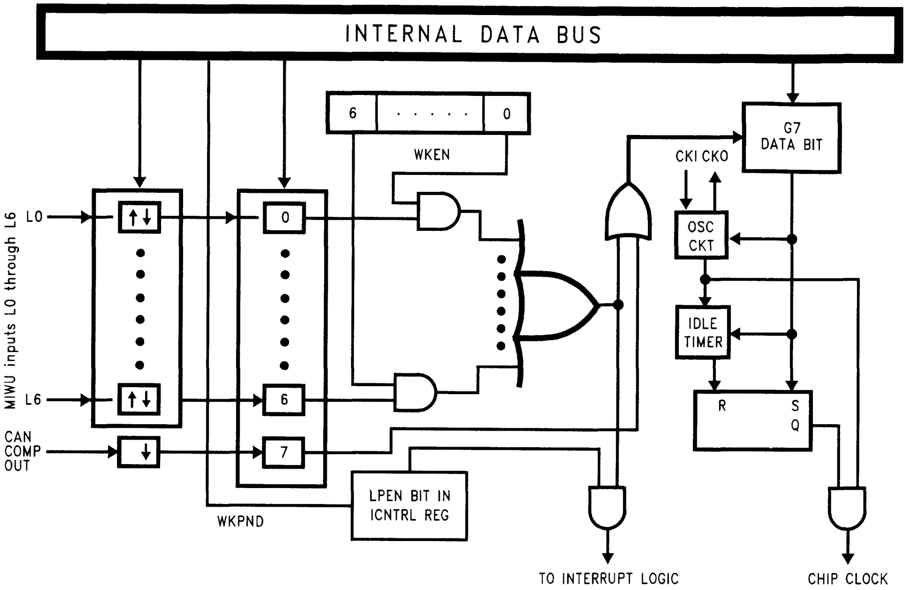
```
RBIT  5, WKEN
SBIT  5, WKEDG
RBIT  5, WKPND
SBIT  5, WKEN
```

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/12067-16

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. This vector can point to the Software Trap (ST) interrupt service routine, or to another special service routine as desired.

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.

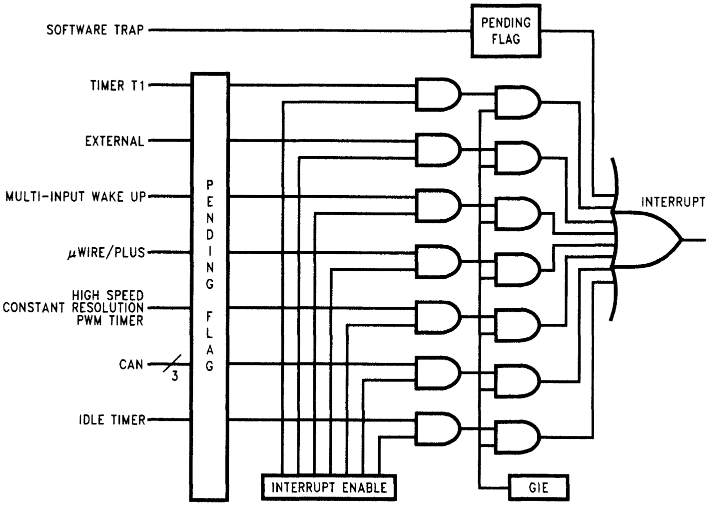
The ST has the highest rank among all interrupts.

Nothing (except another ST) can interrupt an ST being serviced.

CAN Block Description *

This device contains a CAN serial bus interface as described in the CAN Specification Rev. 2.0 part B.

* Patents Pending.



TL/DD/12067-17

FIGURE 17. Interrupt Block Diagram

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 retransmission is supported for messages not longer than 2 bytes. Messages which are longer than two byte 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. However, 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.

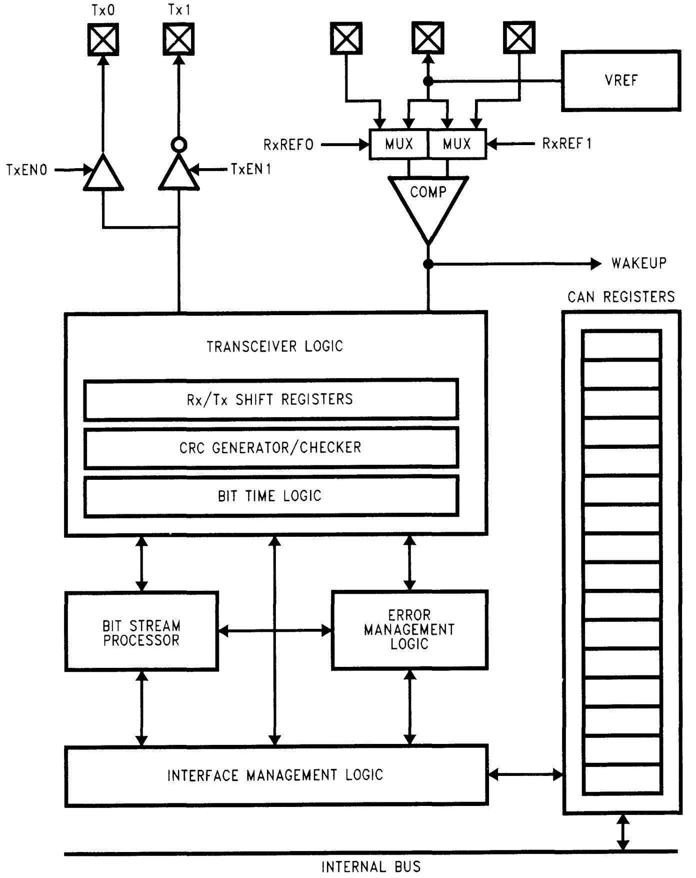


FIGURE 18. CAN Interface Block Diagram

Functional Block Description Of The CAN Interface

INTERFACE MANAGEMENT LOGIC (IML)

The IML executes the CPU's transmission and reception commands and controlling 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 Interface Management Logic (parallel data) and the bus line (serial data). It controls the transceive logic with regard to reception, 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 BSR. 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 resulting bus clock (t_{CAN}) 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).

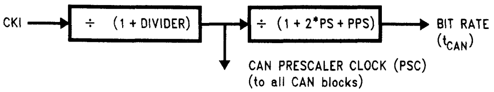
Note: The synchronization jump width (SJ) (see CAN BUS TIMING REGISTER (CTIM)) should be less then the programmed value of PS1. If a soft resynchronization is done during phase segment 1 or the propagation segment, then SJ will always be equal to the programmed value. If soft resynchronization is done during phase segment 2 and the programmed value of SJ is greater than or equal to the programmed PS1 value, PS2 will never be smaller than 1.

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.

TABLE II. Bus Level Definition

Bus Level	Pin Tx0	Pin Tx1
"dominant"	drive low (GND)	drive high (V_{CC})
"recessive"	TRI-STATE	TRI-STATE



TL/DD/12067–19

FIGURE 19. Bit Rate Generation

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 contains always 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 Transmit Data Register 2 contains always 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..TID0 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.

DLC3..TDLC0 Transmit Data Length Code

These bits determine the number of data bytes to be transmitted within a frame.

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

This bit is set if the frame to be transmitted is a remote frame.

TID10..TID4 Transmit Identifier Bits 10..4 (higher 7 bits)

Bits TID10..TID4 are the upper 7 bits of the 11-bit transmit identifier.

Functional Block Description Of The CAN Interface (Continued)

RECEIVE DATA REGISTER 1 (RXD1) (Address X'00B4)

The Receive Data Register 1 (RXD1) contains always 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 always 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.

RECEIVE DATA LENGTH CODE AND IDENTIFIER LOW REGISTER (RIDL) (Address X'00B6)

RID3	RID2	RID1	RID0	RDLC3	RDLC2	TCLC1	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 (see also RECEIVE IDENTIFIER HIGH (RID) (Address X'00B7).

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 registers) 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.

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	PS2	PS1	PS0	SJ1	SJ0

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).

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 2.

SJ1, SJ0 Synchronization Jump Width 0 and 1

The Synchronization Jump Width defines the maximum number of Prescaler clock cycles by which a bit may be shortened, or lengthened, to achieve re-synchronization on "recessive" to "dominant" data transitions on the bus.

TABLE III. Synchronization Jump Width

SJ1	SJ0	Synchronization Jump Width
0	0	1 PSC
0	1	2 PSC
1	0	3 PSC
1	1	4 PSC

LENGTH OF TIME SEGMENTS

- 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

CAN BUS CONTROL REGISTER (CBUS) (00BA)

Re-served	RIAF	TXEN1	TXEN0	RXREF1	RXRED0	Re-served	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 IV. 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

Functional Block Description Of The CAN Interface (Continued)

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 resets the TXEN1 and TXEN0 bits before switching the device into the HALT mode (the CAN receive wake up will still work) in order to reduce current consumption and to assure a proper resynchronization 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.

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 V. Node Status

NS1	NS0	Output
0	0	Error Active
0	1	Error Passive
1	0	Bus Off
I	1	Bus Off

The Node Status bits are read only.

TERR Transmit Error

This bit is automatically set when an error occurred 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 has to be cleared by the user's software.

Note: This is used for messages of 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. Nevertheless, regardless of the number of data bytes: The user's software has to reset this bit if CEIE is enabled. Otherwise a new interrupt will be generated immediately after return from the interrupt service routine.

RERR Receive 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 transmit 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. 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 transmission; 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 cancelled.

Functional Block Description Of The CAN Interface (Continued)

RECEIVE/TRANSMIT STATUS (RTSTAT) (Address X'00BC)

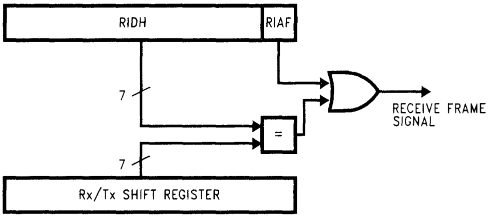
TBE	TXPND	RRTR	ROLB	RORN	RFV	RCV	RBF
1	0	0	0	0	0	0	0

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 or 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.



TL/DD/12067-20

FIGURE 20. Acceptance Filter Block-Diagram

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 cancelled 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 beginning 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 did not maintain the RxDn registers when receiving a frame. 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.

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 was 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	TCLC1	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.

Functional Block Description Of The CAN Interface (Continued)

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)

REC7	REC6	REC5	REC4	REC3	REC2	REC1	REC0

Bit 7 Bit 0

This register is read/write.

MESSAGE IDENTIFICATION

a) Transmitted Messages

The user can select all 11 Transmit Identifier Bits to transmit any message which 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 in 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 tristated (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 over 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 chapters 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 filter bits, are stored 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 a 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. This device is incapable of initiating an overload frame, but will join an 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 21.*

Basic CAN Concepts (Continued)

AUTOMATIC RETRANSMISSION OF FRAMES

If a data or remote frame was 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 acknowledged 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 can not 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 22* and *Figure 23* 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:

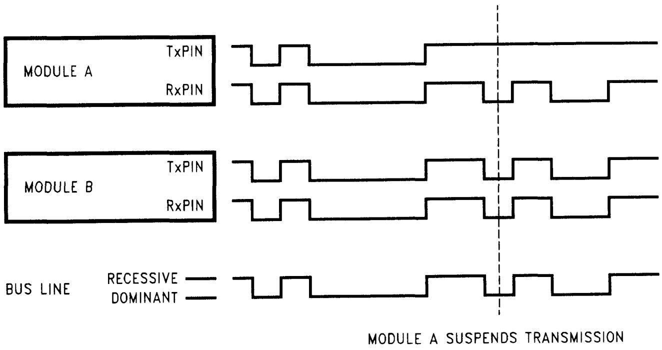
1. Start of Frame (SOF)
2. Arbitration field
3. Control field (IDE bit, R0 bit, and DLC field)
4. Data field (not in remote frame)
5. CRC field
6. ACK field
7. End of Frame (EOF)

A remote frame has no data field and is used for requesting data from other (remote) CAN nodes. *Figure 24* shows the format of a CAN data frame.

FRAME CODING

Remote and Data Frames are NRZ coded 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.



TL/DD/12067-21

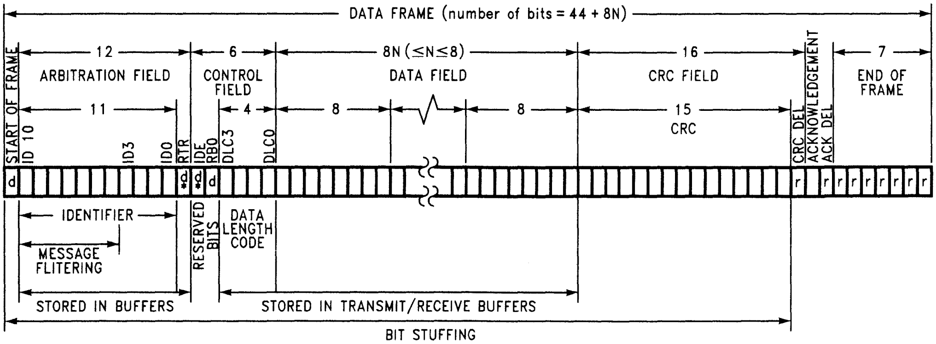
FIGURE 21. CAN Message Arbitration

Frame Formats (Continued)

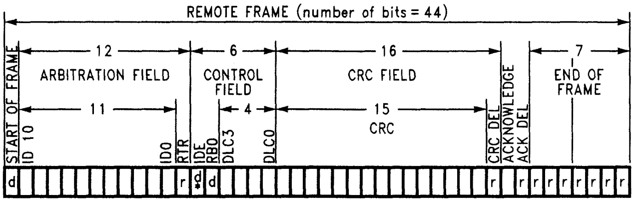
BIT STUFFING

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
Unstuffed Bit Stream	1000001x	0111110x
		x = {0, 1}



TL/DD/12067-22



TL/DD/12067-23

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

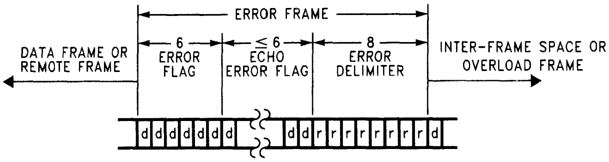
The IDE bit in the standard format is transmitted "dominant", whereas in the extended format the IDE bit is "recessive" and the id is expanded to 29 bits.

r = recessive

d = dominant

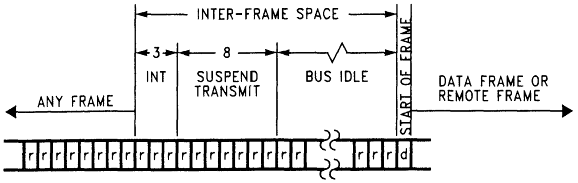
FIGURE 22. CAN Data Transmission Frames

Frame Formats (Continued)



TL/DD/12067-24

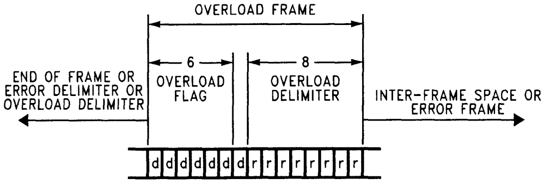
An error frame can start anywhere in the middle of a frame.



TL/DD/12067-25

INT = Intermission

Suspend Transmission is only for error passive nodes.



TL/DD/12067-26

An overload frame can only start at the end of a frame.

FIGURE 23. CAN Control Frames

SOF	Arbitration Field Identifier + RTR	Control Field	Data Field (If Present)	CRC Field	ACK Field	EOF
1-Bit	12-Bit	6-Bit	n * Bit	16-Bit	2-Bit	7-Bit

n ϵ (0,8)

FIGURE 24. CAN Frame Format

Frame Formats (Continued)

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 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" 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 frame 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 admissible 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.

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 separated from every preceding frame (data, remote, error and overload frames) by the interframe space see *Figure 25* and *Figure 26* 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 signalling an overload condition. This means that also 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" startbit 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 flag 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 is detecting 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 27* 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 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 has 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 28* 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.

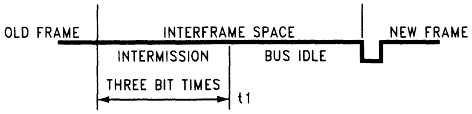
Frame Formats (Continued)

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 this way destroys the fixed form of the intermission field.

ORDER OF BIT TRANSMISSION

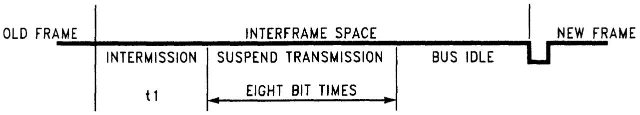
A frame is transmitted starting with the Start of Frame, sequentially followed by the remaining bit fields. In every bit field the MSB is transmitted first.



TL/DD/12067-27

t1 is the first possible start bit of a new frame

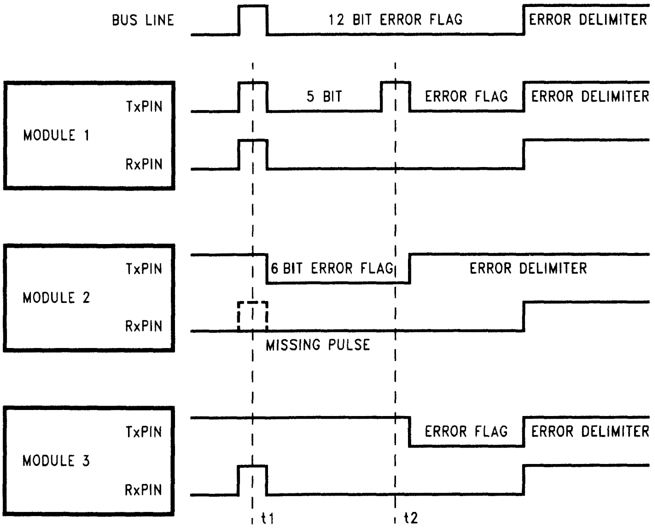
FIGURE 25. Interframe Space for Nodes Which Are Not Error Passive or Have Been Receiver for The Last Frame



TL/DD/12067-28

t1 - any module can start transmission except the error passive module which has transmitted the last frame.

FIGURE 26. Interframe Space for Nodes Which Are Error Passive and Have Been Transmitter for The Last Frame



TL/DD/12067-29

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 27. Error Frame—Error Active Transmitter

Frame Formats (Continued)

FRAME VALIDATION

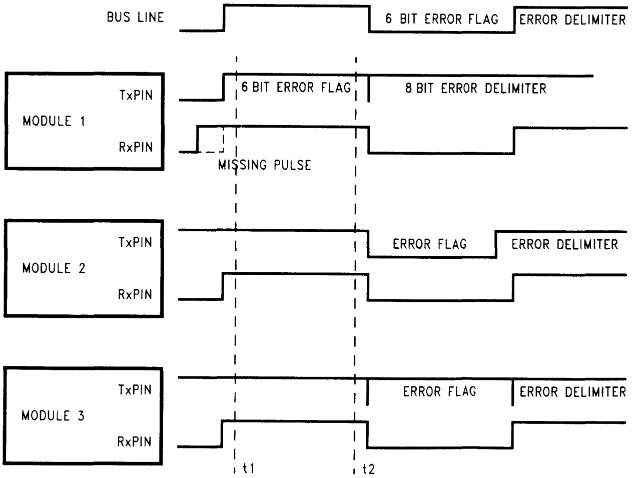
Frames have a different validation point for transmitter and receivers. A frame is valid for the transmitter of a message, if there is no error until the end of the last bit of End of Frame field. A frame is valid for a receiver, if there is no error until and including the end of the penultimate bit 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 collision on the bus:

- Within one system each message must be assigned to 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. Which could happen if two or more modules may 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 would 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 results in bit errors.



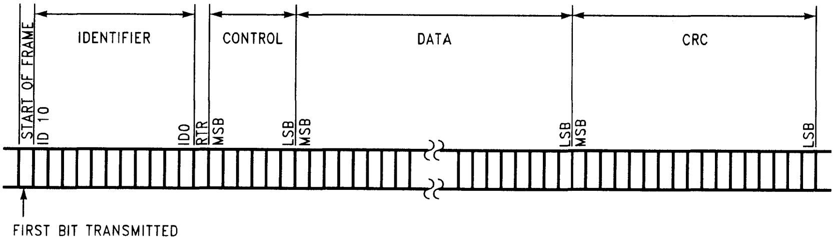
TL/DD/12067–30

module 1 = error passive transmitter detects bit error at t2

module 2 = error active receiver with a local fault at t1

module 3 = error passive receiver detects stuff error at t2

FIGURE 28. Error Frame—Error Passive Transmitter



TL/DD/12067–31

FIGURE 29. Order of Bit Transmission within a CAN Frame

Frame Formats (Continued)

ACCEPTANCE FILTERING

Every node performs 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 this 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 the 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.

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 frame 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 could at least transmit 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-bit 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 also 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.

The counters are modified by the device's hardware according to the following rules:

TABLE VI. 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

Frame Formats (Continued)

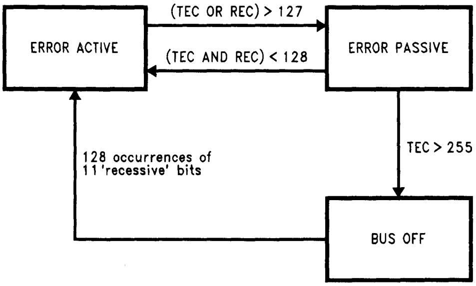
TABLE VII. Transmit Error Counter Handling

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 "dominant" 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 the 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 30 shows the connection of different bus states according to the error counters.



TL/DD/12067-32

FIGURE 30. CAN Bus States

SYNCHRONIZATION

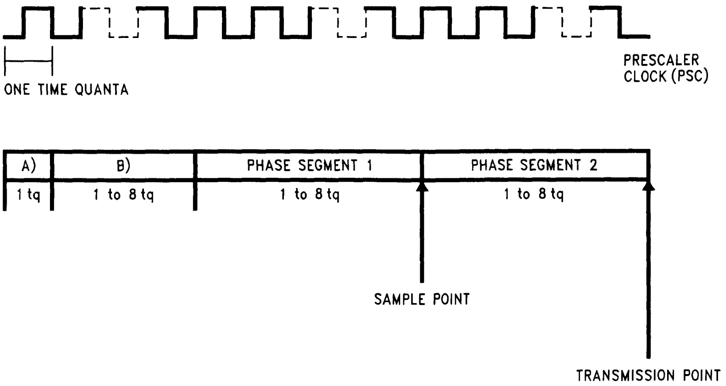
Every receiver starts with a "hard synchronization" on the falling edge of the SOF bit. One bit time consists of four bit segments: Synchronization segment, propagation segment, phase segment 1 and phase segment 2.

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 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 31*).



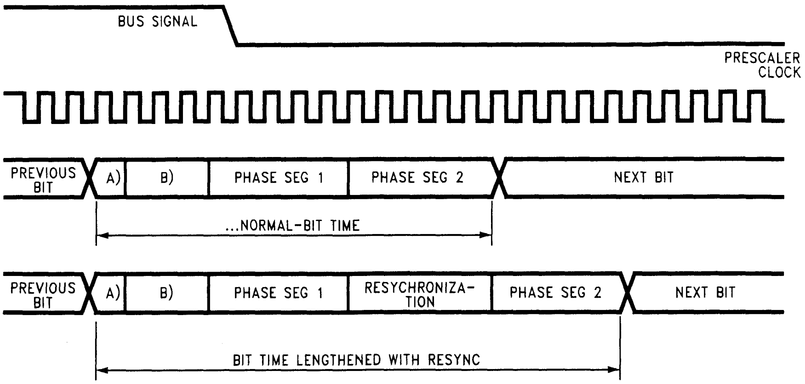
TL/DD/12067-33

A) synchronization segment

B) propagation segment

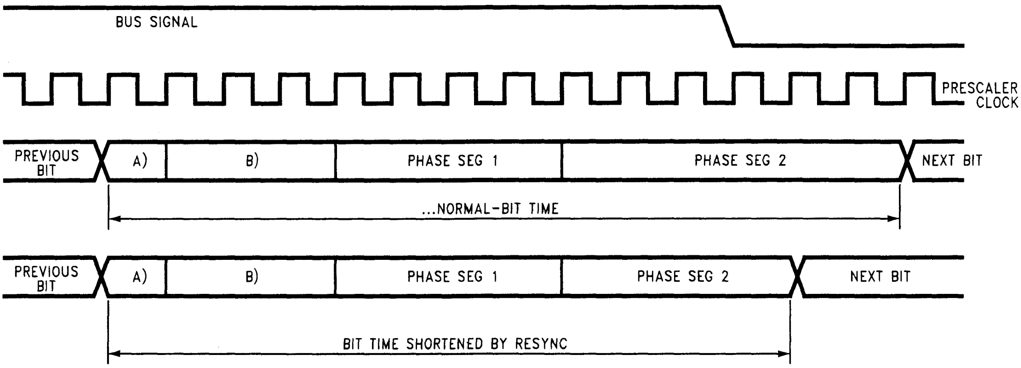
FIGURE 31. Bit Timing

Frame Formats (Continued)



TL/DD/12067-34

FIGURE 32. Resynchronization 1



TL/DD/12067-35

FIGURE 33. Resynchronization 2

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.

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 34*, 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 34 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

The register contains the following bits:

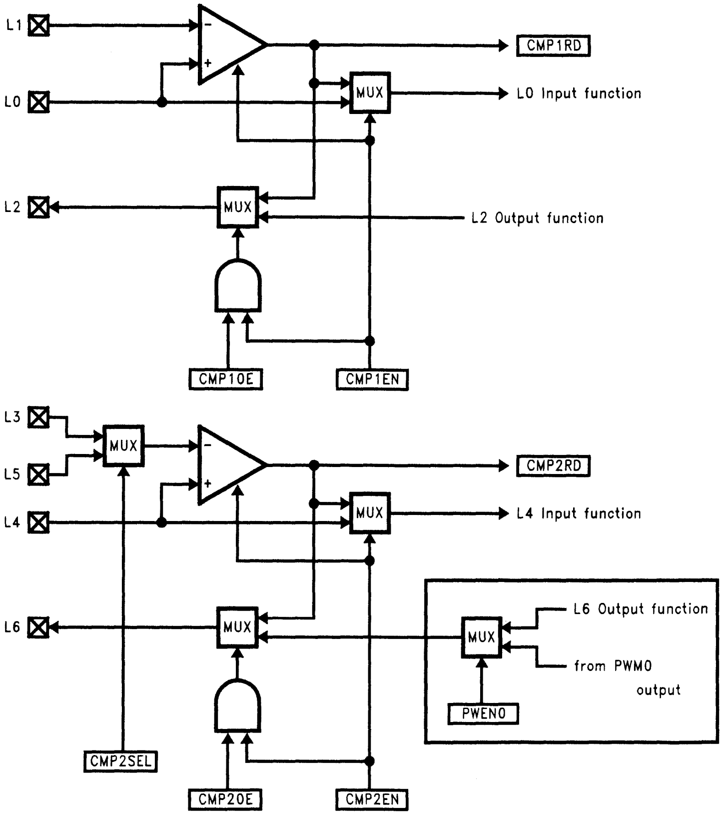
CMP1EN	Enables comparator 1 ("1" = enable). If comparator 1 is disabled the associated L-pins can be used as standard I/O.
CMP1RD	Reads comparator 1 output internally (CMP1EN = 1) Read-only, reads as a "0" if comparator not enabled.

Comparators (Continued)

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 I/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 SHADED area shows logic from PWM Timer. Comparator 2 output (CMP2OE) must be disabled in order to use PWM0 output.

FIGURE 34. 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 34* 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 35* shows how two COP888 family microcontrollers and several peripherals may be interconnected using the MICROWIRE/PLUS arrangements.

Warning:

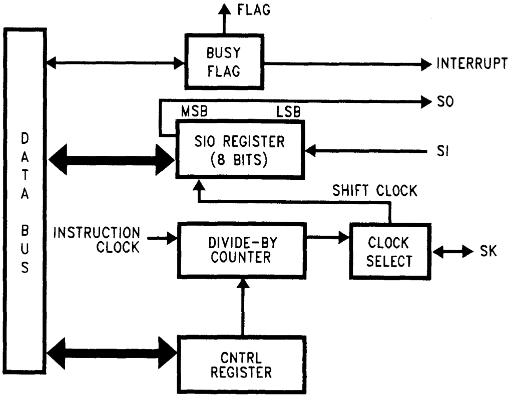
The SIO register should only be loaded when the SK clock is low. Loading the SIO register while the SK clock is high will result in undefined data in the SIO register. SK clock is normally low when not shifting.

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

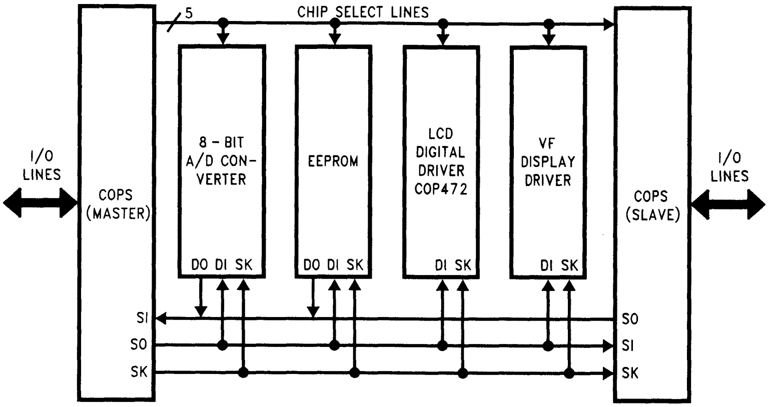
In the MICROWIRE/PLUS Master mode of operation the shift clock (SK) is generated internally. The MICROWIRE Master always initiates all data exchanges. The MSEL bit in the CNTRL register must be set to enable the SO and SK functions onto the G Port. The SO and SK pins must also be selected as outputs 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/12067-37

FIGURE 35. MICROWIRE/PLUS Block Diagram



TL/DD/12067-38

FIGURE 36. MICROWIRE/PLUS Application

MICROWIRE/PLUS (Continued)

TABLE VIII. 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 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

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 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 accumuiator 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]
MemI	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

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.
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	JP − 15	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 − 14	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 − 13	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 − 12	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 − 11	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 − 10	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 − 9	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 − 7	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 − 6	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 − 5	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 − 4	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 − 3	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 − 2	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 − 1	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 − 0	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 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 RESET
- = 1 Enable ON-CHIP RESET
- = 2 Disable ON-CHIP 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

IN-CIRCUIT EMULATOR

The MetaLink iceMASTER™-COP8 Model 400 In-Circuit Emulator for the COP8 family of microcontrollers features high-performance operation, ease of use, and an extremely flexible user-interface or maximum productivity. Interchangeable probe cards, which connect to the standard common base, support the various configurations and packages of the COP8 family.

The iceMASTER provides real time, full speed emulation up to 10 MHz, 32 kbytes of emulation memory and 4k frames of trace buffer memory. The user may define as many as 32k trace and break triggers which can be enabled, disabled, set or cleared. They can be simple triggers based on code address, direct address, opcode value, opcode class or immediate operand. Complex breakpoints can be ANDed and ORed together. Trace information consists of address bus values, opcodes and user selectable probe clips status (external event lines). The trace buffer can be viewed as raw hex or as disassembled instructions. The probe clip bit values can be displayed in binary, hex or digital waveform formats.

During single-step operation the dynamically annotated code feature displays the contents of all accessed (read and write) memory locations and registers, as well as flow-of-control direction change markers next to each instruction executed.

The iceMASTER's performance analyzer offers a resolution of better than 6 μs. The user can easily monitor the time spent executing specific portions of code and find "hot spots" or "dead code". Up to 15 independent memory areas based on code address or label ranges can be defined. Analysis results can be viewed in bar graph format or as actual frequency count.

Emulator memory operations for program memory include single line assembler, disassembler, view, change and write to file. Data memory operations include fill, move, compare, dump to file, examine and modify. The contents of any memory space can be directly viewed and modified from the corresponding window.

The iceMASTER comes with an easy to use window interface. Each window can be sized, highlighted, color-controlled, added, or removed completely. Commands can be accessed via pull-down-menus and/or redefinable hot keys. A context sensitive hypertext/hyperlinked on-line help system explains clearly the options the user has from within any window.

The iceMASTER connects easily to a personal computer via the standard COMM port and its 115.2 kBaud serial link keeps typical program download time shorter.

The following tables list the emulator and probe cards ordering information.

Emulator Ordering Information

Part Number	Description	Current Version
IM-COP8/400/1‡	MetaLink base unit in-circuit emulator for all COP8 devices, symbolic debugger software and RS232 serial interface cable, with 110V @ 60 Hz Power Supply.	Host Software: Ver. 3.3 Rev. 5, Model File Rev 3.050.
IM-COP8/400/2‡	MetaLink base unit in-circuit emulator for all COP8 devices, symbolic debugger software and RS232 serial interface cable, with 220V @ 50 Hz Power Supply.	

‡These parts include National's COP8 Assembler/Linker/Librarian Package (COP8-DEV-IBMA).

Probe Card Ordering Information

Part Number	Package	Voltage Range	Emulates
MHW-884BC28D5PC	28 DIP	4.5V–5.5V	COP884BC
MHW-SOIC28	28 SO	28-Pin SOIC	Adaptor Kit

MACRO CROSS ASSEMBLER

National Semiconductor offers a relocatable COP8 macro cross assembler. It runs on industry standard compatible PCs and supports all of the full-symbolic debugging features of the MetaLink iceMASTER emulators.

Assembler Ordering Information

Part Number	Description	Manual
COP8-DEV-IBMA	COP8 Assembler/Linker/Librarian for IBM® PC/XT®, AT® or compatible.	424410632-001

Development Support (Continued)

SINGLE CHIP EMULATOR DEVICE

The COP8 family is fully supported by single chip form, fit and function emulators. For more detailed information refer to the emulation deviced specific datasheets and the form, fit, function emulator selection table below.

COP684BC/COP884BC Ordering Information

Device Number	Clock Option	Package	Emulates
COP884BCMHEA-X*	Crystal R/C	28 LCC	COP884BC

*Check with the local sales office about the availability.

DIAL-A-HELPER

Dial-A-Helper is a service provided by the Microcontroller Applications group. The Dial-A-Helper is an Electronic Bulletin Board Information system.

PROGRAMMING SUPPORT

Programming of the single chip emulator devices is supported by different sources.

The following programmers are certified for programming EPROM versions of COP8:

INFORMATION SYSTEM

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 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. The minimum requirement for accessing the Dial-A-Helper is a Hayes compatible modem.

If the user has a PC with a communications package then files from the FILE SECTION can be down loaded to disk for later use.

ORDER P/N: MOLE-DIAL-A-HLP

Information System Package contains:
Dial-A-Helper Users Manual
Public Domain Communications Software

FACTORY APPLICATIONS SUPPORT

Dial-A-Helper also provides immediate factor applications support. If a user has questions, he can leave messages on our electronic bulletin board, which we will respond to.

Voice: (800) 272-9959
Modem: CANADA/U.S.: (800) NSC-MICRO
(800) 672-6427
Baud: 14.4k
Set-Up: Length: 8-Bit
Parity: None
Stop Bit: 1
Operation: 24 Hrs., 7 Days

EPROM Programmable Information

Manufacturer and Product	U.S. Phone Number	Europe Phone Number	Asia Phone Number
MetaLink- Debug Module	(602) 926-0797	Germany: +49-8141-1030	Hong Kng: +852-737-1800
Xeltek- Superpro	(408) 745-7974	Germany: +49 2041-684758	Singapore: +65-276-6433
BP Microsystems- Turpro	(800) 225-2102	Germany: +49 2041-684758	Hong Kong: +852-388-0629
Data I/O-Unisite -System 29 -System 39	(800) 322-8246	Europe: +31-20-622866 Germany: +49-89-858020	Japan: +81-33-432-6991
Abcom-COP8 Programmer		Europe: +49-89 808707	
System General- Turpro-1-FX -APRO	(408) 263-6667	Switzerland: +41-31 921-7844	Taiwan: +886-2-917-3005

COP688CL/COP684CL/COP888CL/COP884CL/COP988CL/COP984CL

National Semiconductor

COP688CL/COP684CL, COP888CL/COP884CL, COP988CL/COP984CL Single-Chip microCMOS Microcontroller

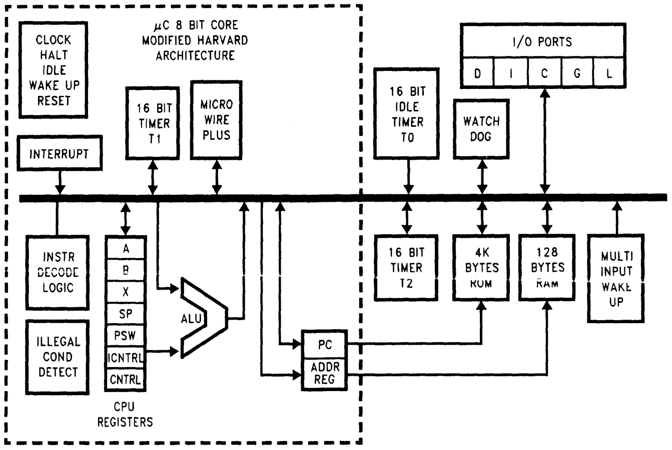
General Description

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)

Features

- Low cost 8-bit microcontroller
- Fully static CMOS, with low current drain
- Two power saving modes: HALT and IDLE
- 1 μs instruction cycle time
- 4096 bytes on-board ROM
- 128 bytes on-board RAM
- Single supply operation: 2.5V–6V
- MICROWIRE/PLUS™ serial I/O
- WATCHDOG™ and Clock Monitor logic
- Idle Timer
- Multi-Input Wakeup (MIWU) with optional interrupts (8)
- Ten multi-source vectored interrupts servicing
  - — External Interrupt
  - — Idle Timer T0
  - — Timers TA, TB (Each with 2 Interrupts)
  - — MICROWIRE/PLUS
  - — Multi-Input Wake Up
  - — Software Trap
  - — Default VIS
- Two 16-bit timers, each with two 16-bit registers supporting:
  - — Processor Independent PWM mode
  - — External Event counter mode
  - — Input Capture mode
- 8-bit Stack Pointer SP (stack in RAM)
- Two 8-bit Register Indirect Data Memory Pointers (B and X)
- Versatile instruction set
- True bit manipulation
- Memory mapped I/O
- BCD arithmetic instructions
- Package:
  - — 44 PLCC with 39 I/O pins
  - — 40 N with 33 I/O pins
  - — 28 SO or 28 N, each with 23 I/O pins
- 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
- Temperature ranges: 0°C to +70°C,
  −40°C to +85°C,
  −55°C to +125°C
- One-Time Programmable (OTP) emulation device
- Fully supported by Metalink's Development Systems

Block Diagram



TL/DD/9766–1

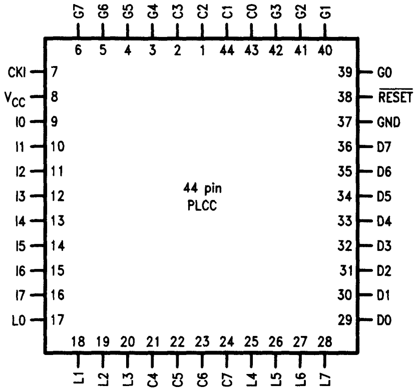
FIGURE 1. Block Diagram

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

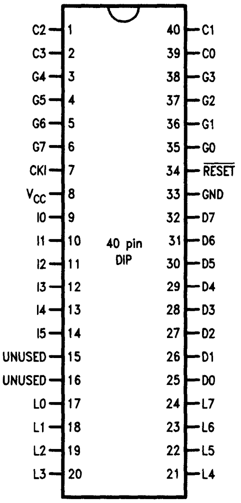


TL/DD/9766-2

Top View

Order Number COP688CL-XXX/V, COP888CL-XXX/V or COP988CL-XXX/V
See NS Plastic Chip Package Number V44A

Dual-In-Line Package

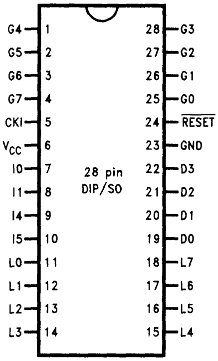


TL/DD/9766-4

Top View

Order Number COP688CL-XXX/N, COP888CL-XXX/N or COP988CL-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 or COP984CL-XXX/N
See NS Molded Package Number N28B

Order Number COP684CL-XXX/WM, COP884CL-XXX/WM or COP984CL-XXX/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: $0°C \leq T_A \leq +70°C$ 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	$V_{CC} = 6V$, $t_c = 1\ \mu s$			12.5	mA
CKI = 4 MHz	$V_{CC} = 4V$, $t_c = 2.5\ \mu s$			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	$V_{CC} = 6V$, $t_c = 1\ \mu s$			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 $0°C \le T_A \le +70°C$ 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 $0°C \le T_A \le +70°C$ unless otherwise specified

Parameter	Conditions	Min	Typ	Max	Units
Instruction Cycle Time (t_c)					
Crystal or 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 5)	$R_L = 2.2k, C_L = 100$ 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: 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	$V_{CC} = 6V$, $t_c = 1\ \mu s$			12.5	mA
CKI = 4 MHz	$V_{CC} = 4V$, $t_c = 2.5\ \mu s$			2.5	mA
HALT Current (Note 3)	$V_{CC} = 6V$, CKI = 0 MHz		<1	10	μA
IDLE Current					
CKI = 10 MHz	$V_{CC} = 6V$, $t_c = 1\ \mu s$			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: $-55°C \leq T_A \leq +125°C$ 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					
$\overline{\text{RESET}}$					
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 $-55°C < T_A < +25°C$ 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)				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

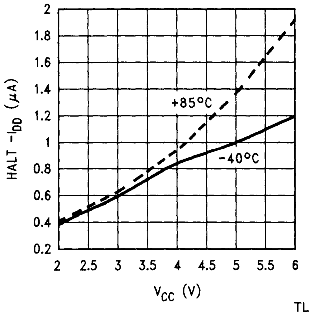
Parameter	Conditions	Min	Typ	Max	Units
Instruction Cycle Time (t_c)					
Crystal, Resonator, or External Oscillator	$V_{CC} \geq 4.5V$	1		DC	μs
R/C Oscillator (div-by 10)	$V_{CC} \geq 4.5V$	3		DC	μs
Inputs					
t_{SETUP}	$V_{CC} \geq 4.5V$	200			ns
t_{HOLD}	$V_{CC} \geq 4.5V$	60			ns
Output Propagation 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.

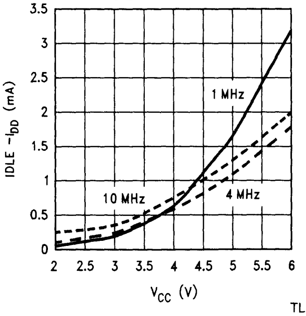
Typical Performance Characteristics ($-40°C \leq T_A \leq +85°C$)

Halt—I_{DD}



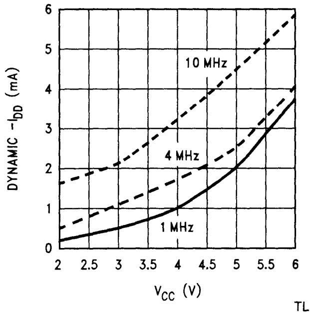
TL/DD/9766-27

Idle—I_{DD} (Crystal Clock Option)



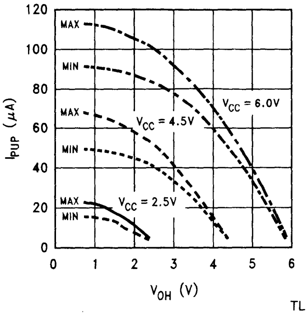
TL/DD/9766-28

Dynamic—I_{DD} vs V_{CC} (Crystal Clock Option)



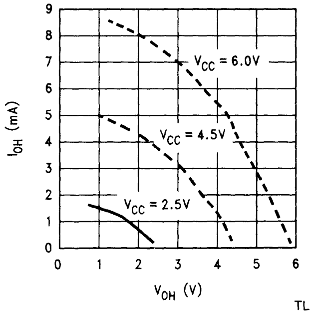
TL/DD/9766-29

Port L/C/G Weak Pull-Up Source Current



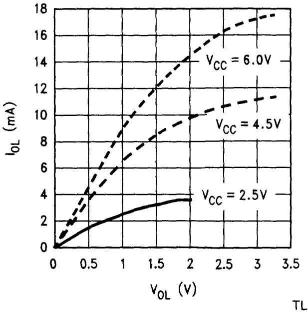
TL/DD/9766-30

Port L/C/G Push-Pull Source Current



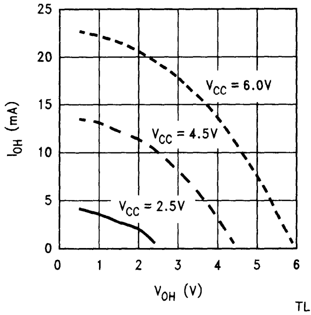
TL/DD/9766-31

Port L/C/G Push-Pull Sink Current



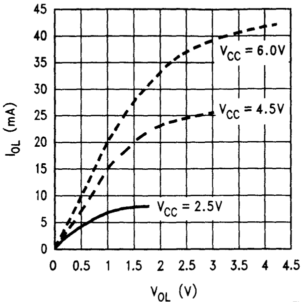
TL/DD/9766-32

Port D Source Current



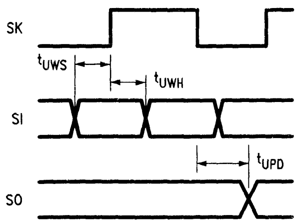
TL/DD/9766-33

Port D Sink Current



TL/DD/9766-34

AC Electrical Characteristics (Continued)



TL/DD/9766-26

FIGURE 2. MICROWIRE/PLUS Timing

Pin Descriptions

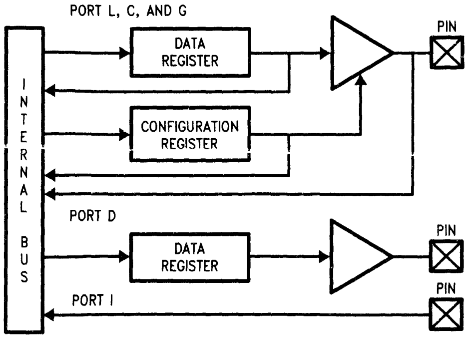
V_{CC} and GND are the power supply pins.

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.

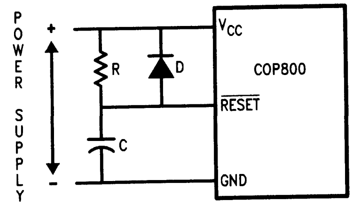
Reset

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

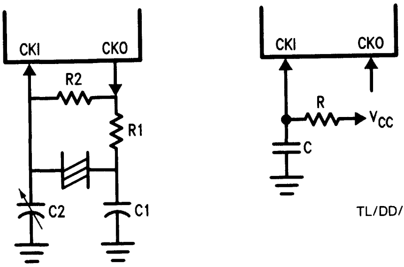
CKI and CKO can be connected to make a closed loop crystal (or resonator) controlled oscillator.

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: $3k \leq R \leq 200k$, $50\ pF \leq C \leq 200\ pF$

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. Clock Monitor current when enabled—I6

Thus the total current drain, It, is given as

$$It = I1 + I2 + I3 + I4 + I5 + I6$$

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

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)

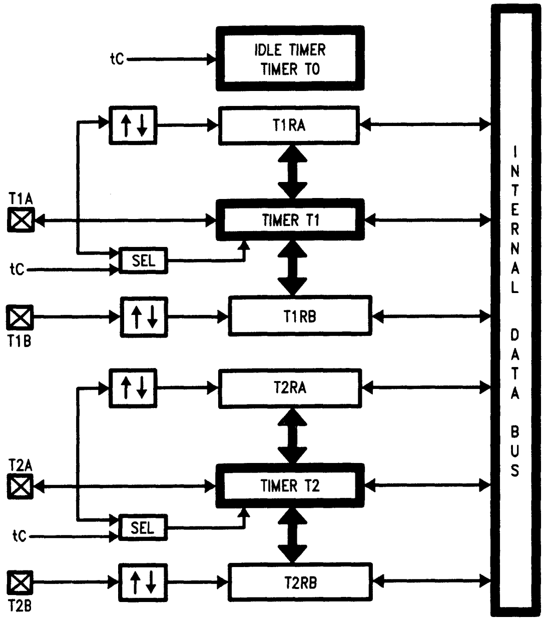


FIGURE 6. Timers

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 ($t_C = 1\ \mu s$). 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)

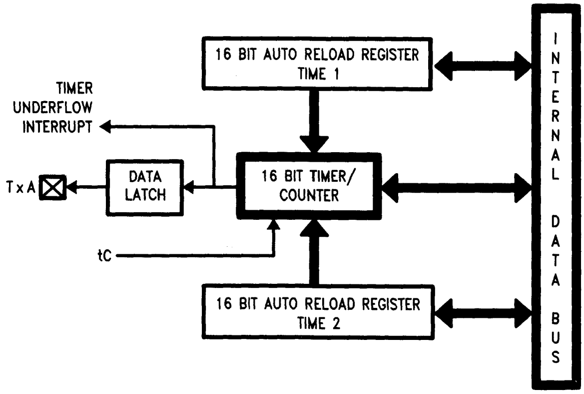


FIGURE 7. Timer in PWM Mode

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.

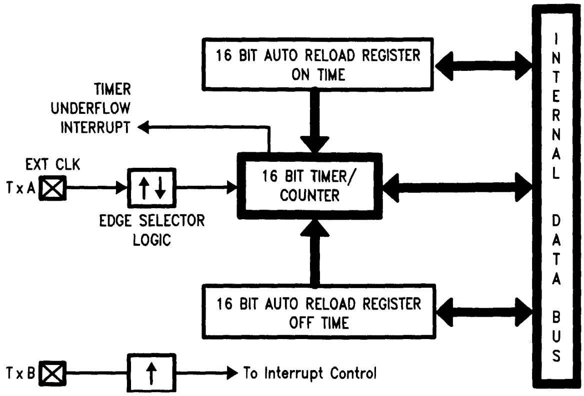


FIGURE 8. Timer in External Event Counter Mode

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.

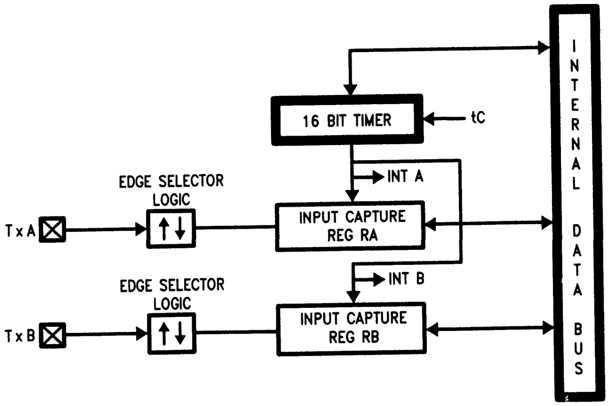
Underflows from the timer can also be programmed to generate interrupts. Underflows are latched into the timer TxC0 pending flag (the TxC0 control bit serves as the timer underflow interrupt pending flag in the Input Capture mode). Consequently, the TxC0 control bit should be reset when entering the 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, $t_C = 1 \mu s$) 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 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.

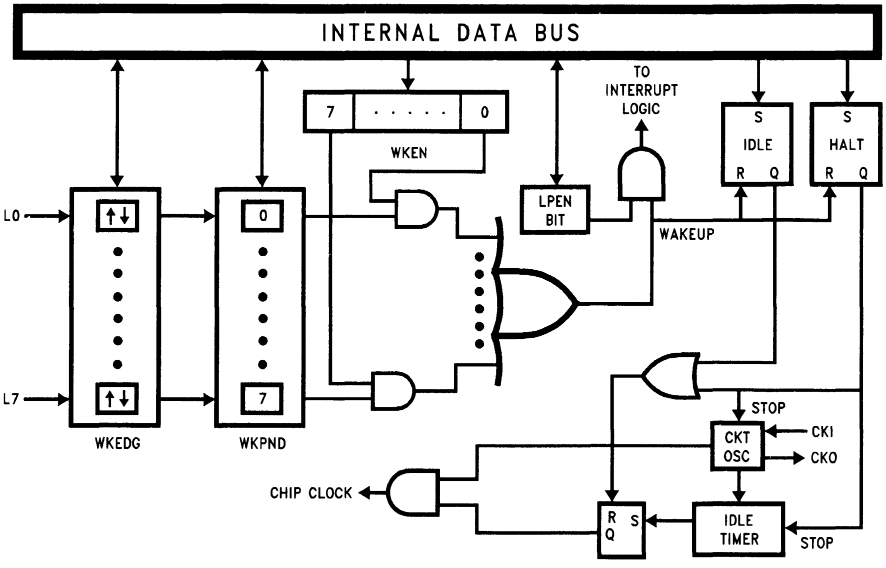
The occurrence of the selected trigger condition for Multi-Input Wakeup is latched into a pending register called WKPND. The respective bits of the WKPND register will be set on the occurrence of the selected trigger edge on the corresponding Port L pin. The user has the responsibility of clearing these 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.

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)

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.

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.

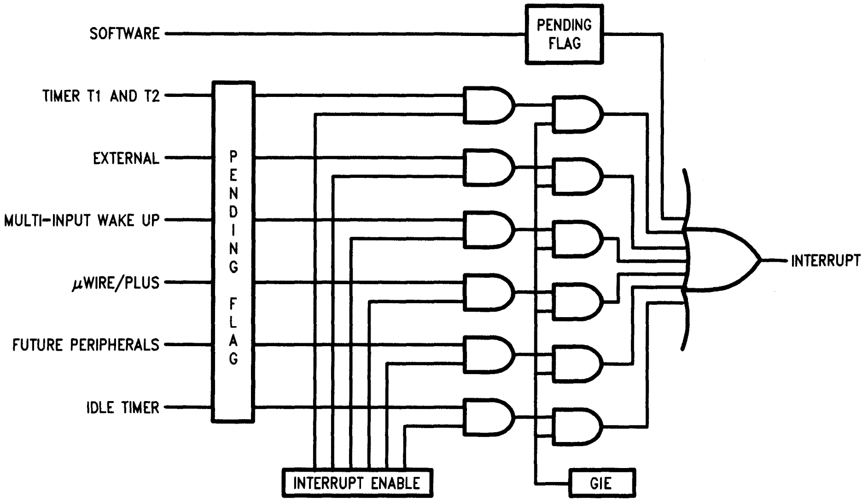
VIS and the vector table must be located in the same 256-byte block (0y00 to 0yFF) except if VIS is located at the last address of a block. In this case, the table must be in the next block. The vector table cannot be inserted in the first 256-byte block.

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. This vector can point to the Software Trap (ST) interrupt service routine, or to another special service routine as desired.

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

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 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

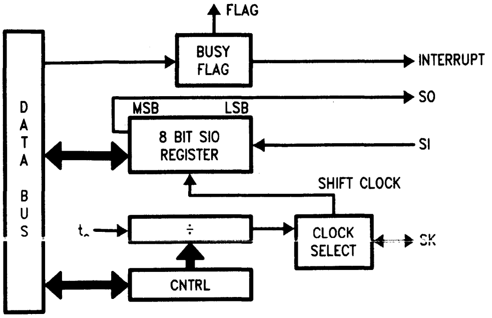
b. Over "POP"ing the stack by having more returns than calls.

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

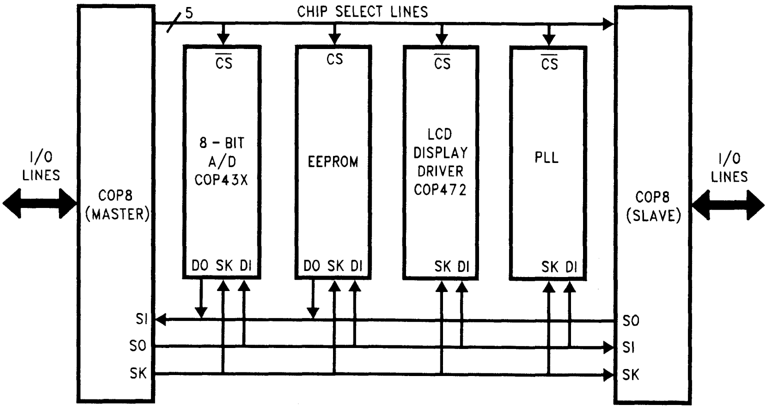
In the MICROWIRE/PLUS Slave mode of operation the SK clock is generated by an external source. Setting the MSEL bit in the CNTRL register enables the SO and SK functions onto the G Port. The SK pin must be selected as an input and the SO pin is selected as an output pin by setting and resetting the appropriate 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]
MemI	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.

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/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
  = 1     44-Pin PCC
  = 2     40-Pin DIP
  = 3     N.A.
  = 4     28-Pin DIP
  = 5     28-Pin SO
```

Development Support

IN-CIRCUIT EMULATOR

The MetaLink iceMASTER™-COP8 Model 400 In-Circuit Emulator for the COP8 family of microcontrollers features high-performance operation, ease of use, and an extremely flexible user-interface for maximum productivity. Interchangeable probe cards, which connect to the standard common base, support the various configurations and packages of the COP8 family.

The iceMASTER provides real-time, full-speed emulation, up to 10 MHz, 32 kBytes of emulation memory and 4k frames of trace buffer memory. The user may define as many as 32k trace and break triggers which can be enabled, disabled, set or cleared. They can be simple triggers based on code or address ranges or complex triggers based on code address, direct address, opcode value, opcode class or immediate operand. Complex breakpoints can be ANDed and ORed together. Trace information consists of address bus values, opcodes and user-selectable probe clips status (external event lines). The trace buffer can be viewed as raw hex or as disassembled instructions. The probe clip bit values can be displayed in binary, hex or digital waveform formats.

During single-step operation the dynamically annotated code feature displays the contents of all accessed (read and write) memory locations and registers, as well as flow-of-control direction change markers next to each instruction executed.

The iceMASTER's performance analyzer offers a resolution of better than 6 μs. The user can easily monitor the time spent executing specific portions of code and find "hot spots" or "dead code". Up to 15 independent memory areas based on code address or label ranges can be defined. Analysis results can be viewed in bar graph format or as actual frequency count.

Emulator memory operations for program memory include single line assembler, disassembler, view, change and write to file. Data memory operations include fill, move, compare, dump to file, examine and modify. The contents of any memory space can be directly viewed and modified from the corresponding window.

The iceMASTER comes with an easy to use windowed interface. Each window can be sized, highlighted, color-controlled, added, or removed completely. Commands can be accessed via pull-down-menus and/or redefinable hot keys. A context sensitive hypertext/hyperlinked on-line help system explains clearly the options the user has from within any window.

The iceMASTER connects easily to a PC® via the standard COMM port and its 115.2 kBaud serial link keeps typical program download time to under 3 seconds.

The following tables list the emulator and probe cards ordering information.

Emulator Ordering Information

Part Number	Description	Current Version
IM-COP8/400/1‡	MetaLink base unit in-circuit emulator for all COP8 devices, symbolic debugger software and RS-232 serial interface cable, with 110V @ 60 Hz Power Supply.	HOST SOFTWARE: VER. 3.3 REV.5, Model File Rev 3.050.
IM-COP8/400/2‡	MetaLink base unit in-circuit emulator for all COP8 devices, symbolic debugger software and RS-232 serial interface cable, with 220V @ 50 Hz Power Supply.	
DM-COP8/888CF‡	MetaLink iceMASTER Debug Module. This is the low cost version of the MetaLink's iceMASTER. Firmware: Ver. 6.07.	

‡These parts include National's COP8 Assembler/Linker/Librarian Package (COP8-DEV-IBMA).

Development Support (Continued)

Probe Card Ordering Information

Part Number	Package	Voltage Range	Emulates
MHW-884CL28D5PC	28 DIP	4.5V–5.5V	COP884CL
MHW-884CL28DWPC	28 DIP	2.3V–6.0V	COP884CL
MHW-888CL40D5PC	40 DIP	4.5V–5.5V	COP888CL
MHW-888CL40DWPC	40 DIP	2.3V–6.0V	COP888CL
MHW-888CL44D5PC	44 PLCC	4.5V–5.5V	COP888CL
MHW-888CL44DWPC	44 PLCC	2.5V–6.0V	COP888CL

MACRO CROSS ASSEMBLER

National Semiconductor offers a COP8 macro cross assembler. It runs on industry standard compatible PCs and supports all of the full-symbolic debugging features of the MetaLink iceMASTER emulators.

Assembler Ordering Information

Part Number	Description	Manual
COP8-DEV-IBMA	COP8 Assembler/ Linker/Librarian for IBM® PC/XT®, AT® or compatible	424410632-001

PROGRAM SUPPORT

Programming of the single-chip emulator devices is supported by different sources. The table below shows the programmers certified for programming the One-Time Programmable (OTP) devices.

EMULATOR DEVICE

The COP8 family is fully supported by One-Time Programmable (OTP) emulators. For more detailed information refer to the emulation device specific data sheets and the emulator selection table below.

OTP Ordering Information

Device Number	Clock Option	Package	Emulates
COP8788CLV-X COP8788CLV-R*	Crystal R/C	44 PLCC	COP888CL
COP8788CLN-X COP8788CLN-R*	Crystal R/C	40 DIP	COP888CL
COP8784CLN-X COP8784CLN-R*	Crystal R/C	28 DIP	COP884CL
COP8784CLWM-X* COP8784CLWM-R*	Crystal R/C	28 SO	COP884CL

*Check with the local sales office about the availability.

EPROM Programmer Information

Manufacturer and Product	U.S. Phone Number	Europe Phone Number	Asia Phone Number
MetaLink- Debug Module	(602) 926-0797	Germany: +49-8141-1030	Hong Kong: +852-737-1800
Xeltek- Superpro	(408) 745-7974	Germany: +49-2041-684758	Singapore: +65-276-6433
BP Microsystems- EP-1140	(800) 225-2102	Germany: +49-89-857-66-67	Hong Kong: +852-388-0629
Data I/O-Unisite; -System 29, -System 39	(800) 322-8246	Europe: +31-20-622866 Germany: +49-89-85-8020	Japan: +33-432-6991
Abcom-COP8 Programmer		Europe: +89-80 8707	
System General Turpro-1-FX; -APRO	(408) 263-6667	Switzerland: +31-921-7844	Taiwan Taipei: +2-9173005

Development Support (Continued)

DIAL-A-HELPER

Dial-A-Helper is a service provided by the Microcontroller Applications group. The Dial-A-Helper is an Electronic Bulletin Board Information System.

Information System

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 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. The minimum requirement for accessing the Dial-A-Helper is a Hayes compatible modem.

If the user has a PC with a communications package then files from the FILE SECTION can be down loaded to disk for later use.

Order P/N: MOLE-DIAL-A-HLP

Information System Package Contents
- Dial-A-Helper User Manual P/N
- Public Domain Communications Software

Factory Applications Support

Dial-A-Helper also provides immediate factor applications support. If a user has questions, he can leave messages on our electronic bulletin board, which we will respond to.

Voice: (800) 272-9959
Modem: CANADA/U.S.: (800) NSC-MICRO
(800) 672-6427
Baud: 14.4k
Set-up: Length: 8-Bit
Parity: None
Stop Bit: 1
Operation: 24 Hrs., 7 Days

National Semiconductor

COP988CF/COP984CF/COP888CF/COP884CF Single-Chip microCMOS Microcontroller

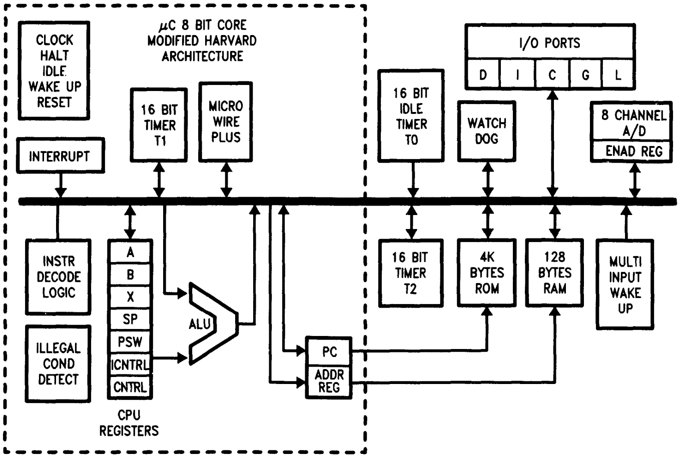
General Description

The COP888 family of microcontrollers uses an 8-bit single chip core architecture fabricated with National Semiconductor's M2CMOS™ process technology. The COP888CF is a member of this expandable 8-bit core processor family of microcontrollers. (Continued)

Features

- Low cost 8-bit microcontroller
- Fully static CMOS, with low current drain
- Two power saving modes: HALT and IDLE
- 1 μs instruction cycle time
- 4096 bytes on-board ROM
- 128 bytes on-board RAM
- Single supply operation: 2.5V–6V
- 8-channel A/D converter with prescaler and both differential and single ended modes
- MICROWIRE/PLUS™ serial I/O
- WATCHDOG™ and Clock Monitor logic
- 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
- Idle Timer
- Multi-Input Wakeup (MIWU) with optional interrupts (8)
- Two 16-bit timers, each with two 16-bit registers supporting:
  - — Processor Independent PWM mode
  - — External Event counter mode
  - — Input Capture mode
- 8-bit Stack Pointer SP (stack in RAM)
- Two 8-bit Register Indirect Data Memory Pointers (B and X)
- Versatile instruction set
- True bit manipulation
- Memory mapped I/O
- BCD arithmetic instructions
- Package:
  - — 44 PLCC with 37 I/O pins
  - — 40 N with 33 I/O pins
  - — 28 SO or 28 N, each with 23 I/O pins
- 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
- Temperature ranges: 0°C to +70°C
  −40°C to+ 85°C
- One-Time Programmable (OTP) emulation devices
- Real time emulation and full program debug offered by Metalink's Development Systems

Block Diagram



TL/DD/9425–1

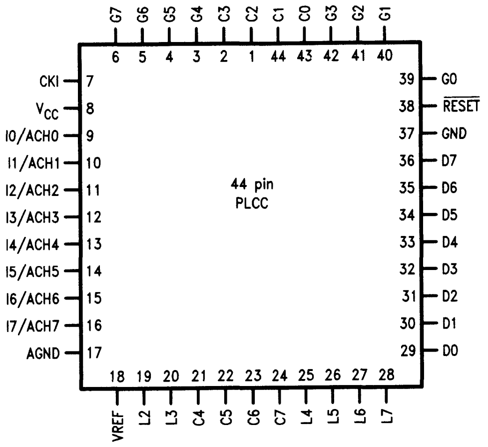
FIGURE 1. Block Diagram

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), 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

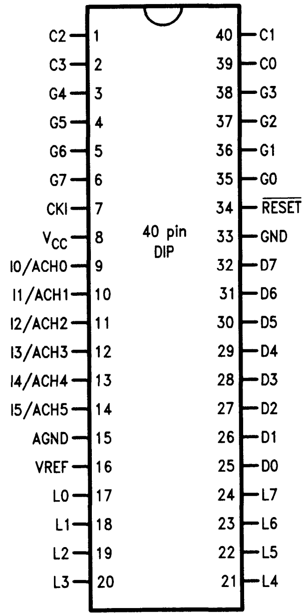


TL/DD/9425-2

Top View

Order Number COP888CF-XXX/V
See NS Plastic Chip Package Number V44A

Dual-In-Line Package

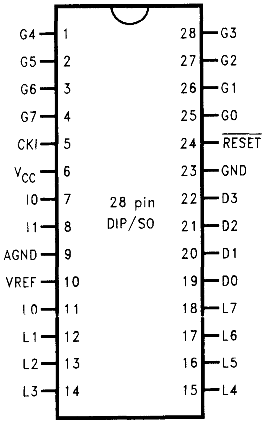


TL/DD/9425-4

Top View

Order Number COP888F-XXX/N
See NS Molded Package Number N40A

Dual-In-Line Package



TL/DD/9425-37

Top View

Order Number COP884CF-XXX/N
or COP884CF-XXX/WM
See NS Package Number D28G 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: $0°C \le T_A \le +70°C$ 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	$V_{CC} = 6V$, $t_c = 1\ \mu s$			12.5	mA
CKI = 4 MHz	$V_{CC} = 6V$, $t_c = 2.5\ \mu s$			5.5	mA
CKI = 4 MHz	$V_{CC} = 4V$, $t_c = 2.5\ \mu s$			2.5	mA
CKI = 1 MHz	$V_{CC} = 4V$, $t_c = 10\ \mu s$			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	$V_{CC} = 6V$, $t_c = 1\ \mu s$			3.5	mA
CKI = 4 MHz	$V_{CC} = 6V$, $t_c = 2.5\ \mu s$			2.5	mA
CKI = 1 MHz	$V_{CC} = 4.0V$, $t_c = 10\ \mu s$			0.7	mA
Input Levels					
$\overline{\text{RESET}}$					
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 ≤ T_A ≤ +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 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 ±10% (V_{SS} − 0.050V) ≤ Any Input ≤ (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.

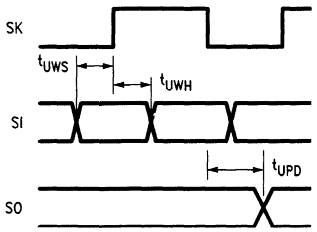
Note 6: Pins G6 and $\overline{\text{RESET}}$ are designed with a high voltage input network for factory testing. These pins allow input voltages greater than V_{CC} and the pins will have sink current to V_{CC} when biased at voltages greater than V_{CC} (the pins do not have source current when biased 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 $0°C < T_A < +70°C$ 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 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: $-40°C \leq T_A \leq +85°C$ 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	$V_{CC} = 6V, t_c = 1\ \mu s$			12.5	mA
CKI = 4 MHz	$V_{CC} = 4V, t_c = 2.5\ \mu s$			2.5	mA
HALT Current (Note 3)	$V_{CC} = 6V$, CKI = 0 MHz		<1	10	μA
IDLE Current					
CKI = 10 MHz	$V_{CC} = 6V, t_c = 1\ \mu s$			3.5	mA
CKI = 1 MHz	$V_{CC} = 4V, t_c = 10\ \mu s$			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)				15	mA
All others				3	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:

$V_{CC} = 5V \pm 10\% \ (V_{SS} - 0.050V) \leq \text{Any Input} \leq (V_{CC} + 0.050V)$

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.

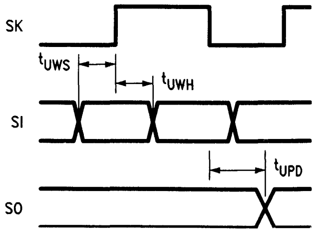
Note 6: Pins G6 and $\overline{RESET}$ are designed with a high voltage input network for factory testing. These pins allow input voltages greater than V_{CC} and the pins will have sink current to V_{CC} when biased at voltages greater than V_{CC} (the pins do not have source current when biased at 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: $-40^{\circ}C \leq T_A \leq +85^{\circ}C$ 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 end of the instruction cycle where the output change occurs.

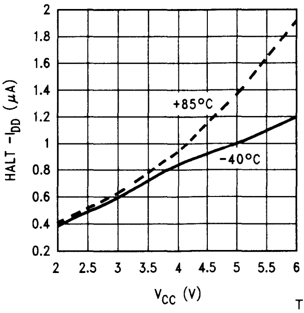


TL/DD/9425-26

FIGURE 3. MICROWIRE/PLUS Timing

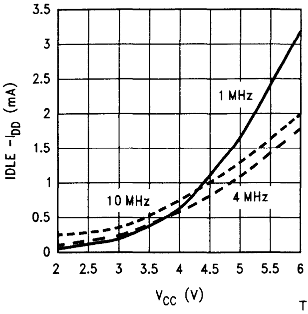
Typical Performance Characteristics (−40°C to +85°C)

Halt—I_{DD} vs V_{CC}



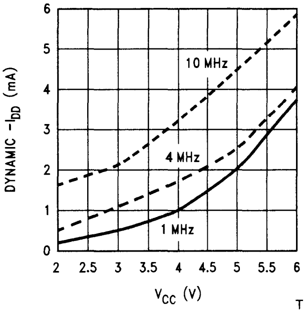
TL/DD/9425-29

Idle—I_{DD} (Crystal Clock Option)



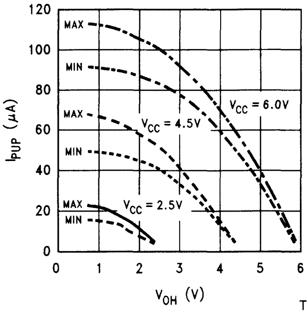
TL/DD/9425-30

Dynamic—I_{DD} (Crystal Clock Option)



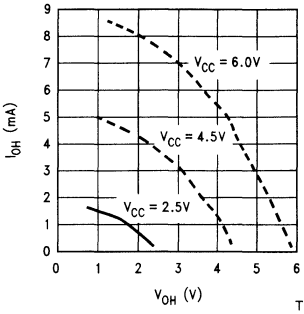
TL/DD/9425-31

Port L/C/G Weak Pull-Up Source Current



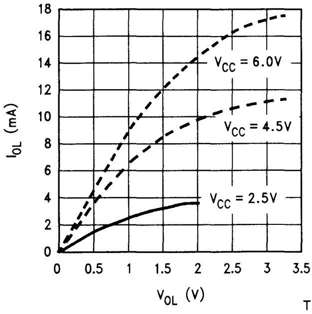
TL/DD/9425-32

Port L/C/G Push-Pull Source Current



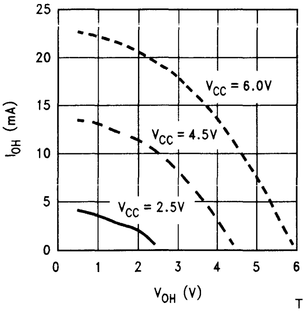
TL/DD/9425-33

Port L/C/G Push-Pull Sink Current



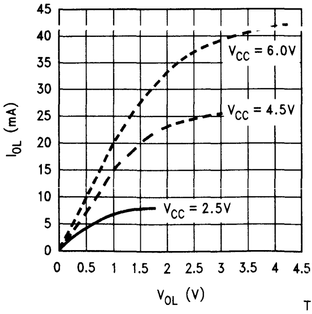
TL/DD/9425-34

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.

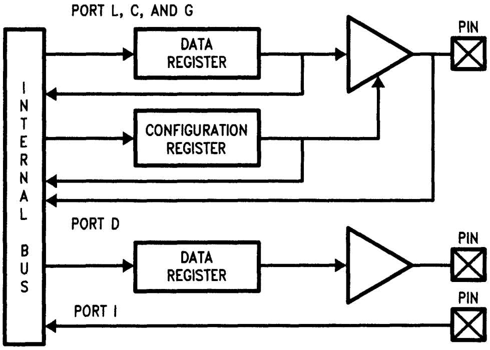
V_{REF} and AGND are the reference voltage pins for the on-board A/D converter.

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.

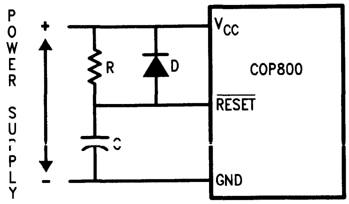
Note: RAM contents are undefined upon power-up.

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.

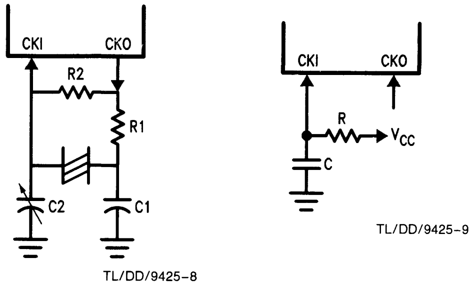


FIGURE 6. Crystal and R/C Oscillator Diagrams

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: $3k \le R \le 200k$
$50\ pF \le C \le 200\ pF$

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. DC reference current contribution from the A/D converter—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, 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

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

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

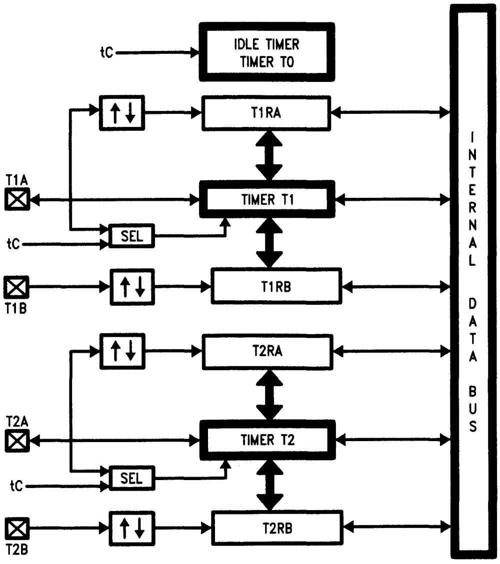
The IDLE Timer T0 can generate an interrupt when the thirteenth bit toggles. This toggle is latched into the T0PND pending flag, and will occur every 4 ms at the maximum clock frequency (t_c = 1 μs). A control flag T0EN allows the interrupt from the thirteenth bit of Timer T0 to be enabled or disabled. 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 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.



TL/DD/9425-11

FIGURE 7. Timers

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.

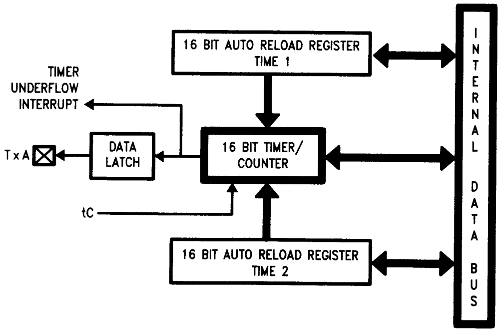
The Tx Timer control bits, TxC3, TxC2 and TxC1 set up the timer for PWM mode operation.

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

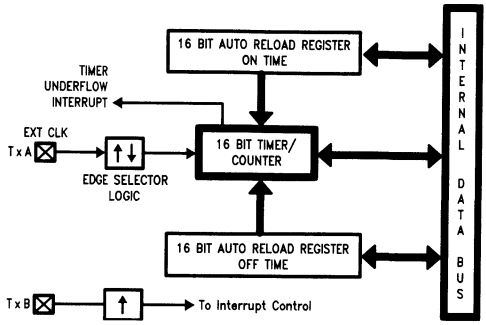
Mode 2. External Event Counter Mode

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.

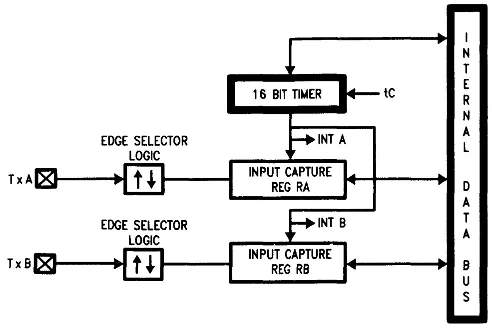
The trigger conditions can also be programmed to generate interrupts. The occurrence of the specified trigger condition on the TxA and TxB pins will be respectively latched into the pending flags, TxPNDA and TxPNDB. The control flag TxENA allows the interrupt on TxA to be either enabled or disabled. 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.

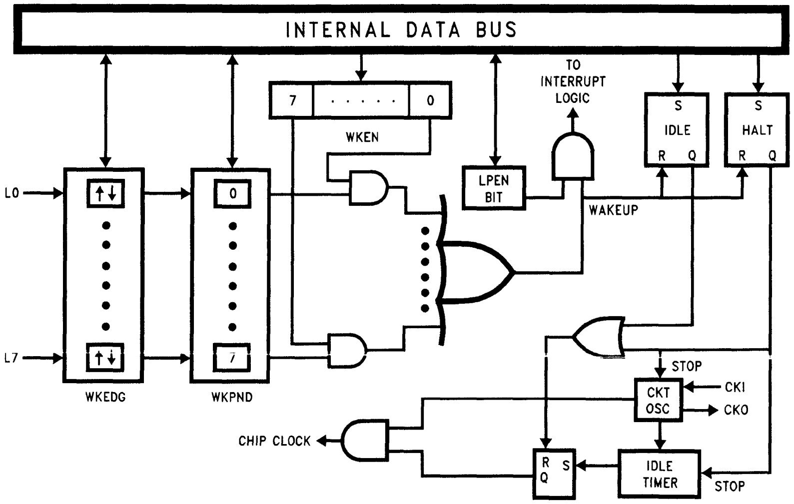
The occurrence of the selected trigger condition for Multi-Input Wakeup is latched into a pending register called WKPND. The respective bits of the WKPND register will be set on the occurrence of the selected trigger edge on the corresponding Port L pin. The user has the responsibility of clearing these 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.

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.

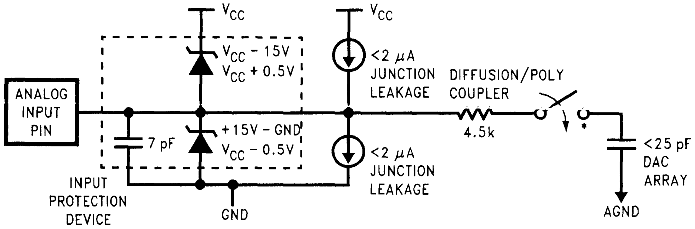
Note: The A/D converter is also powered down when the device is in either the HALT or IDLE modes. If the ADC is running when the device enters the HALT or IDLE modes, the ADC will power down during the HALT or IDLE, and then will reinitialize the conversion when the device comes out of the HALT or IDLE 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.

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.



*The analog switch is closed only during the sample time.

FIGURE 12. A/D Pin Model (Single Ended Mode)

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.

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.

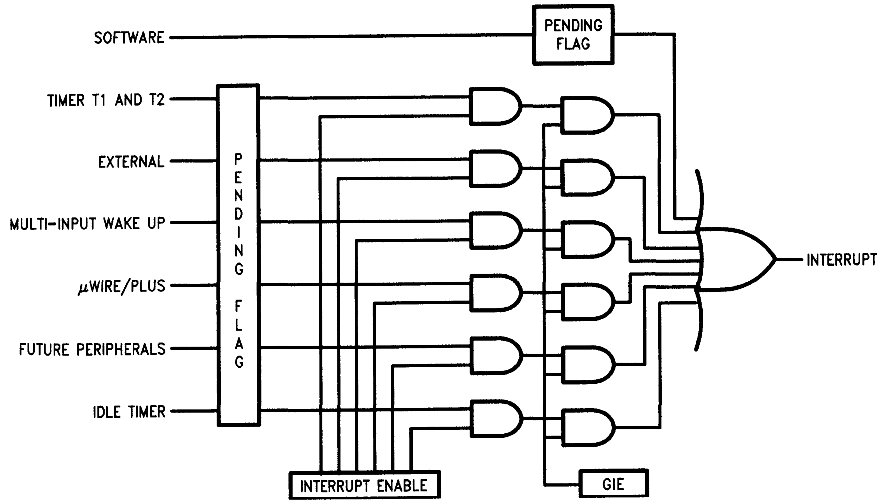
VIS and the vector table must be located in the same 256-byte block (0y00 to 0yFF) except if VIS is located at the last address of a block. In this case, the table must be in the next block. The vector table cannot be inserted in the first 256-byte block.

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

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

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.

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.

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.

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)

- 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:

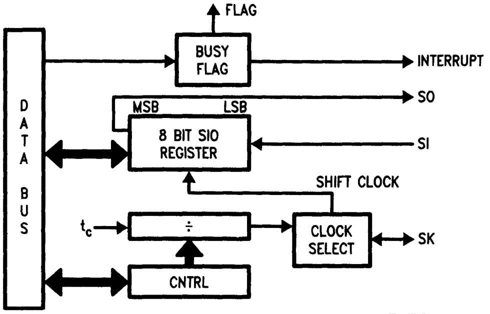
a. Executing from undefined ROM

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.

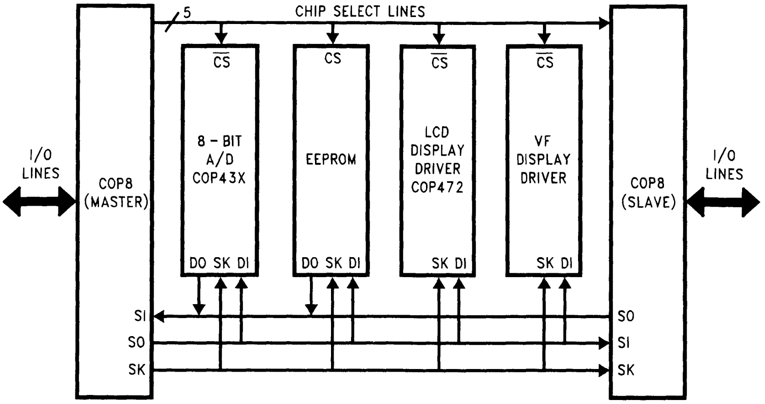
Warning:

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

Instruction Set (Continued)

INSTRUCTION SET

ADD	A,MemI	ADD	A ← A + MemI
ADC	A,MemI	ADD with Carry	A ← A + MemI + C, C ← Carry HC ← Half Carry
SUBC	A,MemI	Subtract with Carry	A ← A $\overline{\text{MemI}}$ + C, C ← Carry HC ← Half Carry
AND	A,MemI	Logical AND	A ← A and MemI
ANDSZ	A,Imm	Logical AND Immed., Skip if Zero	Skip next if (A and Imm) = 0
OR	A,MemI	Logical OR	A ← A or MemI
XOR	A,MemI	Logical EXclusive OR	A ← A xor MemI
IFEQ	MD,Imm	IF EQual	Compare MD and Imm, Do next if MD = Imm
IFEQ	A,MemI	IF EQual	Compare A and MemI, Do next if A = MemI
IFNE	A,MemI	IF Not Equal	Compare A and MemI, Do next if A ≠ MemI
IFGT	A,MemI	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	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,MemI	LoaD A with Memory	A ← MemI
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.

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/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
  = 1 44-Pin PLCC
  = 2 40-Pin DIP
  = 3 N/A
  = 4 28-Pin DIP
  = 5 28-Pin SO

Development Support

IN-CIRCUIT EMULATOR

The MetaLink iceMASTER™-COP8 Model 400 In-Circuit Emulator for the COP8 family of microcontrollers features high-performance operation, ease of use, and an extremely flexible user-interface or maximum productivity. Interchangeable probe cards, which connect to the standard common base, support the various configurations and packages of the COP8 family.

The iceMASTER provides real time, full speed emulation up to 10 MHz, 32 kBytes of emulation memory and 4k frames of trace buffer memory. The user may define as many as 32k trace and break triggers which can be enabled, disabled, set or cleared. They can be simple triggers based on code address, direct address, opcode value, opcode class or immediate operand. Complex breakpoints can be ANDed and ORed together. Trace information consists of address bus values, opcodes and user selectable probe clips status (external event lines). The trace buffer can be viewed as raw hex or as disassembled instructions. The probe clip bit values can be displayed in binary, hex or digital waveform formats.

During single-step operation the dynamically annotated code feature displays the contents of all accessed (read and write) memory locations and registers, as well as flow-of-control direction change markers next to each instruction executed.

The iceMASTER's performance analyzer offers a resolution of better than 6 μs. The user can easily monitor the time spent executing specific portions of code and find "hot spots" or "dead code". Up to 15 independent memory areas based on code address or label ranges can be defined. Analysis results can be viewed in bar graph format or as actual frequency count.

Emulator memory operations for program memory include single line assembler, disassembler, view, change and write to file. Data memory operations include fill, move, compare, dump to file, examine and modify. The contents of any memory space can be directly viewed and modified from the corresponding window.

The iceMASTER comes with an easy to use window interface. Each window can be sized, highlighted, color-controlled, added, or removed completely. Commands can be accessed via pull-down-menus and/or redefinable hot keys. A context sensitive hypertext/hyperlinked on-line help system explains clearly the options the user has from within any window.

The iceMASTER connects easily to a PC® via the standard COMM port and its 115.2 kBaud serial link keeps typical program download time to under 3 seconds.

The following tables list the emulator and probe cards ordering information.

Emulator Ordering Information

Part Number	Description	Current Version
IM-COP8/400/1‡	MetaLink base unit in-circuit emulator for all COP8 devices, symbolic debugger software and RS-232 serial interface cable, with 110V @ 60 Hz Power Supply.	HOST SOFTWARE: VER. 3.3 REV.5, Model File Rev 3.050.
IM-COP8/400/2‡	MetaLink base unit in-circuit emulator for all COP8 devices, symbolic debugger software and RS-232 serial interface cable, with 220V @ 50 Hz Power Supply.	
DM-COP8/888CF‡	MetaLink iceMASTER Debug Module. This is the low cost version of MetaLink's iceMASTER. Firmware: Ver. 6.07.	

‡ These parts include National's COP8 Assembler/Linker/Librarian Package (COP8/DEV-IBMA).

Development Support (Continued)

Probe Card Ordering Information

Part Number	Package	Voltage Range	Emulates
MHW-884CF28D5PC	28 DIP	4.5V–5.5V	COP884CF
MHW-884CF28DWPC	28 DIP	2.5V–6.0V	COP884CF
MHW-888CF40D5PC	40 DIP	4.5V–5.5V	COP888CF
MHW-888CF40DWPC	40 DIP	2.5V–6.0V	COP888CF
MWH-888CF44D5PC	44 PLCC	4.5V–5.5V	COP888CF
MHW-888CF44DWPC	44 PLCC	2.5V–6.0V	COP888CF

MACRO CROSS ASSEMBLER

National Semiconductor offers a COP8 macro cross assembler. It runs on industry standard compatible PCs and supports all of the full-symbolic debugging features of the MetaLink iceMASTER emulators.

Assembler Ordering Information

Part Number	Description	Manual
COP8-DEV-IBMA	COP8 Assembler/ Linker/Librarian for IBM®, PC/XT®, AT® or compatible.	424410632-001

SINGLE CHIP EMULATOR DEVICE

The COP8 family is fully supported by One-Time Programmable (OTP) emulators. For more detailed information refer to the emulation device specific datasheets and the emulator selection table below.

PROGRAMMING SUPPORT

Programming of the single chip emulator devices is supported by different sources. The following programmers are certified for programming the One-Time Programmable (OTP) devices.

EPROM Programmer Information

Manufacturer and Product	U.S. Phone Number	Europe Phone Number	Asia Phone Number
MetaLink-Debug Module	(602) 926-0797	Germany: +49-8141-1030	Hong Kong: +852-737-1800
Zeltek-Superpro	(408) 745-7974	Germany: +49-20-41 684758	Singapore: +65 276 6433
BP Microsystems-EP-1140	(800) 225-2102	Germany: +49-89 857 66 67	Hong Kong: +852 388 0629
Data I/O-Unisite; -System 29, -System 39	(800) 322-8246	Europe: +31-20-622866 Germany: +49-89-85-8020	Japan: +33-432-6991
Abcom-COP8 Programmer		Europe: + 89-80 8707	
System General Turpro-1-FX; -APRO	(408) 263-6667	Switzerland: +31-921-7844	Taiwan Taipei: +2-9173005

OTP Emulator Ordering Information

Device Number	Clock Option	Package	Emulates
COP8788CFV-X COP8788CFV-R*	Crystal R/C	44 LDCC	COP888CF
COP8788CFN-X COP8788CFN-R*	Crystal R/C	40 DIP	COP888CF
COP8784CFN-X COP8784CFN-R*	Crystal R/C	28 DIP	COP884CF
COP8784CFWM-X* COP8784CFWM-R*	Crystal R/C	28 SO	COP884CF

*Check with the local sales office about the availability.

Development Support (Continued)

DIAL-A-HELPER

Dial-A-Helper is a service provided by the Microcontroller Applications group. The Dial-A-Helper is an Electronic Bulletin Board Information system.

INFORMATION SYSTEM

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 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. The minimum requirement for accessing the Dial-A-Helper is a Hayes compatible modem.

If the user has a PC with a communications package then files from the FILE SECTION can be down loaded to disk for later use.

ORDER P/N: MOLE-DIAL-A-HLP

Information System Package Contents:
- Dial-A-Helper Users Manual
- Public Domain Communications Software

FACTORY APPLICATIONS SUPPORT

Dial-A-Helper also provides immediate factor applications support. If a user has questions, he can leave messages on our electronic bulletin board, which we will respond to.

Voice: (800) 272-9959
Modem: Canada/
  U.S.: (800) NSC-MICRO
  (800) 672-6427
  Baud: 14.4k
  Set-Up: Length: 8-Bit
  Parity: None
  Stop Bit: 1
  Operation: 24 Hours, 7 Days

National Semiconductor

COP688CS/COP684CS/COP888CS/COP884CS/COP988CS/COP984CS Single-Chip microCMOS Microcontroller

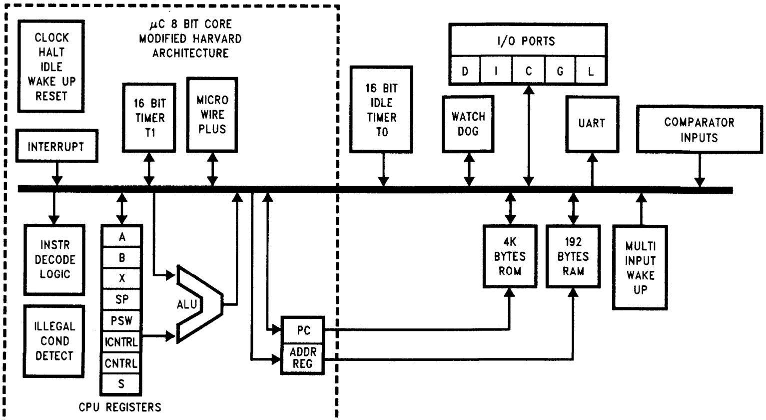
General Description

The COP888 family of microcontrollers uses an 8-bit single chip core architecture fabricated with National Semiconductor's M2CMOS™ process technology. The COP888CS is a member of this expandable 8-bit core processor family of microcontrollers. (Continued)

Features

- Low cost 8-bit microcontroller
- Fully static CMOS, with low current drain
- Two power saving modes: HALT and IDLE
- 1 μs instruction cycle time
- 4096 bytes on-board ROM
- 192 bytes on-board RAM
- Single supply operation: 2.5V–6V
- Full duplex UART
- One analog comparator
- MICROWIRE/PLUS™ serial I/O
- WATCHDOG™ and Clock Monitor logic
- Idle Timer
- Multi-Input Wakeup (MIWU) with optional interrupts (8)
- One 16-bit timer, with two 16-bit registers supporting:
  - Processor Independent PWM mode
  - External Event counter mode
  - Input Capture mode
- 8-bit Stack Pointer SP (stack in RAM)
- Two 8-bit Register Indirect Data Memory Pointers (B and X)
- Ten multi-source vectored interrupts servicing
  - External Interrupt
  - Idle Timer T0
  - Timer (2)
  - MICROWIRE/PLUS
  - Multi-Input Wake Up
  - Software Trap
  - UART (2)
  - Default VIS
- Versatile instruction set
- True bit manipulation
- Memory mapped I/O
- BCD arithmetic instructions
- Package:
  - 44 PLCC with 39 I/O pins
  - 40 N with 35 I/O pins
  - 28 SO or 28 N, each with 23 I/O pins
- 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
- One-Time Programmable (OTP) emulation devices
- Real time emulation and full program debug offered by MetaLink's Development Systems

Block Diagram



TL/DD/10830–1

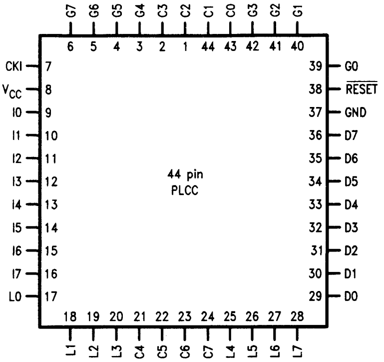
FIGURE 1. Block Diagram

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, 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

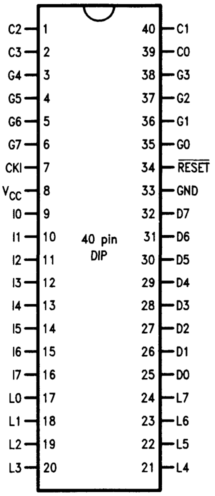


TL/DD/10830-2

Top View

Order Number COP888CS-XXX/V
See NS Package Number V44A

Dual-In-Line Package

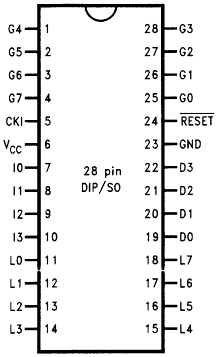


TL/DD/10830-3

Top View

Order Number COP888S-XXX/N
See NS Package Number N40A

Dual-In-Line Package



TL/DD/10830-5

Top View

Order Number COP884CS-XXX/N
See NS Package Number N28B

Order Number COP884CS-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	$V_{CC} = 6V$, $t_c = 1\ \mu s$			12.5	mA
CKI = 4 MHz	$V_{CC} = 6V$, $t_c = 2.5\ \mu s$			5.5	mA
CKI = 4 MHz	$V_{CC} = 4V$, $t_c = 2.5\ \mu s$			2.5	mA
CKI = 1 MHz	$V_{CC} = 4V$, $t_c = 10\ \mu s$			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	$V_{CC} = 6V$, $t_c = 1\ \mu s$			3.5	mA
CKI = 4 MHz	$V_{CC} = 6V$, $t_c = 2.5\ \mu s$			2.5	mA
CKI = 1 MHz	$V_{CC} = 4V$, $t_c = 10\ \mu s$			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 ≤ 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 98XCS: 0°C ≤ T_A ≤ +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.

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	$V_{CC} = 6V, t_c = 1\ \mu s$			12.5	mA
CKI = 4 MHz	$V_{CC} = 6V, t_c = 2.5\ \mu s$			5.5	mA
HALT Current (Note 3)	$V_{CC} = 6V$, CKI = 0 MHz		<1	10	μA
IDLE Current					
CKI = 10 MHz	$V_{CC} = 6V, t_c = 1\ \mu s$			3.5	mA
CKI = 4 MHz	$V_{CC} = 6V, t_c = 2.5\ \mu s$			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: $-40^{\circ}C \leq T_A \leq +85^{\circ}C$ 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)	$T_A = 25^{\circ}C$			±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: $-40^{\circ}C \leq T_A \leq +85^{\circ}C$ unless otherwise specified

Parameter	Conditions	Min	Typ	Max	Units
Instruction Cycle Time (t_c)	$4V \leq V_{CC} \leq 6V$	1		DC	μs
Crystal, Resonator,	$2.5V \leq V_{CC} < 4V$	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 6)	$R_L = 2.2k, C_L = 100$ 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 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 68XCS: $-55°C \le T_A \le +125°C$ 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					
$\overline{\text{RESET}}$					
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: $-55^\circ C \le T_A \le +125^\circ C$ 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)	$T_A = 25^\circ C$			±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: $-55^\circ C \le T_A \le +125^\circ C$ unless otherwise specified

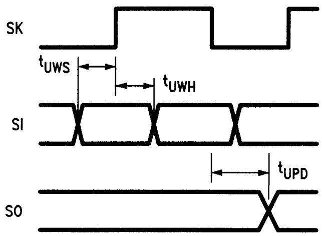
Parameter	Conditions	Min	Typ	Max	Units
Instruction Cycle Time (t_c)	$4.5V \le V_{CC} \le 5.5V$	1		DC	μs
Crystal, Resonator, R/C Oscillator	$4.5V \le V_{CC} \le 5.5V$	3		DC	μs
Inputs					
t_{SETUP}	$4.5V \le V_{CC} \le 5.5V$	200			ns
t_{HOLD}	$4.5V \le V_{CC} \le 5.5V$	60			ns
Output Propagation Delay (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	$0.4V \leq V_{IN} \leq V_{CC} - 1.5V$		±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 (When Enabled)				250	μA
Response Time	TBD mV Step, TBD mV Overdrive, 100 pF Load		1		μs

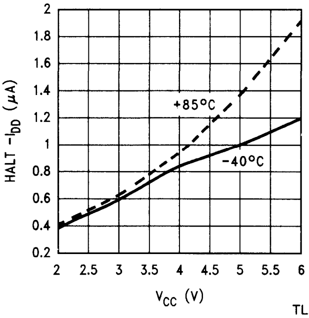


TL/DD/10830-6

FIGURE 3. MICROWIRE/PLUS Timing

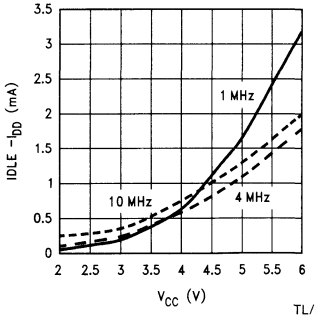
Typical Performance Characteristics (−40°C to +85°C)

Halt—I_{DD}



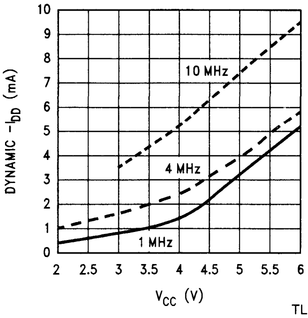
TL/DD/10830-24

Idle—I_{DD} (Crystal Clock Option)



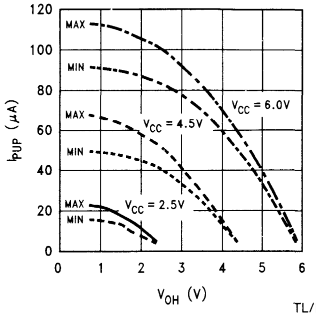
TL/DD/10830-25

Dynamic—I_{DD} (Crystal Clock Option)



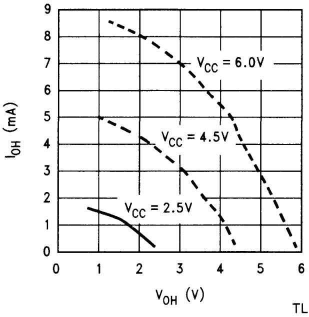
TL/DD/10830-26

Port L/C/G Weak Pull-Up Source Current



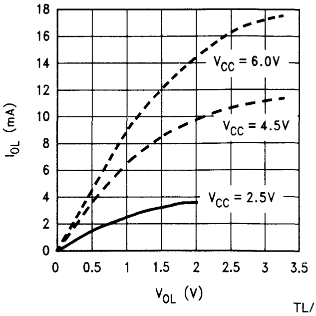
TL/DD/10830-27

Port L/C/G Push-Pull Source Current



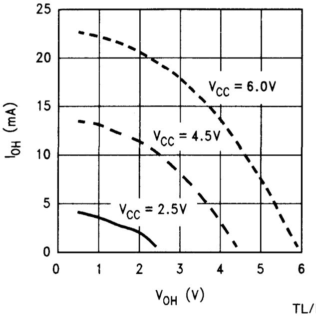
TL/DD/10830-28

Port L/C/G/ Push-Pull Sink Current



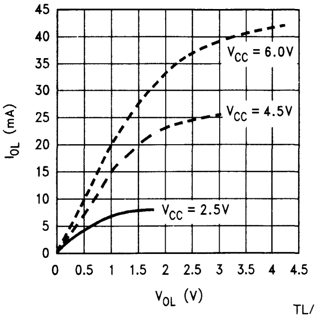
TL/DD/10830-29

Port D Source Current



TL/DD/10830-30

Port D Sink Current



TL/DD/10830-31

Pin Descriptions

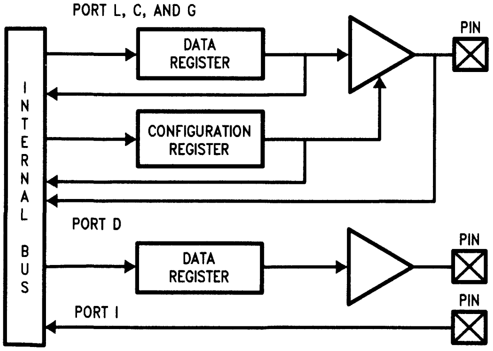
V_{CC} and GND are the power supply pins.

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

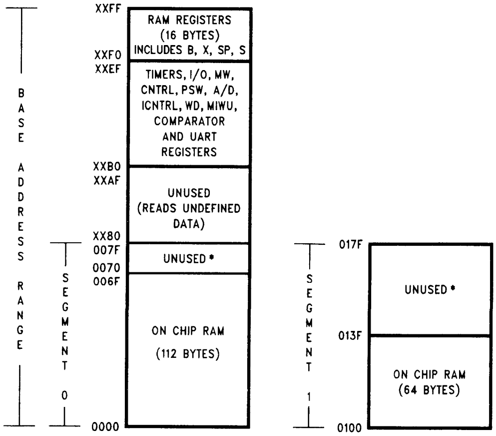
Data Memory Segment RAM Extension (Continued)

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 intitialized 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.



TL/DD/10830-8

*Reads as all ones.

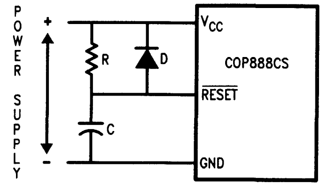
FIGURE 5. RAM Organization

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 0F 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/10830-9

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

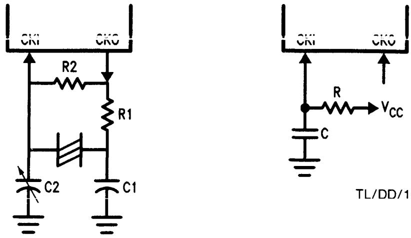
CKI and CKO can be connected to make a closed loop crystal (or resonator) controlled oscillator.

Table A shows the component values required for various standard crystal values.

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

Oscillator Circuits (Continued)

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: $3k \le R \le 200k$

$50 pF \le C \le 200 pF$

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

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 ($t_C = 1\ \mu s$). 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.

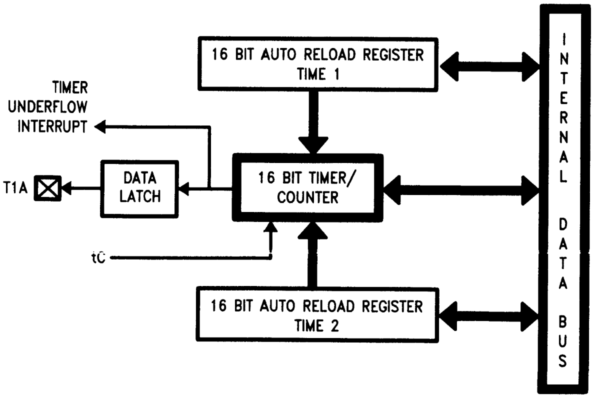
The user only has to define the parameters of the PWM signal (ON time and OFF time). Once begun, the timer block will continuously generate the PWM signal completely independent of the microcontroller. The user software services the timer block only when the PWM parameters require updating.

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.

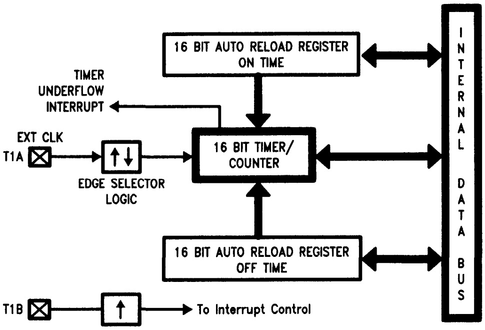
Underflows from the timer can also be programmed to generate interrupts. Underflows are latched into the timer T1C0 pending flag (the T1C0 control bit serves as the timer underflow interrupt pending flag in the Input Capture mode). Consequently, the T1C0 control bit should be reset when entering the 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

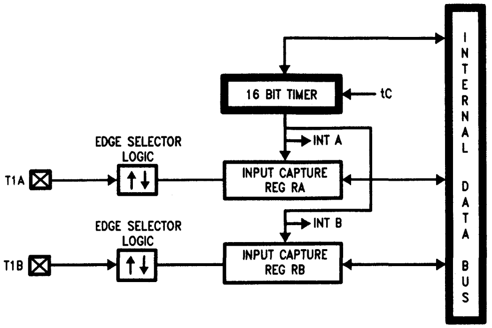
The control bits and their functions are summarized below.

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.

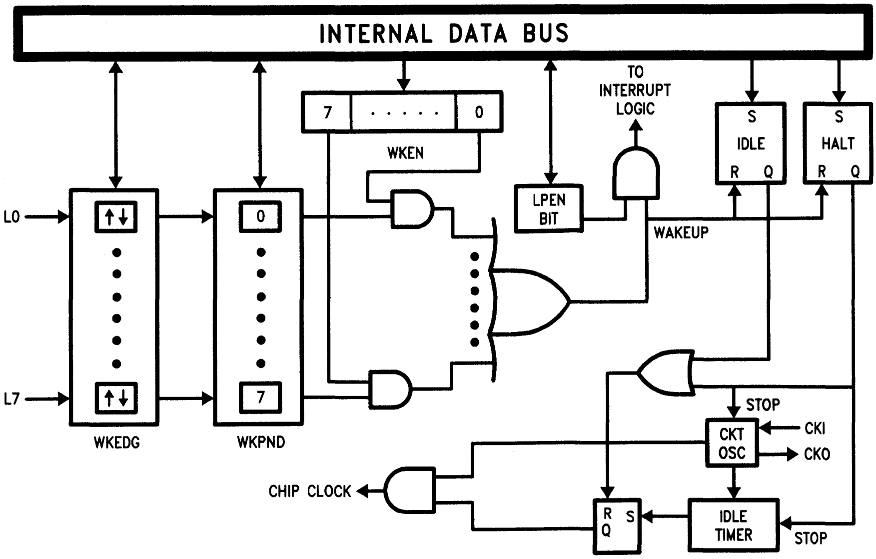
Alternatively, the user can enter the IDLE mode with the IDLE Timer T0 interrupt disabled. In this case, the device will resume normal operation with the instruction immediately following the "Enter IDLE Mode" instruction.

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.

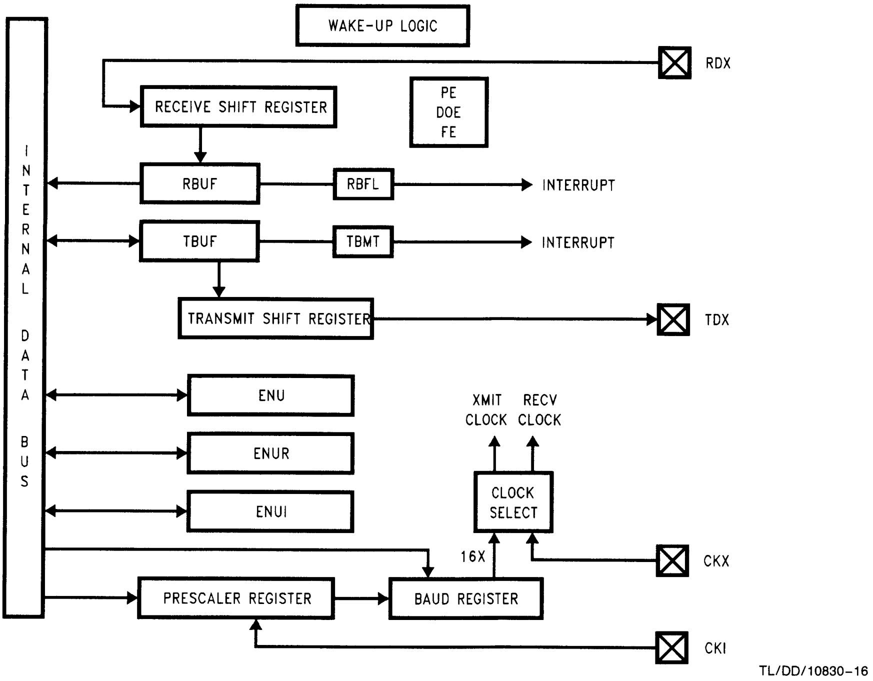


FIGURE 12. UART Block Diagram

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.

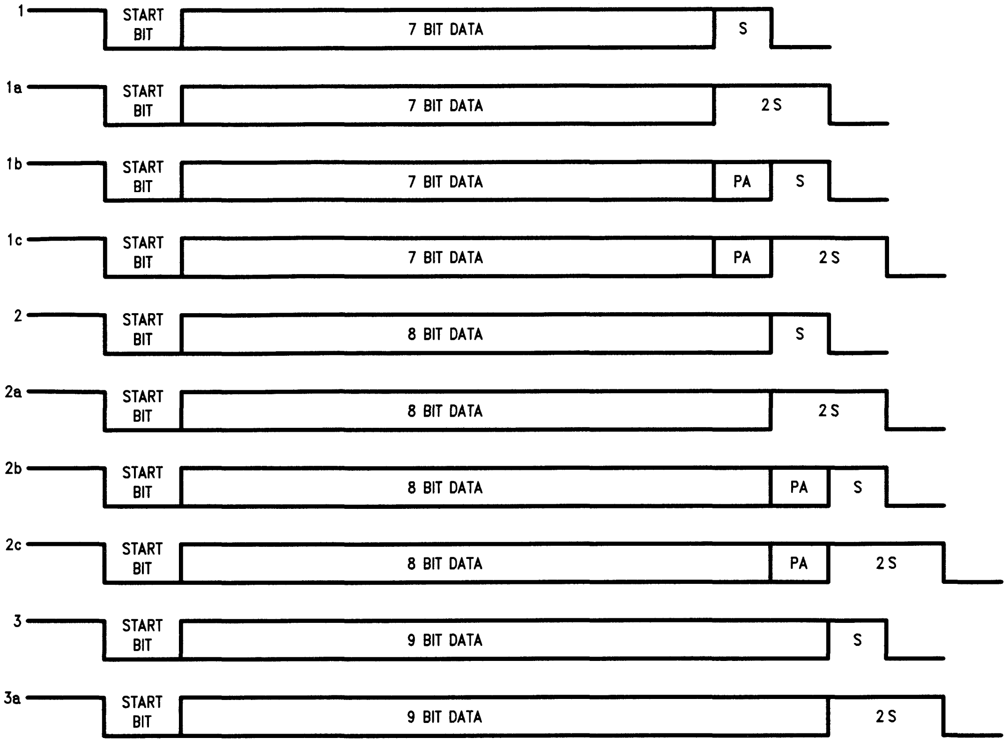
For any of the above framing formats, the last Stop bit can be programmed to be 7/8th of a bit in length. If two Stop bits are selected and the 7/8th bit is set (selected), the second Stop bit will be 7/8th of a bit in length.

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).

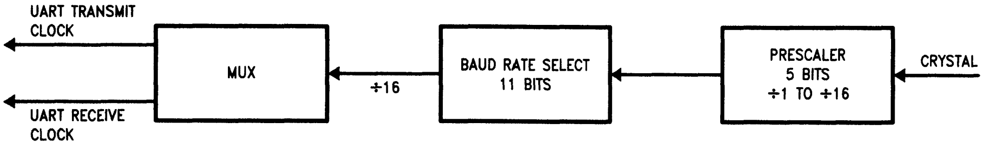
Baud Clock Generation

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.

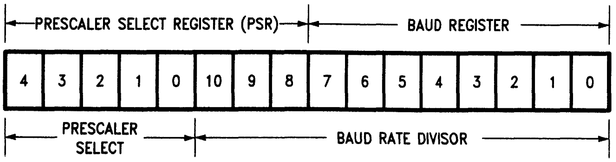
The correspondences between the 5-bit Prescaler Select and Prescaler factors are shown in Table I. Therer are many ways to calculate the two divisor factors, but one particularly effective method would be to achieve a 1.8432 MHz frequency coming out of the first stage. The 1.8432 MHz prescaler output 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:

$$BR = (5 \times 10^6)/(16 \times 5 \times 6.5) = 9615.384$$

$$\%\ error = (9615.385 - 9600)/9600 = 0.16$$

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)
CMP1OE	Selects pin I3 as comparator 1 output provided that CMPIEN is set to enable the comparator

Unused	Unused	Unused	Unused	CMP1OE	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.

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.

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.

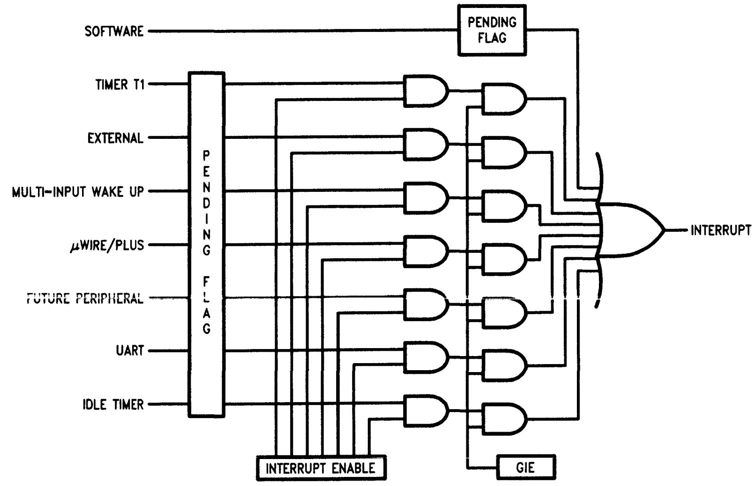
WATCHDOG

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.



TL/DD/10830-20

FIGURE 16. Interrupt Block Diagram

WATCHDOG (Continued)

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:

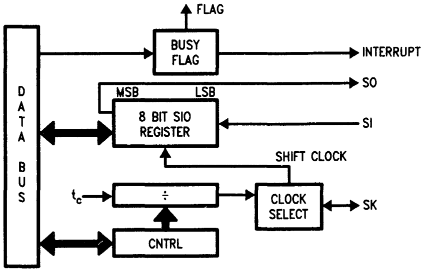
a. Executing from undefined ROM

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^{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/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

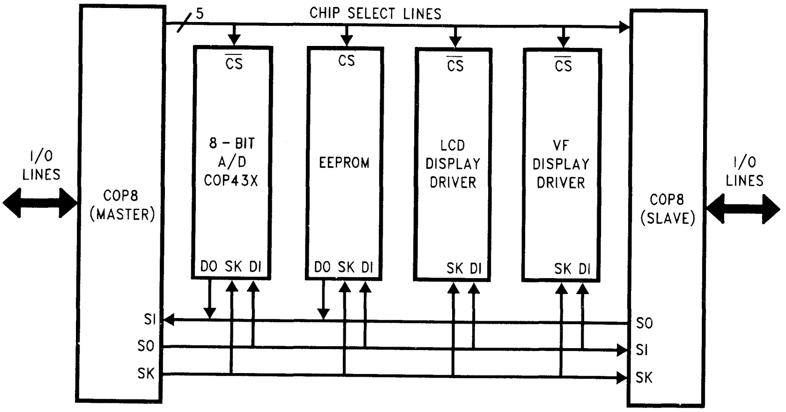
The device allows either the normal SK clock or an alternate phase SK clock to shift data in and out of the SIO register. In both the modes the SK is normally low. In the normal mode data is shifted in on the rising edge of the SK clock and the data is shifted out on the falling edge of the SK clock. 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]
MemI	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,MemI	ADD	A ← A + MemI
ADC	A,MemI	ADD with Carry	A ← A + MemI + C, C ← Carry HC ← Half Carry
SUBC	A,MemI	Subtract with Carry	A ← A − $\overline{\text{MemI}}$ + C, C ← Carry HC ← Half Carry
AND	A,MemI	Logical AND	A ← A and MemI
ANDSZ	A,Imm	Logical AND Immed., Skip if Zero	Skip next if (A and Imm) = 0
OR	A,MemI	Logical OR	A ← A or MemI
XOR	A,MemI	Logical EXclusive OR	A ← A xor MemI
IFEQ	MD,Imm	IF EQual	Compare MD and Imm, Do next if MD = Imm
IFEQ	A,MemI	IF EQual	Compare A and MemI, Do next if A = MemI
IFNE	A,MemI	IF Not Equal	Compare A and MemI, Do next if A ≠ MemI
IFGT	A,MemI	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	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,MemI	LoaD A with Memory	A ← MemI
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.

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/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 NA
= 4 28-Pin DIP
= 5 28-Pin SO

Development Support

IN-CIRCUIT EMULATOR

The MetaLink iceMASTER™-COP8 Model 400 In-Circuit Emulator for the COP8 family of microcontrollers features high-performance operation, ease of use, and an extremely flexible user-interface for maximum productivity. Interchangeable probe cards, which connect to the standard common base, support the various configurations and packages of the COP8 family.

The iceMASTER provides real time, full speed emulation up to 10 MHz, 32 kBytes of emulation memory and 4k frames of trace buffer memory. The user may define as many as 32k trace and break triggers which can be enabled, disabled, set or cleared. They can be simple triggers based on code or address ranges or complex triggers based on code address, direct address, opcode value, opcode class or immediate operand. Complex breakpoints can be ANDed and ORed together. Trace information consists of address bus values, opcodes and user selectable probe clips status (external event lines). The trace buffer can be viewed as raw hex or as diassembled instructions. The probe clip bit values can be displayed in binary, hex or digital waveform formats.

During single-step operation the dynamically annotated code feature displays the contents of all accessed (read and write) memory locations and registers, as well as flow-of-control direction change markers next to each instruction executed.

The iceMASTER's performance analyzer offers a resolution of better than 6 μs. The user can easily monitor the time spent executing specific portions of code and find "hot spots" or "dead code". Up to 15 independent memory areas based on code address or label ranges can be defined. Analysis results can be viewed in bar graph format or as actual frequency count.

Emulator memory operations for program memory include single line assembler, disassembler, view, change and write to file. Data memory operations include fill, move, compare, dump to file, examine and modify. The contents of any memory space can be directly viewed and modified from the corresponding window.

The iceMASTER comes with an easy to use windowed interface. Each window can be sized, highlighted, color-controlled, added, or removed completely. Commands can be accessed via pull-down-menus and/or redefineable hot keys. A context sensitive hypertext/hyperlinked on-line help system explains clearly the options the user has from within any window.

The iceMASTER connects easily to a PC® via the standard COMM port and its 115.2 kBaud serial link keeps typical program download time to under 3 seconds.

The following tables list the emulator and probe cards ordering information.

Emulator Ordering Information

Part Number	Description	Current Version
IM-COP8/400/1‡	MetaLink base unit in-circuit emulator for all COP8 devices, symbolic debugger software and RS-232 serial interface cable, with 110V @ 60 Hz Power Supply.	Host Software: Ver. 3.3 Rev. 5, Model File Rev. 3.050.
IM-COP8/400/2‡	MetaLink base unit in-circuit emulator for all COP8 devices, symbolic debugger software and RS-232 serial interface cable, with 220V @ 50 Hz Power Supply.	
DM-COP8/888EG‡	MetaLink iceMASTER Debug Module. This is the low cost version of MetaLink's iceMASTER. Firmware: Ver. 6.07.	

‡These parts include National's COP8 Assembler/Linker/Librarian Package (COP8-DEV-IBMA).

Development Support (Continued)

MACRO CROSS ASSEMBLER

National Semiconductor offers a COP8 macro cross assembler. It runs on industry standard compatible PCs and supports all of the full-symbolic debugging features of the MetaLink iceMASTER emulators.

SINGLE CHIP EMULATOR DEVICE

The COP8 family is fully supported by One-Time Programmable (OTP) emulators. For more detailed information refer to the emulation device specific data sheets and emulator selection table below. (The COP8788EG/COP8784EG can be used to emulate the COP888CS/COP884CS.)

PROGRAMMING SUPPORT

Programming of the single chip emulator devices is supported by different sources.

Probe Card Ordering Information

Part Number	Package	Voltage Range	Emulates
MHW-884CG28D5PC	28 DIP	4.5V–5.5V	COP884CS
MHW-884CG28DWPC	28 DIP	2.5V–6.0V	COP884CS
MHW-888CG40D5PC	40 DIP	4.5V–5.5V	COP888CS
MHW-888CG40DWPC	40 DIP	2.5V–6.0V	COP888CS
MHW-888CG44D5PC	44 PLCC	4.5V–5.5V	COP888CS
MHW-888CG44DWPC	44 PLCC	2.5V–6.0V	COP888CS

EPROM Programmer Information

Manufacturer and Product	U.S. Phone Number	Europe Phone Number	Asia Phone Number
MetaLink-Debug Module	(602) 926-0797	Germany: +49-8141-1030	Hong Kong: +852-737-1800
Xeltek-Superpro	(408) 745-7974	Germany: +49-2041 684758	Singapore: +65 276 6433
BP Microsystems-EP-1140	(800) 225-2102	Germany: +49 89 857 66 67	Hong Kong: +852 388 0629
Data I/O-Unisite; -System 29, -System 39	(800) 322-8246	Europe: +31-20-622866 Germany: +49-89-85-8020	Japan: +33-432-6991
Abcom-COP8 Programmer		Europe: +89 808707	
System General Turpro-1-FX; -APRO	(408) 263-6667	Switzerland: +31-921-7844	Taiwan Taipei: +2-9173005

Assembler Ordering Information

Part Number	Description	Manual
COP8-DEV-IBMA	COP8 Assembler/Linker/Librarian for IBM® PC/XT®, AT® or compatible	424410632-001

Single Chip Emulator Selection Table

Device Number	Clock Option	Package	Emulates
COP87898EGV-X COP8788EGV-R*	Crystal R/C	44 PLCC	COP888CS
COP8788EGN-X COP8788EGN-R*	Crystal R/C	40 DIP	COP888CS
COP8784EGN-X COP8784EGN-R*	Crystal R/C	28 DIP	COP884CS
COP8784EGWM-X* COP8784EGWM-R*	Crystal R/C	28 SO	COP884CS

*Check with the local sales office about the availability.

Development Support (Continued)

DIAL-A-HELPER

Dial-A-Helper is a service provided by the Microcontroller Applications group. The Dial-A-Helper is an Electronic Bulletin Board Information system.

INFORMATION SYSTEM

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 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. The minimum requirement for accessing the Dial-A-Helper is a Hayes compatible modem.

If the user has a PC with a communications package then files from the FILE SECTION can be down loaded to disk for later use.

ORDER P/N: MOLE-DIAL-A-HLP

Information System Package contains:
- Dial-A-Helper Users Manual
- Public Domain Communications Software

FACTORY APPLICATIONS SUPPORT

Dial-A-Helper also provides immediate factor applications support. If a user has questions, he can leave messages on our electronic bulletin board, which we will respond to.

Voice: (800) 272-9959
Modem: Canada/U.S.
(800) NSC-Micro:
(800) 672-6427
Baud: 14.4k
Set-up: Length: 8-Bit
Parity: None
Stop Bit: 1
Operation: 24 Hrs., 7 Days

National Semiconductor

COP884CG/COP888CG Single-Chip microCMOS Microcontrollers

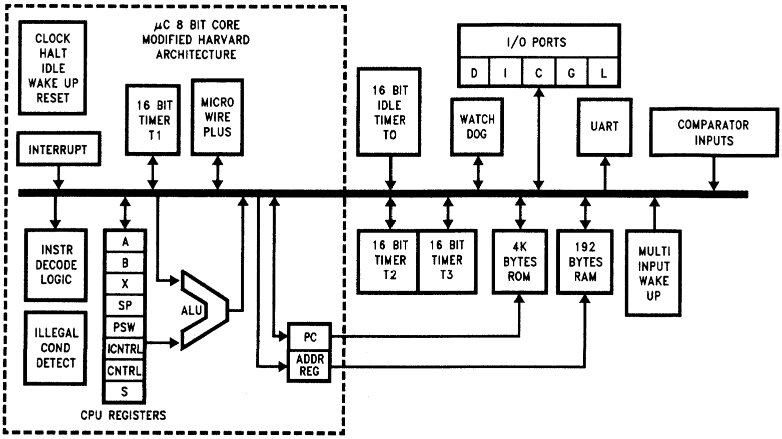
General Description

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)

Features

- Low cost 8-bit microcontroller
- Fully static CMOS, with low current drain
- Two power saving modes: HALT and IDLE
- 1 µs instruction cycle time
- 4096 bytes on-board ROM
- 192 bytes on-board RAM
- Single supply operation: 2.5V–6V
- Full duplex UART
- Two analog comparators
- MICROWIRE/PLUS™ serial I/O
- WATCHDOG™ and Clock Monitor logic
- Idle Timer
- Multi-Input Wakeup (MIWU) with optional interrupts (8)
- Three 16-bit timers, each with two 16-bit registers supporting:
  - — Processor Independent PWM mode
  - — External Event counter mode
  - — Input Capture mode
- 8-bit Stack Pointer SP (stack in RAM)
- Two 8-bit Register Indirect Data Memory Pointers (B and X)
- 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
- Versatile instruction set
- True bit manipulation
- Memory mapped I/O
- BCD arithmetic instructions
- Package:
  - — 44 PLCC with 39 I/O pins
  - — 40 N with 35 I/O pins
  - — 28 N with 23 I/O pins
  - — 28 SO with 23 I/O pins
- 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
- Temperature ranges: −40°C to +85°C
- One-Time Programmable emulation devices
- Real time emulation and full program debug offered by MetaLink's Development Systems

Block Diagram



TL/DD/9765-1

FIGURE 1. Block Diagram

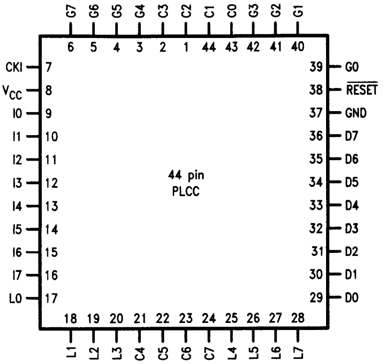
General Description (Continued)

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

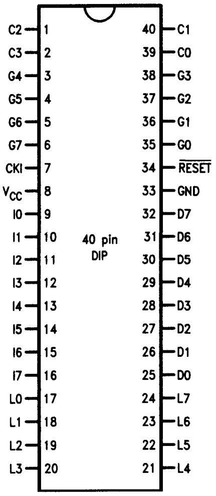


TL/DD/9765-2

Top View

Order Number COP888CG-XXX/V
See NS Plastic Chip Package Number V44A

Dual-In-Line Package

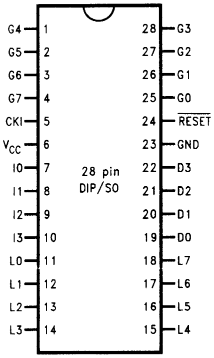


TL/DD/9765-4

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 $-40°C \leq T_A \leq +85°C$ 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	$V_{CC} = 6V, t_c = 1\ \mu s$			8.0	mA
CKI = 4 MHz	$V_{CC} = 6V, t_c = 2.5\ \mu s$			4.5	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)	$V_{CC} = 6V$, CKI = 0 MHz		<1	10	μA
	$V_{CC} = 4.0V$, CKI = 0 MHz		<0.5	6	μA
IDLE Current					
CKI = 10 MHz	$V_{CC} = 6V, t_c = 1\ \mu s$			3.5	mA
CKI = 4 MHz	$V_{CC} = 6V, t_c = 2.5\ \mu s$			2.5	mA
CKI = 1 MHz	$V_{CC} = 4.0V, t_c = 10\ \mu s$			0.7	mA
Input Levels					
$\overline{\text{RESET}}$					
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 $-40^{\circ}C \leq T_A \leq +85^{\circ}C$ 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	$T_A = 25^{\circ}C$			±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

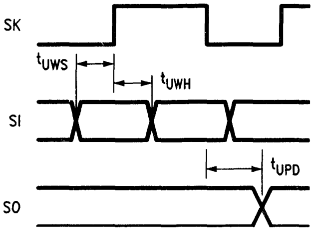
AC Electrical Characteristics $-40^{\circ}C \leq T_A \leq +85^{\circ}C$ unless otherwise specified

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 4)	$R_L = 2.2k, C_L = 100$ 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: 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	$0.4V \leq V_{IN} \leq V_{CC} - 1.5V$		±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	TBD mV Step, TBD mV Overdrive, 100 pF Load		1		μs



TL/DD/9765-7

FIGURE 2. MICROWIRE/PLUS Timing

Pin Descriptions

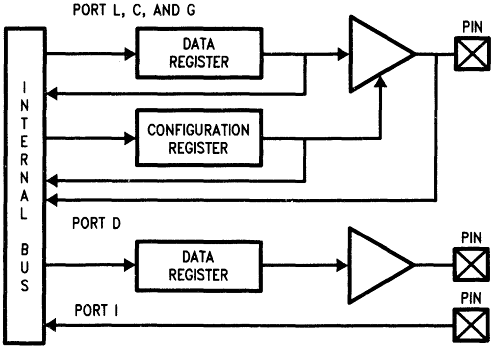
V_{CC} and GND are the power supply pins.

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).

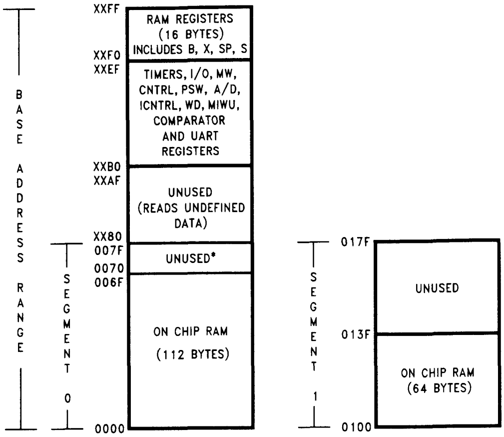
The data store memory is either addressed directly by a single byte address within the instruction, or indirectly relative to the reference of the B, X, or SP pointers (each contains a single-byte address). This single-byte address allows an addressing range of 256 locations from 00 to FF hex. The upper 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 intitialized 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

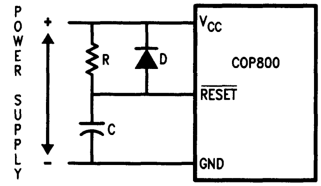
Reset

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

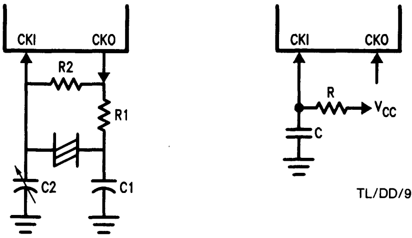
CKI and CKO can be connected to make a closed loop crystal (or resonator) controlled oscillator.

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: $3k \leq R \leq 200k$

$50\text{ pF} \leq C \leq 200\text{ pF}$

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

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.

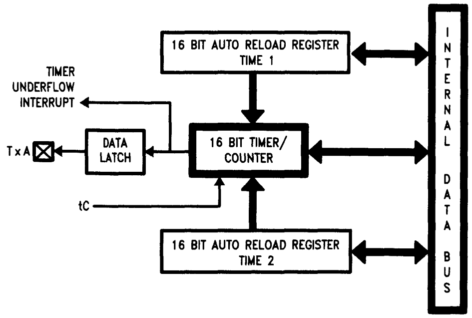
The Tx Timer control bits, TxC3, TxC2 and TxC1 set up the timer for PWM mode operation.

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

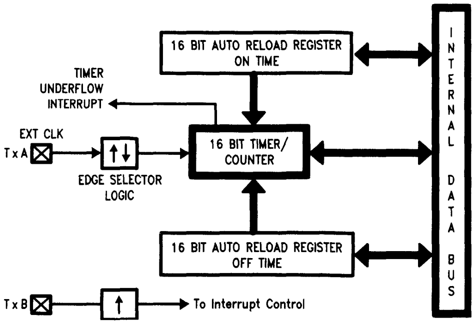
Mode 2. External Event Counter Mode

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.

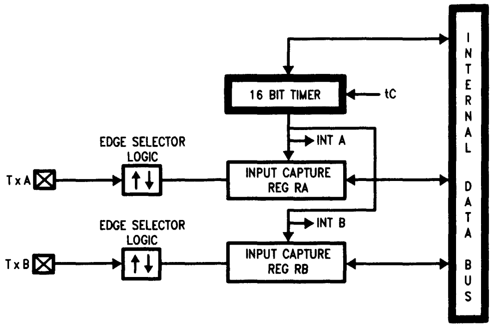
Timers (Continued)

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.

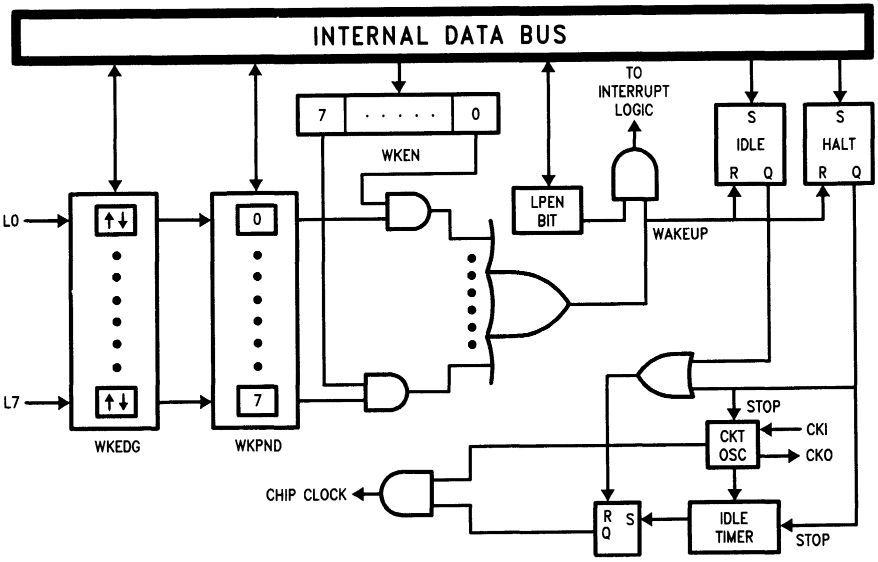
Alternatively, the user can enter the IDLE mode with the IDLE Timer T0 interrupt disabled. In this case, the device will resume normal operation with the instruction immediately following the "Enter IDLE Mode" instruction.

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.

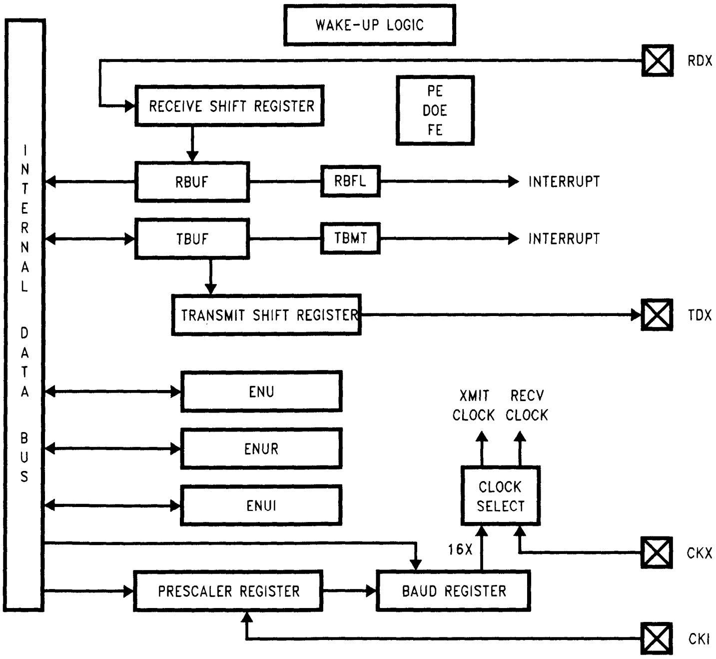
The Wakeup signal will not start the chip running immediately since crystal oscillators or ceramic resonators have a finite start up time. The IDLE Timer (T0) generates a fixed delay to ensure that the oscillator has indeed stabilized before allowing the device to execute instructions. In this case, 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.



TL/DD/9765-18

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.

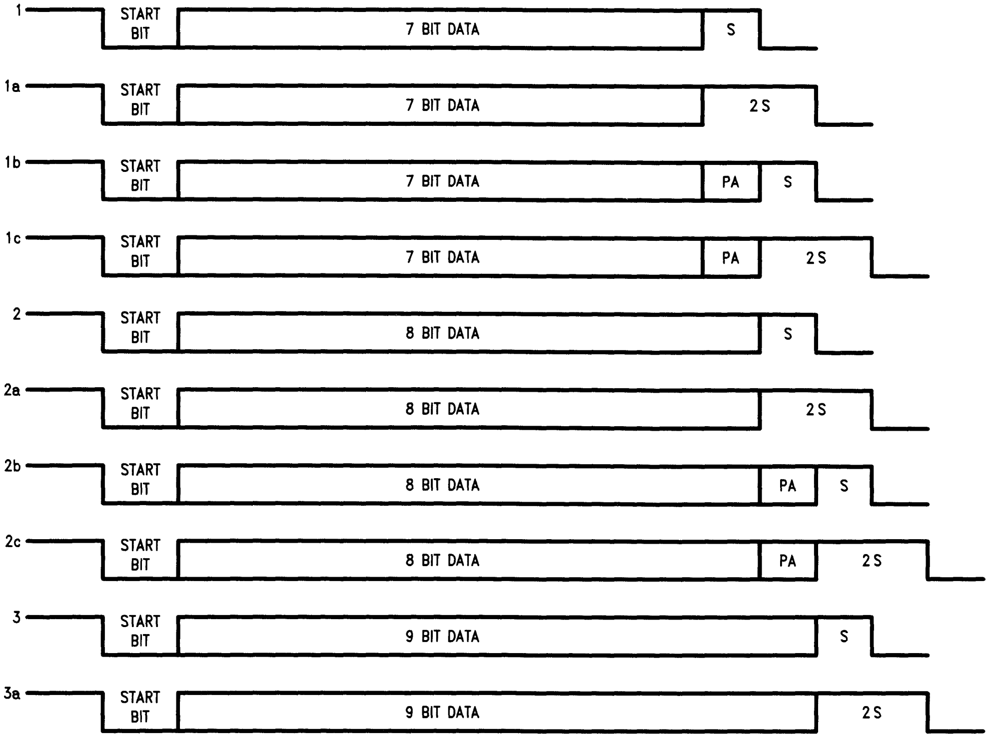
For any of the above framing formats, the last Stop bit can be programmed to be 7/8th of a bit in length. If two Stop bits are selected and the 7/8th bit is set (selected), the second Stop bit will be 7/8th of a bit in length.

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).

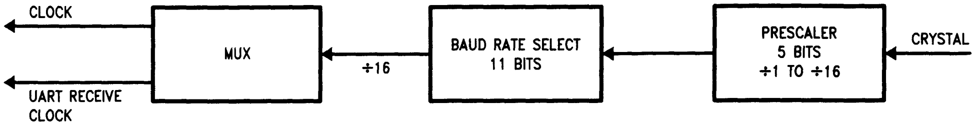
Baud Clock Generation

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.

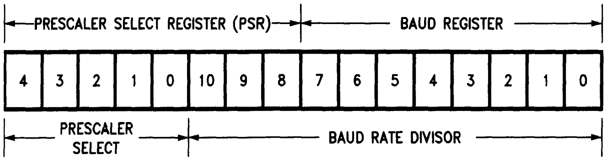
The correspondences between the 5-bit Prescaler Select and Prescaler factors are shown in Table I. Therer are many ways to calculate the two divisor factors, but one particularly effective method would be to achieve a 1.8432 MHz frequency coming out of the first stage. The 1.8432 MHz prescaler output 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)

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 zero.)

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 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)
CMP1OE	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)
CMP2OE	Selects pin I6 as comparator 2 output provided that CMP2EN is set to enable the comparator

Unused	CMP2OE	CMP2RD	CMP2EN	CMP1OE	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.

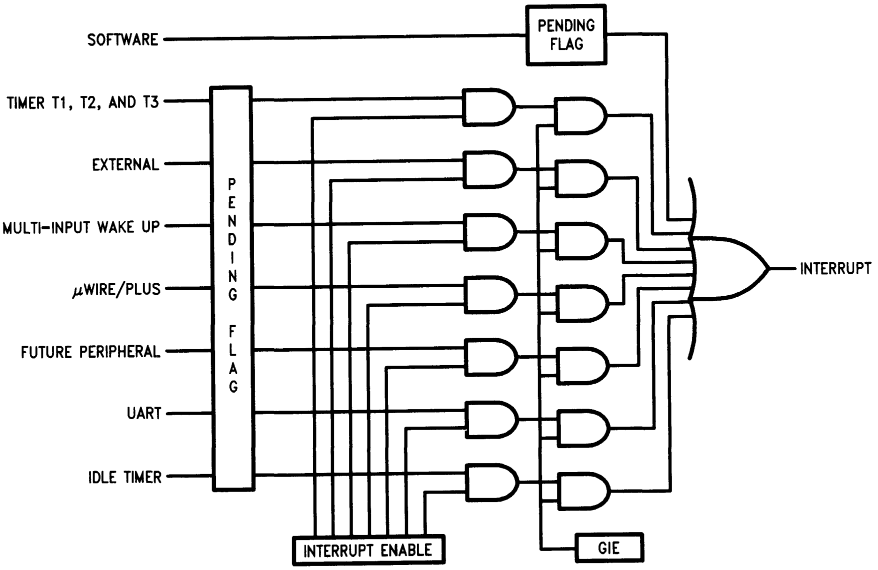
Interrupts

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. This vector can point to the Software Trap (ST) interrupt service routine, or to another special service routine as desired.

Figure 15 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.

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 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:

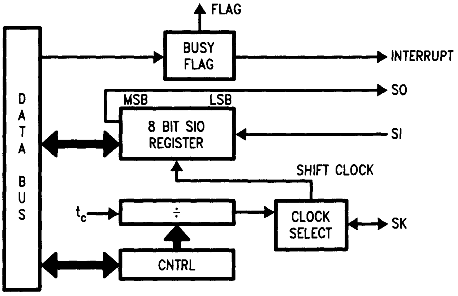
a. Executing from undefined ROM

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^{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/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

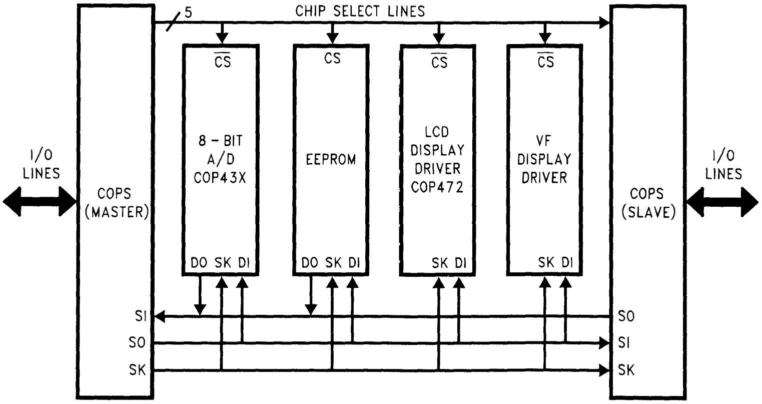
The device allows either the normal SK clock or an alternate phase SK clock to shift data in and out of the SIO register. In both the modes the SK is normally low. In the normal mode data is shifted in on the rising edge of the SK clock and the data is shifted out on the falling edge of the SK clock. 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]
MemI	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.

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/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

IN-CIRCUIT EMULATOR

The MetaLink iceMASTER™-COP8 Model 400 In-Circuit Emulator for the COP8 family of microcontrollers features high-performance operation, ease of use, and an extremely flexible user-interface or maximum productivity. Interchangeable probe cards, which connect to the standard common base, support the various configurations and packages of the COP8 family.

The iceMASTER provides real time, full speed emulation up to 10 MHz, 32 kBytes of emulation memory and 4k frames of trace buffer memory. The user may define as many as 32k trace and break triggers which can be enabled, disabled, set or cleared. They can be simple triggers based on code or address ranges or complex triggers based on code address, direct address, opcode value, opcode class or immediate operand. Complex breakpoints can be ANDed and ORed together. Trace information consists of address bus values, opcodes and user selectable probe clips status (external event lines). The trace buffer can be viewed as raw hex or as disassembled instructions. The probe clip bit values can be displayed in binary, hex or digital waveform formats.

During single-step operation the dynamically annotated code feature displays the contents of all accessed (read and write) memory locations and registers, as well as flow-of-control direction change markers next to each instruction executed.

The iceMASTER's performance analyzer offers a resolution of better than 6 μs. The user can easily monitor the time spent executing specific portions of code and find "hot spots" or "dead code". Up to 15 independent memory areas based on code address or label ranges can be defined. Analysis results can be viewed in bar graph format or as actual frequency count.

Emulator memory operations for program memory include single line assembler, disassembler, view, change and write to file. Data memory operations include fill, move, compare, dump to file, examine and modify. The contents of any memory space can be directly viewed and modified from the corresponding window.

The iceMASTER comes with an easy to use window interface. Each window can be sized, highlighted, color-controlled, added, or removed completely. Commands can be accessed via pull-down-menus and/or redefinable hot keys. A context sensitive hypertext/hyperlinked on-line help system explains clearly the options the user has from within any window.

The iceMASTER connects easily to a PC® via the standard COMM port and its 115.2 kBaud serial link keeps typical program download time to under 3 seconds.

The following tables list the emulator and probe cards ordering information.

Probe Card Ordering Information

Part Number	Package	Voltage Range	Emulates
MHW-884CG28D5PC	28 DIP	4.5V–5.5V	COP884CG
MHW-884CG28DWPC	28 DIP	2.5V–6.0V	COP884CG
MHW-888CG40D5PC	40 DIP	4.5V–5.5V	COP888CG
MHW-888CG40DWPC	40 DIP	2.5V–6.0V	COP888CG
MWH-888CG44D5PC	44 PLCC	4.5V–5.5V	COP888CG
MHW-888CG44DWPC	44 PLCC	2.5V–6.0V	COP888CG

MACRO CROSS ASSEMBLER

National Semiconductor offers a COP8 macro cross assembler. It runs on industry standard compatible PCs and supports all of the full-symbolic debugging features of the MetaLink iceMASTER emulators.

Assembler Ordering Information

Part Number	Description	Manual
COP8-DEV-IBMA	COP8 Assembler/ Linker/Librarian for IBM®, PC-/XT®, AT® or compatible.	424410632-001

Emulator Ordering Information

Part Number	Description	Current Version
IM-COP8/400/1‡	MetaLink base unit in-circuit emulator for all COP8 devices, symbolic debugger software and RS232 serial interface cable, with 110V @ 60 Hz Power Supply.	HOST SOFTWARE: VER. 3.3 REV.5, Model File Rev 3.050.
IM-COP8/400/2‡	MetaLink base unit in-circuit emulator for all COP8 devices, symbolic debugger software and RS32 serial interface cable, with 220V @ 50 Hz Power Supply.	
DM-COP8/888EG‡	MetaLink iceMASTER Debug Module. This is the low cost version of the MetaLink iceMASTER. Firmware Ver. 6.07.	

‡These parts include National's COP8 Assembler/Linker/Librarian Package (COP8-DEV-IBMA).

Development Support (Continued)

SINGLE CHIP EMULATOR DEVICE

The COP8 family is fully supported by One-Time Programmable (OTP) emulators. For more detailed information refer to the emulation device specific datasheets and the single chip emulator selection table below.

PROGRAMMING SUPPORT

Programming of the single chip emulator devices is supported by different sources. The following programmers are certified for programming the One-Time Programmable (OTP) devices:

EPROM Programmer Information

Manufacturer and Product	U.S. Phone Number	Europe Phone Number	Asia Phone Number
MetaLink- Debug Module	(602) 926-0797	Germany: +49-8141-1030	Hong Kong: +852-737-1800
Xeltek- Superpro	(408) 745-7974	Germany: +49 2041-684758	Singapore: +65-276-6433
BP Microsystems- Turpro	(800) 225-2102	Germany: +49 2041-684758	Hong Kong: +852-388-0629
Data I/O-Unisite -System 29 -System 39	(800) 322-8246	Europe: +31-20-622866 Germany: +49-89-858020	Japan: +81-33-432-6991
Abcom-COP8 Programmer		Europe: +49-89 808707	
System General- Turpro-1-FX -APRO	(408) 263-6667	Switzerland: +41-31 921-7844	Taiwan: +886-2-917-3005

The COP8788EG/COP8784EG can be used to emulate the COP8788CG/COP8784CG.

Single Chip Emulator Ordering Information

Device Number	Clock Option	Package	Emulates
COP8788EGV-X COP8788EGV-R*	Crystal R/C	44 PLCC	COP888EG
COP8788EGN-X COP8788EGN-R*	Crystal R/C	40 DIP	COP888EG
COP8784EGN-X COP8784EGN-R*	Crystal R/C	28 DIP	COP884EG
COP8784EGWM-X* COP8784EGWM-R*	Crystal R/C	28 SO	COP884EG

*Check with the local sales office about the availability.

Development Support (Continued)

DIAL-A-HELPER

Dial-A-Helper is a service provided by the Microcontroller Applications group. The Dial-A-Helper is an Electronic Bulletin Board Information system.

INFORMATION SYSTEM

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 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. The minimum requirement for accessing the Dial-A-Helper is a Hayes compatible modem.

If the user has a PC with a communications package then files from the FILE SECTION can be down loaded to disk for later use.

ORDER P/N: MOLE-DIAL-A-HLP

Information System Package contains:
  Dial-A-Helper Users Manual
  Public Domain Communications Software

FACTORY APPLICATIONS SUPPORT

Dial-A-Helper also provides immediate factor applications support. If a user has questions, he can leave messages on our electronic bulletin board, which we will respond to.

Voice: (800) 272-9959
Modem: Canada/U.S.: (800) NSC-MICRO
(800) 672-6427
Baud: 14.4 k
Set-up: Length: 8-Bit
Parity: None
Stop Bit: 1
Operation: 24 Hrs., 7 Days

National Semiconductor

COP888EK/COP884EK
Single-Chip microCMOS Microcontrollers

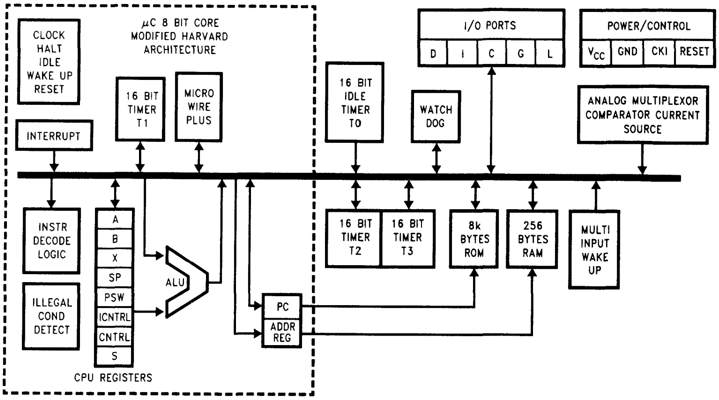
General Description

The COP888 family of microcontrollers uses an 8-bit single chip core architecture fabricated with National Semiconductor's M2CMOS™ process technology. The COP888EK/COP884EK is a member of this expandable 8-bit core processor family of microcontrollers. (Continued)

Features

- Low cost 8-bit microcontroller
- Fully static CMOS, with low current drain
- Two power saving modes: HALT and IDLE
- 1 μs instruction cycle time
- 8k bytes on-board ROM
- 256 bytes on-board RAM
- Single supply operation: 2.5V–6V
- Analog function block with
  - — Analog comparator with seven input multiplexor
  - — Constant current source and $V_{CC/2}$ reference
- MICROWIRE/PLUS™ serial I/O
- WATCHDOG™ and Clock Monitor logic
- Idle Timer
- Multi-Input Wakeup (MIWU) with optional interrupts (8)
- Three 16-bit timers, each with two 16-bit registers supporting:
  - — Processor Independent PWM mode
  - — External Event counter mode
  - — Input Capture mode
- 8-bit Stack Pointer SP (stack in RAM)
- Two 8-bit Register Indirect Data Memory Pointers (B and X)
- 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
- Versatile instruction set
- True bit manipulation
- Memory mapped I/O
- BCD arithmetic instructions
- Package:
  - — 44 PLCC with 39 I/O pins
  - — 40 N with 35 I/O pins
  - — 28 SO or 28 N, each with 23 I/O pins
- 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
- Quiet design (low radiated emissions)
- Temperature range: −40°C to +85°C
- Single chip emulation devices
- Real time emulation and full program debug offered by MetaLink's Development Systems

Block Diagram



TL/DD/12094–1

FIGURE 1. Block Diagram

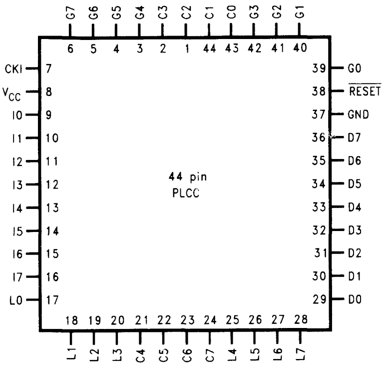
General Description (Continued)

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), 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 of 1 μs 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.

Connection Diagrams

Plastic Chip Carrier

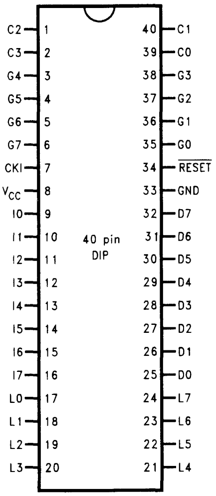


TL/DD/12094-2

Top View

Order Number COP888EK-XXX/V
See NS Plastic Chip Package Number V44A

Dual-In-Line Package

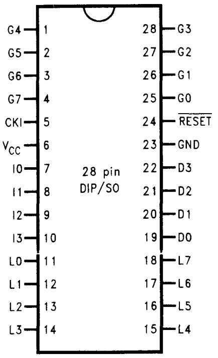


TL/DD/12094-3

Top View

Order Number COP888EK-XXX/N
See NS Molded Package Number N40A

Dual-In-Line Package



TL/DD/12094-4

Top View

Order Number COP884EK-XXX/WM or COP884EK-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		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 888EK: $-40°C \leq T_A \leq +85°C$ 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	$V_{CC} = 6V$, $t_c = 1\ \mu s$			12.5	mA
CKI = 4 MHz	$V_{CC} = 6V$, $t_c = 2.5\ \mu s$			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)	$V_{CC} = 6V$, CKI = 0 MHz		< 5	10	μA
	$V_{CC} = 4.0V$, CKI = 0 MHz		< 3	6	μA
IDLE Current					
CKI = 10 MHz	$V_{CC} = 6V$, $t_c = 1\ \mu s$			3.5	mA
CKI = 4 MHz	$V_{CC} = 6V$, $t_c = 2.5\ \mu s$			2.5	mA
CKI = 1 MHz	$V_{CC} = 4.0V$, $t_c = 10\ \mu s$			0.7	mA
Input Levels					
$\overline{\text{RESET}}$					
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 oscillator, 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 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.

DC Electrical Characteristics 888EK: $-40^\circ C \leq T_A \leq +85^\circ C$ 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)	$T_A = 25^\circ C$			±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 888EK: $-40^\circ C \leq T_A \leq +85^\circ C$ 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 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}) (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 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: 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: 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.

1

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 7)		0.4		$V_{CC} - 1.5$	V
Voltage Gain			300k		V/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	$4.0V < V_{CC} < 6.0V$ Temp = Constant			2	μA
Current Source Enable Time			1.5	2	μs
Comparator Response Time	100 mV Overdrive, 100 pF Load			1	μs

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.

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 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.

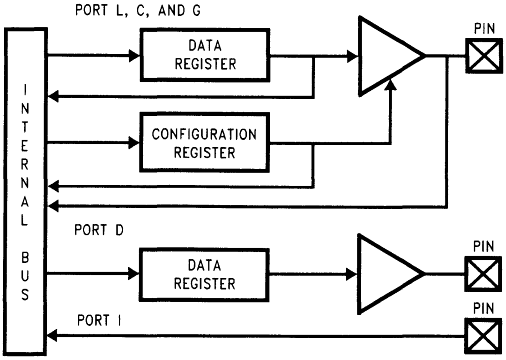
The Port L supports Multi-Input Wake Up on all eight pins. L4 and L5 are used for the timer input functions T2A and T2B. L6 and L7 are used for the timer input functions T3A and T3B.

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 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 on the next page. Reading the G6 and G7 data bits will return zeros.



TL/DD/12094-5

FIGURE 3. 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 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).

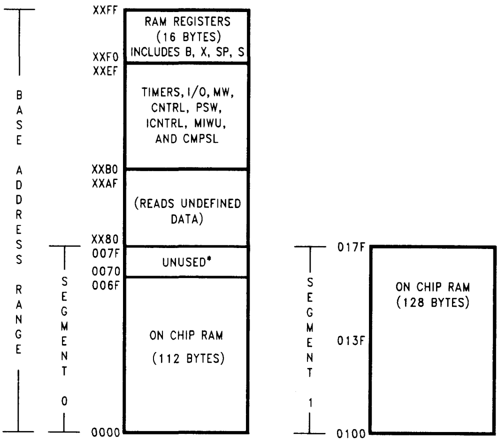
The data store memory is either addressed directly by a single byte address within the instruction, or indirectly relative to the reference of the B, X, or SP pointers (each contains a single-byte address). This single-byte address allows an addressing range of 256 locations from 00 to FF hex. The upper 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 intitialized 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/12094-6

*Reads as all ones.

FIGURE 4. RAM Organization

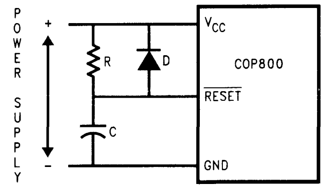
Reset

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 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 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/12094-7

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 oscillator diagrams.

CRYSTAL OSCILLATOR

CKI and CKO can be connected to make a closed loop crystal (or resonator) controlled oscillator.

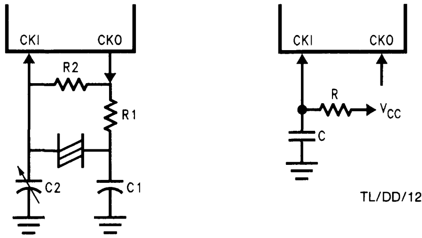
Table A shows the component values required for various standard crystal values.

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 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: $3k \leq R \leq 200k$

$50\ pF \leq C \leq 200\ pF$

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

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.

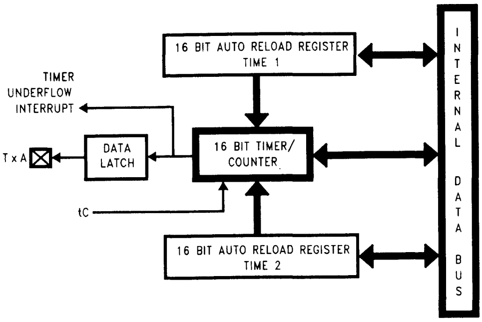
The Tx Timer control bits, TxC3, TxC2 and TxC1 set up the timer for PWM mode operation.

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/12094–10

FIGURE 7. Timer in PWM Mode

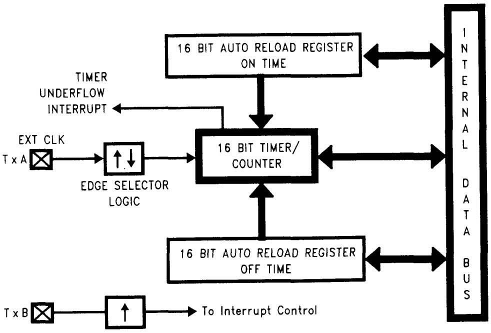
Mode 2. External Event Counter Mode

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/12094–11

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.

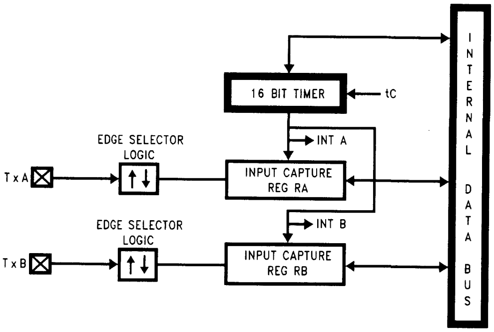
Timers (Continued)

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/12094–12

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 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.

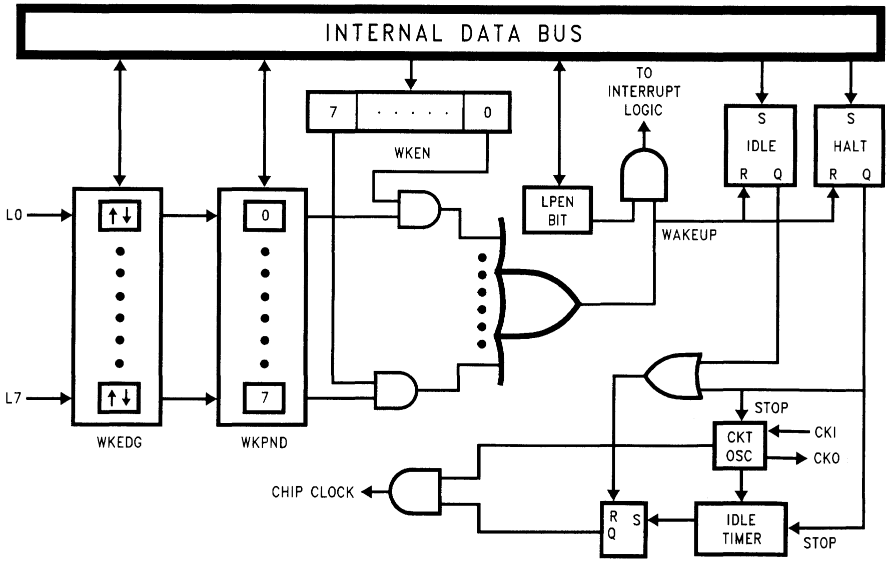
Alternatively, the user can enter the IDLE mode with the IDLE Timer T0 interrupt disabled. In this case, the device will resume normal operation with the instruction immediately following the "Enter IDLE Mode" instruction.

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 10 shows the Multi-Input Wakeup logic.



TL/DD/12094-13

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.

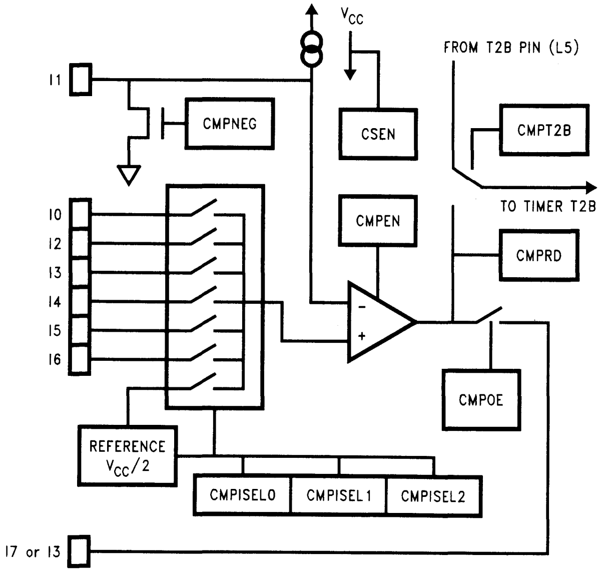
A control flag, LPEN, functions as a global interrupt enable for Port L interrupts. Setting the LPEN flag will enable interrupts and vice versa. A separate global pending flag is not needed since the register WKPND is adequate.

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.

Analog Function Block



TL/DD/12094-14

FIGURE 11. 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.

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.

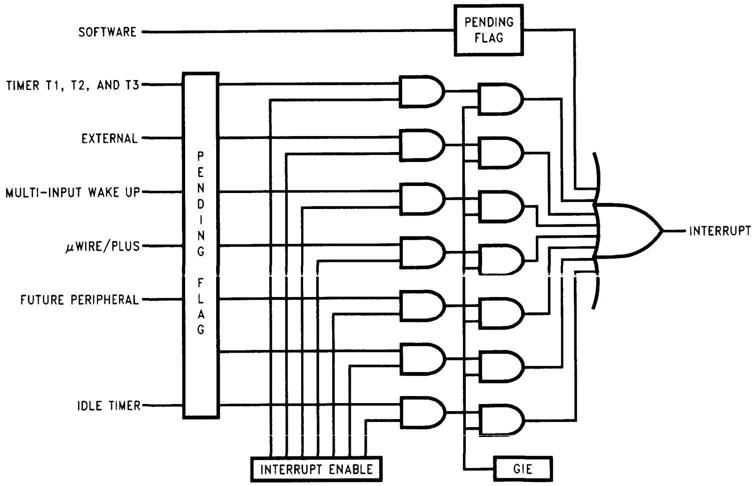
Interrupts

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/12094–15

FIGURE 12. 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)	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 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. This vector can point to the Software Trap (ST) interrupt service routine, or to another special service routine as desired.

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.

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.

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 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

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), 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:

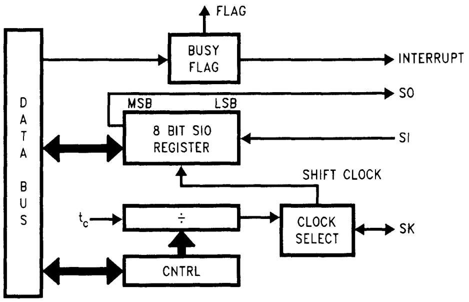
a. Executing from undefined ROM

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 13* shows a block diagram of the MICROWIRE/PLUS logic.



TL/DD/12094-16

FIGURE 13. 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 14* 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 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

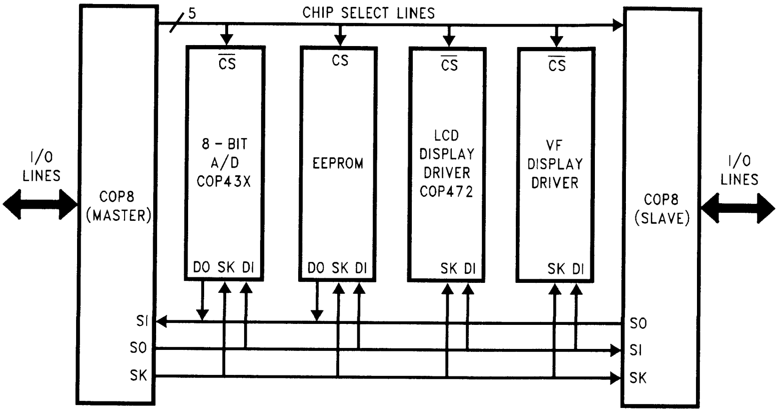
The device allows either the normal SK clock or an alternate phase SK clock to shift data in and out of the SIO register. In both the modes the SK is normally low. In the normal mode data is shifted in on the rising edge of the SK clock and the data is shifted out on the falling edge of the SK clock. 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 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	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
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]
MemI	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.

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/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

IN-CIRCUIT EMULATOR

The MetaLink iceMASTER™-COP8 Model 400 In-Circuit Emulator for the COP8 family of microcontrollers features high-performance operation, ease of use, and an extremely flexible user-interface for maximum productivity. Interchangeable probe cards, which connect to the standard common base, support the various configurations and packages of the COP8 family.

The iceMASTER provides real time, full speed emulation up to 10 MHz, 32 kBytes of emulation memory and 4k frames of trace buffer memory. The user may define as many as 32k trace and break triggers which can be enabled, disabled, set or cleared. They can be simple triggers based on code or address ranges or complex triggers based on code address, direct address, opcode value, opcode class or immediate operand. Complex breakpoints can be ANDed and ORed together. Trace information consists of address bus values, opcodes and user selectable probe clips status (external event lines). The trace buffer can be viewed as raw hex or as diassembled instructions. The probe clip bit values can be displayed in binary, hex or digital waveform formats.

During single-step operation the dynamically annotated code feature displays the contents of all accessed (read and write) memory locations and registers, as well as flow-of-control direction change markers next to each instruction executed.

The iceMASTER's performance analyzer offers a resolution of better than 6 μs. The user can easily monitor the time spent executing specific portions of code and find "hot spots" or "dead code". Up to 15 independent memory areas based on code address or label ranges can be defined. Analysis results can be viewed in bar graph format or as actual frequency count.

Emulator memory operations for program memory include single line assembler, disassembler, view, change and write to file. Data memory operations include fill, move, compare, dump to file, examine and modify. The contents of any memory space can be directly viewed and modified from the corresponding window.

The iceMASTER comes with an easy to use windowed interface. Each window can be sized, highlighted, color-controlled, added, or removed completely. Commands can be accessed via pull-down-menus and/or redefineable hot keys. A context sensitive hypertext/hyperlinked on-line help system explains clearly the options the user has from within any window.

The iceMASTER connects easily to a PC® via the standard COMM port and its 115.2 kBaud serial link keeps typical program download time shorter.

The following tables list the emulator and probe cards ordering information.

Emulator Ordering Information

Part Number	Description	Current Version
IM-COP8/400/1‡	MetaLink base unit in-circuit emulator for all COP8 devices, symbolic debugger software and RS 232 serial interface cable, with 110V @ 60 Hz Power Supply.	Host Software: Ver. 3.3 Rev. 5, Model File Rev 3.050.
IM-COP8/400/2‡	MetaLink base unit in-circuit emulator for all COP8 devices, symbolic debugger software and RS 232 serial interface cable, with 220V @ 50 Hz Power Supply.	

‡These parts include National's COP8 Assembler/Linker/Librarian Package (COP8-DEV-IBMA).

Probe Card Ordering Information

Part Number	Package	Voltage Range	Emulates
MHW-888EK44DWPC	44 PLCC	2.5V–5.5V	COP888EK
MHW-888EK40DWPC	40 DIP	2.5V–5.5V	COP888EK
MHW-884EK28DWPC	28 DIP	2.5V–5.5V	COP884EK
MHW-SOIC28	28 SO	28-pin SOIC Adaptor Kit	

MACRO CROSS ASSEMBLER

National Semiconductor offers a relocatable COP8 macro cross assembler. It runs on industry standard compatible PCs and supports all of the full-symbolic debugging features of the MetaLink IceMASTER emulators.

Assembler Ordering Information

Part Number	Description	Manual
COP8-DEV-IBMA	COP8 Assembler/ Linker/Librarian for IBM®, PC/XT®, AT® or compatible.	424410632-001

SINGLE CHIP EMULATOR DEVICE

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.

Development Support (Continued)

PROGRAMMING SUPPORT

Programming of the single chip emulator devices is supported by different sources.

The following programmers are certified for programming EPROM versions of COP8.

EPROM Programmer Information

Manufacturer and Product	U.S. Phone Number	Europe Phone Number	Asia Phone Number
MetaLink— Debug Module	(602) 926-0797	Germany: (49-81-41) 1030	Hong Kong: 852-737-1800
Xeltek— Superpro	(408) 745-7974	Germany: (49-20-41) 684758	Singapore: (65) 276-6433
BP Microsystems— EP-1140	(800) 225-2102	Germany: (49-89-85) 76667	Hong Kong: (852) 388-0629
Data I/O—Unisite; —System 29 —System 39	(800) 322-8246	Europe: (31-20) 622866 Germany: (49-89-85) 8020	Japan: (33) 432-6991
Abcom—COP8 Programmer		Europe: (89-80) 8707	
System General— Turpro-1—FX; —APRO	(408) 263-6667	Switzerland: (31) 921-7844	Taiwan: (2) 917-3005

DIAL-A-HELPER

Dial-A-Helper is a service provided by the Microcontroller Applications group. The Dial-A-Helper is an Electronic Bulletin Board Information system.

INFORMATION SYSTEM

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 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. The minimum requirement for accessing the Dial-A-Helper is a Hayes compatible modem.

If the user has a PC with a communications package then files from the FILE SECTION can be down loaded to disk for later use.

ORDER P/N: MOLE-DIAL-A-HLP

Information System Package contains:
- Dial-A-Helper Users Manual
- Public Domain Communications Software

FACTORY APPLICATIONS SUPPORT

Dial-A-Helper also provides immediate factor applications support. If a user has questions, he can leave messages on our electronic bulletin board, which we will respond to.

Voice:	(800) 272-9959		
Modem:	CANADA/U.S.:	(800) NSC-MICRO	
	Baud:	14.4k	
	Set-up:	Length:	8-Bit
		Parity:	None
		Stop Bit:	1
	Operation:	24 Hrs., 7 Days	

National Semiconductor

COP688EG/COP684EG/COP888EG/COP884EG/ COP988EG/COP984EG Single-Chip microCMOS Microcontrollers

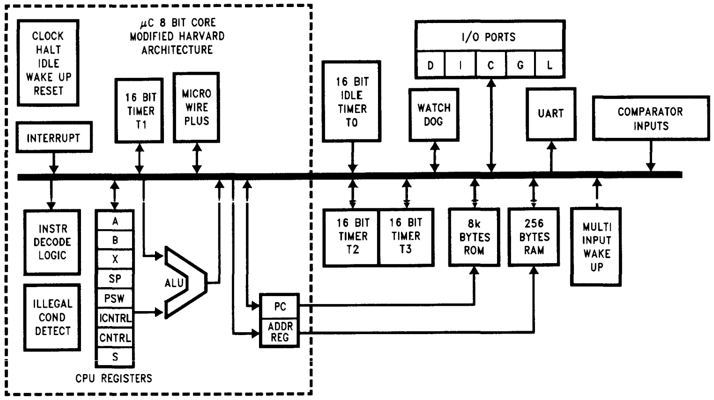
General Description

The COP888 family of microcontrollers uses an 8-bit single chip core architecture fabricated with National Semiconductor's M2CMOS™ process technology. The COP888EG/COP884EG is a member of this expandable 8-bit core processor family of microcontrollers. (Continued)

Features

- Low cost 8-bit microcontroller
- Fully static CMOS, with low current drain
- Two power saving modes: HALT and IDLE
- 1 μs instruction cycle time
- 8k bytes on-board ROM
- 256 bytes on-board RAM
- Single supply operation: 2.5V–6V
- Full duplex UART
- Two analog comparators
- MICROWIRE/PLUS™ serial I/O
- WATCHDOG™ and Clock Monitor logic
- Idle Timer
- Multi-Input Wakeup (MIWU) with optional interrupts (8)
- Three 16-bit timers, each with two 16-bit registers supporting:
  - — Processor Independent PWM mode
  - — External Event counter mode
  - — Input Capture mode
- 8-bit Stack Pointer SP (stack in RAM)
- Two 8-bit Register Indirect Data Memory Pointers (B and X)
- 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
- Versatile instruction set
- True bit manipulation
- Memory mapped I/O
- BCD arithmetic instructions
- Package:
  - — 44 PLCC with 39 I/O pins
  - — 40 N with 35 I/O pins
  - — 28 SO or 28 N, each with 23 I/O pins
- 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
- Temperature ranges: 0°C to +70°C,
  −40°C to +85°C
  −55°C to +125°C
- One-Time Programmable emulation devices
- Real time emulation and full program debug offered by MetaLink's Development Systems

Block Diagram



TL/DD/11214–1

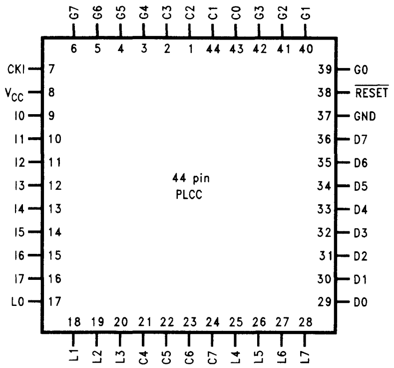
FIGURE 1. Block Diagram

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 of 1 μs per instruction rate.

Connection Diagrams

Plastic Chip Carrier

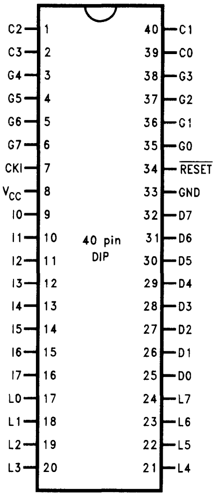


TL/DD/11214-2

Top View

Order Number COP888EG-XXX/V
See NS Plastic Chip Package Number V44A

Dual-In-Line Package

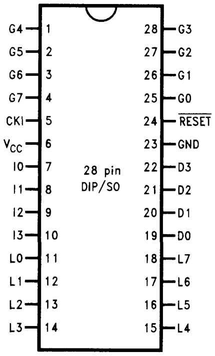


TL/DD/11214-3

Top View

Order Number COP888EG-XXX/N
See NS Molded Package Number N40A

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 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 98XEG: $0°C \leq T_A \leq +70°C$ 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	$V_{CC} = 6V$, $t_c = 1\ \mu s$			12.5	mA
CKI = 4 MHz	$V_{CC} = 6V$, $t_c = 2.5\ \mu s$			5.5	mA
CKI = 4 MHz	$V_{CC} = 4V$, $t_c = 2.5\ \mu s$			2.5	mA
CKI = 1 MHz	$V_{CC} = 4V$, $t_c = 10\ \mu s$			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	$V_{CC} = 6V$, $t_c = 1\ \mu s$			3.5	mA
CKI = 4 MHz	$V_{CC} = 6V$, $t_c = 2.5\ \mu s$			2.5	mA
CKI = 1 MHz	$V_{CC} = 4V$, $t_c = 10\ \mu s$			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)	$4V \le V_{CC} \le 6V$	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.

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: $-40°C \leq T_A \leq +85°C$ 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	$V_{CC} = 6V, t_c = 1 \mu s$			12.5	mA
CKI = 4 MHz	$V_{CC} = 6V, t_c = 2.5 \mu s$			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)	$V_{CC} = 6V$, CKI = 0 MHz		<1	10	μA
	$V_{CC} = 4.0V$, CKI = 0 MHz		<0.5	6	μA
IDLE Current					
CKI = 10 MHz	$V_{CC} = 6V, t_c = 1 \mu s$			3.5	mA
CKI = 4 MHz	$V_{CC} = 6V, t_c = 2.5 \mu s$			2.5	mA
CKI = 1 MHz	$V_{CC} = 4.0V, t_c = 10 \mu s$			0.7	mA
Input Levels					
$\overline{\text{RESET}}$					
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: $-40°C \leq T_A \leq +85°C$ 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	$T_A = 25°C$			±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: $-40°C \leq T_A \leq +85°C$ 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
R/C Oscillator	$2.5V \leq V_{CC} < 4V$	2.5		DC	μs
	$4V \leq V_{CC} \leq 6V$	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 4)	$R_L = 2.2k, C_L = 100$ 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

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: $-55°C \leq T_A \leq +125°C$ 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					
$\overline{\text{RESET}}$					
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 ≤ T_A ≤ +125°C unless otherwise specified (Continued)

Parameter	Conditions	Min	Typ	Max	Units
Allowable Sink/Source Current per Pin D Outputs (Sink) All others				 12 2.5	 mA 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

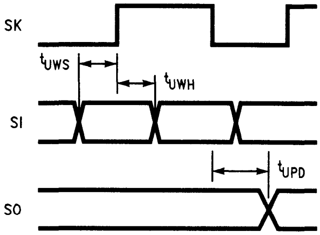
AC Electrical Characteristics 688EG: −55°C ≤ T_A ≤ +125°C unless otherwise specified

Parameter	Conditions	Min	Typ	Max	Units
Instruction Cycle Time (t_c)					
Crystal, Resonator,	$V_{CC} \geq 4.5V$	1		DC	μs
R/C Oscillator	$V_{CC} \geq 4.5V$	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 4)	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: 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

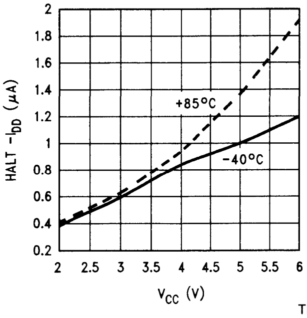


TL/DD/11214-5

FIGURE 2. MICROWIRE/PLUS Timing

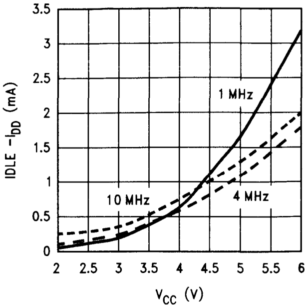
Typical Performance Characteristics ($-40°C \leq T_A \leq +85°C$)

Halt—I_{DD}



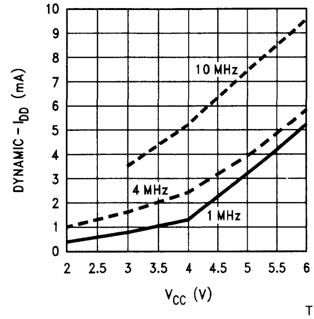
TL/DD/11214-7

Idle—I_{DD} (Crystal Clock Option)



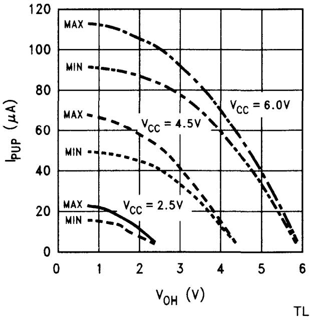
TL/DD/11214-8

Dynamic—I_{DD} vs V_{CC} (Crystal Clock Option)



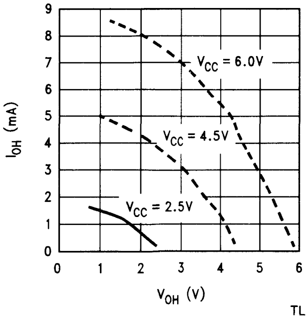
TL/DD/11214-9

Port L/C/G Weak Pull-Up Source Current



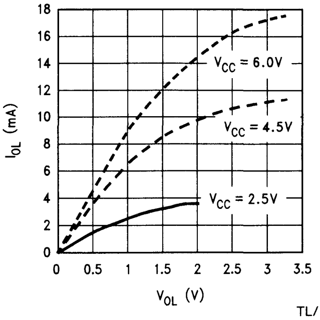
TL/DD/11214-10

Port L/C/G Push-Pull Source Current



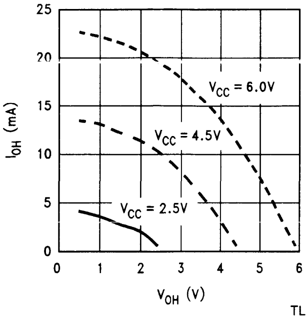
TL/DD/11214-11

Port L/C/G Push-Pull Sink Current



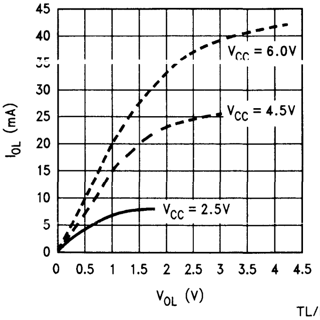
TL/DD/11214-12

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

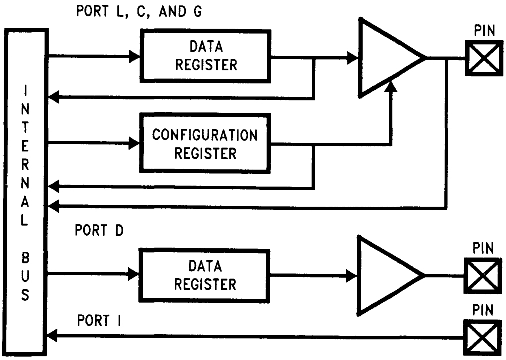
PORT L is an 8-bit I/O port. All L-pins have Schmitt triggers on the inputs.

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

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).

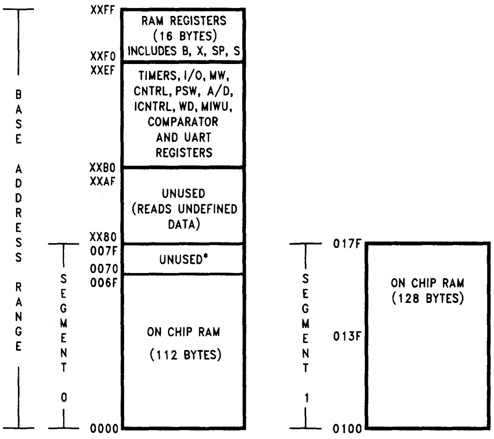
The data store memory is either addressed directly by a single byte address within the instruction, or indirectly relative to the reference of the B, X, or SP pointers (each contains a single-byte address). This single-byte address allows an addressing range of 256 locations from 00 to FF hex. The upper 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 intitialized 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

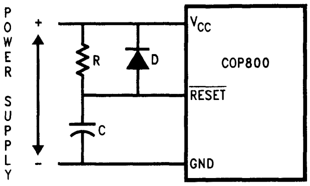
Reset

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.

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

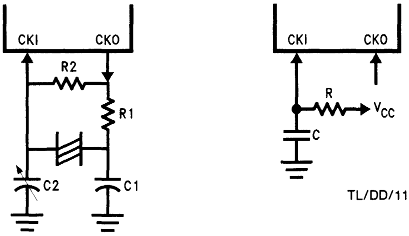
CKI and CKO can be connected to make a closed loop crystal (or resonator) controlled oscillator.

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/11214-18

TL/DD/11214-17

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: $3k \leq R \leq 200k$

$50\text{ pF} \leq C \leq 200\text{ pF}$

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

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.

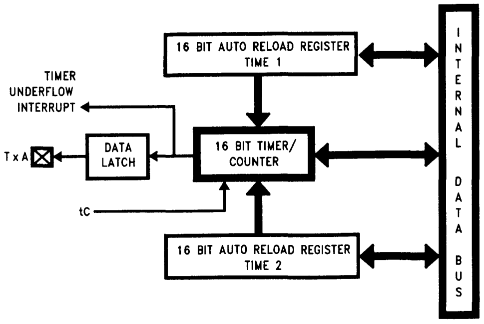
In this mode the timer Tx counts down at a fixed rate of t_C. Upon every underflow the timer is alternately reloaded with the contents of supporting registers, RxA and RxB. The very first underflow of the timer causes the timer to reload from the register RxA. Subsequent underflows cause the timer to 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/11214–19

FIGURE 7. Timer in PWM Mode

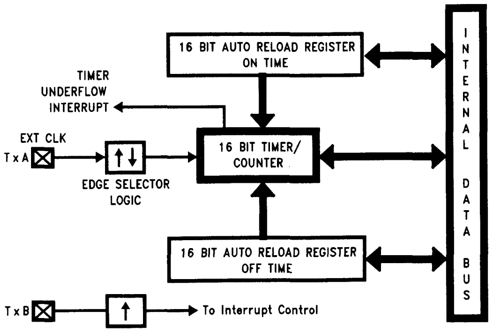
Mode 2. External Event Counter Mode

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.

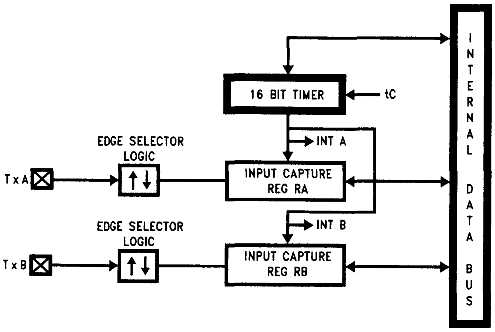
Timers (Continued)

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, $t_C = 1\ \mu s$) 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.

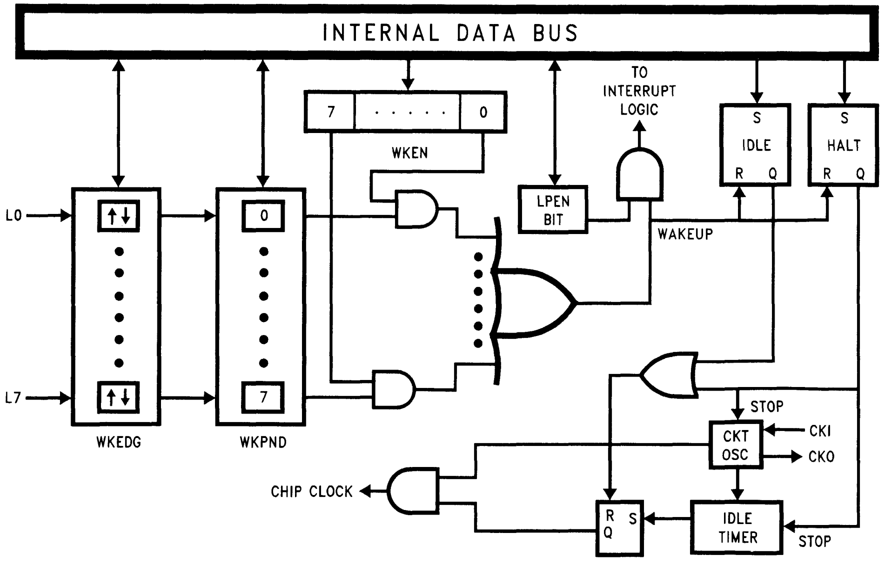
Alternatively, the user can enter the IDLE mode with the IDLE Timer T0 interrupt disabled. In this case, the device will resume normal operation with the instruction immediately following the "Enter IDLE Mode" instruction.

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/11214-22

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.

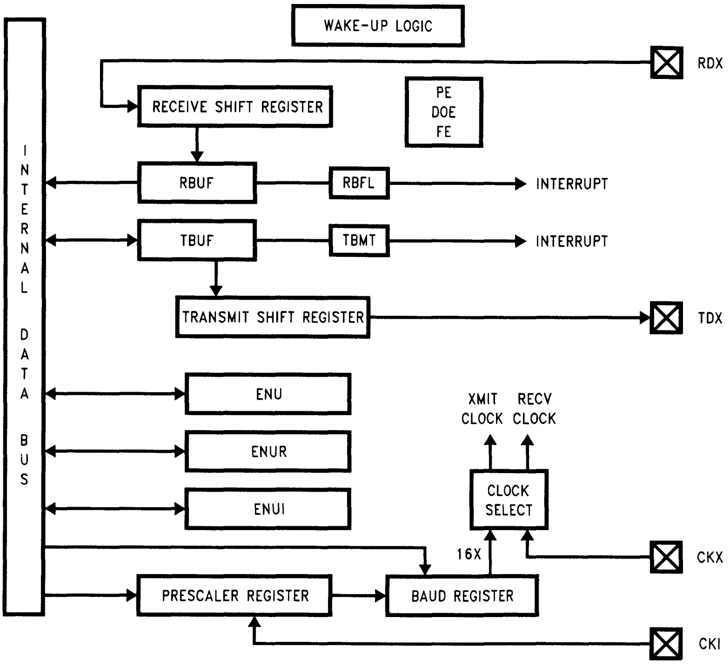
The Wakeup signal will not start the chip running immediately since crystal oscillators or ceramic resonators have a finite start up time. The IDLE Timer (T0) generates a fixed delay to ensure that the oscillator has indeed stabilized before allowing the device to execute instructions. In this case, 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.

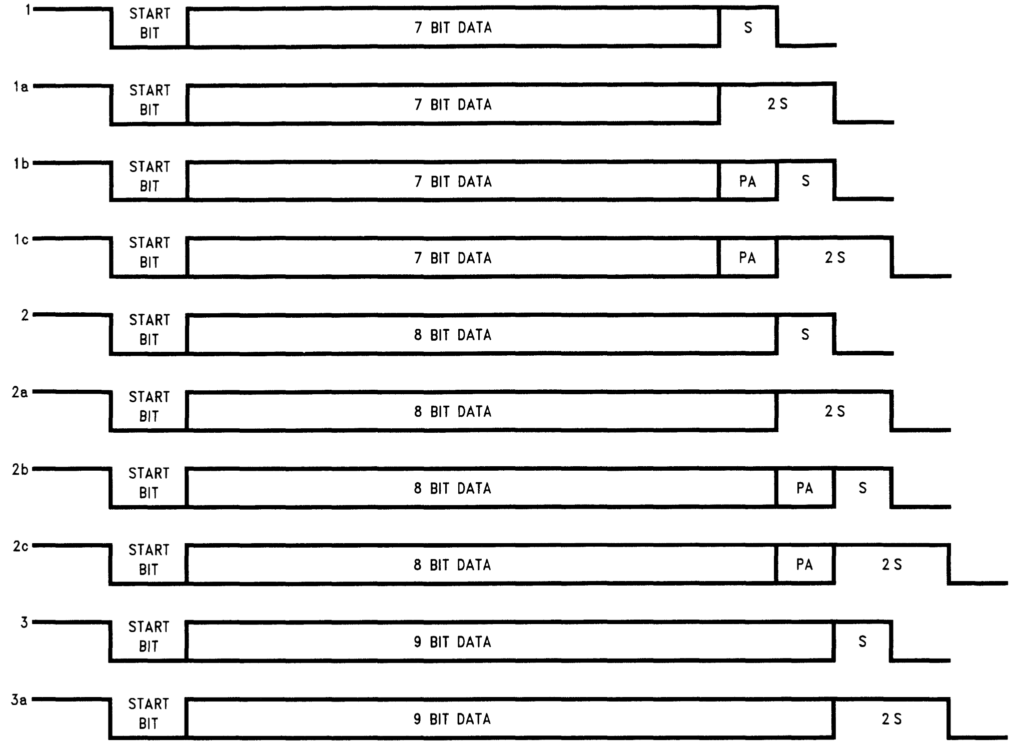
For any of the above framing formats, the last Stop bit can be programmed to be 7/8th of a bit in length. If two Stop bits are selected and the 7/8th bit is set (selected), the second Stop bit will be 7/8th of a bit in length.

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).

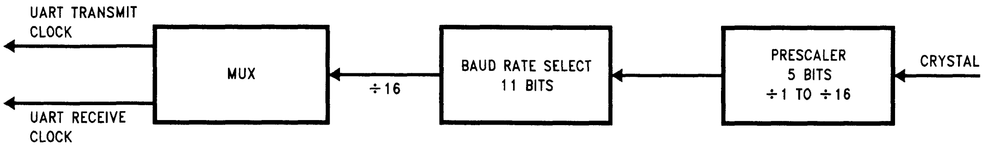
Baud Clock Generation

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.

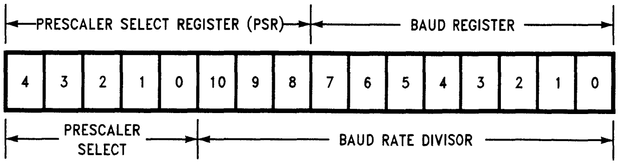
The correspondences between the 5-bit Prescaler Select and Prescaler factors are shown in Table I. Therer are many ways to calculate the two divisor factors, but one particularly effective method would be to achieve a 1.8432 MHz frequency coming out of the first stage. The 1.8432 MHz prescaler output 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:

$$BR = Fc/(16 \times N \times P)$$

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 zero.)

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 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.

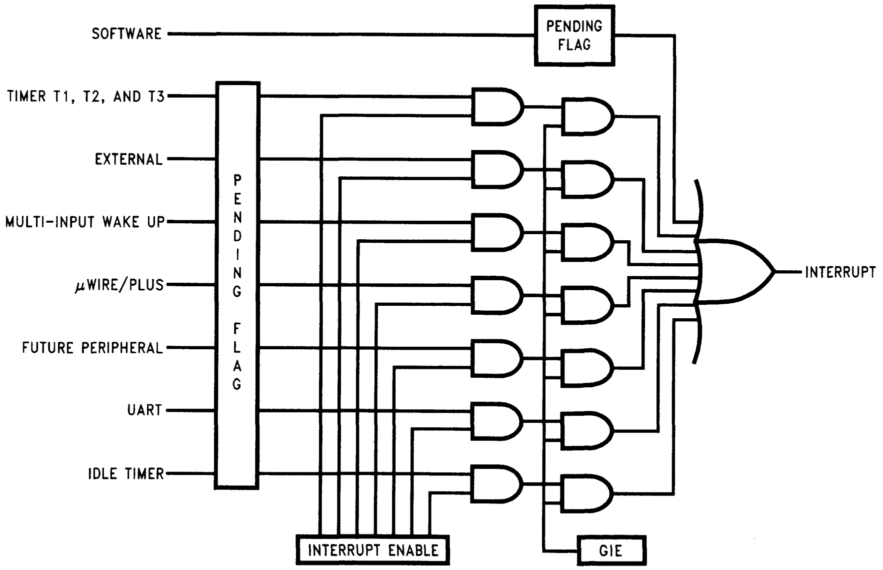
Interrupts

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.



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 ≠ 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 ≠ 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.

Figure 15 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.

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.

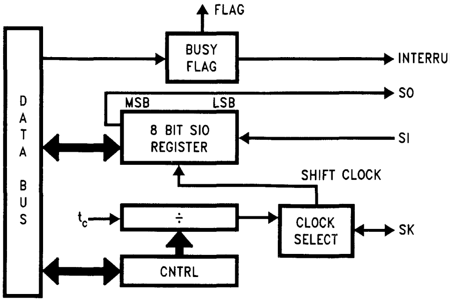
Thus, the chip can detect the following illegal conditions:

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, 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/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

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

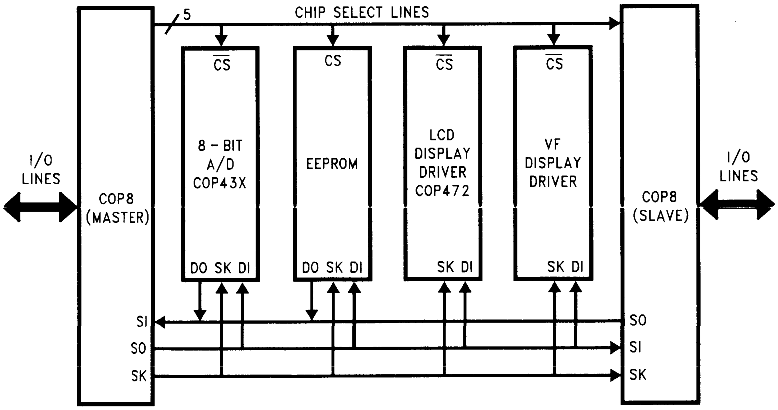
The device allows either the normal SK clock or an alternate phase SK clock to shift data in and out of the SIO register. In both the modes the SK is normally low. In the normal mode data is shifted in on the rising edge of the SK clock and the data is shifted out on the falling edge of the SK clock. 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/11214-29

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
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 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]
MemI	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.

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/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

IN-CIRCUIT EMULATOR

The MetaLink iceMASTER™-COP8 Model 400 In-Circuit Emulator for the COP8 family of microcontrollers features high-performance operation, ease of use, and an extremely flexible user-interface for maximum productivity. Interchangeable probe cards, which connect to the standard common base, support the various configurations and packages of the COP8 family.

The iceMASTER provides real time, full speed emulation up to 10 MHz, 32 kBytes of emulation memory and 4k frames of trace buffer memory. The user may define as many as 32k trace and break triggers which can be enabled, disabled, set or cleared. They can be simple triggers based on code or address ranges or complex triggers based on code address, direct address, opcode value, opcode class or immediate operand. Complex breakpoints can be ANDed and ORed together. Trace information consists of address bus values, opcodes and user selectable probe clips status (external event lines). The trace buffer can be viewed as raw hex or as diassembled instructions. The probe clip bit values can be displayed in binary, hex or digital waveform formats.

During single-step operation the dynamically annotated code feature displays the contents of all accessed (read and write) memory locations and registers, as well as flow-of-control direction change markers next to each instruction executed.

The iceMASTER's performance analyzer offers a resolution of better than 6 μs. The user can easily monitor the time spent executing specific portions of code and find "hot spots" or "dead code". Up to 15 independent memory areas based on code address or label ranges can be defined. Analysis results can be viewed in bar graph format or as actual frequency count.

Emulator memory operations for program memory include single line assembler, disassembler, view, change and write to file. Data memory operations include fill, move, compare, dump to file, examine and modify. The contents of any memory space can be directly viewed and modified from the corresponding window.

The iceMASTER comes with an easy to use windowed interface. Each window can be sized, highlighted, color-controlled, added, or removed completely. Commands can be accessed via pull-down-menus and/or redefineable hot keys. A context sensitive hypertext/hyperlinked on-line help system explains clearly the options the user has from within any window.

The iceMASTER connects easily to a PC® via the standard COMM port and its 115.2 kBaud serial link keeps typical program download time to under 3 seconds.

The following tables list the emulator and probe cards ordering information.

Emulator Ordering Information

Part Number	Description	Current Version
IM-COP8/400/1‡	MetaLink base unit in-circuit emulator for all COP8 devices, symbolic debugger software and RS 232 serial interface cable, with 110V @ 60 Hz Power Supply.	Host Software: Ver. 3.3 Rev. 5, Model File Rev 3.050.
IM-COP8/400/2‡	MetaLink base unit in-circuit emulator for all COP8 devices, symbolic debugger software and RS 232 serial interface cable, with 220V @ 50 Hz Power Supply.	
DM-COP8/888EG‡	MetaLink iceMASTER Debug Module. This is the low cost version of the MetaLink iceMASTER. Firmware: Ver. 6.07.	

‡These parts include National's COP8 Assembler/Linker/Librarian Package (COP8-DEV-IBMA).

Probe Card Ordering Information

Part Number	Package	Voltage Range	Emulates
MHW-888EG44DWPC	44 PLCC	2.5V–5.5V	COP888EG
MHW-888EG40DWPC	40 DIP	2.5V–5.5V	COP888EG
MHW-884EG28DWPC	28 DIP	2.5V–5.5V	COP884EG
MHW-SOIC28	28 SO	28-Pin SOIC Adaptor Kit	

Assembler Ordering Information

Part Number	Description	Manual
COP8-DEV-IBMA	COP8 Assembler/Linker/Librarian for IBM® PC/XT®, AT® or compatible.	424410632-001

SINGLE CHIP EMULATOR DEVICE

The COP8 family is fully supported by One-Time Programmable (OTP) emulators. For more detailed information refer to the emulation device specific datasheets and the emulator selection table below.

Single Chip Emulator Ordering Information

Device Number	Clock Option	Package	Emulates
COP8788EGV-X COP8788EGV-R*	Crystal R/C	44 PLCC	COP888EG
COP8788EGN-X COP8788EGN-R*	Crystal R/C	40 DIP	COP888EG
COP8784EGN-X COP8784EGN-R*	Crystal R/C	28 DIP	COP884EG
COP8784EGWM-X COP8784EGWM-R*	Crystal R/C	28 SO	COP884EG

*Check with the local sales office about the availability.

Development Support (Continued)

PROGRAMMING SUPPORT

Programming of the single chip emulator devices is supported by different sources. The following programmers are certified for programming these One-Time Programmable emulator devices:

EPROM Programmer Information

Manufacturer and Product	U.S. Phone Number	Europe Phone Number	Asia Phone Number
MetaLink- Debug Module	(602) 926-0797	Germany: +49-8141-1030	Hong Kong: 852-737-1800
Xeltek- Superpro	(408) 745-7974	Germany: +49-2041 684758	Singapore: +65 276-6433
BP Microsystems- Turpro	(800) 225-2102	Germany: +49 89 857 66 67	Hong Kong: +852 388-0629
Data I/O-Unisite -System 29 -System 39	(800) 322-8246	Europe: +31-20-622866 Germany: +49-89-85-8020	Japan: +33-432-6991
Abcom-COP8 Programmer		Europe: +89 808707	
System General- Turpro-1-FX -APRO	(408) 263-6667	Switzerland: +31-921-7844	Taiwan: +2-917-3005

DIAL-A-HELPER

Dial-A-Helper is a service provided by the Microcontroller Applications group. The Dial-A-Helper is an Electronic Bulletin Board Information system.

INFORMATION SYSTEM

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 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. The minimum requirement for accessing the Dial-A-Helper is a Hayes compatible modem.

If the user has a PC with a communications package then files from the FILE SECTION can be down loaded to disk for later use.

ORDER P/N: MOLE-DIAL-A-HLP

Information System Package contains:
- Dial-A-Helper Users Manual
- Public Domain Communications Software

FACTORY APPLICATIONS SUPPORT

Dial-A-Helper also provides immediate factor applications support. If a user has questions, he can leave messages on our electronic bulletin board, which we will respond to.

Voice: (800) 272-9959
Modem: CANADA/U.S.: (800) NSC-MICRO
Baud: 14.4k
Set-up: Length: 8-Bit
Parity: None
Stop Bit: 1
Operation: 24 Hrs., 7 Days

National Semiconductor

PRELIMINARY

COP888GW
Single-Chip microCMOS Microcontroller

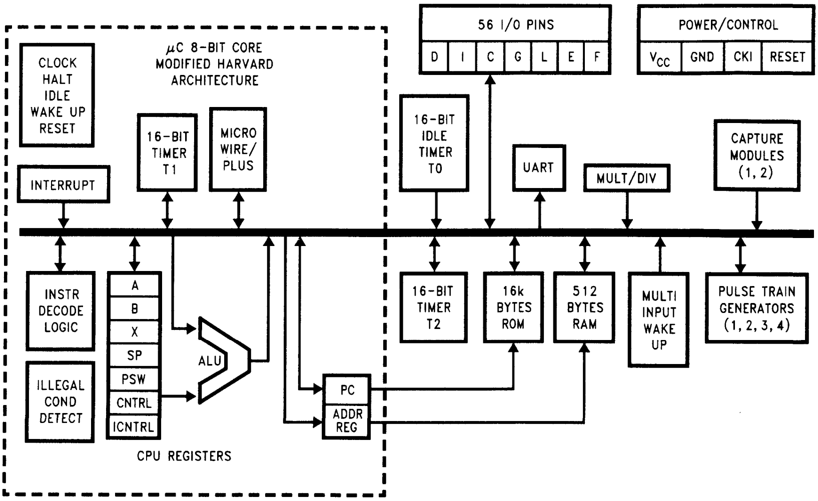
General Description

The COP888 family of microcontrollers uses an 8-bit single chip core architecture fabricated with National Semiconductor's M2CMOS™ process technology. The COP888GW is a member of this expandable 8-bit core processor family of microcontrollers. (Continued)

Features

- Low cost 8-bit microcontroller
- Fully static CMOS, with low current drain
- Two power saving modes: HALT and IDLE
- 1 µs instruction cycle time
- 16 kbytes on-board ROM
- 512 bytes on-board RAM
- Single supply operation: 2.5V–6V
- Full duplex UART
- MICROWIRE/PLUS™ serial I/O
- Idle Timer
- Two 16-bit timers, each with two 16-bit registers supporting:
  - — Processor independent PWM mode
  - — External event counter mode
  - — Input capture mode
- Four pulse train generators with 16-bit prescalers
- Two 16-bit input capture modules with 8-bit prescalers
- Multi-Input Wake Up (MIWU) with optional interrupts (8)
- 8-bit Stack Pointer SP (stack in RAM)
- Two 8-bit register indirect data memory pointers (B and X)
- 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 instruction set
- True bit manipulation
- Memory mapped I/O
- BCD arithmetic instructions
- Multiply/Divide Functions
- 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
- Temperature range: −40°C to +85°C
- Real time emulation and full program debug offered by MetaLink Development System

Block Diagram



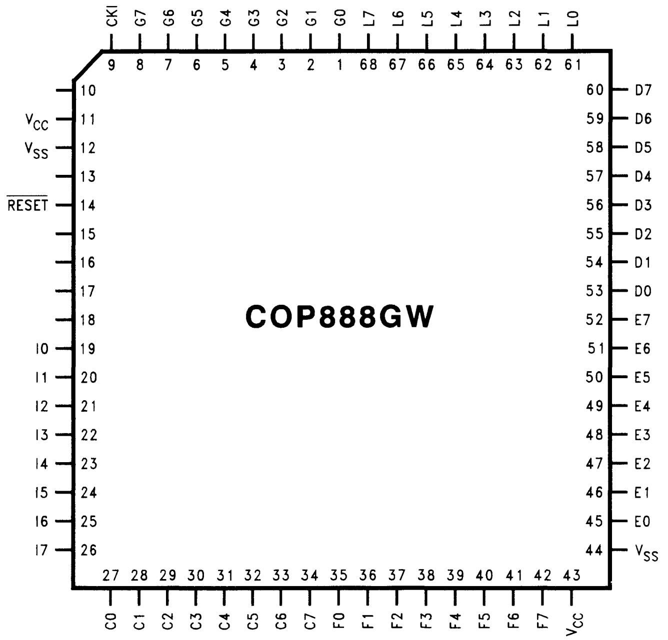
TL/DD12065-1

FIGURE 1. COP888GW 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 2.5V–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.

Connection Diagram



TL/DD12065-2

Top View

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: $-40°C \leq T_A \leq +85°C$ 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: $-40°C < T_A < +85°C$ 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			
t_{HOLD}	$V_{CC} \ge 4V$	60			
	$2.5V \le V_{CC} < 4V$	150			
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	
All Others	$V_{CC} \ge 4V$			1	
	$2.5V \le V_{CC} < 4V$			2.5	
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			
MICROWIRE Output Propagation Delay (t_{UPD})	$V_{CC} \ge 4V$			220	
Input Pulse Width (Note 7)					
Interrupt Input High Time		1			t_c
Interrupt Input Low Time		1			
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			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.

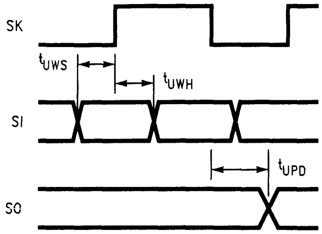
Note 4: Pins G6 and $\overline{\text{RESET}}$ are designed with a high voltage input network. These pins allow input voltages greater than V_{CC} and the pins will have sink current to V_{CC} when biased at voltages greater than V_{CC} (the pins do not have source current when biased at a voltage below 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/DD12065-3

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 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 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

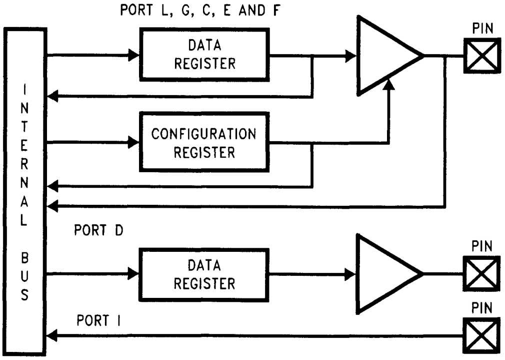
PORT L is an 8-bit I/O port. All L-pins have Schmitt triggers on the inputs.

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/DD12065-4

FIGURE 3. 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.

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 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

Data Memory Segment RAM Extension (Continued)

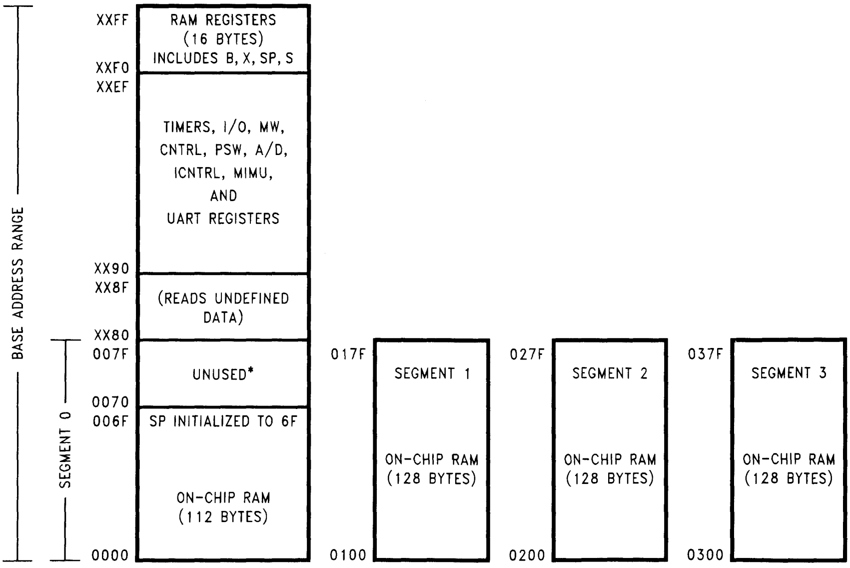
register S is used to extend the base address range (from 0000 to 007F) from XX00 to XX7F, where XX represents the 8 bits from the S register. Thus the 128-byte data segment extensions are located from addresses 0100 to 017F for data segment 1, 0200 to 027F for data segment 2, etc., up to FF00 to FF7F 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:

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

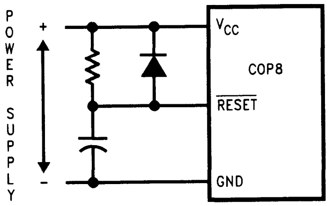
RANDOM after RESET at power-on

RAM:

UNAFFECTED after RESET with power already applied

RANDOM after RESET at power-on

The external RC network shown in *Figure 5* should be used to ensure that the $\overline{\text{RESET}}$ pin is held low until the power supply to the chip stabilizes.



TL/DD12065-6

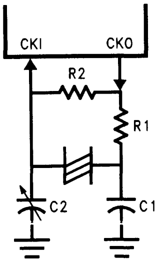
RC > 5 × POWER SUPPLY RISE TIME

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/DD12065-7

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	$V_{CC} = 5V$

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

Thus the total current drain, It, is given as

$$It = I1 + I2 + I3 + I4 + I5$$

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

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 ($t_C = 1\ \mu s$). 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.

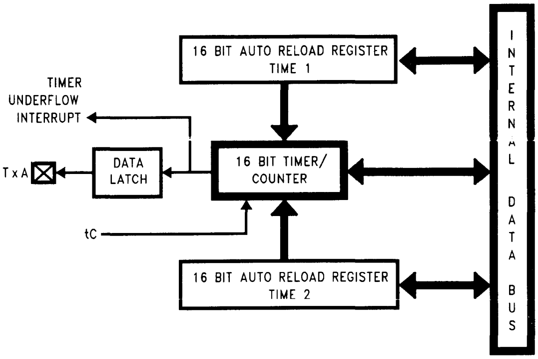
The underflows can be programmed to toggle the TxA output pin. The underflows can also be programmed to generate interrupts.

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

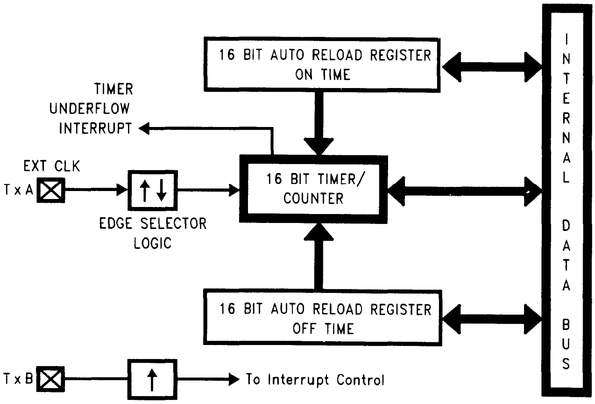
This mode is quite similar to the processor independent PWM mode described above. The main difference is that the timer, Tx, is clocked by the input signal from the TxA pin. The Tx timer control bits, TxC3, TxC2 and TxC1 allow the timer to be clocked either on a positive or negative edge from the TxA 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/DD12065-8

FIGURE 7. Timer in PWM Mode

Timers (Continued)



TL/DD12065-9

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.

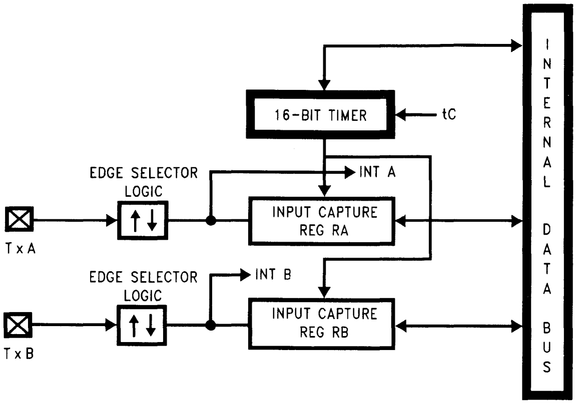
In this mode, the timer Tx is constantly running at the fixed t_C rate. The two registers, RxA and RxB, act as capture registers. Each register acts in conjunction with a pin. The register RxA acts in conjunction with the TxA pin and the register RxB acts in conjunction with the TxB pin.

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/DD12065-10

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

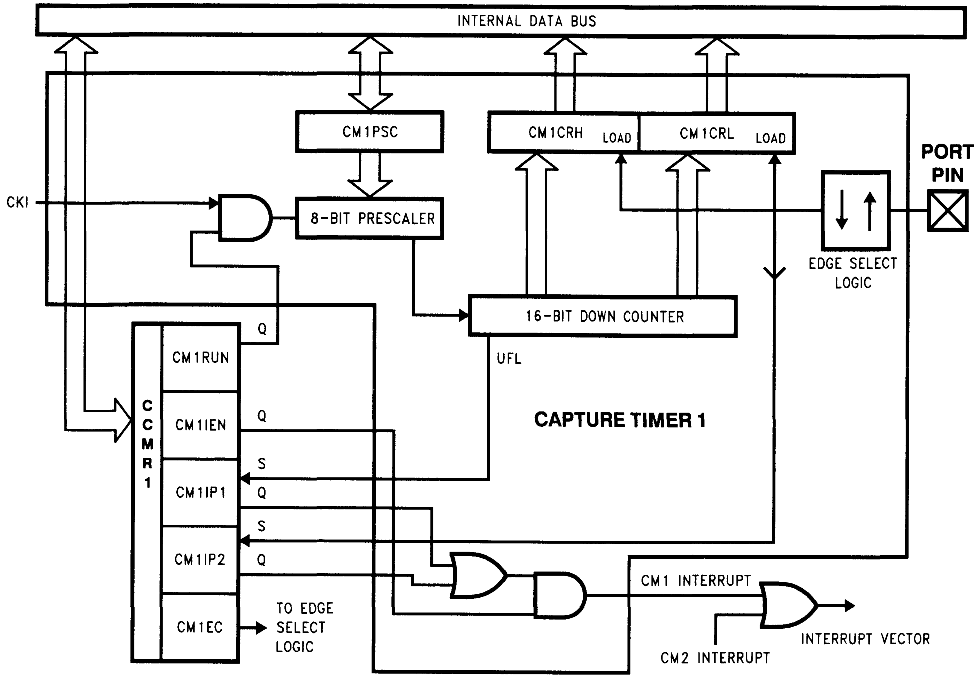
This device contains two independent capture timers, Capture Timer 1 and Capture Timer 2. Each capture timer contains an 8-bit programmable prescaler register, a 16-bit down counter, a 16-bit input capture register, and capture edge select logic. The 16-bit down counter is clocked at a specific frequency 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 TxB 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/DD12065-11

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	unused	unused	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:

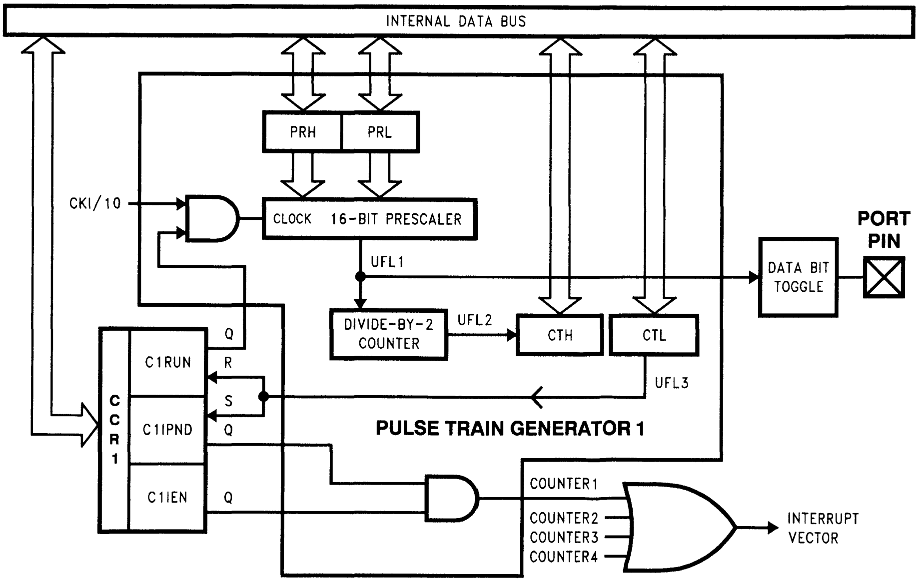
1. Reset the CMxRUN bit
2. Configure the corresponding Port bits as inputs
3. Set the edge control bits CMxEC
4. Reset CMxIP1 (CMxIP1 = 0)
5. Reset CMxIP2 (CMxIP2 = 0)
6. Load the 8-bit prescaler register CMxPSC with the desired value (from 0 to 255)
7. Set CMxIEN (if interrupts are to be used)
8. Set 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.



TL/DD12065-12

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:

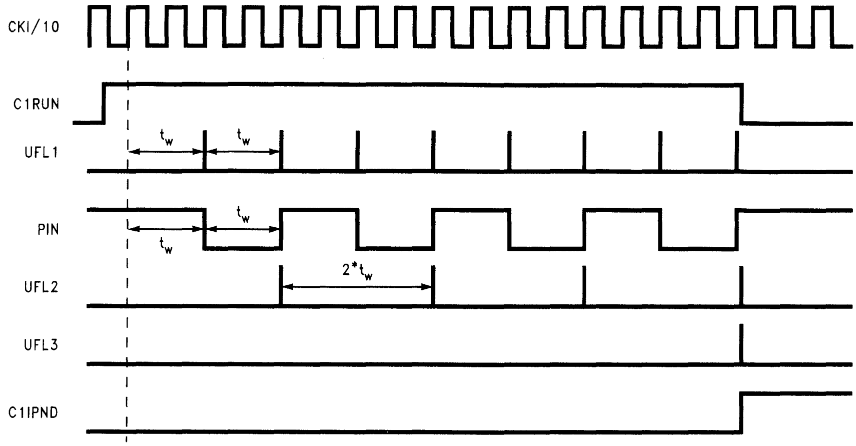
1. Load PRL register
2. Load PRH register
3. Load CTL register
4. Load CTH register
5. Reset CxIPND bit
6. Set CxIEN (if interrupt is to be used)
7. Configure the associated port bit as an output (if OUT is to be used)
8. Set the Global Interrupt Enable (GIE) bit (if interrupt is to be used)
9. Set CxRUN 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.

Figure 13 contains the block diagram of the multiply/divide block. It shows the registers contained within the multiply/divide block.

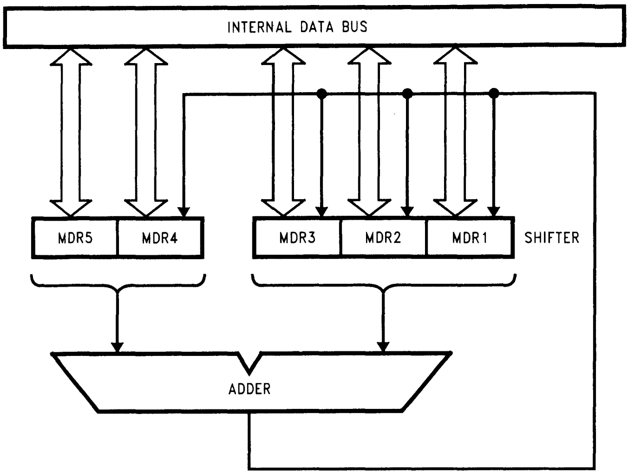
The registers shown in the block diagram are assigned according to Table III.



TL/DD12065-13

FIGURE 12. Timing Diagram for PRL = 1, PRH = 0, CTL = 3, CTH = 0

Multiply/Divide (Continued)



TL/DD12065-14

FIGURE 13. Multiply/Divide Block Diagram

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 Ihe 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 14 shows the Multi-Input Wake Up logic.

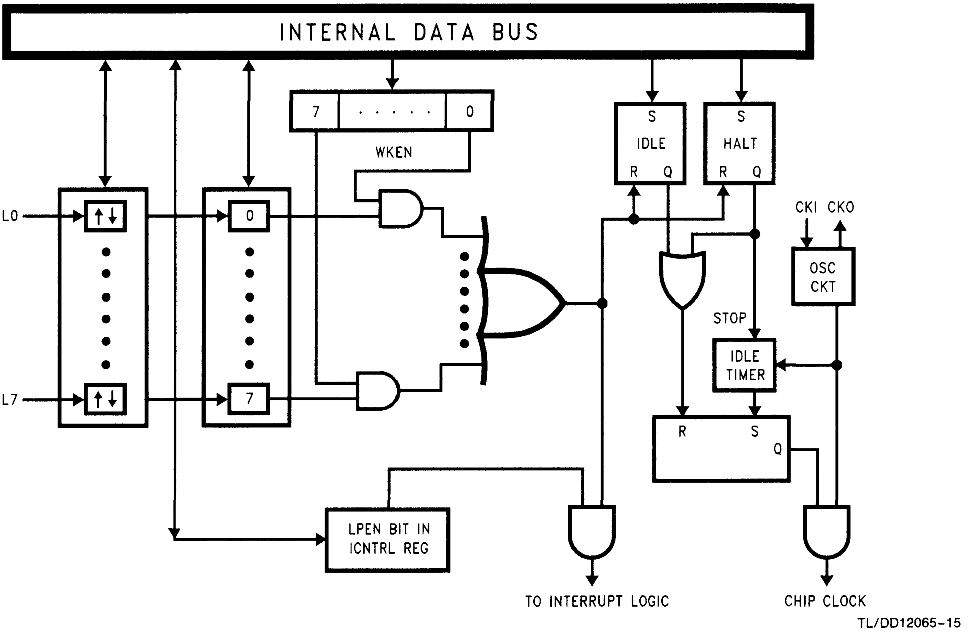


FIGURE 14. Multi-Input Wake Up Logic

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.

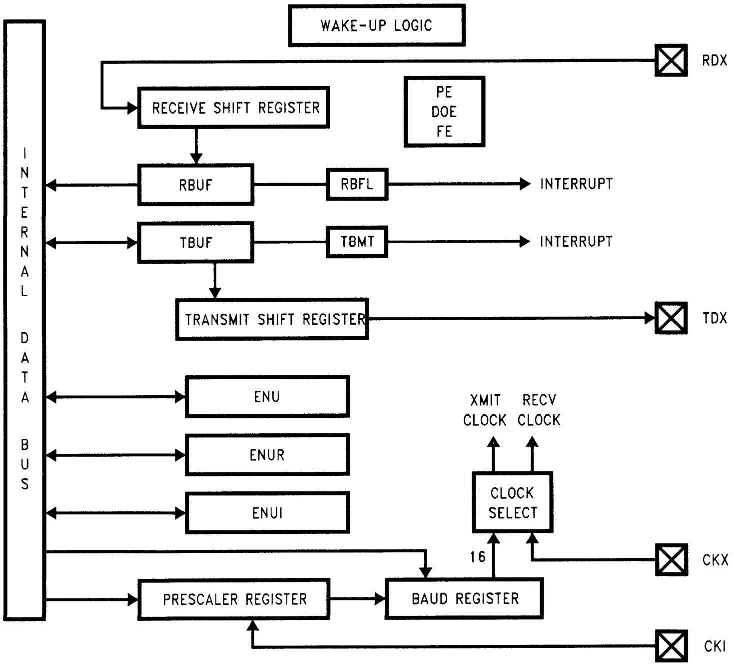
A control flag, LPEN, functions as a global interrupt enable for Port L interrupts. Setting the LPEN flag will enable interrupts and vice versa. A separate global pending flag is not needed since the register WKPND is adequate.

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 15*) 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/DD12065-16

FIGURE 15. 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 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

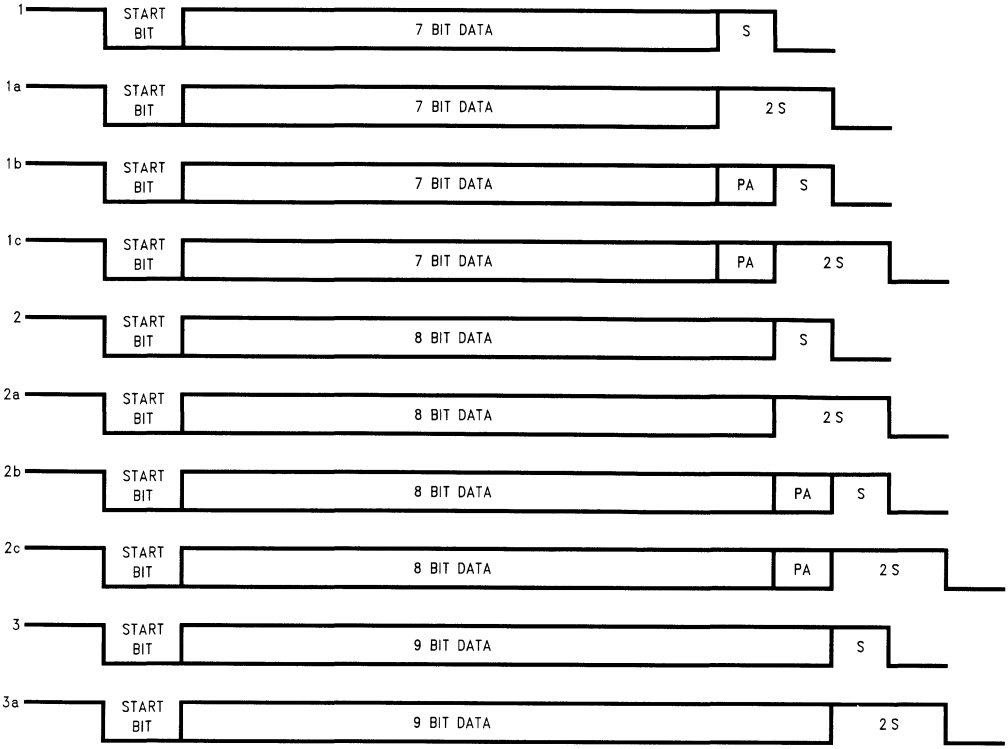
The UART supports several serial framing formats *(Figure 16)*. The format is selected using control bits in the ENU, ENUR and ENUI registers.

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/DD12065-17

FIGURE 16. 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.

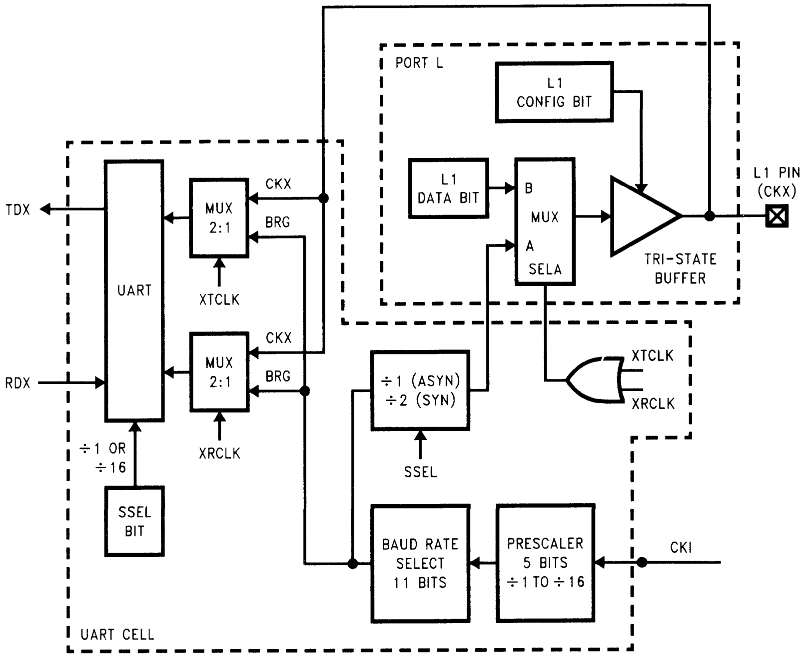
The interrupt from the Transmitter is set pending, and remains pending, as long as both the TBMT and ETI bits are set. To remove this interrupt, software must either clear the ETI bit or write to the TBUF register (thus clearing the TBMT bit).

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 17)*.The divide factors are specified through two read/write registers shown in *Figure 18*. 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/DD12065-18

FIGURE 17. UART BAUD Clock Generation

PRESCALER SELECT REGISTER (PSR)					BAUD REGISTER										
4	3	2	1	0	10	9	8	7	6	5	4	3	2	1	0
PRESCALER SELECT					BAUD RATE DIVISOR										

TL/DD12065-19

FIGURE 18. 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.

$N - 1 = 5$ ($N - 1$ is the value from Table IV)

$N = 6$ (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:

$$BR = Fc/(16 \times N \times P)$$

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 $N \times P$ 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 ($P = 6.5$).

$$N = 32.552/6.5 = 5.008 \ (N = 5)$$

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 "0".)

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 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. This vector can point to the Software Trap (ST) interrupt service routine, or to another special service routine as desired.

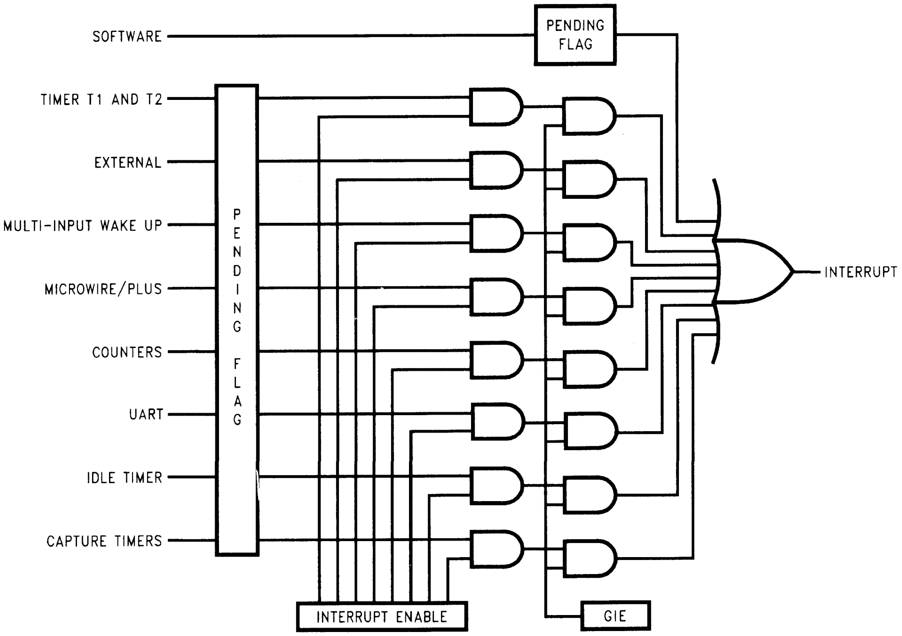
Figure 19 shows the Interrupt block diagram.

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.



TL/DD12065–20

FIGURE 19. Interrupt Block Diagram

Interrupts (Continued)

The ST has the highest rank among all interrupts.

Nothing (except another ST) can interrupt an ST being serviced.

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 20* shows a block diagram of the MICROWIRE/PLUS logic.

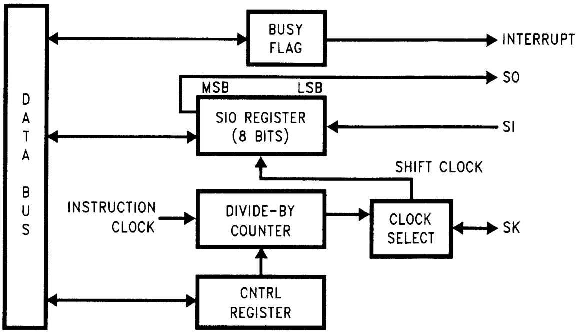
The shift clock can be selected from either an internal source or an external source. Operating the MICROWIRE/PLUS arrangement with the internal clock source is called the Master mode of operation. Similarly, operating the MICROWIRE/PLUS arrangement with an external shift clock is called the Slave mode 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/DD12065-21

FIGURE 20. 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 21* 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

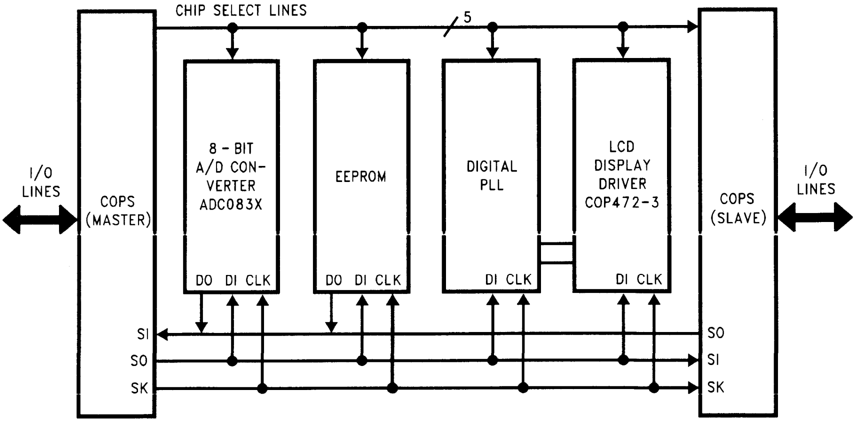
This table assumes that the control flag MSEL is set.

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/DD12065-22

FIGURE 21. 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 accumuiator 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 l 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 in the current 8k 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
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]
MemI	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 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.

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	Bits 3-0
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

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

OPTION 2: HALT

= 1 Enable HALT mode

= 2 Disable HALT mode

OPTION 3: BONDING OPTIONS

= 1 68 Pins PLCC

Development Support

IN-CIRCUIT EMULATOR

The MetaLink iceMASTER™-COP8 Model 400 In-Circuit Emulator for the COP8 family of microcontrollers features high-performance operation, ease of use, and an extremely flexible user-interface for maximum productivity. Interchangeable probe cards, which connect to the standard common base, support the various configurations and packages of the COP8 family.

The iceMASTER provides real-time, full-speed emulation up to 10 MHz, 32 kBytes of emulation memory and 4k frames of trace buffer memory. The user may define as many as 32k trace and break triggers which can be enabled, disabled, set or cleared. They can be simple triggers based on code or address ranges or complex triggers based on code address, direct address, opcode value, opcode class or immediate operand. Complex breakpoints can be ANDed and ORed together. Trace information consists of address bus values, opcodes and user-selectable probe clips status (external event lines). The trace buffer can be viewed as raw hex or as disassembled instructions. The probe clip bit values can be displayed in binary, hex or digital waveform formats.

During single-step operation the dynamically annotated code feature displays the contents of all accessed (read and write) memory locations and registers, as well as flow-of-control direction change markers next to each instruction executed.

The iceMASTER's performance analyzer offers a resolution of better than 6 μs. The user can easily monitor the time spent executing specific portions of code and find "hot spots" or "dead code". Up to 15 independent memory areas based on code address or label ranges can be defined. Analysis results can be viewed in bar graph format or as actual frequency count.

Emulator memory operations for program memory include single line assembler, disassembler, view, change and write to file. Data memory operations include fill, move, compare, dump to file, examine and modify. The contents of any memory space can be directly viewed and modified from the corresponding window.

The iceMASTER comes with an easy to use windowed interface. Each window can be sized, highlighted, color-controlled, added, or removed completely. Commands can be accessed via pull-down-menus and/or redefinable hot keys. A context sensitive hypertext/hyperlinked on-line help system explains clearly the options the user has from within any window.

The iceMASTER connects easily to a PCRM via the standard COMM port and its 115.2 kBaud serial link keeps typical program download time to under 3 seconds.

The following tables list the emulator and probe cards ordering information.

Emulator Ordering Information

Part Number	Description	Current Version
IM-COP8/400/1‡	MetaLink base unit in-circuit emulator for all COP8 devices, symbolic debugger software and RS232 serial interface cable, with 110V @ 60 Hz Power Supply.	Host Software: Ver 3.3 Rev. 5, Model File Rev 3.050.
IM-COP8/400/2‡	MetaLink base unit in-circuit emulator for all COP8 devices, symbolic debugger software and RS232 serial interface cable, with 220V @ 50 Hz Power Supply.	

‡These parts include National's COP8 Assembler/Linker/Librarian Package (COP8-DEV-IBMA).

Probe Card Ordering Information

Part Number	Package	Voltage Range	Emulates
MHW-888GW68PWPC	68 PLCC	2.5V–6.0V	COP888GW

MACRO CROSS ASSEMBLER

National Semiconductor offers a COP8 macro cross assembler. It runs on industry standard compatible PCs and supports all of the full-symbolic debugging features of the MetaLink iceMASTER emulators.

Development Support (Continued)

Assembler Ordering Information

Part Number	Description	Manual
COP8-DEV-IBMA	COP8 Assembler/ Linker/Librarian for IBM® PC/XT®, AT® or compatible.	424410632-001

DIAL-A-HELPER

Dial-A-Helper is a service provided by the Microcontroller Applications group. The Dial-A-Helper is an Electronic Bulletin Board Information System.

INFORMATION SYSTEM

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 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. The minimum requirement for accessing the Dial-A-Helper is a Hayes compatible modem.

If the user has a PC with a communications package then files from the FILE SECTION can be down loaded to disk for later use.

Order P/N: MDS-DIAL-A-HLP

Information System Package Contains
Dial-A-Helper User's Manual
Public Domain Communications Software

Factory Applications Support

Dial-A-Helper also provides immediate factor applications support. If a user has questions, he can leave messages on our electronic bulletin board, which we will respond to.

Voice:	(800) 272-9959	
Modem:	CANADA/US.:	(800) NSC-MICRO (800) 672-6427
	Baud:	14.4k
	Set-Up:	Length: 8-Bit Parity: None Stop Bit 1
	Operation:	24 Hours, 7 Days

National Semiconductor

COP8780C/COP8781C/COP8782C Single-Chip EPROM/OTP Microcontrollers

General Description

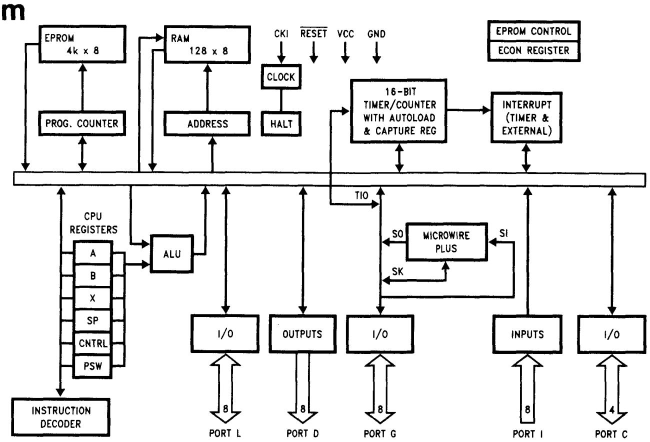
The COP8780C, COP8781C and COP8782C are members of the COPS™ 8-bit microcontroller family. They are fully static microcontrollers, fabricated using double-metal, double poly silicon gate microCMOS EPROM technology. These devices are available as UV erasable or One Time Programmable (OTP). These low cost 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.

Features

- Low cost 8-bit microcontroller
- Fully static CMOS
- 4096 x 8 on-chip UV erasable or OTP EPROM
- EPROM security
- 128 or 64 bytes of on-chip RAM, user configurable
- Crystal, RC or External Oscillator, user configurable
- 1 μs instruction time (10 MHz clock)
- Low current drain
- Extra-low current static HALT mode
- Single supply operation: 4.5V to 6.0V
- 8-bit stack pointer (stack in RAM)
- 16-bit read/write timer operates in a variety of modes
  - — PWM (Pulse Width Modulation) mode with 16-bit autoreload register
  - — External Event Counter mode, with selectable edge
  - — Input Capture mode (selectable edge) with 16-bit capture register
- Multi-source interrupt
  - — External interrupt with selectable edge
  - — Timer interrupt or capture interrupt
  - — Software interrupt
- Powerful instruction set, with most instructions single byte
- Many single byte, single cycle instructions
- BCD arithmetic instructions
- MICROWIRE/PLUS serial I/O
- Software selectable I/O options (TRI-STATE, push-pull, weak pull-up)
- Temperature ranges: −40°C to +85°C
- Schmitt trigger inputs on G port
- COP8780C EPROM Programming fully supported by different sources
- 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

Block Diagram



TL/DD/11299–1

FIGURE 1

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)	$V_{CC} = 6V$, $t_c = 1\ \mu s$			21	mA
HALT Current (Note 3)	$V_{CC} = 6V$, CKI = 0 MHz			10	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	$V_{CC} = 6.0V$	−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 $\overline{\text{RESET}}$ 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.

Note 6: Parameter characterized but not tested.

COP8780C/COP8781C/COP8782C

AC Electrical Characteristics

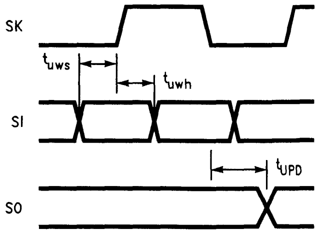
$-40°C < T_A < +85°C$ unless otherwise specified

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.

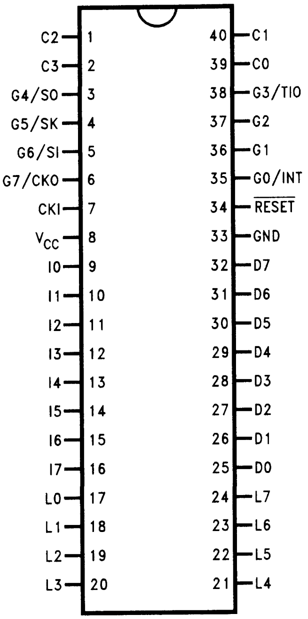
Timing Diagram



TL/DD/10802-2

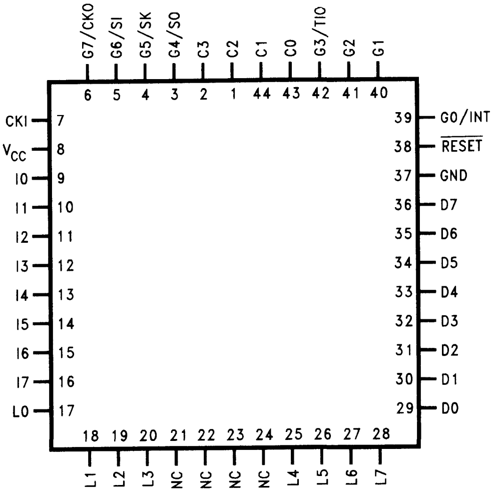
FIGURE 2. MICROWIRE/PLUS Timing

Connection Diagrams



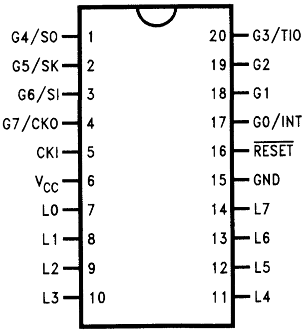
TL/DD/11299-3

Top View
COP8780CN, COP8780CJ



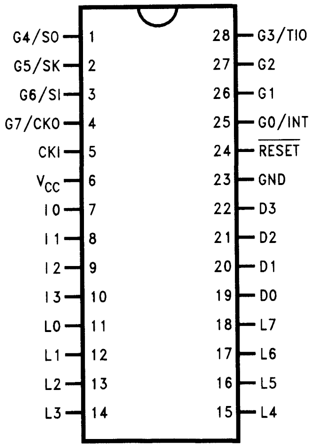
TL/DD/11299-4

Top View
COP8780CV, COP8780CEL



TL/DD/11299-5

Top View
COP8782CN, COP8782CWM,
COP8782CJ



TL/DD/11299-6

Top View
COP8781CN, COP8781CWM
COP8781CJ

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 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.

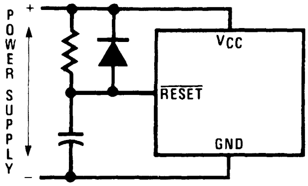
= 0 Selects 64 byte RAM option. This emulates COP820.

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.



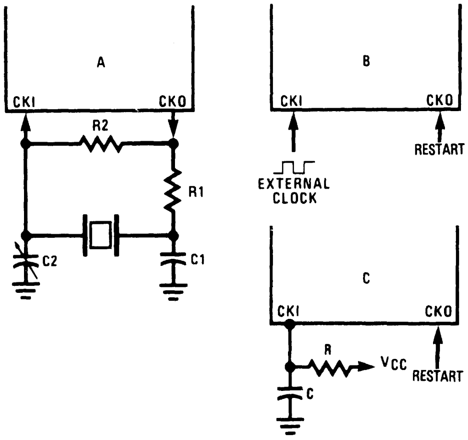
TL/DD/11299-7

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, $T_A = 25^\circ C$

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, $T_A = 25^\circ C$

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.

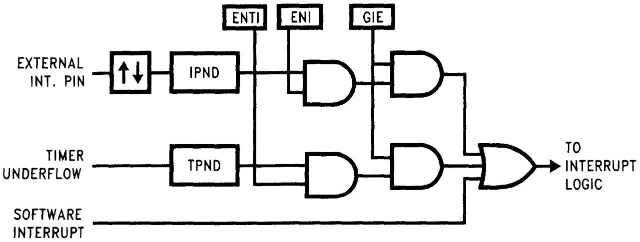
IPND and TPND bits signal which interrupt is pending. After an interrupt is acknowledged, the user can check these two bits to determine which interrupt is pending. This permits the interrupts to be prioritized under software. The pending flags have to be cleared by the user. Setting the GIE bit high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



TL/DD/11299-9

FIGURE 6. Interrupt Block Diagram

DETECTION OF ILLEGAL CONDITIONS

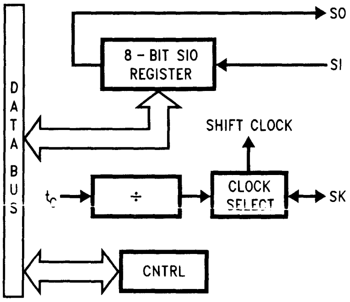
The device incorporates a hardware mechanism that allows it to detect illegal conditions which may occur from coding errors, noise and "brown out" voltage drop situations. Specifically, it detects cases of executing out of undefined EPROM area and unbalanced stack situations.

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)

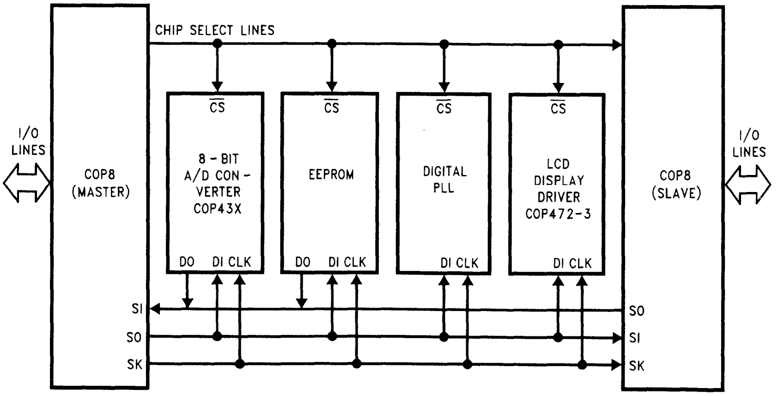


FIGURE 8. MICROWIRE/PLUS Application

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).*

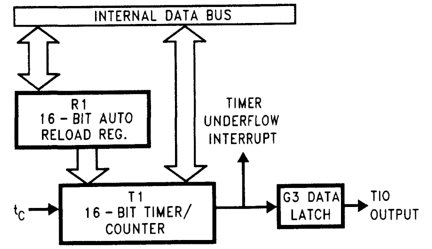
MODE 3. TIMER WITH CAPTURE REGISTER

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

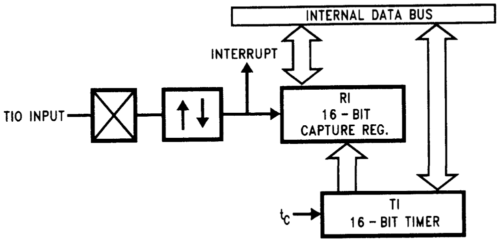
CNTRL Bits 7 6 5	Operation Mode	T Interrupt	Timer Counts On
0 0 0	External Counter w/Auto-Load Reg.	Timer Underflow	TIO Pos. Edge
0 0 1	External Counter w/Auto-Load Reg.	Timer Underflow	TIO Neg. Edge
0 1 0	Not Allowed	Not Allowed	Not Allowed
0 1 1	Not Allowed	Not Allowed	Not Allowed
1 0 0	Timer w/Auto-Load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

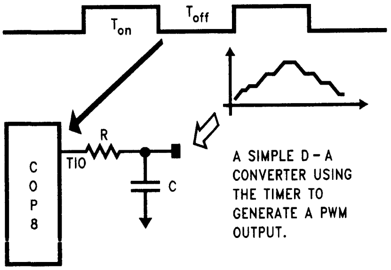


TL/DD/11299-13

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	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

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 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.

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 [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			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		

Programming Support

Programming of the single-chip emulator devices is supported by different sources. The following programmers are certified for programming the One-Time Programmable (OTP) and UV-erasable devices:

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 inhouse 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

EPROM Programmer Information

Manufacturer and Product	U.S. Phone Number	Europe Phone Number	Asia Phone Number
MetaLink- Debug Module	(602) 926-0797	Germany: +49-8141-1030	Hong Kong: 852-737-1800
Xeltek- Superpro	(408) 745-7974	Germany: +49-2041-684758	Singapore: +65-276-6433
BP Microsystems- EP-1140	(800) 225-2102	Germany: +49-89-857-6667	Hong Kong: +852-388-0629
Data I/O-Unisite; -System 29, -System 39	(800) 322-8246	Europe: +31-20-622866 Germany: +49-89-85-8020	Japan: +33-4326991
Abcom-COP8 Programmer		Europe: +89-808707	
System General Turpro-1-FX; -APRO	(408) 263-6667	Switzerland: +31-921-7844	Taiwan Taipei: +2-9173005

Programming Support (Continued)

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/cm²)	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.

Development Support

IN-CIRCUIT EMULATOR

The MetaLink iceMASTER™-COP8 Model 400 In-Circuit Emulator for the COP8 family of microcontrollers features high-performance operation, ease of use, and an extremely flexible user-interface for maximum productivity. Interchangeable probe cards, which connect to the standard common base, support the various configurations and packages of the COP8 family.

The iceMASTER provides real time, full speed emulation up to 10 MHz, 32 kbytes of emulation memory and 4k frames of trace buffer memory. The user may define as many as 32k trace and break triggers which can be enabled, disabled, set or cleared. They can be simple triggers based on code or address ranges, or complex triggers based on code address, direct address, opcode value, opcode class or immediate operand. Complex breakpoints can be ANDed and ORed together. Trace information consists of address bus values, opcodes, and user selectable probe clips status (external event lines). The trace buffer can be viewed as raw hex or as disassembled instructions. The probe clip bit values can be displayed in binary, hex or digital waveform formats.

During single-step operation the dynamically annotated code feature displays the contents of all accessed (read and write) memory locations and registers, as well as flow-of-control direction change markers next to each instruction executed.

The iceMASTER's performance analyzer offers a resolution of better than 6 μs. The user can easily monitor the time spent executing specific portions of code and find "hot spots" or "dead code". Up to 15 independent memory areas based on code address or label ranges can be defined. Analysis results can be viewed in bar graph format or as actual frequency count.

Emulator memory operations for program memory include single line assembler, disassembler, view, change and write to file. Data memory operations include fill, move, compare, dump to file, examine and modify. The contents of any memory space can be directly viewed and modified from the corresponding window.

The iceMASTER comes with an easy-to-use windowed interface. Each window can be sized, highlighted. color-controlled, added, or removed completely. Commands can be accessed via pull-down-menus and/or redefinable hot keys. A context sensitive hypertext/hyperlinked on-line help system explains clearly the options the user has from within any window.

The iceMASTER connects easily to a PC via the standard COMM port and its 115.2 kBaud serial link keeps typical program download to under 3 seconds.

The following tables list the emulator and probe cards ordering information.

Emulator Ordering Information

Part Number	Description	Current Version
IM-COP8/400/1‡	MetaLink base unit in-circuit emulator for all COP8 devices, symbolic debugger software and RS232 serial interface cable, with 110V @ 60 Hz Power Supply.	Host Software: Ver 3.3 Rev. 5, Model File Rev 3.050.
IM-COP8/400/2‡	MetaLink base unit in-circuit emulator for all COP8 devices, symbolic debugger software and RS232 serial interface cable, with 220V @ 50Hz Power Supply.	
DM-COP8/880C‡	MetaLink iceMASTER Debug Module. This is the low cost version of the MetaLink's iceMASTER. Firmware: Ver. 6.07	

‡These parts include National's COP8 Assembler/Linker/Librarian Package (COP8-DEV-IBMA).

Development Support (Continued)

Probe Card Ordering Information

Part Number	Package	Voltage Range	Emulator
MHW-880C28D5PC	28 DIP	4.5V–5.5V	COP820C, 840C, 881C, 8781C
MHW-880C28DWPC	28 DIP	2.5V–6.0V	COP820C, 840C, 881C, 8781C
MHW-880C40D5PC	40 DIP	4.5V–5.5V	COP880C, 8780C
MHW-880C40DWPC	40 DIP	2.5V–6.0V	COP880C, 8780C
MHW-880C44D5PC	44 PLCC	4.5V–5.5V	COP880C, 8780C
MHW-880C44DWPC	44 PLCC	2.5V–6.0V	COP880C, 8780C

MACRO CROSS ASSEMBLER

National Semiconductor offers a COP8 macro cross assembler. It runs on industry standard compatible PCs and supports all of the full symbolic debugging features of the MetaLink iceMASTER emulators.

Assembler Ordering Information

Part Number	Description	Manual
COP8-DEV-IBMA	COP8 Assembler/ Linker/Librarian for IBM® PC/XT®, AT® or compatible.	424410632-001

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

DIAL-A-HELPER

Dial-A-Helper is a service provided by the Microcontroller Applications Group. The Dial-A-Helper is an Electronic Bulletin Board information system.

INFORMATION SYSTEM

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 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. The minimum requirement for accessing the Dial-A-Helper is a Hayes compatible modem.

If the user has a PC with a communications package then files from the FILE SECTION can be down-loaded to disk for later use.

FACTORY APPLICATIONS SUPPORT

Dial-A-Helper also provides immediate factory applications support. If a user has questions, he can leave messages on our electronic bulletin board.

Voice:	(800) 272-9959	
Modem:	CANADA/U.S.:	(800) NSC-MICRO (800) 672-6427
	Baud:	14.4k
	Setup:	Length: 8-Bit Parity: None Stop Bit: 1
	Operation:	24 Hrs. 7 Days

National Semiconductor

COP8640CMH/COP8642CMH Microcontroller Emulator

General Description

The COP8640CMH/COP8642CMH hybrid emulators are members of the COPS™ microcontroller family. The devices (offered in 28-pin DIP LCC and 20-pin DIP) contain transparent windows which allow the EPROM to be erased and reprogrammed. They are fully static parts, fabricated using double-metal silicon gate microCMOS technology. These microcontrollers are complete microcomputers containing all system timing, interrupt logic, EPROM, RAM, EEPROM, 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 COP8640CMH/COP8642CMH to the specific application. The part operates over a voltage range of 4.5V to 6.0V. High throughput is achieved with an efficient, regular instruction set operating at a 1 microsecond per instruction rate.

COP8640CMH and COP8642CMH are intended primarily as a prototyping design tool. The Electrical Performance Characteristics are not tested but are included for reference only.

Features

- Form fit and function emulation devices for COP8640C/COP8642C/COP8620C/COP8622C
- Fully static CMOS
- 1 µs instruction time
- Single supply operation: 4.5V to 6.0V
- 8k bytes EPROM/64 bytes RAM/64 bytes EEPROM
- 16-Bit read/write timer operates in a variety of modes
  - — Timer with 16-bit auto reload register
  - — 16-bit external event counter
  - — Timer with 16-bit capture register (selectable edge)
- Multi-source interrupt
  - — Reset master clear
  - — External interrupt with selectable edge
  - — Timer interrupt or capture interrupt
  - — Software interrupt
- 8-bit stack pointer (stack in RAM)
- Powerful instruction set, most instructions single byte
- BCD arithmetic instructions
- MICROWIRE/PLUS serial I/O
- 28-pin and 20-pin DIP packages
- 24 input/output pins (28-pin package)
- Software selectable I/O options (TRI-STATE®, push-pull, weal pull-up)
- Schmitt trigger inputs on Port G
- Fully supported by National's Development Systems

Ordering Information

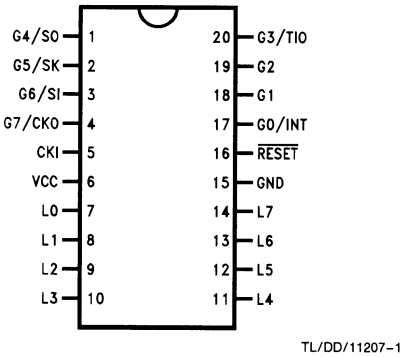
Hybrid Emulator	Package Type	Part Emulated
COP8640CMHD-x	28-DIP	COP8640C-XXX/N COP8620C-XXX/N
COP8642CMHD-x	20-DIP	COP8642C-XXX/N COP8622C-XXX/N

x = 1, 2, 3 corresponds to oscillator option.

Connection Diagrams

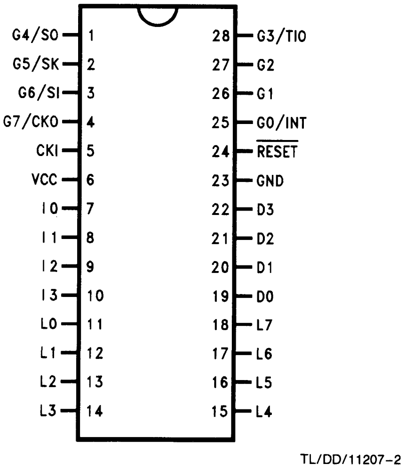
DUAL-IN-LINE PACKAGES

20-Pin DIP



Top View

28-Pin DIP



Top View

FIGURE 1. COP8640CMH/COP8642CMH Connection Diagrams

COP8640CMH/COP8642CMH Pinouts

Port	Type	Alternate Function	20-Pin DIP	28-Pin DIP/LCC
L0	I/O		7	11
L1	I/O		8	12
L2	I/O		9	13
L3	I/O		10	14
L4	I/O		11	15
L5	I/O		12	16
L6	I/O		13	17
L7	I/O		14	18
G0	I/O	Interrupt	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	Halt Restart	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 (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)	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.

The following AC and DC Electrical Characteristics are not tested but are for reference only.

DC Electrical Characteristics $0°C \le T_A \le +70°C$ 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	$V_{CC} = 6V, t_c = 1\ \mu s$			19	mA
Supply Current during Write Operation (Note 2)					
CKI = 10 MHz	$V_{CC} = 6V, t_c = 1\ \mu s$			25	mA
HALT Current (Note 3)	$V_{CC} = 6V$, CKI = 0 MHz		500		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	$V_{CC} = 6.0V$	−2		+2	μA
Input Pullup Curent	$V_{CC} = 6.0V$	40		250	μA
G Port Input Hysteresis			$0.05\ V_{CC}$		V
Outut 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 (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

COP8640CMH/COP8642CMH (Continued)

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

Parameter	Condition	Min	Typ	Max	Units
EEPROM Characteristics					
EEPROM Write Cycle Time				10	ms
EEPROM Number of Write Cycles				10,000	Cycle
EEPROM Data Retention				10	Years

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 at TRI-STATE and tied to ground, all outputs low and tied to ground.

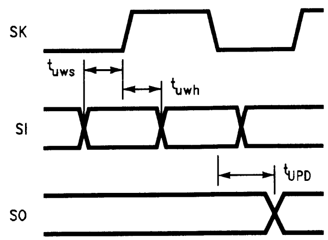
Note 4: Pins G6 and $\overline{\text{RESET}}$ are designed with a high voltage input network for factory testing. These pins allow input voltages greater than V_{CC} and the pins will have sink current to V_{CC} when biased at voltages greater than V_{CC} (the pins do not have source current when biased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

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

Parameter	Condition	Min	Typ	Max	Units
Instruction Cycle Time (t_c)					
Ext, Crystal/Resonator (Div-by 10)		1		DC	μs
R/C Oscillator Mode (Div-by 10)		3		DC	μs
CKI Clock Duty Cycle (Note 4)		40		60	%
Rise Time (Note 4)	fr = 10 MHz Ext Clock			12	ns
Fall Time (Note 4)	fr = 10 MHz Ext Clock			8	ns
Inputs					
t_{SETUP}		200			ns
t_{HOLD}		60			ns
Output Propagation Delay	$C_L = 100$ pF, $R_L = 2.2$ kΩ				
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 Time (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 4: Parameter sampled but not 100% tested.

Timing Diagram



TL/DD/11207-3

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 inupt. 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 Inupt (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.

Functional Description

OSCILLATOR CIRCUITS

Figure 3 shows the three clock oscillator configurations. Table III shows the clock options per package.

A. CRYSTAL OSCILLATOR

The COP8640CMH/COP8642CMH can be driven by a crystal clock. The crystal network is cnonected 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. CKI 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 (due to the part) as functions of the R/C component values (R/C tolerances not included).

TABLE I. Crystal Oscillator Configuration
T_A = 25°C, V_{CC} = 5.0V

R1 (kΩ)	R2 (MΩ)	C1 (pF)	C2 (pF)	CKI Freq (MHz)
0	1	30	30–36	10
0	1	30	30–36	4
5.5	1	100	100	0.455

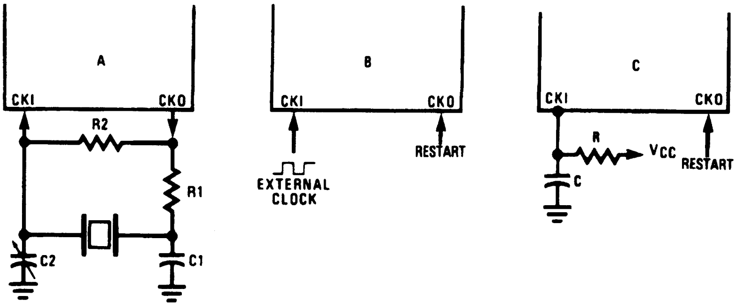
TABLE II. RC Oscillator Configuration
T_A = 25°C, V_{CC} = 5.0V

R (kΩ)	C (pF)	CKI Freq. (MHz)	Instr. 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 9.0
6.8	100	0.9 to 1.1	8.8 to 10.8

Note: $3k \leq R \leq 200k$

$50\ pF \leq C \leq 200\ pF$

Functional Description (Continued)



TL/DD/11207-4

FIGURE 3. Crystal and R-C Connection Diagrams

TABLE III. Clock Option per Package

Order Part Number	Package	Clock Option
COP8640CMHD-1 COP8642CMHD-1	28 DIP 20 DIP	Crystal Oscillator ÷ 10
COP8640CMHD-2 COP8642CMHD-2	28 DIP 20 DIP	External Oscillator ÷ 10
COP8640CMHD-3 COP8642CMHD-3	28 DIP 20 DIP	R/C Oscillator ÷ 10

Programming the COP8640CMH/COP8642CMH

Programming the hybrid emulators is accomplished through the duplicator board which is a stand alone programmer capable of supporting different package types. It works in conjunction with a pre-programmed EPROM (either via the NSC development system or a standard programmer) holding the application program. The duplicator board essentially copies the information in the EPROM into the hybrid emulator.

The last byte of program memory (EPROM location 01FFF Hex) must contain the proper value specified in the following table:

TABLE IV

Device	Package Type	Contents of Last Byte (Address 01FFF)
COP8640CMHD	28 DIP	6F
COP8642CMHD	20 DIP	E7

ERASING THE PROGRAM MEMORY

Erasure of the EPROM program memory is achieved by removing the device from its socket and exposing the transparent window to an ultra-violet light source.

The erasure characteristics of the device are such that 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, opaque labels should be placed over the window of the device to prevent temporary functional failure due to the generation of photo currents, erasure, and excessive HALT current. 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 devices 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 15 W-sec/cm^2.

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.

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:

TABLE V. Minimum Erasure Time

Package Type	Light Intensity (Micro-Watts/cm^2)	Erasure Time (Minutes)
28 DIP	15,000	20
	10,000	25
	5,000	50
20 DIP	15,000	40
	10,000	50
	5,000	100

Development Support

IN-CIRCUIT EMULATOR

The MetaLink iceMASTER™-COP8 Model 400 In-Circuit Emulator for the COP8 family of microcontrollers features high-performance operation, ease of use, and an extremely flexible user-interface for maximum productivity. Interchangeable probe cards, which connect to the standard common base, support the various configurations and packages of the COP8 family.

The iceMASTER provides real time, full speed emulation up to 10 MHz, 32 kbytes of emulation memory and 4k frames of trace buffer memory. The user may define as many as 32k trace and break triggers which can be enabled, disabled, set or cleared. They can be simple triggers based on code or address ranges or complex triggers based on code address, direct address, opcode value, opcode class or immediate operand. Complex breakpoints can be ANDed and ORed together. Trace information consists of address bus values, opcodes and user selectable probe clips status (external event lines). The trace buffer can be viewed as raw hex or as disassembled instructions. The probe clip bit values can be displayed in binary, hex or digital waveform formats.

During single-step operation the dynamically annotated code feature displays the contents of all accessed (read and write) memory locations and registers, as well as flow-of-control direction change markers next to each instruction executed.

The iceMASTER's performance analyzer offers a resolution of better than 6 μs. The user can easily monitor the time spent executing specific portions of code and find "hot spots" or "dead code". Up to 15 independent memory areas based on code address or label ranges can be defined. Analysis results can be viewed in bar graph format or as actual frequency count.

Emulator memory operations for program memory include single line assembler, disassembler, view, change and write to file. Data memory operations include fill, move, compare, dump to file, examine and modify. The contents of any memory space can be directly viewed and modified from the corresponding window.

The iceMASTER comes with an easy to use windowed interface. Each window can be sized, highlighted, color-controlled, added, or removed completely. Commands can be accessed via pull-down-menus and/or redefineable hot keys. A context sensitive hypertext/hyperlinked on-line help system explains clearly the options the user has from within any window.

The iceMASTER connects easily to a PC® via the standard COMM port and its 115.2 kBaud serial link keeps typical program download time to under 3 seconds.

The following tables list the emulator and probe cards ordering information.

Emulator Ordering Information

Part Number	Description
IM-COP8/400	MetaLink base unit in-circuit emulator for all COP8 devices, symbolic debugger software and RS 232 serial interface cable
MHW-PS3	Power Supply 110V/60 Hz
MHW-PS4	Power Supply 220V/50 Hz

Probe Card Ordering Information

Part Number	Package	Voltage Range	Emulates
MHW-8640C20D5PC	20 DIP	4.5V–5.5V	COP8642C, 8622C
MHW-8640C20DWPC	20 DIP	2.5V–6.0V	COP8642C, 8622C
MHW-8640CG28D5PC	28 DIP	4.5V–5.5V	COP8640C, 8620C
MHW-8640CG28DWPC	28 DIP	2.5V–6.0V	COP8640C, 8620C

Development Support (Continued)

MACRO CROSS ASSEMBLER

National Semiconductor offers a COP8 macro cross assembler. It runs on industry standard compatible PCs and supports all of the full-symbolic debugging features of the MetaLink iceMASTER emulators.

SIMULATOR

The COP8 Designers' Toolkit is available for evaluating National Semiconductor's COP8 microcontroller family. The kit contains programmer's manuals, device datasheets, pocket reference guides, assembler and simulator which allow the user to write, test, debug and run code on an industry standard compatible PC. The simulator has a windowed user interface and can handle script files that simulate hardware inputs, interrupts and automatic command processing. The capture file feature enables the user to record to a file current cycle count and output port changes which are caused by the program under test.

SINGLE CHIP EMULATOR DEVICE

The COP8 family is fully supported by single chip form, fit and function emulators. For more detailed information refer to the emulation device specific data sheets and the form, fit, function emulator selection table below.

PROGRAMMING SUPPORT

Programming of the single chip emulator devices is supported by different sources. National Semiconductor offers a duplicator board which allows the transfer of program code from a standard programmed EPROM to the single chip emulator and vice versa. Data I/O supports COP8 emulator device programming with its uniSite 48 and System 2900 programmers. Further information on Data I/O programmers can be obtained from any Data I/O sales office or the following USA numbers:

Telephone: (206) 881-6444 FAX: (206) 882-1043

Assembler Ordering Information

Part Number	Description	Manual
MOLE-COP8-IBM	COP8 Macro Cross Assembler for IBM® PC-XT®, PC-AT® or Compatible	424410527-001

Simulator Ordering Information

Part Number	Description	Manual
COP8-TOOL-KIT	COP8 Designer's Tool Kit Assembler and Simulator	420420270-001 424420269-001

Single Chip Emulator Selection Table

Device Number	Clock Option	Package	Description	Emulates
COP8640CMHD-X	X = 1 : Crystal X = 2 : External X = 3 : R/C	28 DIP	Multi-Chip Module (MCM), UV Erasable	COP8640C, 8620C
COP8640CMHEA-X	X = 1 : Crystal X = 2 : External X = 3 : R/C	28 LCC	MCM (Same Footprint as 28 SO), UV Erasable	COP8640C, 8620C
COP8642CMHD-X	X = 1 : Crystal X = 2 : External X = 3 : R/C	20 DIP	MCM, UV Erasable	COP8642C, 8622C

Duplicator Board Ordering Information

Part Number	Description	Devices Supported
COP8-PRGM-28D	Duplicator Board for 28 DIP and for use with Scrambler Boards	COP8640CMHD
COP8-SCRM-DIP	Scrambler Board for 20 DIP Socket	COP8642CMHD
COP8-SCRM-SBX	Scrambler Board for 28 LCC Socket	COP8640CMHEA
COP8-PRGM-DIP	Duplicator Board with COP8-SCRM-DIP Scrambler Board	COP8642CMHD, COP8640CMHD

Development Support (Continued)

DIAL-A-HELPER

Dial-A-Helper is a service provided by the Microcontroller Applications group. The Dial-A-Helper is an Electronic Bulletin Board Information system.

INFORMATION SYSTEM

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 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. The minimum requirement for accessing the Dial-A-Helper is a Hayes compatible modem.

If the user has a PC with a communications package then files from the FILE SECTION can be down loaded to disk for later use.

ORDER P/N: MOLE-DIAL-A-HLP

Information System Package contains:
- Dial-A-Helper Users Manual
- Public Domain Communications Software

FACTORY APPLICATIONS SUPPORT

Dial-A-Helper also provides immediate factor applications support. If a user has questions, he can leave messages on our electronic bulletin board, which we will respond to.

Voice:	(408) 721-5582		
Modem:	(408) 739-1162		
	Baud:	300 or 1200 Baud	
	Set-up:	Length:	8-Bit
		Parity:	None
		Stop Bit:	1
	Operation:	24 Hrs., 7 Days	

PRELIMINARY

COP8788CL/COP8784CL microCMOS One-Time Programmable (OTP) Microcontrollers

General Description

The COP8788CL/COP8784CL programmable microcontrollers are members of the COPS™ microcontroller family. Each device is a two chip system in a plastic package. Within the package is the COP888CL and a 8k EPROM with port recreation logic. The code executes out of the EPROM. These devices are offered in four packages: 44-pin PLCC, 40-pin DIP, 28-pin DIP and 28-pin SO.

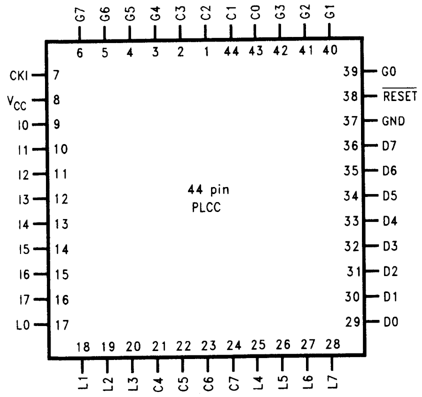
The COP8788CL/COP8784CL are fully static parts, fabricated using double-metal silicon gate microCMOS technology. Features include an 8-bit memory mapped architecture, MICROWIRE/PLUS™ serial I/O, two 16-bit timer/counters supporting three modes (Processor Independent PWM generation, External Event counter, and Input Capture mode capabilities). Each I/O pin has software selectable configurations. The devices operates over a voltage range of 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.

Features

- Low cost 8-bit microcontroller
- Fully static CMOS, with low current drain
- 1 μs instruction cycle time
- 8192 bytes on-board EPROM
- 128 bytes on-board RAM
- Single supply operation: 4.5V–5.5V
- MICROWIRE/PLUS serial I/O
- WATCHDOG™ and Clock Monitor logic
- Idle timer
- Multi-Input Wakeup (MIWU) with optional interrupts (8)
- 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
- Two 16-bit timers, each with two 16-bit registers supporting:
  - — Processor independent PWM mode
  - — External event counter mode
  - — Input capture mode
- 8-bit Stack Pointer SP (stack in RAM)
- Two 8-bit register indirect data memory pointers (B and X)
- Versatile instruction set with True bit manipulation
- Memory mapped I/O
- BCD arithmetic instructions
- Package:
  - — 44 PLCC with 39 I/O pins
  - — 40 DIP with 33 I/O pins
  - — 28 DIP with 23 I/O pins
  - — 28 SO with 23 I/O pins (contact local sales office for availability)
- 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
- Form fit and function emulation device for the COP888CL/COP884CL
- Real time emulation and full program debug offered by Metalink's Development Systems

Connection Diagrams

Plastic Chip Carrier

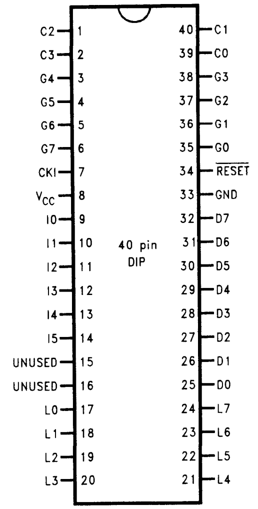


TL/DD12063-1

Top View

Order Number COP8788CLV-X, COP8788CLFV-R
See NS Package Number V44A

Dual-In-Line Package

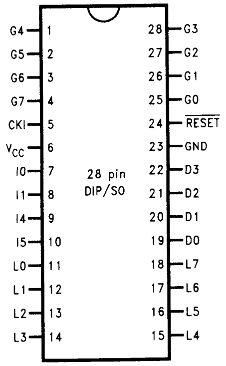


TL/DD12063-2

Top View

Order Number COP8788CLN-X, COP8788CLN-R
See NS Package Number N40A

Dual-In-Line Package



TL/DD12063-3

Top View

Order Number COP8784CLN-X, COP8784CLN-R,
COP8784CLWM-X and COP8784CLWM-R
See NS Package Number M28B or N28B

FIGURE 1. COP8788CL/COP8784CL 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	ALE	25	35	39
G1	WDOUT			26	36	40
G2	I/O	T1B	$\overline{WR}$	27	37	41
G3	I/O	T1A	$\overline{RD}$	28	38	42
G4	I/O	SO		1	3	3
G5	I/O	SK		2	4	4
G6	I	SI	ME	3	5	5
G7	I/CKO	Halt Restart		4	6	6
D0	O		AD0	19	25	29
D1	O		AD1	20	26	30
D2	O		AD2	21	27	31
D3	O		AD3	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		AD4		29	33
D5	O		AD5		30	34
D6	O		AD6		31	35
D7	O		AD7		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
$\overline{RESET}$			V_{PP}	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 $-40°C \leq T_A \leq +85°C$ 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)				25	mA
CKI = 10 MHz	$V_{CC} = 5.5V$, $t_c = 1\ \mu s$				
HALT Current (Note 3)	$V_{CC} = 5.5V$, CKI = 0 MHz		250		μA
IDLE Current, CKI = 10 MHz	$V_{CC} = 5.5V$, $t_c = 1\ \mu s$			15	mA
Input Levels					V
$\overline{\text{RESET}}$					
Logic High		$0.8\ V_{CC}$			
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	$V_{CC} = 4.5V$, $V_{OL} = 1V$	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 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

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.

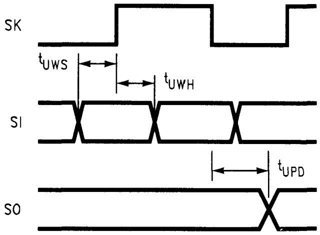
Note 3: The HALT mode will stop CKI from oscillating in the RC and the Crystal configurations. Test conditions: All inputs tied to V_{CC}, L and G ports in the TRI-STATE mode and tied to ground, all outputs low and tied to ground. The clock monitor is disabled.

Note 4: Pins G5 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 $-40°C \le T_A \le +85°C$ 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	
CKI Clock Duty Cycle (Note 5)	f_r = Max	40		60	%
Rise Time (Note 5)	f_r = 10 MHz Ext Clock			5	ns
Fall Time (Note 5)	f_r = 10 MHz Ext Clock			5	ns
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	$4V \le V_{CC} \le 6V$			0.7	
All Others	$4V \le V_{CC} \le 6V$			1	
MICROWIRE™ Setup Time (t_{UWS})		20			ns
MICROWIRE Hold Time (t_{UWH})		56			
MICROWIRE Output Propagation Delay (t_{UPD})				220	
Input Pulse Width					
Interrupt Input High Time		1			t_c
Interrupt Input Low Time		1			
Timer Input High Time		1			
Timer Input Low Time		1			
Reset Pulse Width		1			µs

Note 5: Parameter sampled (not 100% tested).



TL/DD12063-4

FIGURE 2. MICROWIRE/PLUS Timing

Pin Descriptions

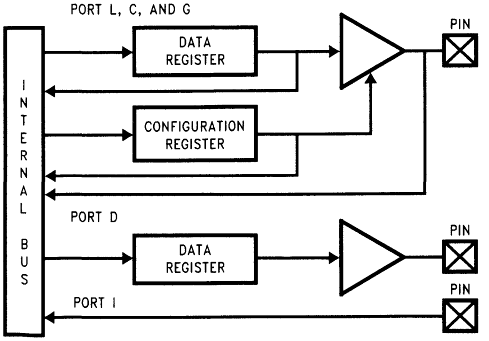
V_{CC} and GND are the power supply pins.

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/DD12063-5

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 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 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 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

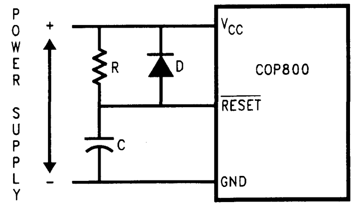
The $\overline{\text{RESET}}$ input when pulled low initializes the microcontroller. Initialization will occur whenever the $\overline{\text{RESET}}$ input is pulled low. Upon initialization, the data and configuration registers for Ports L, G, and C are cleared, resulting in these Ports being initialized 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 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.

Note: In continual state of reset, the device will draw excessive current.

Reset (Continued)



TL/DD12063-6

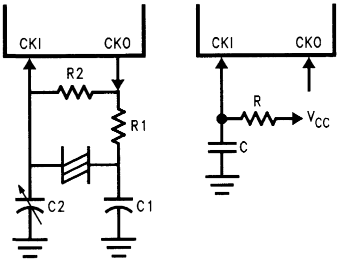
RC > 5 × Power Supply Rise Time

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/DD12063-7

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: $3k \leq R \leq 200k$, $50\ pF \leq C \leq 200\ pF$

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. Clock Monitor current when enabled—I6

Thus the total current drain, It, is given as

$$It = I1 + I2 + I3 + I4 + I5 + I6$$

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, 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

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 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

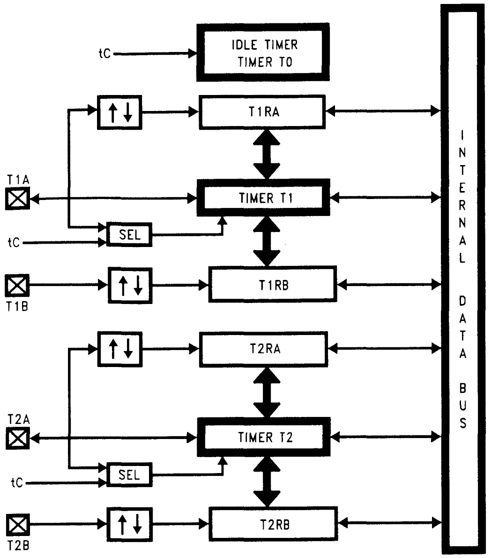
T2C3	T2C2	T2C1	T2C0	T2PNDA	T2ENA	T2PNDB	T2ENB

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.

Figure 6 shows a block diagram for the timers.



TL/DD12063-8

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 ($t_C = 1\ \mu s$). 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

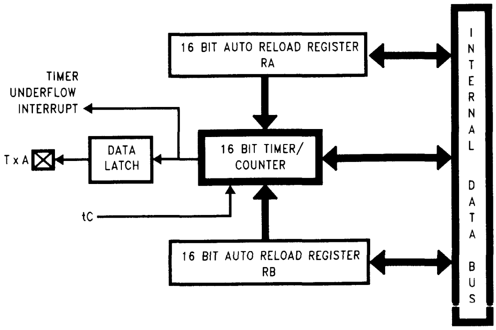
As the name suggests, this mode allows the device to generate a PWM signal with very minimal user intervention.

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.



TL/DD12063-9

FIGURE 7. Timer in PWM Mode

Timers (Continued)

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.

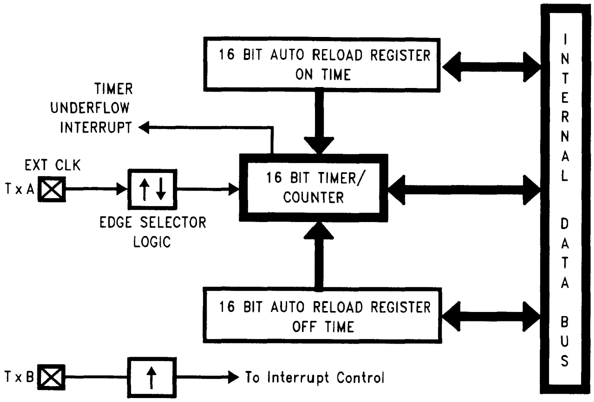
Mode 2. External Event Counter Mode

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.



TL/DD12063-10

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.

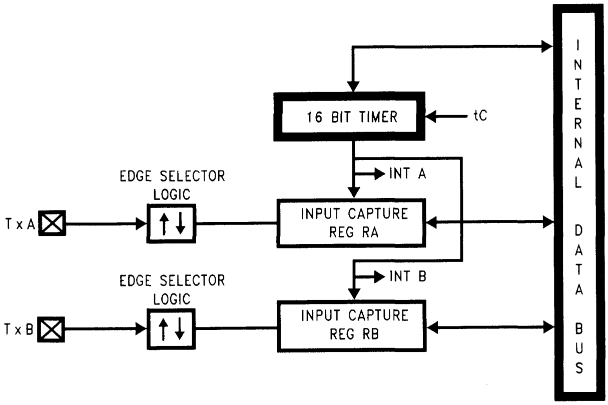
Underflows from the timer can also be programmed to generate interrupts. Underflows are latched into the timer TxC0 pending flag (the TxC0 control bit serves as the timer underflow interrupt pending flag in the Input Capture mode). Consequently, the TxC0 control bit should be reset when entering the 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 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/DD12063-11

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 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, $t_C = 1\ \mu s$) 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.

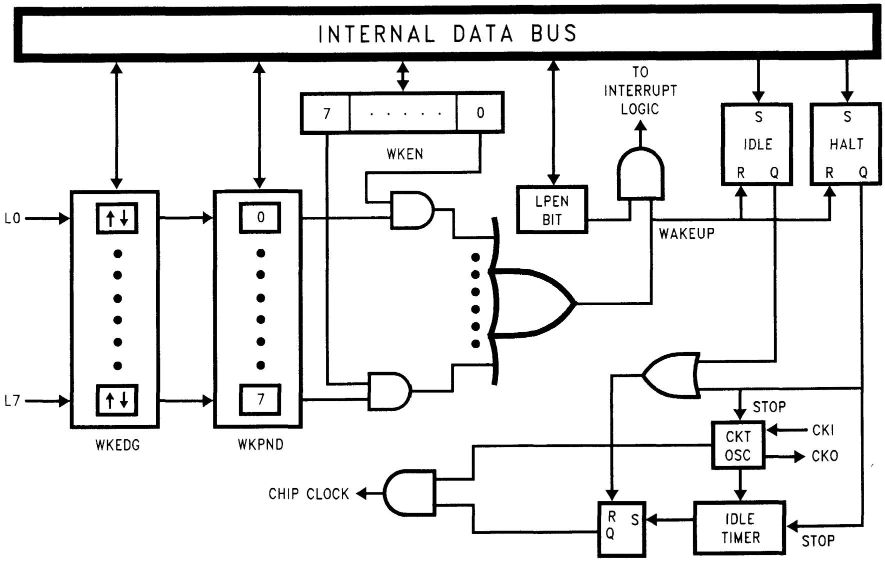
Alternatively, the user can enter the IDLE mode with the IDLE Timer T0 interrupt disabled. In this case, the device will resume normal operation with the instruction immediately following the "Enter IDLE Mode" instruction.

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 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 10 shows the Multi-Input Wakeup logic.



TL/DD12063-12

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:

```
RMRBIT  5, WKEN
RMSBIT  5, WKEDG
RMRBIT  5, WKPND
RMSBIT  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
	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.

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.

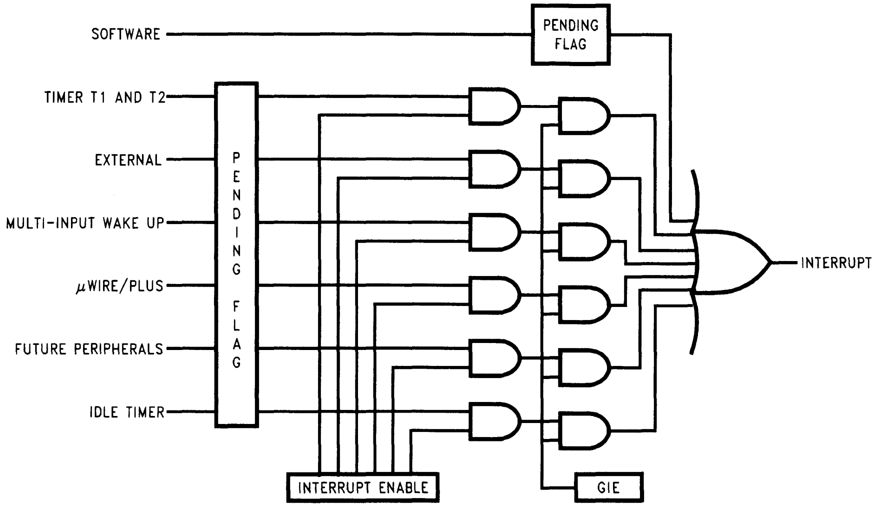
VIS and the vector table must be located in the same 256-byte block (0y00 to 0yFF) except if VIS is located at the last address of a block. In this case, the table must be in the next block. The vector table cannot be inserted in the first 256-byte block.

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. This vector can point to the Software Trap (ST) interrupt service routine, or to another special service routine as desired.

Figure 11 shows the Interrupt block diagram.



TL/DD12063-13

FIGURE 11. 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:

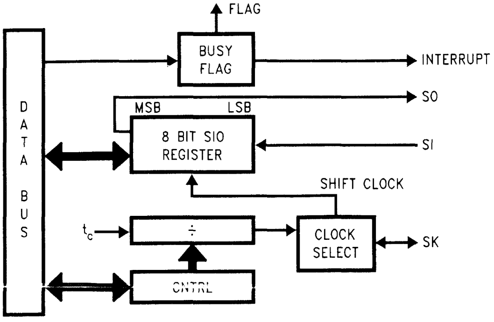
1. Executing from undefined ROM
2. Over "POP"ing the stack by having more returns than calls.

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 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/PLUS arrangement with an external shift clock is called the Slave mode of operation.



TL/DD12063-14

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 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 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 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 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.

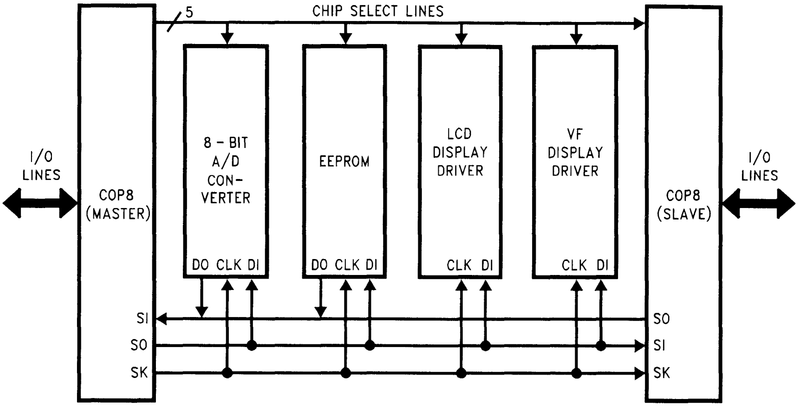
A control flag, SKSEL, allows either the normal SK clock or the alternate SK clock to be selected. Resetting SKSEL causes the MICROWIRE/PLUS logic to be clocked from the normal SK signal. Setting the SKSEL flag selects the alternate SK clock. The SKSEL is mapped into the G6 configuration bit. The SKSEL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

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/DD1206315

FIGURE 13. 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 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]
MemI	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,MemI	ADD	A ← A + MemI
ADC	A,MemI	ADD with Carry	A ← A + MemI + C, C ← Carry, HC ← Half Carry
SUBC	A,MemI	Subtract with Carry	A ← A − MemI + C, C ← Carry, HC ← Half Carry
AND	A,MemI	Logical AND	A ← A and MemI
ANDSZ	A,Imm	Logical AND Immed., Skip if Zero	Skip next if (A and Imm) = 0
OR	A,MemI	Logical OR	A ← A or MemI
XOR	A,MemI	Logical EXclusive OR	A ← A xor MemI
IFEQ	MD,Imm	IF EQual	Compare MD and Imm, Do next if MD = Imm
IFEQ	A,MemI	IF EQual	Compare A and MemI, Do next if A = MemI
IFNE	A,MemI	IF Not Equal	Compare A and MemI, Do next if A ≠ MemI
IFGT	A,MemI	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	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,MemI	LoaD A with Memory	A ← MemI
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.

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.

COP8788CL/COP8784CL 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]	IFBITL 0,[B]	ANDSZ A, #i	LD B,#0F	IFBNE 0	JSR x000–x0FF	JMP x000–x0FF	JP +17	JP − 15	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 − 14	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 − 13	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 − 12	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 − 11	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 − 10	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 − 9	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 − 7	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 − 6	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 − 5	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 − 4	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 − 3	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 − 2	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 − 1	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 − 0	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.

Ordering Information and Development Support

COP8788CL/CIP8784CL Ordering Information

Device Number	Clock Option	Package	Emulates
COP8788CLV-X COP8788CLV-R*	Crystal R/C	44 PLCC	COP888CL
COP8788CLN-X COP8788CLN-R*	Crystal R/C	40 DIP	COP888CL
COP8784CLN-X COP8784CLN-R*	Crystal R/C	28 DIP	COP884CL
COP8784CLWM-X* COP8784CLWM-R*	Crystal R/C	28 SO	COP884CL

*Check with the local sales office about the availability.

PROGRAMMING SUPPORT

Programming of these emulator devices is supported by different sources. The following programmers are certified for programming these One-Time Programmable emulator devices:

EPROM Programmer Information

Manufacturer and Product	U.S. Phone Number	Europe Phone Number	Asia Phone Number
Metalink-Debug Module	(602)926-0797	Germany: +49-8141-1030	Hong Kong: 852-737-1800
Xeltek-Superpro	(408)745-7974	Germany: +49-20-41-684758	Singapore: 65-276-6433
BP Microsystems-Turpro	(800)225-2102	Germany: +49-89-85-76667	Hong Kong: 852-388-0629
Data I/O-Unisite - System 29 - System 39	(800)322-8246	Europe: +31-20-622866 Germany: +49-89-85-8020	Japan: +33-432-6991
Abcom-COP8 Programmer		Europe: +89-808707	
System General-Turpro-1-FX -APRO	(408)263-6667	Switzerland: +31-921-7844	Taiwan: +2-917-3005

IN-CIRCUIT EMULATOR

The MetaLink iceMASTER™-COP8 Model 400 In-Circuit Emulator for the COP8 family of microcontrollers features high-performance operation, ease of use, and an extremely flexible user-interface for maximum productivity. Interchangeable probe cards, which connect to the standard common base, support the various configurations and packages of the COP8 family.

The iceMASTER provides real-time, full-speed emulation up to 10 MHz, 32 kBytes of emulation memory and 4k frames of trace buffer memory. The user may define as many as 32k trace and break triggers which can be enabled, disabled, set or cleared. They can be simple triggers based on code address, direct address, opcode value, opcode class or immediate operand. Complex breakpoints can be ANDed and ORed together. Trace information consists of address bus values, opcodes and user-selectable probe clips status (external event lines). The trace buffer can be viewed as raw hex or as disassembled instructions. The probe clip bit values can be displayed in binary, hex or digital waveform formats.

During single-step operation the dynamically annotated code feature displays the contents of all accessed (read and write) memory locations and registers, as well as flow-of-control direction change markers next to each instruction executed.

The iceMASTER's performance analyzer offers a resolution of better than 6 μs. The user can easily monitor the time spent executing specific portions of code and find "hot spots" or "dead code". Up to 15 independent memory areas based on code address or label ranges can be defined. Analysis results can be viewed in bar graph format or as actual frequency count.

Emulator memory operations for program memory include single line assembler, disassembler, view, change and write to file. Data memory operations include fill, move, compare, dump to file, examine and modify. The contents of any memory space can be directly viewed and modified from the corresponding window.

The iceMASTER comes with an easy to use windowed interface. Each window can be sized, highlighted, color-controlled, added, or removed completely. Commands can be accessed via pull-down-menus and/or redefinable hot keys. A context sensitive hypertext/hyperlinked on-line help system explains clearly the options the user has from within any window.

The iceMASTER connects easily to a PCRM via the standard COMM port and its 115.2 kBaud serial link keeps typical program download time to under 3 seconds.

The following tables list the emulator and probe cards ordering information.

Emulator Ordering Information

Part Number	Description	Current Version
IM-COP8/400/1‡	MetaLink base unit in-circuit emulator for all COP8 devices, symbolic debugger software and RS 232 serial interface cable, with 110V @ 60 Hz Power Supply.	Host Software: Ver 3.3 Rev. 5, Model File Rev 3.050.
IM-COP8/400/2‡	MetaLink base unit in-circuit emulator for all COP8 devices, symbolic debugger software and RS 232 serial interface cable, with 220V @ 50 Hz Power Supply.	
DM-COP8/888CF‡	MetaLink iceMASTER Debug Modul. This is the low cost version of the MetaLink iceMASTER. Firmware: Ver. 6.07	

‡These parts include National's COP8 Assembler/Linker/Librarian Package (COP8-DEV-IBMA).

Probe Card Ordering Information

Part Number	Package	Voltage Range	Emulates
MHW-884CL28D5PC	28 DIP	4.5V–5.5V	COP884CL
MHW-884CL28DWPC	28 DIP	2.5V–6.0V	COP884CL
MHW-888CL40D5PC	40 DIP	4.5V–5.5V	COP888CL
MHW-888CL40DWPC	40 DIP	2.5V–6.0V	COP888CL
MHW-888CL44D5PC	44 PLCC	4.5V–5.5V	COP888CL
MHW-888CL44DWPC	44 PLCC	2.5V–6.0V	COP888CL

MACRO CROSS ASSEMBLER

National Semiconductor offers a COP8 macro cross assembler. It runs on industry standard compatible PCs and supports all of the full-symbolic debugging features of the MetaLink iceMASTER emulators.

Assembler Ordering Information

Part Number	Description	Manual
COP8-DEV-IBMA	COP8 Assembler/ Linker/Librarian for IBM® PC/XT®, AT® or compatible.	424410632-001

DIAL-A-HELPER

Dial-A-Helper is a service provided by the Microcontroller Applications group. The Dial-A-Helper is an Electronic Bulletin Board Information System.

Information System

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 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. The minimum requirement for accessing the Dial-A-Helper is a Hayes compatible modem.

If the user has a PC with a communications package then files from the FILE SECTION can be down loaded to disk for later use.

Order P/N: MOLE-DIAL-A-HLP

Information System Package Contents
- Dial-A-Helper User Manual
- Public Domain Communications Software

Factory Applications Support

Dial-A-Helper also provides immediate factor applications support. If a user has questions, he can leave messages on our electronic bulletin board, which we will respond to.

Voice:	(800) 272-9959		
Modem:	CANADA/U.S.:	(800) NSC-MICRO	
		(800) 672-6427	
	Baud:	14.4k	
	Set-Up:	Length:	8-Bit
		Parity:	None
		Stop Bit 1	
	Operation:	24 Hours, 7 Days	

National Semiconductor

PRELIMINARY

COP8788CF/COP8784CF microCMOS One-Time Programmable (OTP) Microcontrollers

General Description

The COP8788CF/COP8784CF programmable microcontrollers are members of the COPS™ microcontroller family. Each device is a two chip system in a plastic package. Within the package is the COP888CF and an 8k EPROM with port recreation logic. The code executes out of the EPROM. The device is offered in four packages: 44-pin PLCC, 40-pin DIP, 28-pin DIP and 28-pin SO.

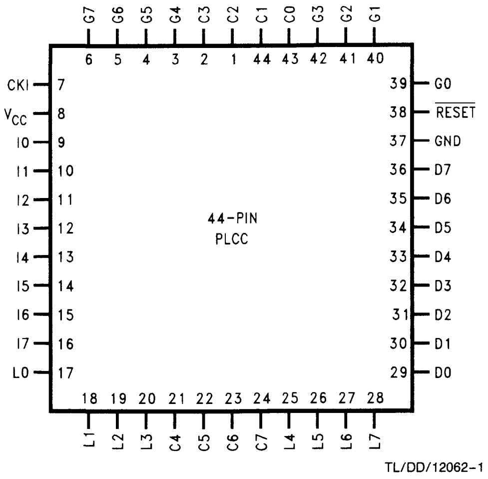
The device is a fully static part, fabricated using double-metal silicon gate microCMOS technology. Features include an 8-bit memory mapped architecture, MICROWIRE/PLUS™ serial I/O, two 16-bit timer/counters supporting three modes (Processor Independent PWM generation, External Event counter, and Input 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 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.

Features

- Low cost 8-bit microcontroller
- Fully static CMOS, with low current drain
- 1 μs instruction cycle time
- 8192 bytes on-board EPROM
- 128 bytes on-board RAM
- Single supply operation: 4.5V–5.5V
- 8-channel A/D converter with prescaler and both differential and single ended modes
- MICROWIRE/PLUS serial I/O
- WATCHDOG™ and Clock Monitor logic
- Idle Timer
- Multi-Input Wake Up (MIWU) with optional interrupts (8)
- 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
- Two 16-bit timers, each with two 16-bit registers supporting:
  - — Processor Independent PWM mode
  - — External Event counter mode
  - — Input Capture mode
- 8-bit Stack Pointer SP (stack in RAM)
- Two 8-bit Register Indirect Data Memory Pointers (B and X)
- Versatile instruction set with True bit manipulation
- Memory mapped I/O
- BCD arithmetic instructions
- Package:
  - — 44 PLCC with 37 I/O pins
  - — 40 DIP with 33 I/O pins
  - — 28 DIP with 21 I/Opins
  - — 28 SO with 21 I/O pins (contact local sales office for availability)
- 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
- Form fit and function emulation device for the COP888CF/COP884CF
- Real time emulation and full program debug offered by MetaLink's Development Systems

Connection Diagrams

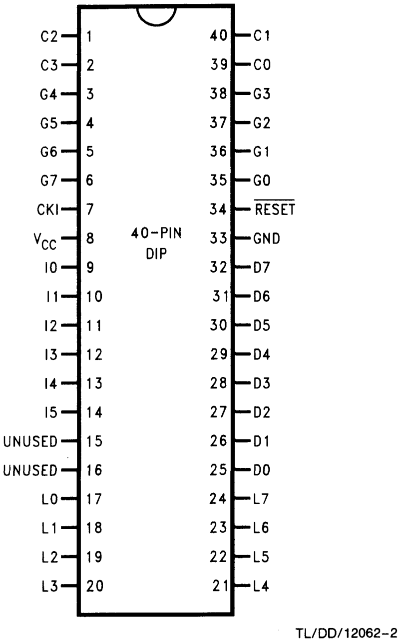
Plastic Chip Carrier



Top View

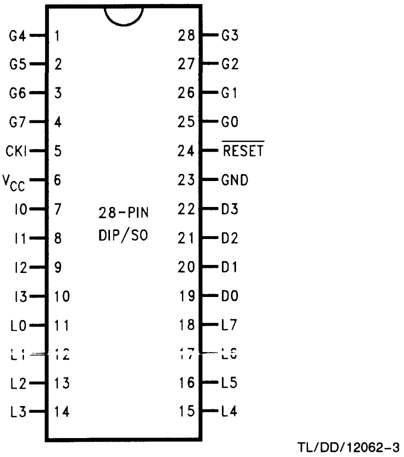
Order Number COP8788CFV-X or C0P8788CFV-R
See NS Package Number V44A

Dual-In-Line Package



Top View
Order Number COP8788CFN-X, COP8788CFN-R
See NS Package Number N40A

Dual-In-Line Package



Top View

Order Number COP8784CFN-X, COP8788CFN-R, COP8784CFWM-X or COP8784CFWM-R
See NS Package Number M28B or N28A

FIGURE 1. COP8788CF/COP8784CF 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	ALE	25	35	39
G1	WDOUT			26	36	40
G2	I/O	T1B	$\overline{WR}$	27	37	41
G3	I/O	T1A	$\overline{WD}$	28	38	42
G4	I/O	SO		1	3	3
G5	I/O	SK		2	4	4
G6	I	SI	ME	3	5	5
G7	I/CKO	HALT Restart		4	6	6
D0	O		AD0	19	25	29
D1	O		AD1	20	26	30
D2	O		AD2	21	27	31
D3	O		AD3	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		AD4		29	33
D5	O		AD5		30	34
D6	O		AD6		31	35
D7	O		AD7		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
$\overline{RESET}$			V_{PP}	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 $-40°C \leq T_A \leq +85°C$ 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	$V_{CC} = 5.5V$, $t_c = 1\ \mu s$			25	mA
HALT Current (Note 3)	$V_{CC} = 5.5V$, CKI = 0 MHz		250		μA
IDLE Current CKI = 10 MHz	$V_{CC} = 5.5V$, $t_c = 1\ \mu s$			15	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	$V_{CC} = 4.5V$, $V_{OL} = 1V$	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 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

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 A/D is disabled. V_{REF} is tied to AGND (effectively shorting the Reference resistor). The clock monitor is disabled.

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.

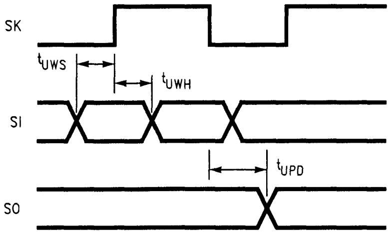
Note 6: Pins G6 and $\overline{\text{RESET}}$ are designed with a high voltage input network for factory testing. These pins allow input voltages greater than V_{CC} and the pins will have sink current to V_{CC} when biased at voltages greater than V_{CC} (the pins do not have source current when biased 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.

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
CKI Clock Duty Cycle (Note 8)	f_r = Max	40		60	%
Rise Time (Note 8)	f_r = 10 MHz Ext Clock			5	ns
Fall Time (Note 8)	f_r = 10 MHz Ext Clock			5	ns
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	$4V \le V_{CC} \le 6V$			0.7	μs
All Others	$4V \le V_{CC} \le 6V$			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

Note 8: Parameter sample (not 100% tested).



TL/DD/12062-4

FIGURE 2. MICROWIRE/PLUS Timing

Pin Descriptions

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

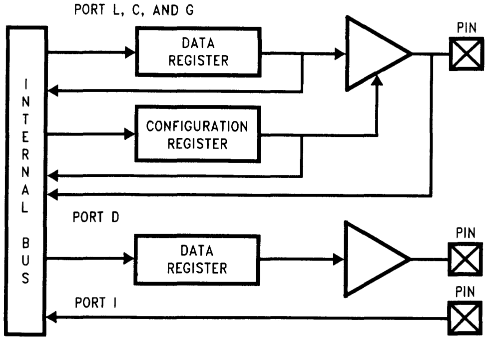
V_{REF} and AGND are the reference voltage pins for the on-board A/D converter.

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



TL/DD/12062-5

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. 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 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 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 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.

Reset

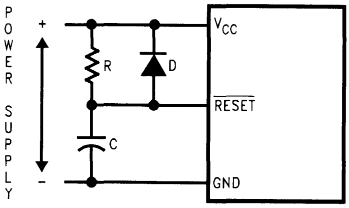
The $\overline{\text{RESET}}$ input when pulled low initializes the microcontroller. Initialization will occur whenever the $\overline{\text{RESET}}$ input is pulled low. Upon initialization, the data and configuration registers for Ports L, G, and C are cleared, resulting in these Ports being initialized 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 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.

Reset (Continued)



TL/DD/12062-6

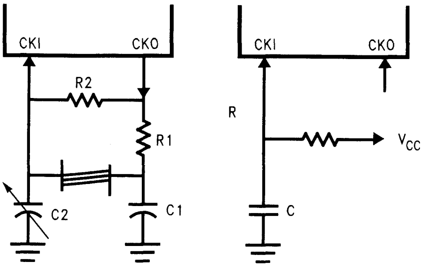
RC > 5 × Power Supply Rise Time

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/12062-7

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: $3k \le R \le 200k$

$50\ pF \le C \le 200\ pF$

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. DC reference current contribution from the A/D converter—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, 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

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.

Figure 6 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 ($t_C = 1s$). 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.

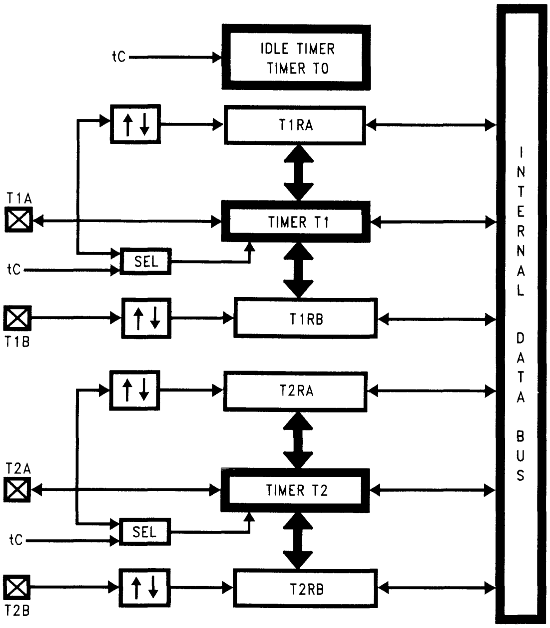
Mode 1. Processor Independent PWM Mode

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.

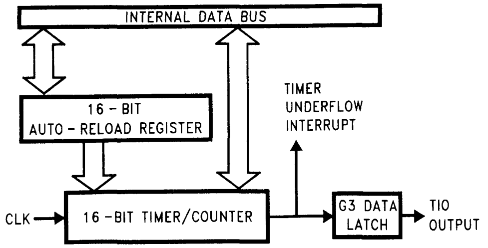


TL/DD/12062-8

FIGURE 6. Timers

Timers (Continued)

Figure 7 shows a block diagram of the timer in PWM mode.



TL/DD/12062-9

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.

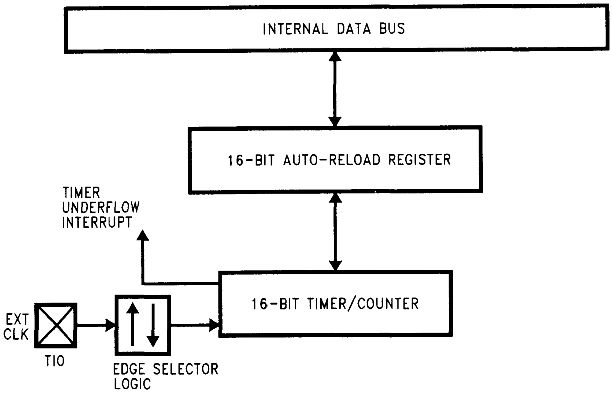
In this mode the input pin TxB can be used as an independent positive edge sensitive interrupt input if the TxENB control flag is set. The occurrence of a positive edge on the TxB input pin is latched into the TxPNDB flag.

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.

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.



TL/DD/12062-10

FIGURE 8. Timer in External Event Counter Mode

Timers (Continued)

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 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

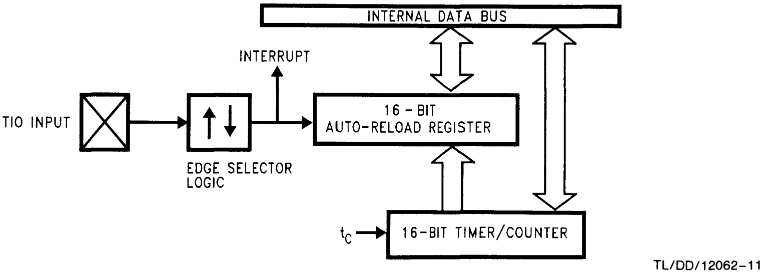


FIGURE 9. Timer in Input Capture Mode

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, 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.

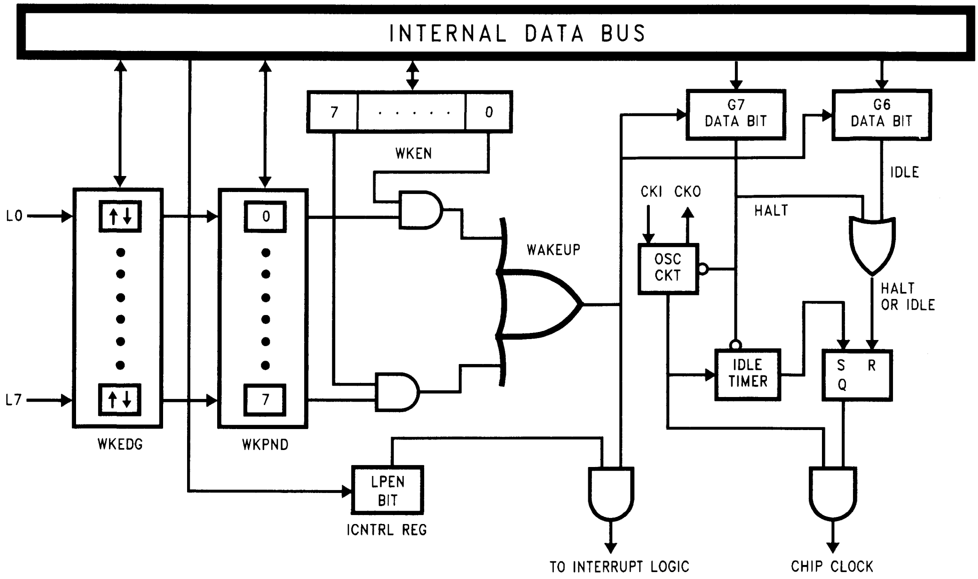
Multi-Input Wake Up

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 10 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/12062-12

FIGURE 10. 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:

```
RMRBIT 5, WKEN
RMSBIT 5, WKEDG
RMRBIT 5, WKPND
RMSBIT 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.

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.

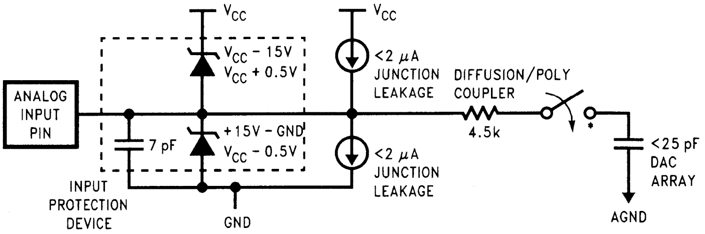
Note: The A/D converter is also powered down when the device is in either the HALT or IDLE modes. If the ADC is running when the device enters the HALT or IDLE modes, the ADC will power down during the HALT or IDLE, and then will reinitialize the conversion when the device comes out of the HALT or IDLE modes.

Analog Input and Source Resistance Considerations

Figure 11 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 11,* 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.



TL/DD/12062-13

*The analog switch is closed only during the sample time.

FIGURE 11. A/D Pin Model (Single Ended Mode)

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.

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
	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	0yE0–0yE1
		without Any Interrupts	

y is VIS page, y ≠ 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 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.

Figure 12 shows the device Interrupt block diagram.

SOFTWARE TRAP

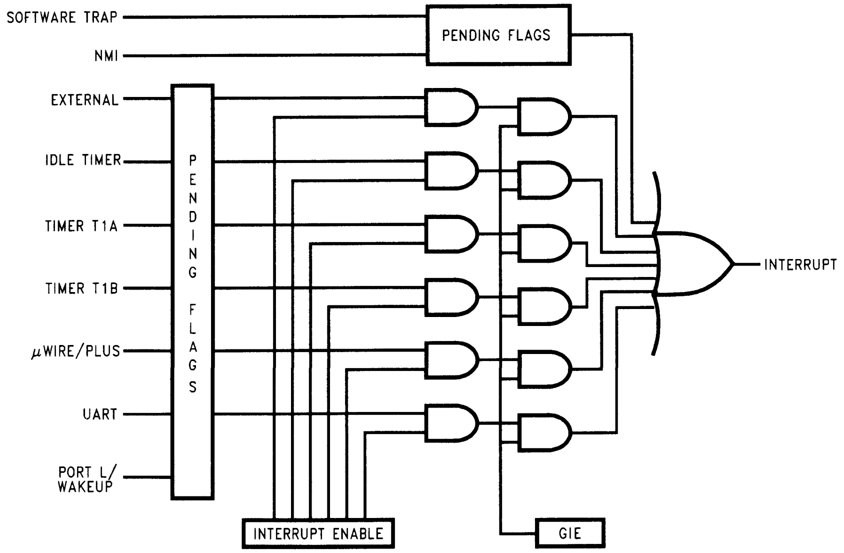
The Software Trap (ST) is a special kind of non-maskable interrupt which occurs when the INTR instruction (used to acknowledge interrupts) is fetched from ROM and placed inside the instruction register. This may happen when the PC is pointing beyond the available ROM address space or when the stack is 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/12062–14

FIGURE 12. 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.

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 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.

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:

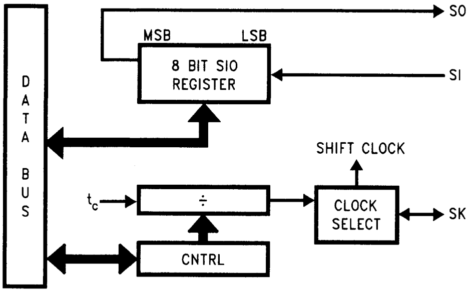
1. Executing from undefined ROM.
2. Over "POP"ing the stack by having more returns than calls.

Detection of Illegal Conditions
(Continued)

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 13* shows a block diagram of the MICROWIRE/PLUS logic.



TL/DD/12062-15

FIGURE 13. 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

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 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.

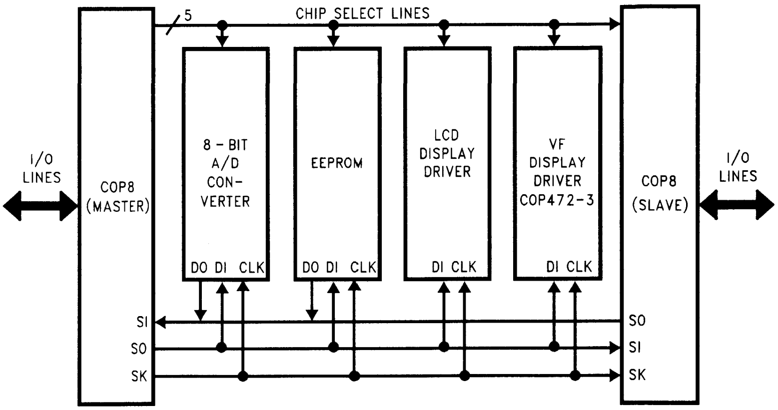
MICROWIRE/PLUS Master Mode Operation

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/12062-16

FIGURE 14. 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.

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 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]
MemI	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.

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	JP−15	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−14	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−13	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−12	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−11	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−10	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−9	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−7	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−6	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−5	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−4	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−3	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−2	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−1	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−0	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

Ordering and Development Support

COP8788CF1COP8784CF Ordering Information

Device Number	Clock Option	Package	Emulates
COP8788CFV-X COP8788CFV-R*	Crystal R/C	44 PLCC	COP888CF
COP8788CFN-X COP8788CFN-R*	Crystal R/C	40 DIP	COP888CF
COP8784CFN-X COP8784CFN-R*	Crystal R/C	28 DIP	COP884CF
COP8784CFWM-X* COP8784CFWM-R*	Crystal R/C	28 SO	COP884CF

*Check with the local sales office about the availability.

PROGRAMMING SUPPORT

Programming of these emulator devices is supported by different sources. The following programmers are certified for programming these One-Time Programmable emulator devices:

EPROM Programmer Information

Manufacturer and Product	U.S. Phone No.	Europe Phone No.	Asia Phone No.
MetaLink— Debug Module	(602) 926-0797	Germany: +49-8141-1030	Hong Kong: 852-737-1800
Xeltek— Superpro	(408) 745-7974	Germany: (49-20-41) 684758	Singapore: (65) 276-6433
BP Microsystems— Turpro	(800) 225-2102	Germany: (49-89-85) 76667	Hong Kong: (852) 388-0629
Data I/O—Unisite —System 29 —System 39	(800) 322-8246	Europe: +31-20-622866 Germany: +49-89-85-8020	Japan: +33-432-6991
Abcom—COP8 programmer		Europe: +89 808707	
System General— Turpro-1—FX —APRO	(408) 263-6667	Switzerland: +31-921-7844	Taiwan: +2-917-3005

Development System Support

IN-CIRCUIT EMULATOR

The MetaLink iceMASTER™-COP8 Model 400 In-Circuit Emulator for the COP8 family of microcontrollers features high-performance operation, ease of use, and an extremely flexible user-interface or maximum productivity. Interchangeable probe cards, which connect to the standard common base, support the various configurations and packages of the COP8 family.

The iceMASTER provides real time, full speed emulation up to 10 MHz, 32 kbytes of emulation memory and 4k frames of trace buffer memory. The user may define as many as 32k trace and break triggers which can be enabled, disabled, set or cleared. They can be simple triggers based on code address, direct address, opcode value, opcode class or immediate operand. Complex breakpoints can be ANDed and ORed together. Trace information consists of address bus values, opcodes and user selectable probe clips status (external event lines). The trace buffer can be viewed as raw hex or as disassembled instructions. The probe clip bit values can be displayed in binary, hex or digital waveform formats.

During single-step operation the dynamically annotated code feature displays the contents of all accessed (read and write) memory locations and registers, as well as flow-of-control direction change markers next to each instruction executed.

The iceMASTER's performance analyzer offers a resolution of better than 6s. The user can easily monitor the time spent executing specific portions of code and find "hot spots" or "dead code". Up to 15 independent memory areas based on code address or label ranges can be defined. Analysis results can be viewed in bar graph format or as actual frequency count.

Emulator memory operations for program memory include single line assembler, disassembler, view, change and write to file. Data memory operations include fill, move, compare, dump to file, examine and modify. The contents of any memory space can be directly viewed and modified from the corresponding window.

The iceMASTER comes with an easy to use window interface. Each window can be sized, highlighted, color-controlled, added, or removed completely. Commands can be accessed via pull-down menus and/or redefinable hot keys. A context sensitive hypertext/hyperlinked on-line help system explains clearly the options the user has from within any window.

The iceMASTER connects easily to a PCRM via the standard COMM port and its 115.2 kBaud serial link keeps typical program download time to under 3 seconds.

The following tables list the emulator and probe cards ordering information.

Emulator Ordering Information

Part Number	Description	Current Version
IM-COP8/ 400/1†	MetaLink base unit in-circuit emulator for all COP8 devices, symbolic debugger software and RS-232 serial interface cable, with 110V @ 60 Hz Power Supply.	Host Software: Ver. 3.3 Rev. 5, Model File Rev 3.050.
IM-COP8/ 400/2†	MetaLink base unit in-circuit emulator for all COP8 devices, symbolic debugger software and RS-232 serial interface cable, with 220V @ 50 Hz Power Supply.	
DM-COP8/ 888CF	MetaLink iceMASTER Debug Module. This is the low cost version of the MetaLink iceMASTER. Firmware: Ver. 6.07.	

†These parts include National's COP8 Assembler/Linker/Librarian Package (COP8-DEV-IBMA).

Probe Card Ordering Information

Part Number	Package	Voltage Range	Emulates
MHW-884CF28D5PC	28 DIP	4.5V–5.5V	COP884CF
MHW-884CF28DWPC	28 DIP	2.5V–6.0V	COP884CF
MHW-888CF40D5PC	40 DIP	4.5V–5.5V	COP888CF
MHW-888CF40DWPC	40 DIP	2.5V–6.0V	COP888CF
MWH-888CF44D5PC	44 PLCC	4.5V–5.5V	COP888CF
MHW-888CF44DWPC	44 PLCC	2.5V–6.0V	COP888CF

MACRO CROSS ASSEMBLER

National Semiconductor offers a relocatable COP8 macro cross assembler. It runs on industry standard compatible PCs and supports all of the full-symbolic debugging features of the MetaLink iceMASTER emulators.

Assembler Ordering Information

Part Number	Description	Manual
COP8-DEV-IBMA	COP8Assembler/ Linker/Librarian for IBM® PC/XT®, AT® or compatible.	424410632-001

Development System Support
(Continued)

DIAL-A-HELPER

Dial-A-Helper is a service provided by the Microcontroller Applications group. The Dial-A-Helper is an Electronic Bulletin Board Information system.

INFORMATION SYSTEM

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 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. The minimum requirement for accessing the Dial-A-Helper is a Hayes compatible modem.

If the user has a PC with a communications package then files from the FILE SECTION can be down loaded to disk for later use.

ORDER PIN: MOLE-DIAL-A-HLP

Information System Package Contents:

Dial-A-Helper Users Manual

Public Domain Communications Software

FACTORY APPLICATIONS SUPPORT

Dial-A-Helper also provides immediate factor applications support. If a user has questions, he can leave messages on our electronic bulletin board, which we will respond to.

Voice:	(800) 272-9959		
Modem:	CANADA/U.S.:	(800) NSC-MICRO (800) 672-6427	
	Baud:	14.4k	
	Set-Up:	Length:	8-Bit
		Parity:	None
		Stop Bit:	1
	Operation:	24 Hours,	7 Days

National Semiconductor

PRELIMINARY

COP8788EG/COP8784EG microCMOS One-Time Programmable (OTP) Microcontrollers

General Description

The COP8788EG/COP8784EG programmable microcontrollers are members of the COPS™ microcontroller family. Each device is a two chip system in a plastic package. Within the package is the COP888EG and an 8k EPROM with port recreation logic. The code executes out of the EPROM. The device is offered in four packages: 44-pin PLCC, 40-pin DIP, 28-pin DIP and 28-pin SO.

The COP8788EG/COP8784EG are fully static, 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 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 COP8788EG/COP8784EG devices can be used to provide form fit and function emulation for the COP888EG/COP884EG, COP888CG/COP884CG and COP888CS/COP884CS family of mask programmable devices. The user must pay special attention, since the COP8788EG/COP8784EG devices contain additional features and are supersets of COP888CG/COP884CG and COP888CS/COP884CS. The following table shows the differences between the various devices.

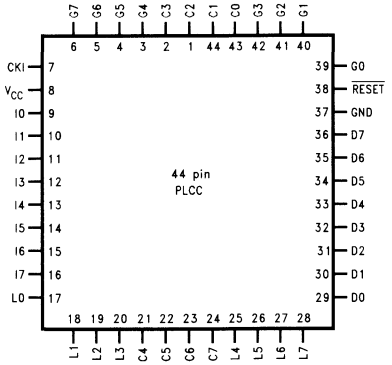
	ROM (Bytes)	RAM (Bytes)	Timers	# of Comparators
COP8788EG/ COP8784EG	8k	256	T0, T1, T2, T3	2
COP888EG/ COP884EG	8k	256	T0, T1, T2, T3	2
COP888CG/ COP884CG	4k	192	T0, T1, T2, T3	2
COP888CS/ COP884CS	4k	192	T0, T1	1

Features

- Low cost 8-bit microcontroller
- Fully static CMOS, with low current drain
- 1 μs instruction cycle time
- 8192 bytes on-board EPROM
- 256 bytes on-board RAM
- Single supply operation: 4.5V–5.5V
- Full duplex UART
- Two analog comparators
- MICROWIRE/PLUS™ serial I/O
- WATCHDOG™ and Clock monitor logic
- Idle Timer
- Multi-Input Wake Up (MIWU) with optional interrupts (8)
- 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
- Three 16-bit timers, each with two 16-bit registers supporting:
  - Processor Independent PWM mode
  - External Event counter mode
  - Input Capture mode
- 8-bit Stack Pointer SP (stack in RAM)
- Two 8-bit Register Indirect Data Memory Pointers (B and X)
- Versatile instruction set with true bit manipulation
- Memory mapped I/O
- BCD arithmetic instructions
- Package:
  - 44 PLCC with 39 I/O pins
  - 40 DIP with 35 I/O pins
  - 28 DIP with 23 I/O pins
  - 28 SO with 23 I/O pins (contact local sales office for availability)
- 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
- Form fit and function emulation device for the COP888EG/COP884EG, COP888CG/COP884CG and COP888CS/COP884CS
- Real time emulation and full program debug offered by MetaLink's Development Systems

Connection Diagrams

Plastic Chip Carrier

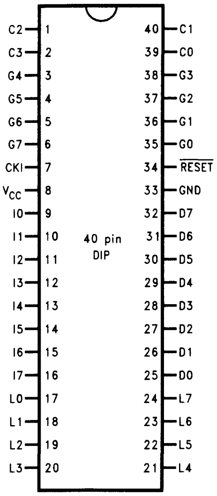


TL/DD12064-1

Top View

Order Number COP8788EGV-X, COP8788EGFV-R
See NS Package Number V44A

Dual-In-Line Package

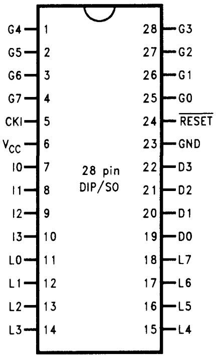


TL/DD12064-2

Top View

Order Number COP8788EGN-X, COP8788EGN-R
See NS Package Number N40A

Dual-In-Line Package



TL/DD12064-3

Top View

Order Number COP8784EGN-X. COP8784EGN-R.
COP8784EGWM-X or COP8784EGWM-R
See NS Package Number M28B or N28A

FIGURE 1. COP8788EG/COP8784EG 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	ALE	25	35	39
G1	WDOUT			26	36	40
G2	I/O	T1B	$\overline{WR}$	27	37	41
G3	I/O	T1A	$\overline{RD}$	28	38	42
G4	I/O	SO		1	3	3
G5	I/O	SK		2	4	4
G6	I	SI	ME	3	5	5
G7	I/CKO	HALT Restart		4	6	6
D0	O		AD0	19	25	29
D1	O		AD1	20	26	30
D2	O		AD2	21	27	31
D3	O		AD3	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		AD4		29	33
D5	O		AD5		30	34
D6	O		AD6		31	35
D7	O		AD7		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
$\overline{RESET}$			V_{PP}	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 $-40°C \leq T_A \leq +85°C$ 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	$V_{CC} = 5.5V, t_c = 1 \mu s$			25	mA
HALT Current (Note 3)	$V_{CC} = 5.5V$, CKI = 0 MHz		250		μA
IDLE Current					
CKI = 10 MHz	$V_{CC} = 5.5V, t_c = 1 \mu s$			15	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	$V_{CC} = 4.5V, V_{OL} = 1V$	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 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

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 is 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.

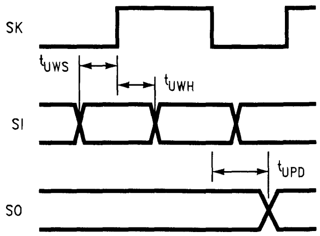
AC Electrical Characteristics $-40^\circ C \leq T_A \leq +85^\circ C$ unless otherwise specified

Parameter	Conditions	Min	Typ	Max	Units
Instruction Cycle Time (t_c)					
Crystal, Resonator,		1		DC	μs
R/C Oscillator		3		DC	μs
CKI Clock Duty Cycle (Note 5)	f_r = Max	40		60	%
Rise Time (Note 5)	f_r = 10 MHz Ext Clock			5	ns
Fall Time (Note 5)	f_r = 10 MHz Ext Clock			5	ns
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

Note 5: Parameter sample (not 100% tested).

Comparators AC and DC Characteristics $V_{CC} = 5V, T_A = 25^\circ C$

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	TBD mV Step, TBD mV Overdrive, 100 pF Load		1		μs



TL/DD12064-4

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 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

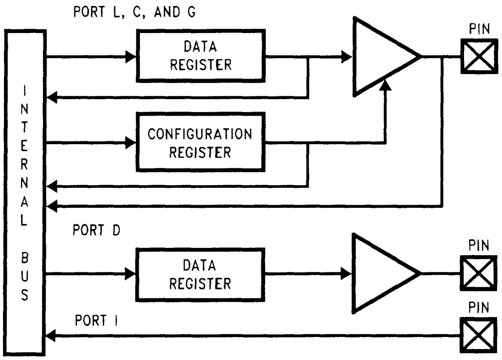
PORT L is an 8-bit I/O port. All L-pins have Schmitt triggers on the inputs.

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/DD12064-5

FIGURE 3. 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 8092 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.

Data Memory Segment RAM Extension

Data memory address 0FF is used as a memory mapped location for the Data Segment Address Register (S).

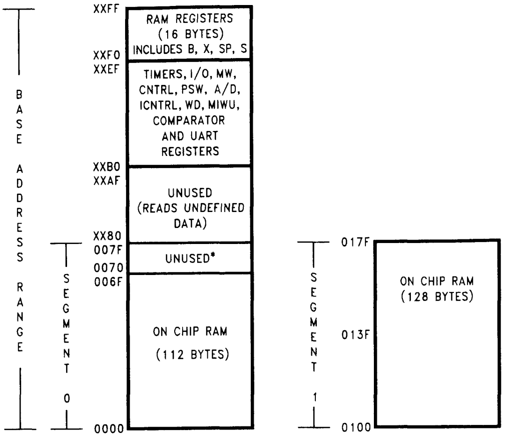
The data store memory is either addressed directly by a single byte address within the instruction, or indirectly relative to the reference of the B, X, or SP pointers (each contains a single-byte address). This single-byte address allows an addressing range of 256 locations from 00 to FF hex. The upper 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 intitialized 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/DD12064-6

*Reads as all ones.

FIGURE 4. RAM Organization

Reset

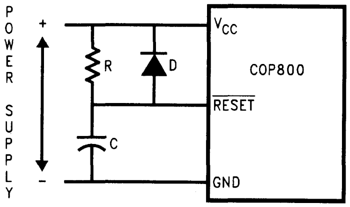
The $\overline{\text{RESET}}$ input when pulled low initializes the microcontroller. Initialization will occur whenever the $\overline{\text{RESET}}$ input is pulled low. Upon initialization, the data and configuration registers for ports L, G and C are cleared, resulting in these Ports being initialized 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 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 5* 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.

Reset (Continued)



TL/DD/12064-7

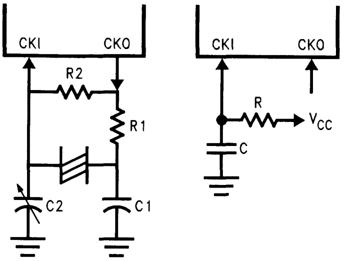
RC > 5 × Power Supply Rise Time

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.



TL/DD12064-8

FIGURE 6. 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

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. Clock Monitor current when enabled—I6
7. Clock Monitor current when enabled—I7

Thus the total current drain, It, is given as

$$It = I1 + I2 + I3 + I4 + I5 + I6 + I7$$

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

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 ($t_C = 1\ \mu s$). 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.

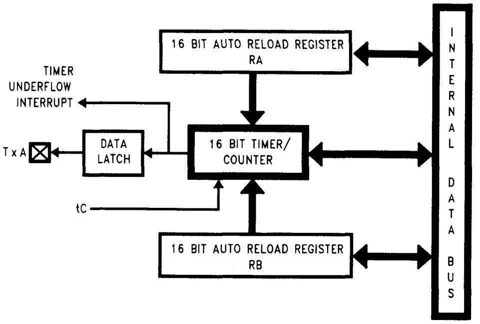
The Tx Timer control bits, TxC3, TxC2 and TxC1 set up the timer for PWM mode operation.

Figure 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/DD12064-9

FIGURE 7. Timer in PWM Mode

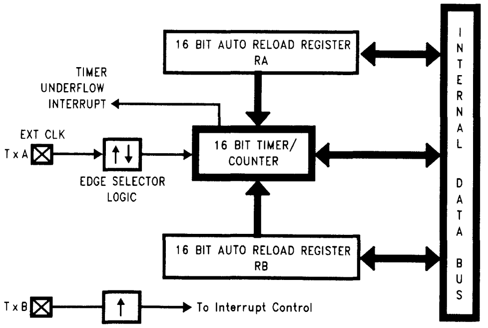
Mode 2. External Event Counter Mode

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/DD12064-10

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.

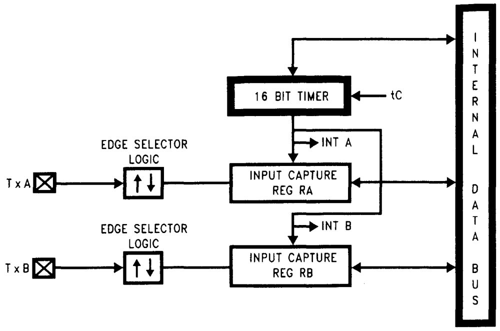
Timers (Continued)

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/DD12064-11

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 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.

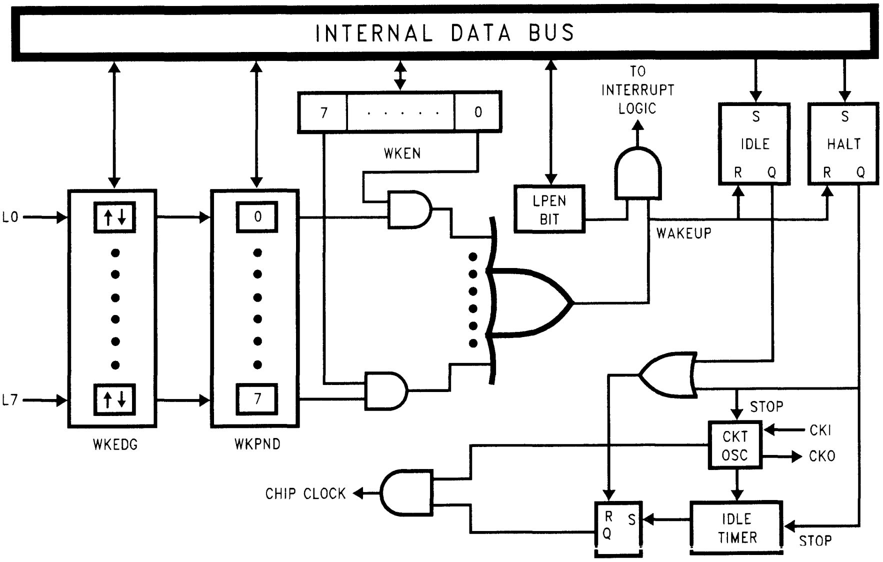
Note: It is necessary to program two NOP instructions following both the set HALT mode and set IDLE mode instructions. These NOP instructions are necessary to allow clock resynchronization following the HALT or IDLE modes.

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 10 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/DD12064-12

FIGURE 10. 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:

```
    RMRBIT  5, WKEN
    RMSBIT  5, WKEDG
    RMRBIT  5, WKPND
    RMSBIT  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.

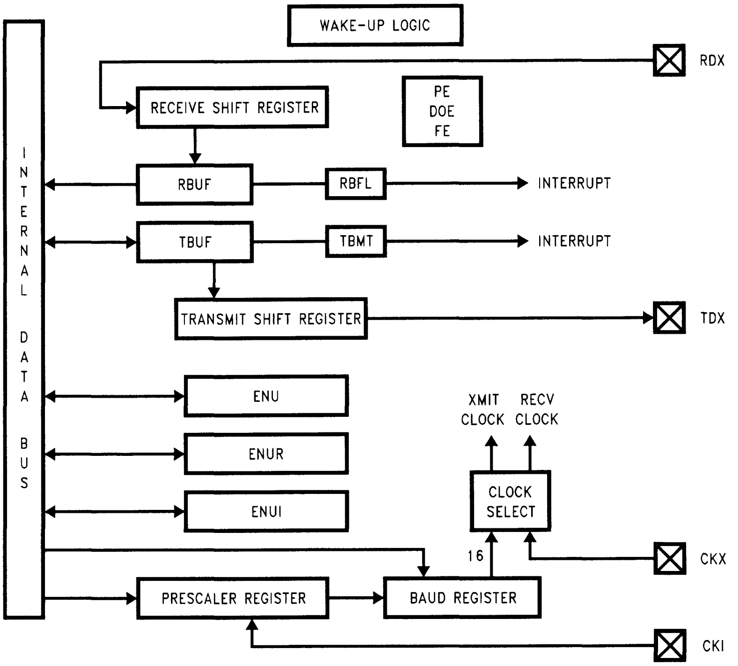
The Wake Up signal will not start the chip running immediately since crystal oscillators or ceramic resonators have a finite start up time. The IDLE Timer (T0) generates a fixed delay to ensure that the oscillator has indeed stabilized before allowing the device to execute instructions. In this case, 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 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 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/DD12064-13

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 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 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.

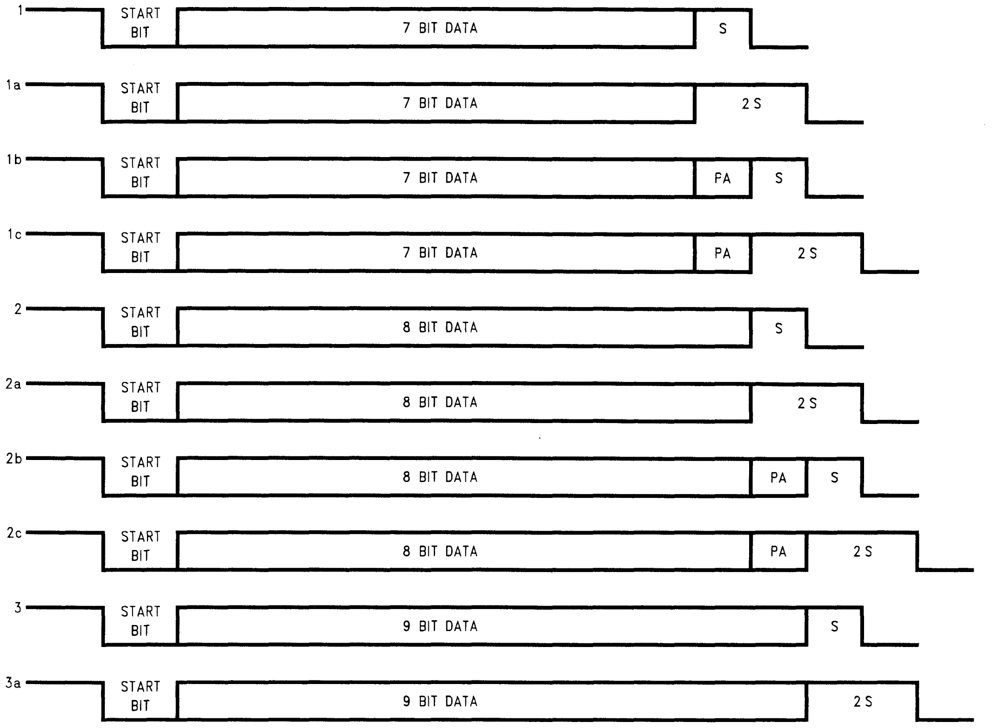
For any of the above framing formats, the last Stop bit can be programmed to be 7/8th of a bit in length. If two Stop bits are selected and the 7/8th bit is set (selected), the second Stop bit will be 7/8th of a bit in length.

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/DD12064–14

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).

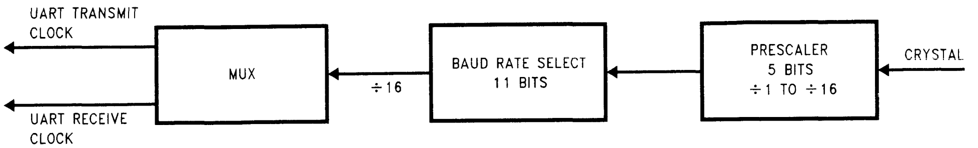
Baud Clock Generation

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 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.

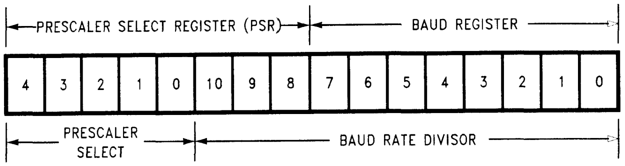
The correspondences between the 5-bit Prescaler Select and Prescaler factors are shown in Table III. There are many ways to calculate the two divisor factors, but one particularly effective method would be to achieve a 1.8432 MHz frequency coming out of the first stage. The 1.8432 MHz prescaler output 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/DD12064-15

FIGURE 13. UART BAUD Clock Generation



TL/DD12064-16

FIGURE 14. 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 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 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 zero.)

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 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)
CMP1OE	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)
CMP2OE	Selects pin I6 as comparator 2 output provided that CMP2EN is set to enable the comparator

Unused	CMP2OE	CMP2RD	CMP2EN	CMP1OE	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
	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/Wake Up	Port L Edge	0yE2-0yE3
(14) 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).

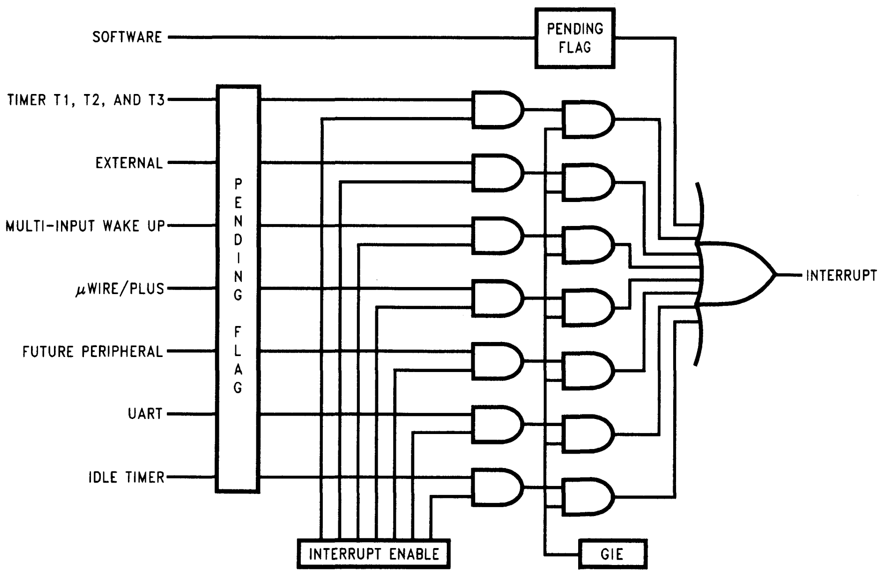
The Software Trap has the highest rank and its vector is located at 0yFE and 0yFF.

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. This vector can point to the Software Trap (ST) interrupt service routine, or to another special service routine as desired.

Figure 15 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/DD12064-17

FIGURE 15. 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.

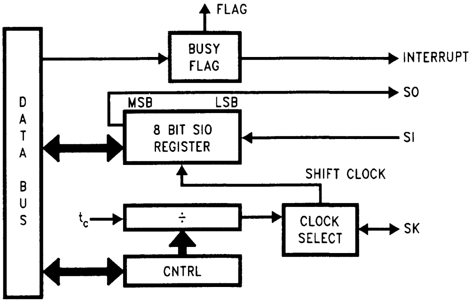
Thus, the chip can detect the following illegal conditions:

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 16* shows a block diagram of the MICROWIRE/PLUS logic.



TL/DD12064-18

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 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 17* 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

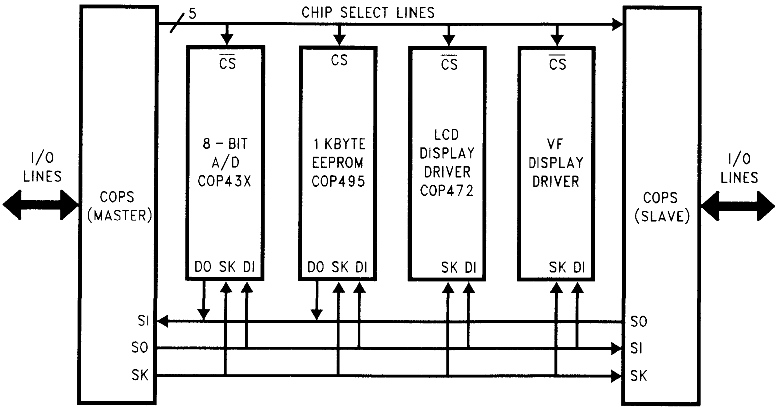
The device allows either the normal SK clock or an alternate phase SK clock to shift data in and out of the SIO register. In both the modes the SK is normally low. In the normal mode data is shifted in on the rising edge of the SK clock and the data is shifted out on the falling edge of the SK clock. 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/DD12064-19

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

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 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]
MemI	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)
Æ	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.

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.

COP8788EG/COP8784EG 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	JP − 15	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 − 14	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 − 13	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 − 12	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 − 11	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 − 10	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 − 9	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 − 7	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 − 6	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 − 5	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 − 4	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 − 3	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 − 2	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 − 1	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 − 0	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

Ordering Information and Development Support

COP8788EG/COP8784EG Ordering Information

Device Number	Clock Option	Package	Emulates
COP8788EGV-X COP8788EGV-R*	Crystal R/C	44 PLCC	COP888EG
COP8788EGN-X COP8788EGN-R*	Crystal R/C	40 DIP	COP888EG
COP8784EGN-X COP8784EGN-R*	Crystal R/C	28 DIP	COP884EG
COP8784EGWM-X* COP8784EGWM-R*	Crystal R/C	28 SO	COP884EG

*Check with the local sales office about the availability.

PROGRAMMING SUPPORT

Programming of these emulator devices is supported by different sources. The following programmers are certified for programming these One-Time Programmable emulator devices:

EPROM Programmer Information

Manufacturer and Product	U.S. Phone Number	Europe Phone Number	Asia Phone Number
MetaLink-Debug Module	(602) 926-0797	Germany: +49-8141-1030	Hong Kong: 852-737-1800
Xeltek-Superpro	(408) 745-7974	Germany: +49-20-41-684758	Singapore: 65-276-6433
BP Microsystems-Turpro	(800) 225-2102	Germany: +49-89-85-76667	Hong Kong: 852-388-0629
Data I/O-Unisite -System 29 -System 39	(800) 322-8246	Europe: +31-20-622866 Germany: +49-89-85-8020	Japan: +33-432-6991
Abcom-COP8 Programmer		Europe: +49-89-808707	
System General-Turpro-1-FX -APRO	(408) 263-6667	Switzerland: +41-31-921-7844	Taiwan: +2-917-3005

Development Support

IN-CIRCUIT EMULATOR

The MetaLink iceMASTER™-COP8 Model 400 In-Circuit Emulator for the COP8 family of microcontrollers features high-performance operation, ease of use, and an extremely flexible user-interface or maximum productivity. Interchangeable probe cards, which connect to the standard common base, support the various configurations and packages of the COP8 family.

The iceMASTER provides real time, full speed emulation up to 10 MHz, 32 kbytes of emulation memory and 4k frames of trace buffer memory. The user may define as many as 32k trace and break triggers which can be enabled, disabled, set or cleared. They can be simple triggers based on code address, direct address, opcode value, opcode class or immediate operand. Complex breakpoints can be ANDed and ORed together. Trace information consists of address bus values, opcodes and user selectable probe clips status (external event lines). The trace buffer can be viewed as raw hex or as disassembled instructions. The probe clip bit values can be displayed in binary, hex or digital waveform formats.

During single-step operation the dynamically annotated code feature displays the contents of all accessed (read and write) memory locations and registers, as well as flow-of-control direction change markers next to each instruction executed.

The iceMASTER's performance analyzer offers a resolution of better than 6 μs. The user can easily monitor the time spent executing specific portions of code and find "hot spots" or "dead code". Up to 15 independent memory areas based on code address or label ranges can be defined. Analysis results can be viewed in bar graph format or as actual frequency count.

Emulator memory operations for program memory include single line assembler, disassembler, view, change and write to file. Data memory operations include fill, move, compare, dump to file, examine and modify. The contents of any memory space can be directly viewed and modified from the corresponding window.

The iceMASTER comes with an easy to use window interface. Each window can be sized, highlighted, color-controlled, added, or removed completely. Commands can be accessed via pull-down-menus and/or redefinable hot keys. A context sensitive hypertext/hyperlinked on-line help system explains clearly the options the user has from within any window.

The iceMASTER connects easily to a PC® via the standard COMM port and its 115.2 kBaud serial link keeps typical program download time to under 3 seconds.

The following tables list the emulator and probe cards ordering information.

Emulator Ordering Information

Part Number	Description	Current Version
IM-COP8/400/1‡	MetaLink base unit in-circuit emulator for all COP8 devices, symbolic debugger software and RS 232 serial interface cable, with 110V @ 60 Hz Power Supply.	Host Software: Ver. 3.3 Rev. 5, Model File Rev 3.050.
IM-COP8/400/2‡	MetaLink base unit in-circuit emulator for all COP8 devices, symbolic debugger software and RS 232 serial interface cable, with 220V @ 50 Hz Power Supply.	
DM-COP8/888EG‡	MetaLink IceMaster Debug Modul. This is the low cost version of the MetaLink IceMaster. Firmware: Ver. 6.07	

‡These parts include National's COP8 Assembler/Linker/Librarian Package (COP8-DEV-IBMA).

Development Support (Continued)

Probe Card Ordering Information

Part Number	Package	Voltage Range	Emulates
MHW-884EG28D5PC	28 DIP	4.5V–5.5V	COP884EG
MHW-884EG28DWPC	28 DIP	2.5V–6.0V	COP884EG
MHW-888EG40D5PC	40 DIP	4.5V–5.5V	COP888EG
MHW-888EG40DWPC	40 DIP	2.5V–6.0V	COP888EG
MWH-888EG44D5PC	44 PLCC	4.5V–5.5V	COP888EG
MHW-888EG44DWPC	44 PLCC	2.5V–6.0V	COP888EG

MACRO CROSS ASSEMBLER

National Semiconductor offers a relocatable COP8 macro cross assembler. It runs on industry standard compatible PCs and supports all of the full-symbolic debugging features of the MetaLink iceMASTER emulators.

Assembler Ordering Information

Part Number	Description	Manual
COP8-DEV-IBMA	COP8 Assembler/ Linker/Librarian for IBM® PC/XT®, AT® or compatible.	424410632-001

DIAL-A-HELPER

Dial-A-Helper is a service provided by the Microcontroller Applications group. The Dial-A-Helper is an Electronic Bulletin Board Information system.

INFORMATION SYSTEM

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 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. The minimum requirement for accessing the Dial-A-Helper is a Hayes compatible modem.

If the user has a PC with a communications package then files from the FILE SECTION can be down loaded to disk for later use.

ORDER P/N: MOLE-DIAL-A-HLP

Information System Package contains:
Dial-A-Helper Users Manual
Public Domain Communications Software

FACTORY APPLICATIONS SUPPORT

Dial-A-Helper also provides immediate factor applications support. If a user has questions, he can leave messages on our electronic bulletin board, which we will respond to.

Voice:	(800) 272-9959	
Modem:	CANADA/U.S.:	(800) NSC-MICRO
		(800) 672-6427
	Baud:	14.4k
	Set-Up:	Length: 8-Bit
		Parity: None
		Stop Bit: 1
	Operation:	24 Hrs., 7 Days

Section 2

COP8 Applications

Section 2 Contents

Dual Tone Multiple Frequency (DTMF)

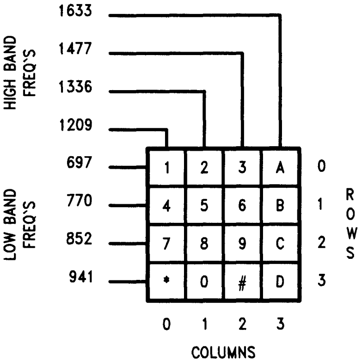
National Semiconductor
Application Note 521
Verne H. Wilson

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
   *                                 * HEX DIGIT IS RRCC,
  0xD0       000             1           WHERE R = ROW #
  0xD1       004             2             AND C = COLUMN #
  0xD2       008             3 -  -  - EXAMPLE: KEY 3 IS ROW #0,
  0xD3       00C             A           COLUMN #2, SO HEX DIGIT
  0xD4       040             4           IS 0010 = 2
  0xD5       044             5              RRCC
  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
```

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

                                  .FORM
                         ;
                         ;MAGIC:         CORE VECTOR
                         ;                 XX00TT00
                         ;
                         ;    TIMER      T      TT00
                         ;    R1         F      XX11
                         ;    R2         Q      XX10
                         ;    R3         R      XX01
                         ;
                         ;DECLARATIONS:
          00D0             PORTLD = 0D0              ; PORTL DATA REG
          00D1             PORTLC = 0D1              ; PORTL CONFIG REG
          00D4             PORTGD = 0D4              ; PORTG DATA REG
          00D5             PORTGC = 0D5              ; PORTG CONFIG REG
          00DC              PORTD = 0DC              ; PORTD REG
          00EA            TIMERLO = 0EA              ; TIMER LOW COUNTER
          00EE              CNTRL = 0EE              ; CONTROL REG
          00EF                PSW = 0EF              ; PROC STATUS WORD
          00F0                 R0 = 0F0              ; LB FREQ LOOP COUNTER
          00F1                 R1 = 0F1              ; LB FREQ LOOP COUNT
          00F2                 R2 = 0F2              ; LB FREQ Q COUNT
          00F3                 R3 = 0F3              ; LB FREQ R COUNT
          00F4                 R4 = 0F4              ; LB FREQ TABLE VECTOR
                         ;
     0000 DD2F           START:     LD         SP,#02F          ; HEX DIGIT MATRIX
     0002 BCD1FF                    LD         PORTLC,#0FF      ; 1   2   3   A
     0005 BCD080                    LD         PORTLD,#080      ; 4   5   6   B
     0008 DEDC                      LD         B,#PORTD         ; 7   8   9   C
     000A 9E00                      LD         [B],#0           ; *   0   #   D
     000C AE             LOOP:      LD         A,[B]            ; DTMF TEST LOOP
     000D 3160                      JSR        DTMF             ; HEX MATRIX DIGIT
     000F DEDC                      LD         B,#PORTD         ; TO SUBROUTINE IS
     0011 AE                        LD         A,[B]            ; OUTPUT TO PORTD
     0012 9405                      ADD        A,#5             ; D0 WILL TOGGLE
     0014 A6                        X          A,[B]            ; FOR EACH CALL OF
     0015 6C                        RBIT       4,[B]            ; DTMF SUBROUTINE
     0016 9DD0                      LD         A,PORTLD         ; PORTL OUTPUTS
     0018 A1                        SC                          ; PROVIDE SYNC
     0019 B0                        RRC        A                ; OUTPUT ORDER IS
     001A 9CD0                      X          A,PORTLD         ; 1,5,9,D,4,8,#,A,
     001C EF                        JP         LOOP             ; 7,0,3,B,*,2,6,C
                         ;
                         ;
```

TL/DD/9662-3

```
     0160                .=0160
                 ;
0160 DED5    DTMF:       LD      B,#PORTGC
0162 9B3F                LD      [B-],#03F
0164 6B                  RBIT    3,[B]           ; OPTIONAL
0165 6A                  RBIT    2,[B]           ; OPTIONAL
                 ;
0166 94D0                ADD     A,#0D0
0168 A4                  LAID                    ; DIGIT MATRIX TABLE
                 ;
0169 5F                  LD      B,#0
016A A6                  X       A,[B]
016B AE                  LD      A,[B]
017B 65                  SWAP    A
016C 97C3                OR      A,#0C3
016E DEF1                LD      B,#R1
0170 DCF4                LD      X,#R4
0172 B6                  X       A,[X]
0173 BE      LUP:        LD      A,[X]
0174 A4                  LAID                    ; LB FREQ TABLES
0175 A2                  X       A,[B+]          ;   (3 PARAMETERS)
0176 C4                  DRSZ    R4
0177 44                  IFBNE   #4
0178 FA                  JP      LUP
                 ;
0179 5F                  LD      B,#0
017A AE                  LD      A,[B]
017C 97C0                OR      A,#0C0
017E A4                  LAID                    ; HB FREQ TABLE
017F DEEA                LD      B,#TIMERLO      ;   (1 PARAMETER)
0181 9A0F                LD      [B+],#15
0183 9A00                LD      [B+],#0
0185 A2                  X       A,[B+]
0186 9A01                LD      [B+],#1
0188 9EB0                LD      [B],#0B0        ; START TIMER PWM
                 ;
018A DED4                LD      B,#PORTGD
018C DCF1                LD      X,#R1
                 ;
018E BB      LUP1:       LD      A,[X-]
018F 72                  IFBIT   2,[B]           ; TEST LB OUTPUT
0190 03                  JP      BYP1
0191 B2                  X       A,[X+]
0192 7A                  SBIT    2,[B]           ; SET LB OUTPUT
0193 03                  JP      BYP2
0194 B8      BYP1:       NOP
0195 6A                  RBIT    2,[B]           ; RESET LB OUTPUT
0196 B2                  X       A,[X+]
0197 C2      BYP2:       DRSZ    R2              ; DECR. QUOT. COUNT
0198 01                  JP      LUP2
0199 0C                  JP      FINI            ; Q COUNT FINISHED
                 ;
019A C0      LUP2:       DRSZ    R0              ; DECR. F COUNT
019B FE                  JP      LUP2            ; LB (HALF PERIOD)
                 ;
019C B8                  NOP                     ; *************
019D BE                  LD      A,[X]           ; BALANCE
019E 9268                IFEQ    A,#104          ; LB FREQUENCY
01A0 ED                  JP      LUP1            ; HALF PERIOD
                 ;                               ; RESIDUE
01A1 B8                  NOP                     ; DELAY FOR
01A2 925D                IFEQ    A,#93           ; EACH OF 4
01A4 E9      BACK:       JP      LUP1            ; LB FREQ'S
01A5 FE                  JP      BACK            ; *************
                 ;
01A6 C3      FINI:       DRSZ    R3              ; DECR. REM. COUNT
01A7 FE                  JP      FINI            ; R CNT NOT FINISHED
                 ;
01A8 BDEE6C              RBIT    4,CNTRL         ; STOP TIMER
01AB 6B                  RBIT    3,[B]           ; CLR HB OUTPUT
01AC 6A                  RBIT    2,[B]           ; CLR LB OUTPUT
                 ;
01AD 8E                  RET
                 ;
```

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

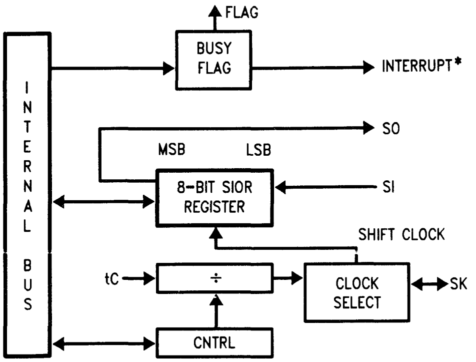
INTRODUCTION

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.



TL/DD/10252-1

*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

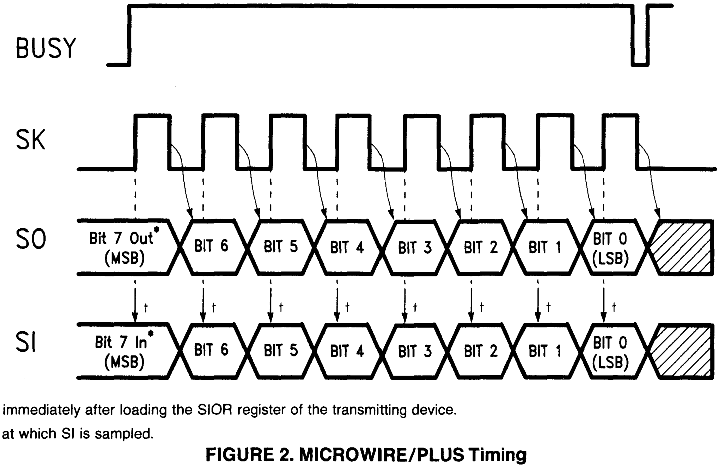
In the MICROWIRE/PLUS master mode, the BUSY flag of PSW (Processor Status Word) is used to control the shifting of the MICROWIRE/PLUS 8-bit shift register. Setting the BUSY flag causes the SIOR register to shift out 8 bits of data from SO at the high order end of the shift register. During the same time, 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)*.

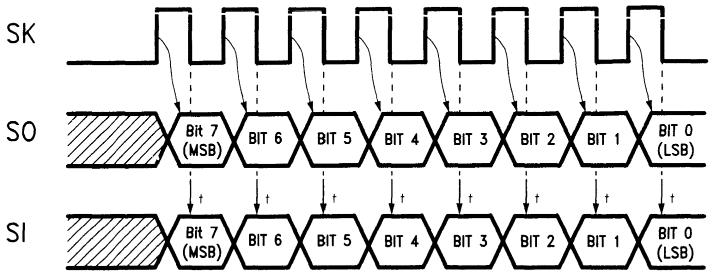


TL/DD/10252-2

*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



TL/DD/10252-3

↑ 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.

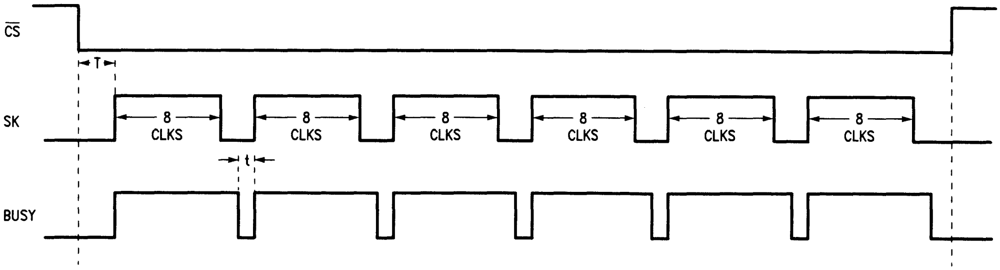
SLAVE MODE OPERATING PROCEDURE

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.



TL/DD/10252-4

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^{2}PROM
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

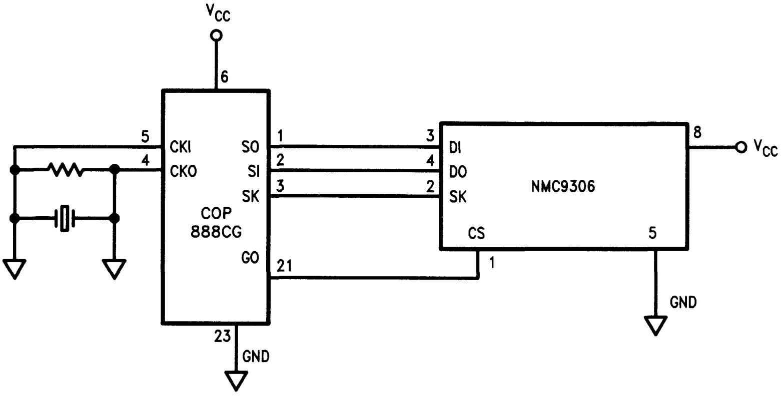
A whole family of off-the shelf devices exist that are directly compatible with MICROWIRE/PLUS protocol. This allows direct interface with the COP800 family of microcontrollers. Table III provides a summary of the existing devices, their function and specification.

NMC9306-COP888CG INTERFACE

The pin connection involved in interfacing an NMC9306 (COP494), a 256 bit E^{2}PROM, 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. $\overline{\text{CS}}$ 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 $\overline{\text{CS}}$ enable ("0" to "1") when executing any instruction. One $\overline{\text{CS}}$ 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 $\overline{\text{CS}}$ is held on after all 16 of the data bits have been outputed, the DO will output the state of DI until another $\overline{\text{CS}}$ LO to HI transition starts a new instruction cycle.
6. After a read cycle, the $\overline{\text{CS}}$ 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 $\overline{\text{CS}}$ low for 10 ms.

All instructions are initiated by a LO-HI transition on $\overline{\text{CS}}$ 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$\overline{\text{CS}}$ pulsed low for 10 ms minimum.

ERASE/ERASE ALL— Command shifted in followed by $\overline{\text{CS}}$ low.

WRITE ALL— Pulsing $\overline{\text{CS}}$ low for 10 ms.

ENABLE/DISABLE— Command shifted in.

A detailed explanation of the E^{2}PROM 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:

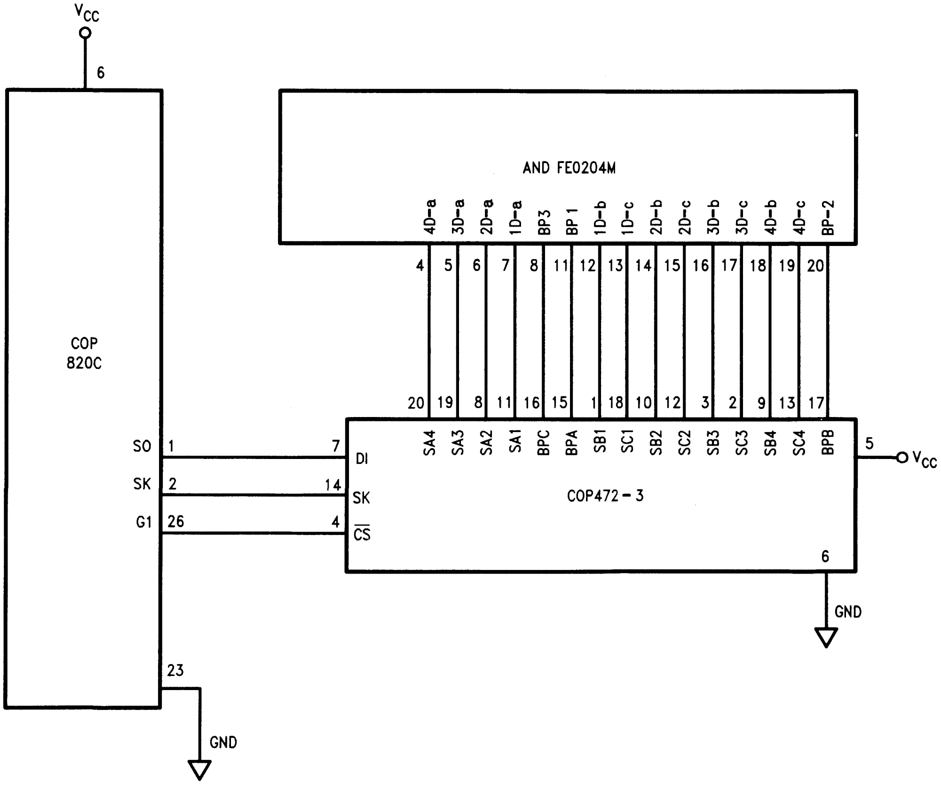
SA	SB	SC	SD	SE	SF	SG	SH

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 8½ 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

```
          36 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:

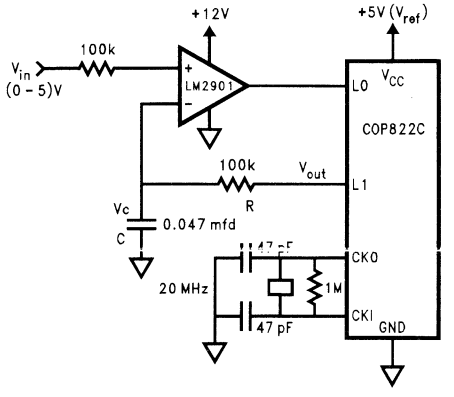
$$V_{in} = V_{ref}[T_{on}/(T_{on} + T_{off})] \tag{1}$$

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:

$$V_{in} = V_{ref}[\text{Sum of } T_{on}/(\text{Sum of } (T_{on} + T_{off}))] \tag{2}$$

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 $V_{in} = V_{in} + K_nV_{out} - K_nV_{in} - K_mV_{in} + K_mV_O$

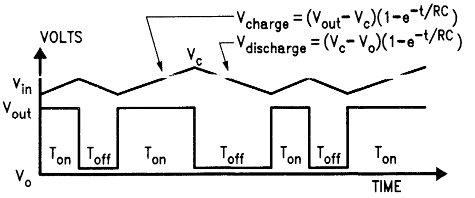
$0 = K_nV_{out} + K_mV_o - KV_{in}(n + m)$

let $V_{out} = V_{ref} - V_{os}$

solving for V_{in}:

$$V_{in} = nV_{ref}/(n + m) - (nV_{os} - mV_o)(1/(n + m)) \tag{3}$$

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.

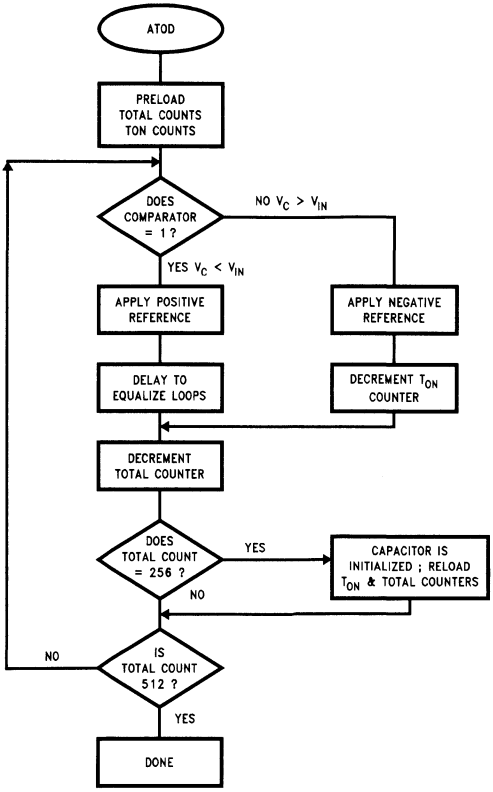
At this point the capacitor is equal to V_{in} and the actual conversion is started. When the TOTAL register is decremented to zero (255 samples later), the conversion is complete. T_{on} will not be reloaded since 0F1 was decremented and IFEQ A,0F1 will no longer be true. The accumulator is then loaded 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.

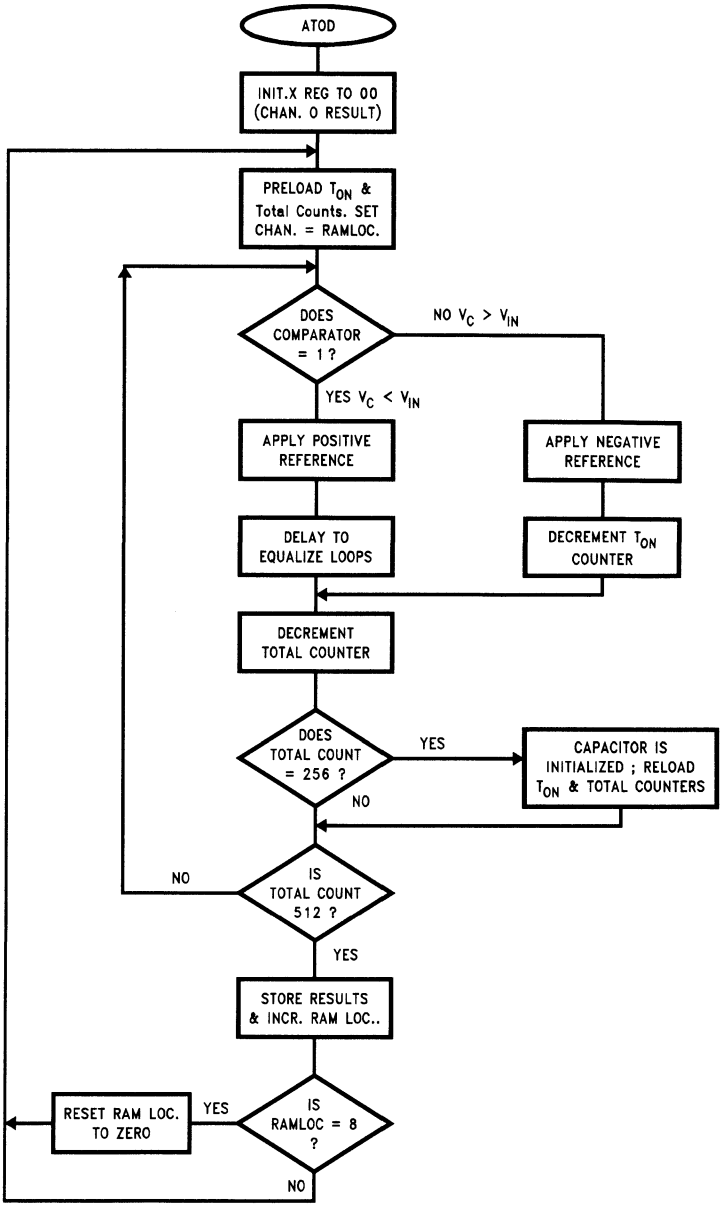


TL/DD/10407-7

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 0F1,#02          ;TOTAL LOOP COUNTER
        LD TON,#0FF         ;PRELOAD Ton
        LD 0FE,#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,0F1          ;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 0F1            ;DECREMENT TOTAL LOOP UNTIL ZERO
        JMP LOOP            ;DONE WHEN 0F1 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*).

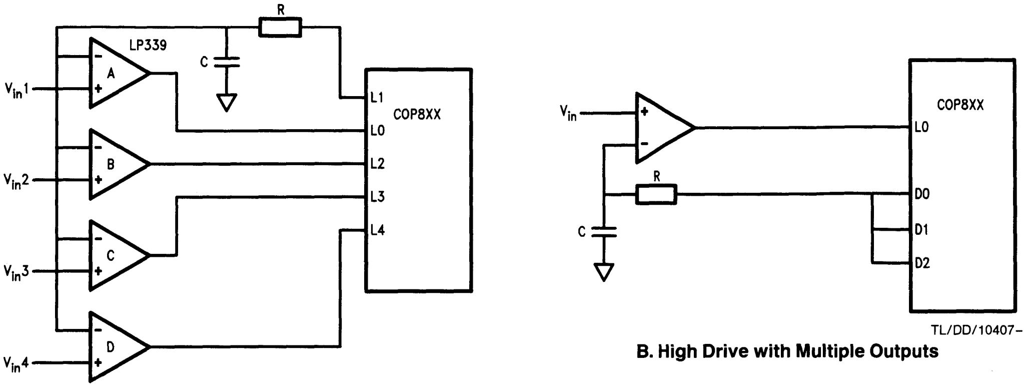
By using the LM604 *(Figure 7B)* the basic software can be easily extended for converting several channels. This will only require a control line to be selected before a conversion is started. Since the LM604 needs to be powered from a higher voltage than the input voltage range, the output voltage will 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.



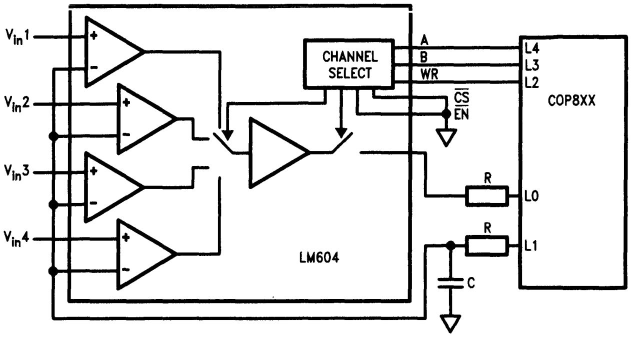
TL/DD/10407–4

A. Multiple Channels with LP339 Low I_{bos} Comparator

TL/DD/10407–5

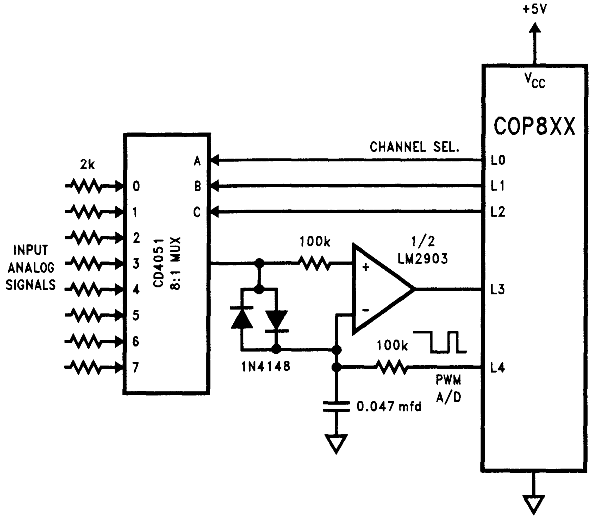
B. High Drive with Multiple Outputs

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.

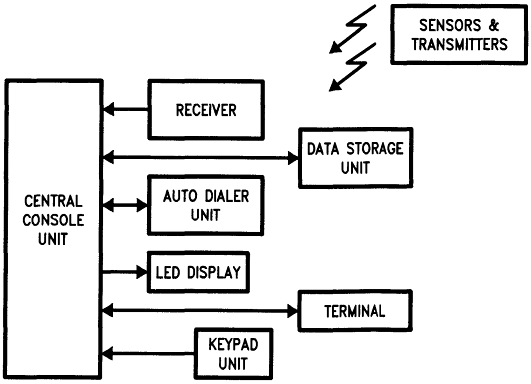
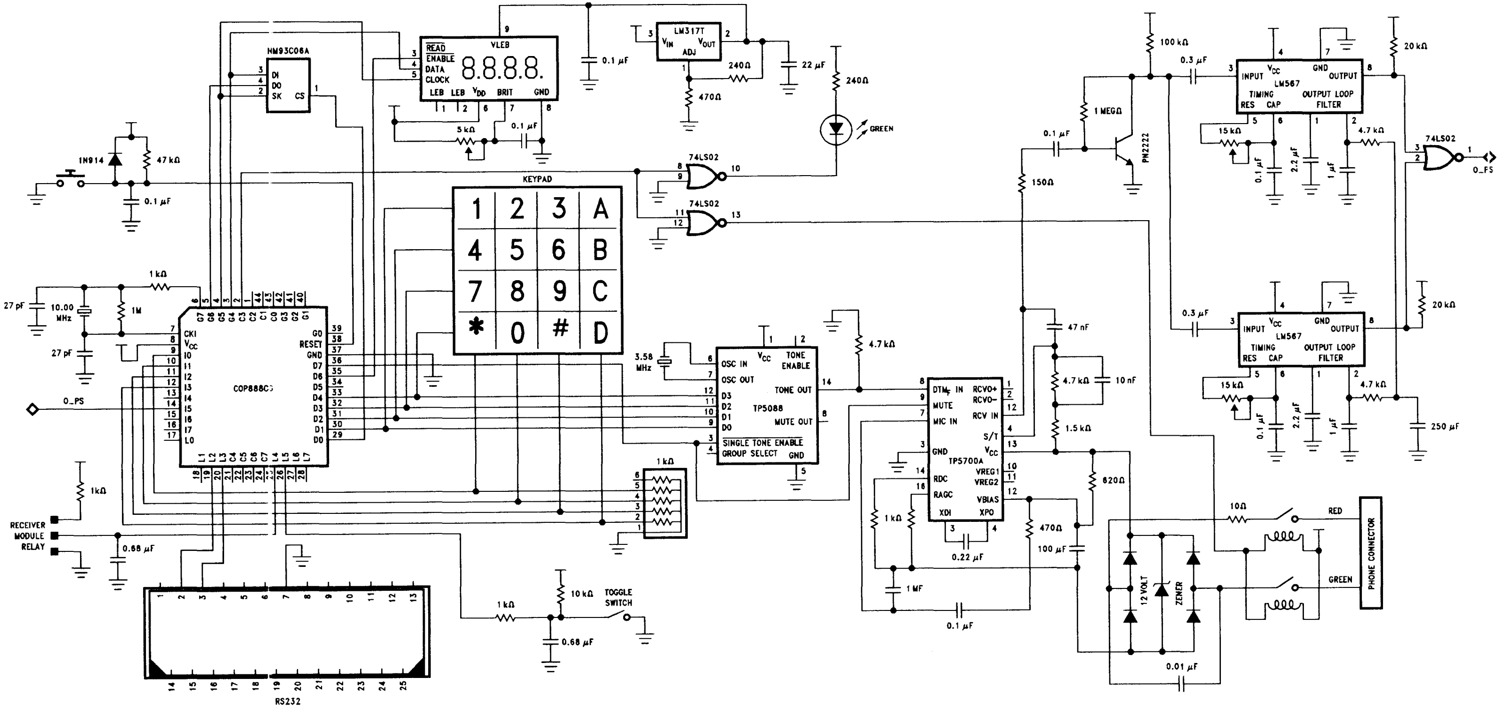


FIGURE 1. Block Diagram of Security/Monitoring System



TL/DD/10607-2

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 Central 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.

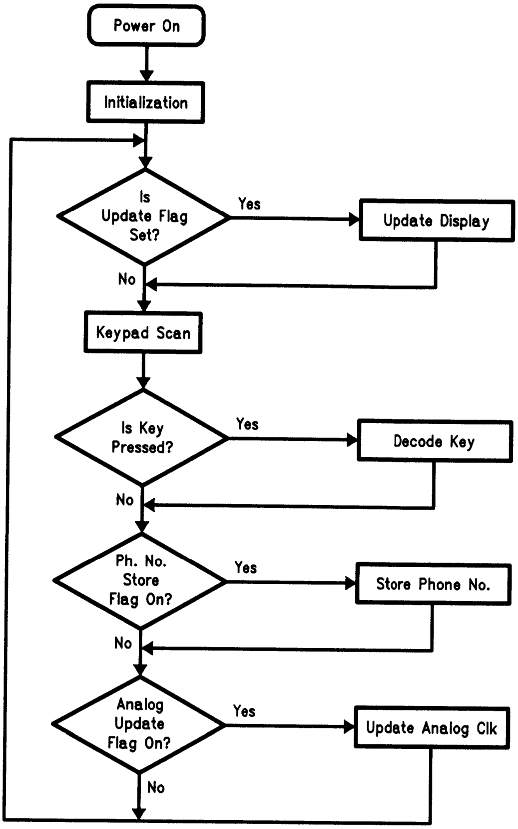
The architecture of COP800 family is based on a modified Harvard type architecture, where the Control Store Program (in ROM) is separated from the Data Store Memory (in RAM). Both types of memory have their own separate addressing space and separate address busses. This architecture allows the overlap 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).

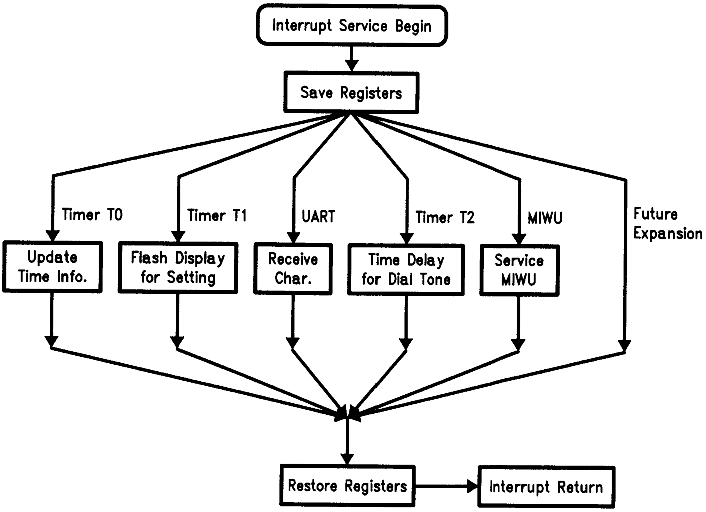
The COP800 family has the ability to detect various illegal conditions resulting from coding errors, transient noise, power supply voltage drops, runaway programs, etc. Reading an undefined ROM location gets zeroes, which results in a non-maskable software interrupt thus signalling an illegal condition 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.



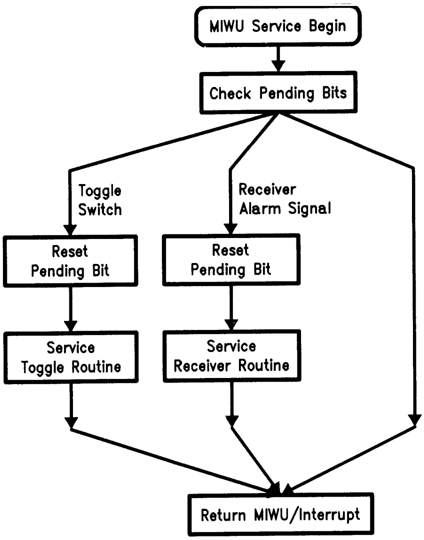
TL/DD/10607–3

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 ⅞, 1, 1 and ⅞, 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

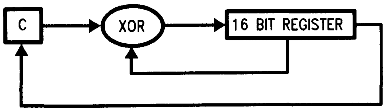
The bomb sound effect combines the descending whistle with an explosion at the end. The TIMER I/O (TIO) and serial input (SI) pins must be tied to the serial output (SO) pin. The explosion portion of this routine was altered slightly in that the first tone control register (FIRSTR) was removed. The first 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.

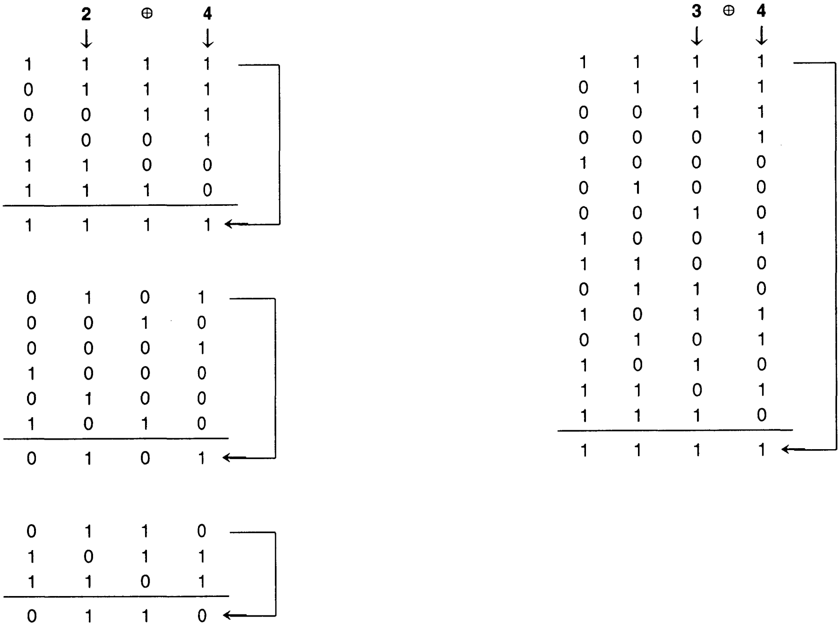
The registers used for program control are EXITR, TCNTR, and the TIMER. By adjusting the value in EXITR the duration of the laser "shot" can be shortened or lengthened. (A value larger than 03F hex may create problems.) By adjusting the TIMER values (TVALO, TVALHI) and the tone counter (TCNTR) value, 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



	2 ↓	⊕	4 ↓
1	1	1	1
0	1	1	1
0	0	1	1
1	0	0	1
1	1	0	0
1	1	1	0
1	1	1	1 ←

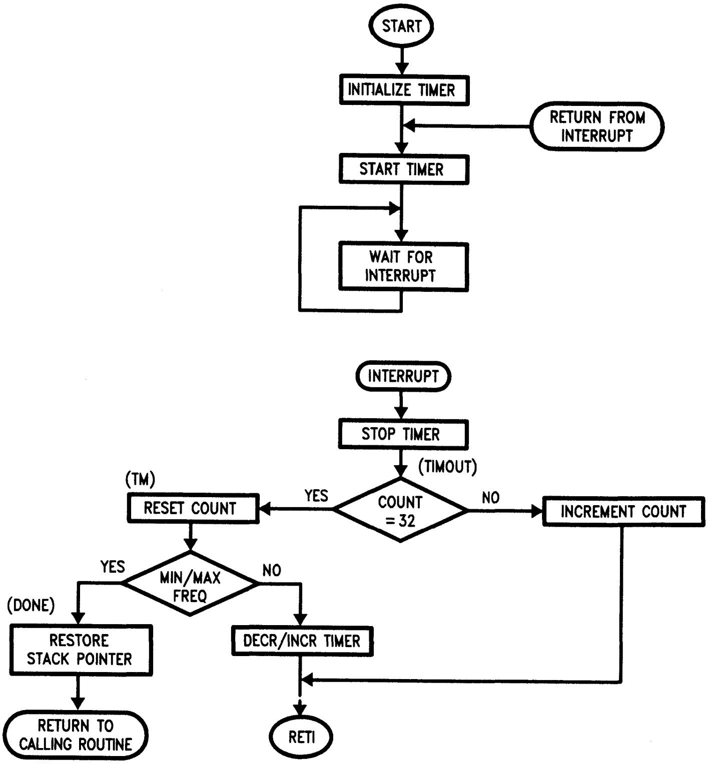
0	1	0	1
0	0	1	0
0	0	0	1
1	0	0	0
0	1	0	0
1	0	1	0
0	1	0	1 ←

0	1	1	0
1	0	1	1
1	1	0	1
0	1	1	0 ←

		3 ↓ ⊕	4 ↓
1	1	1	1
0	1	1	1
0	0	1	1
0	0	0	1
1	0	0	0
0	1	0	0
0	0	1	0
1	0	0	1
1	1	0	0
0	1	1	0
1	0	1	1
0	1	0	1
1	0	1	0
1	1	0	1
1	1	1	0
1	1	1	1 ←

FIGURE 2. Example Showing Possible Cycles from a 4 Stage Ring Counter

Whistle Flow Diagram



TL/DD/10716–2

Descending Whistle

```
                  ;
                  ;
                  ; TIMER INTERRUPT IS USED.
                  ; OUTPUT ON TIMER I/O (TIO) PIN.
                  ; USE 20 MHz XTAL, 1 µs INSTR CYCLE FOR THIS DEMO.
                  ;
                  ; WRITTEN BY: JERRY LEVENTER
                  ; DATE:       OCTOBER 4, 1989
                  ;
                          .TITLE WHISTLE1
                          .CHIP   820
                  ;
      00D5                PORTGC  =   0D5          ; PORT G CONFIGURATION
      00E9                SIOR    =   0E9          ; SIO SHIFT REGISTER
      00EA                TMRLO   =   0EA          ; TIMER LOW BYTE
      00EB                TMRHI   =   0EB          ; TIMER HIGH BYTE
      00EC                TAULO   =   0EC          ; TIMER REGISTER LOW BYTE
      00ED                TAUHI   =   0ED          ; TIMER REGISTER HIGH BYTE
      00EE                CNTRL   =   0EE          ; CONTROL REGISTER
      00EF                PSW     =   0EF          ; PSW REGISTER
      0004                TRUN    =   4
      0005                TPND    =   5
      0002                BUSY    =   2
      0000                GIE     =   0
                  ;
                  ; ****  SPECIAL REGISTERS AND CONSTANTS ****
                  ;
      002F                WSLO    =   02F          ; TIMER VALUES
      0000                WSLHI   =   000
      00F0                FCNTR   =   0F0          ; FREQUENCY COUNT REGISTER
      0000                FCNT    =   000
      00FF                MINFREQ =   0FF          ; MIN FREQUENCY CONSTANT
                  ;
                  ; ********************************
                  ; **** BEGIN DEMO PROGRAM HERE ****
                  ; ********************************
                  ;
0000  DD2F        MAIN:   LD      SP,#02F          ; DEFAULT INITIALIZATION OF SP
0002  3005                JSR     WHISTLE          ; ***CALLING ROUTINE FOR DEMO***
0004  FF                  JP      .
0005  BCD508      WHISTLE:LD      PORTGC,#008      ; TIO PIN (G3) AS OUTPUT
0008  BCEEA2              LD      CNTRL,#0A2       ; PWM WITH TIO TOGGLE, 8Tc
000B  BCEA2F              LD      TMRLO,#WSLO      ; WHISTLE VALUE FOR TIMER
000E  BCEB00              LD      TMRHI,#WSLHI
0011  BCEC2F              LD      TAULO,#WSLO
0014  BCED00              LD      TAUHI,#WSLHI
0017  D000                LD      FCNTR,#FCNT      ; INIT FREQ COUNT
0019  BCEF11      LUP:    LD      PSW,#011         ; ENTI, GIE = 1, TPND = 0
001C  BDEE7C              SBIT    TRUN,CNTRL       ; START TIMER
001F  FF                  JP      .                ; SELF LOOP TIL TIMER INTERRUPT
0020  F8                  JP      LUP              ; RUN TIL LAST HISTLE FREQ
                  ;
                  ; **** INTERRUPT ROUTINE ****
                  ;
      00FF                .=0FF
00FF  BDEF75              IFBIT   TPND, PSW        ; TEST TIMER PENDING FLAG
0102  01                  JP      TIMOUT
0103  FF                  JP      .                ; ERROR
```

Descending Whistle (Continued)

```
0104  BDEE6C    TIMOUT: RBIT     TRUN,CNTRL        ; STOP THE TIMER
0107  BDF075            IFBIT    5,FCNTR           ; COUNT CYCLES
010A  06                JP       TM
010B  9DF0              LD       A,FCNTR           ; INCREMENT COUNT
010D  8A                INC      A
010E  9CF0              X        A,FCNTR
0110  8D                RETSK
0111  D000      TM:     LD       FCNTR,#FCNT       ; RESET COUNT
0113  DEEC              LD       B,#TAULO
0115  AE                LD       A,[B]             ; CHANGE FREQUENCY
0116  92FF              IFEQ     A,#MINFREQ        ; TIMER = MIN FREQ?
0118  03                JP       DONE              ; YES
0119  8A                INC      A
011A  A6                X        A,[B]             ; STORE FREQ IN AUTO RELOAD
011B  8D                RETSK
011C  9DFD      DONE:   LD       A,SP              ; *** RESTORE STACK POINTER ***
011E  9402              ADD      A,#002            ; *** AND RETURN TO CALLING ***
0120  9CFD              X        A,SP              ; *** ROUTINE.              ***
0122  8E                RET
                        .END
```

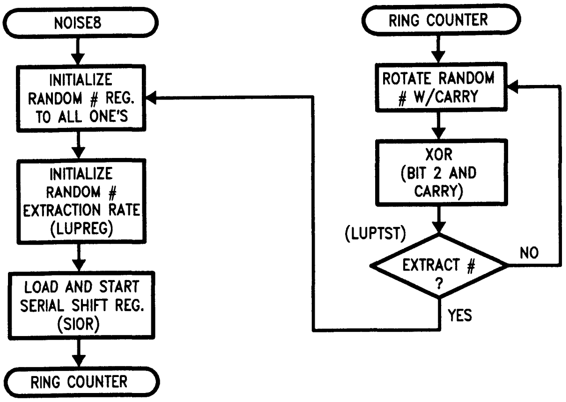
Ascending Whistle

```
                  ;
                  ;
                  ; OUTPUT ON TIMER I/O (TIO) PIN.
                  ; USES TIMER INTERRUT.
                  ; USE 20 MHz XTAL, 1 µs INSTR CYCLE FOR THIS DEMO.
                  ;
                  ; WRITTEN BY: JERRY LEVENTER
                  ; DATE:       OCTOBER 4, 1989
                  ;
                  ;
                        .TITLE WHISTLE2
                        .CHIP   820
                  ;
      00D5              PORTGC  =   0D5             ; PORT G CONFIGURATION
      00EA              TMRLO   =   0EA             ; TIMER LOW BYTE
      00EB              TMRHI   =   0EB             ; TIMER HIGH BYTE
      00EC              TAULO   =   0EC             ; TIMER REGISTER LOW BYTE
      00ED              TAUHI   =   0ED             ; TIMER REGISTER HIGH BYTE
      00EE              CNTRL   =   0EE             ; CONTROL REGISTER
      00EF              PSW     =   0EF             ; PSW REGISTER
      0004              TRUN    =   4
      0005              TPND    =   5
      0002              BUSY    =   2
      0000              GIE     =   0
                  ;
                  ; **** SPECIAL REGISTERS AND CONSTANTS ****
                  ;
      00FF              WSLO    =   0FF             ; TIMER VALUES
      0001              WSLHI   =   001
      000A              MAXFREQ =   00A             ; LAST FREQUENCY CONSTANT
      00F0              FCNTR   =   0F0             ; TIMER COUNT REGISTER
      0010              FCNT    =   010             ; COUNTER CONSTANT
                  ;
                  ; **********************************
                  ; ****    BEGIN PROGRAM HERE    ****
                  ; ***************************** ****
                  ;
0000  DD2F        MAIN: LD      SP,#02F             ; DEFAULT INITIALIZATION OF SP
0002  3005              JSR     WHISTLE2            ; *** CALLING ROUTINE FOR DEMO ***
0004  FF                JP      .
                  WHISTLE2:
0005  BCD508            LD      PORTGC,#008         ; TIO PIN (G3) AS OUTPUT
0008  BCEEA0            LD      CNTRL,#0A0          ; PWM WITH TIO TOGGLE,
000B  BCEAFF            LD      TMRLO,#WSLO         ; WHISTLE VALUE FOR TIMER
000E  BCEB01            LD      TMRHI,#WSLHI
0011  BCECFF            LD      TAULO,#WSLO
0014  BCED01            LD      TAUHI,#WSLHI
0017  D010              LD      FCNTR,#FCNT         ; INITIALIZE COUNTER
0019  BCEF11      LUP:  LD      PSW,#011            ; ENTI, GIE = 1, TPND = 0
001C  BDEE7C            SBIT    TRUN,CNTRL          ; START TIMER
001F  FF                JP      .                   ; SELF LOOP UNTIL TIMER
0020  F8                JP      LUP                 ; INTERRUPT
```

Ascending Whistle (Continued)

```
                              ;
                              ; **** INTERRUPT ROUTINE ****
                              ;
           00FF                       .=0FF
      00FF BDEF75                     IFBIT   TPND,PSW        ; TEST TIMER PENDING FLAG
      0102 01                         JP      TIMOUT
      0103 FF                         JP      .
      0104 BDEE6C             TIMOUT: RBIT    TRUN,CNTRL      ; STOP THE TIMER
      0107 BDF075                     IFBIT   5,FCNTR         ; FREQUENCY TIMED OUT?
      010A 06                         JP      TM              ; YES, CHANGE FREQUENCY
      010B 9DF0                       LD      A,FCNTR         ; NO, KEEP GOING
      010D 8A                         INC     A               ; INCREMENT COUNT
      010E 9CF0                       X       A,FCNTR
      0110 8D                         RETSK                   ; RETURN
      0111 D010               TM:     LD      FCNTR,#FCNT     ; RESET COUNTER
      0113 9DEC                       LD      A,TAULO         ; CHANGE FREQUENCY
      0115 920A                       IFEQ    A,#MAXFREQ      ; TIMER = MAX FREQUENCY ?
      0117 05                         JP      DONE            ; YES
      0118 94FF                       ADD     A,#0FF          ; INCREMENT FREQUENCY
      011A 9CEC                       X       A,TAULO         ; STORE FREQ IN AUTO RELOAD
      011C 8D                         RETSK
      011D 9DFD               DONE:   LD      A,SP            ; *** RESTORE STACK POINTER ***
      011F 9402                       ADD     A,#002          ; *** AND RETURN TO CALLING ***
      0121 9CFD                       X       A,SP            ; *** ROUTINE.              ***
      0123 8E                         RET
                                      .END
```

White Noise



TL/DD/10716–3

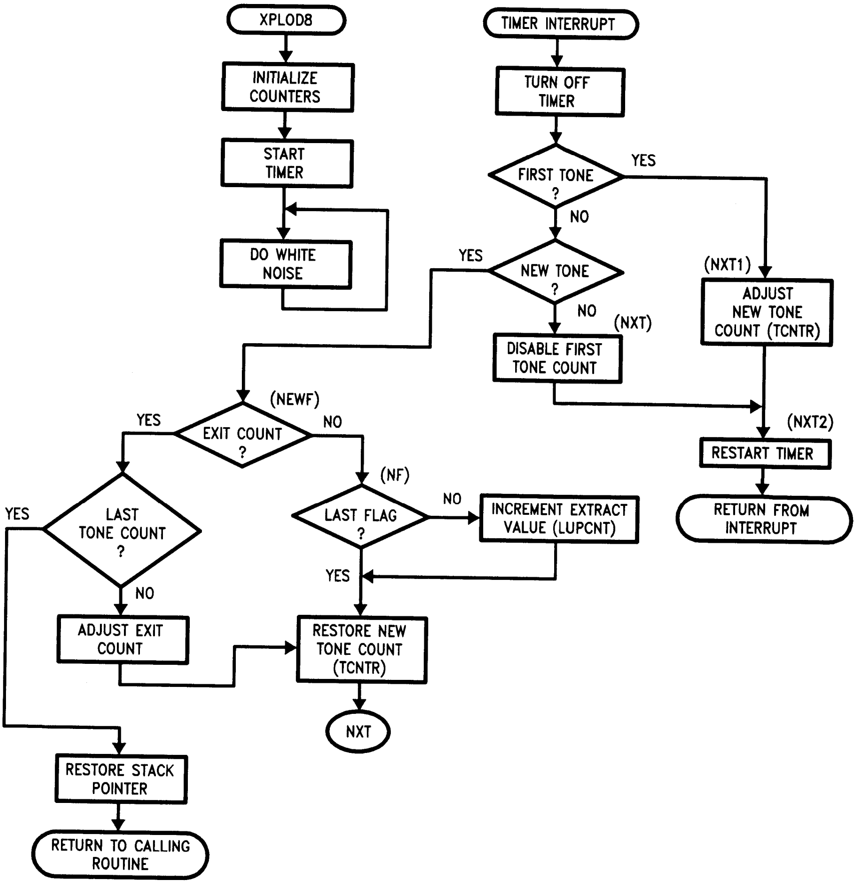
White Noise (Continued)

```
                ;
                ;
                ;
                ;
                ; TIE SERIAL INPUT (SI)PIN TO SERIAL OUTPUT (SO) PIN.
                ; OUTPUT IS ON THE SERIAL OUTPUT (SO) PIN.
                ; NO INTERRUPT IS USED.
                ; USE 20 MHz XTAL, 1 µs INSTR CYCLE FOR THIS DEMO.
                ;
                ; WRITTEN BY: JERRY LEVENTER
                ; DATE:       OCTOBER 4, 1989
                ;
                        .TITLE NOISE8
                        .CHIP   820
                ;
      00D5              PORTGC  =   0D5          ; PORT G CONFIGURATION
      00E9              SIOR    =   0E9          ; SERIAL SHIFT REGISTER
      00EA              TMRLO   =   0EA          ; TIMER LOW BYTE
      00EB              TMRHI   =   0EB          ; TIMER HIGH BYTE
      00EC              TAULO   =   0EC          ; TIMER REGISTER LOW BYTE
      00ED              TAUHI   =   0ED          ; TIMER REGISTER HIGH BYTE
      00EE              CNTRL   =   0EE          ; CONTROL REGISTER
      00EF              PSW     =   0EF          ; PSW REGISTER
      0002              BUSY    =   2            ; BUSY BIT
                ;
                ; ****  SPECIAL REGISTERS AND CONSTANTS ****
                ;
      0002              RNGVAL  =   002          ; RANDOM NUMBER LOCATION
      00FF              LUPREG  =   0FF          ; EXTRACTION RATE REGISTER
      0000              FLAG    =   000          ; RANDOM NUMBER BYTE FLAG
      0004              COUNT   =   4            ; EXTRACTION RATE CONSTANT
                ;
                ; *********************************
                ; ****    BEGIN PROGRAM HERE       ****
                ; *****************************    ****
                ;
0000  DD2F              LD      SP,#02F          ; DEFAULT INITIALIZATION OF SP
0002  BCD530    NOISE:  LD      PORTGC,#030      ; SO AND SK AS OUTPUTS
0005  BCEE8B            LD      CNTRL,#08B       ; SK = DIV BY 8, TIMER RELOAD
0008  A1                SC                       ; INIT STAGE 1
0009  5D                LD      B,#RNGVAL        ; POINT TO RANDOM # LOCATION
000A  9AFF              LD      [B+],#0FF        ; INIT RING VAL TO ONE'S
000C  9EFF              LD      [B],#0FF         ; B POINTS TO UPPER BYTE
000E  9CE9      SHIFT:  X       A,SIOR           ; PLACE # IN SIOR
0010  BDEF7A            SBIT    BUSY,PSW         ; START SHIFTING
0013  DF04              LD      LUPREG,#004      ; RESTORE EXTRACTION COUNT
                ;
```

White Noise (Continued)

```
                    ; ***********************************************
                    ; RING COUNTER (17 STAGE)
                    ; THIS IS A SEVENTEEN STAGE RING COUNTER (LINEAR
                    ; FEEDBACK SHIFT REGISTER) WITH THE RRC COMMAND.
                    ; THE COUNTER'S 14TH AND 17TH STAGES THROUGH AN
                    ; EXCLUSIVE-OR SERVE AS THE FEEDBACK FUNCTION.
                    ; THIS 14, 17 RING COUNTER BREAKS DOWN INTO
                    ; 1 CYCLE OF [(2 ** 17) - 1] COUNTS. SINCE THE EXCLUSIVE OR
                    ; OCCURS AFTER THE ROTATE, IT IS THE 15TH AND CARRY
                    ; STAGES THAT ARE XOR'D (BIT 2 AND CARRY).
                    ;
                    ;                      STAGE
                    ;                    ---------------
                    ; BEFORE ROTATE:      14   17
                    ; AFTER ROTATE:       15   CARRY
                    ;
                    ; CARRY BIT = STAGE ONE
                    ; LOW ORDER BIT = STAGE 17
                    ; ***********************************************
                    ;
0015  AE            RING:   LD      A,[B]            ; GET RANDOM #
0016  B0                    RRC     A                ; ROTATE UPPER BYTE
0017  A3                    X       A,[B-]
0018  AE                    LD      A,[B]
0019  B0                    RRC     A                ; ROTATE LOWER BYTE
001A  A6                    X       A,[B]
001B  9804                  LD      A,#004           ; PERFORM XOR
001D  85                    AND     A,[B]
001E  9200                  IFEQ    A,#000
0020  05                    JP      LUPTST
0021  88                    IFC
0022  02                    JP      RC
0023  A1                    SC
0024  01                    JP      LUPTST
0025  A0            RC:     RC
0026  AA            LUPTST: LD      A,[B+]           ; POINT TO UPPER BYTE
0027  CF                    DRSZ    LUPREG           ; EXTRACT THIS NUMBER ?
0028  EC                    JP      RING             ; NO, KEEP ROTATING
0029  E4                    JP      SHIFT            ; YES, SEND IT
                            .END
```

Explosion



TL/DD/10716–4

Explosion (Continued)

```
                 ;
                 ;
                 ; TIMER INTERRUPT IS USED.
                 ; SI MUST BE TIED TO SO. OUTPUT ON SO.
                 ; USE 20 MHz XTAL, 1 µs INSTR CYCLE FOR THIS DEMO.
                 ;
                 ; WRITTEN BY: JERRY LEVENTER
                 ; DATE:       OCTOBER 4, 1989
                 ;
                         .TITLE XPLOD8
                         .CHIP  820
                 ;
      00D5               PORTGC =   0D5           ; PORT G CONFIGURATION
      00E9               SIOR   =   0E9           ; SIO SHIFT REGISTER
      00EA               TMRLO  =   0EA           ; TIMER LOW BYTE
      00EB               TMRHI  =   0EB           ; TIMER HIGH BYTE
      00EC               TAULO  =   0EC           ; TIMER REGISTER LOW BYTE
      00ED               TAUHI  =   0ED           ; TIMER REGISTER HIGH BYTE
      00EE               CNTRL  =   0EE           ; CONTROL REGISTER
      00EF               PSW    =   0EF           ; PSW REGISTER
      0004               TRUN   =   4
      0005               TPND   =   5
      0002               BUSY   =   2
                 ;
                 ; ****  SPECIAL REGISTERS AND CONSTANTS ****
                 ;
                 ;       ANY REGISTER USED FOR THE DRSZ TEST MUST
                 ;       BE INITIALIZED TO AT LEAST "1".
                 ;
      00F5               FIRSTR =   0F5           ; FIRST TONE CONTROL REGISTER
      0002                FIRST =   002           ;  FIRST TONE CONSTANT
      00F6               LASTR  =   0F6           ; LAST TONE CONTROL REGISTER
      0002                LAST  =   002           ;  LAST TONE CONSTANT
      00F7               EXITR  =   0F7           ; ROUTINE DURATION REGISTER
      0010                EXIT  =   010           ;  EXIT CONSTANT
      0002               RNGVAL =   002           ; HOLDS CURRENT RANDOM #
      00F8               TCNTR  =   0F8           ; TONE DURATION REGISTER
      000A                TCNT  =   0A            ;  TONE CONSTANT
      0020                TCNT1 =   020           ;   "FIRST" TONE CONSTANT
      00F9               LUPREG =   0F9           ; EXTRACTION RATE REGISTER
      0004                XTRCT =   004           ;  EXTRACT CONSTANT
      00FA               LUPCNT =   0FA           ; EXTRACTION VARIABLE REGISTER
      0000               TEMP   =   000           ; LAST TONE FLAG
      00FF               TVALO  =   0FF           ; TIMER VALUES
      0010               TVALHI =   010
                 ;
                 ; *********************************
                 ; ****    BEGIN PROGRAM HERE    ****
                 ; *********************************
                 ;
0000  DD2F       MAIN:   LD     SP,#02F           ; DEFAULT INITIALIZATION OF SP
0002  3005               JSR    XPLOD             ; **** XPLOD CALLING ROUTNE ****
0004  FF                 JP     .                 ; **** SELF LOOP FOR DEMO ****
0005  BCD530     XPLOD:  LD     PORTGC,#030
0008  BCEE8A             LD     CNTRL,#08A        ; SK = DIV BY 8, PWM ON
000B  BCEF11             LD     PSW,#011          ; ENABLE TIMER INTERRUPT
000E  BCEAFF             LD     TMRLO,#TVALO      ; INITIALIZE TIMER
0011  BCEB10             LD     TMRHI,#TVALHI
0014  BCECFF             LD     TAULO,#TVALO
0017  BCED10             LD     TAUHI,#TVALHI
```

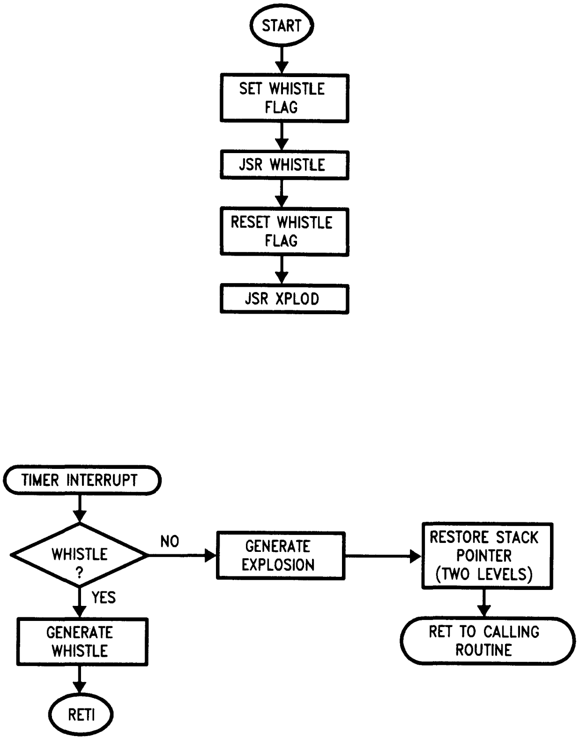
Explosion (Continued)

```
001A  D502              LD      FIRSTR,#FIRST     ; LENGTHEN FIRST TONE
001C  D602              LD      LASTR,#LAST       ; LENGTHEN LAST TONE
001E  D710              LD      EXITR,#EXIT       ; INITIALIZE EXIT COUNT
0020  D80A              LD      TCNTR,#TCNT       ; INITIALIZE TONE COUNT
0022  DA04              LD      LUPCNT,#XTRCT     ; INITIALIZE EXTRACTION RATE
0024  BD0068            RBIT    0,TEMP            ; RESET LAST TONE FLAG
0027  BDEE7C            SBIT    TRUN,CNTRL        ; START TIMER
002A  A1        NOISE:  SC                        ; INIT. STAGE 1
002B  5D                LD      B,#RNGVAL         ; POINT TO RANDOM NUMBER
002C  9AFF              LD      [B+],#0FF         ; INIT TO ALL ONE'S
002E  9EFF              LD      [B],#0FF
0030  9CE9      SHIFT:  X       A,SIOR            ; LOAD AND START SIOR
0032  BDEF7A            SBIT    BUSY,PSW
0035  9DFA              LD      A,LUPCNT          ; RESTORE EXTRACTION COUNT
0037  9CF9              X       A,LUPREG
                ;
                ; **********************************
                ; RING COUNTER (17 STAGE)
                ;
                ; THIS IS A SEVENTEEN STAGE RING COUNTER (LINEAR
                ; FEEDBACK SHIFT REGISTER) WITH THE RRC COMMAND.
                ; THE COUNTER'S 14th AND 17th STAGES THROUGH AN
                ; EXCLUSIVE-OR SERVE AS THE FEEDBACK FUNCTION.
                ; THIS 14, 17 RING COUNTER BREAKS DOWN INTO
                ; 1 CYCLE OF [(2 ** 17) - 1] COUNTS. SINCE THE EXCLUSIVE OR
                ; OCCURS AFTER THE ROTATE, IT IS THE 15th AND CARRY
                ; STAGES THAT ARE XOR'D (BIT 2 AND CARRY).
                ;
                ;                  STAGE
                ;               ---------------
                ; BEFORE ROTATE:    14   17
                ; AFTER ROTATE:     15   CARRY
                ;
                ;  CARRY BIT = STAGE 1
                ;  LOW ORDER BIT OF 16 BIT REGISTER = STAGE 17
                ;  *********************************
                ;
0039  AE        RING:   LD      A,[B]             ; GET RANDOM #
003A  B0                RRC     A                 ; ROTATE UPPER BYTE
003B  A3                X       A,[B-]
003C  AE                LD      A,[B]
003D  B0                RRC     A                 ; ROTATE LOWER BYTE
003E  A6                X       A,[B]
003F  9804              LD      A,#004            ; PERFORM XOR
0041  85                AND     A,[B]
0042  9200              IFEQ    A,#000
0044  05                JP      TSLUP
0045  88                IFC
0046  02                JP      RC
0047  A1                SC
0048  01                JP      TSTLUP
0049  A0        RC:     RC
004A  AA        TSTLUP: LD      A,[B+]            ; POINT TO UPPER BYTE
004B  C9                DRSZ    LUPREG            ; EXTRACT THIS # ?
004C  EC                JP      RING              ; NO, KEEP ROTATING
004D  AE                LD      A,[B]             ; YES
004E  E1                JP      SHIFT
```

Explosion (Continued)

```
                ;
                ; **** TIMER INTERRUPT ROUTINE ****
                ;
      00FF      ;         .=      OFF
00FF  BDEF75              IFBIT   TPND,PSW         ; TEST TIMER PND FLAG
0102  02                  JP      TMOUT
0103  2005                JMP     XPLOD
0105  BDEE6C    TMOUT:    RBIT    TRUN,CNTRL       ; STOP TIMER
0108  DEFA                LD      B,#LUPCNT
010A  C5                  DRSZ    FIRSTR           ; TEST FOR FIRST TONE
010B  213B                JMP     NXT1             ; AND ADJUST
010D  C8                  DRSZ    TCNTR            ; TEST FOR NEW TONE
010E  01                  JP      NXT              ; NO
010F  0D                  JP      NEWF
0110  D501      NXT:      LD      FIRSTR,#1        ; DISABLE FIRST TONE REG
0112  BDEF7C    NXT2:     SBIT    4,PSW            ; ENABLE TIMER INTERRUPT
0115  BDEF6D              RBIT    5,PSW            ; RESET TPND FLAG
0118  5D                  LD      B,#RNGVAL        ; POINT TO RANDOM#
0119  BDEE7C              SBIT    TRUN,CNTRL       ; RESTART TIMER
011C  8F                  RETI                     ; RETURN
011D  C7        NEWF:     DRSZ    EXITR            ; TEST EXIT COUNT
011E  10                  JP      NF               ; NO
011F  C6                  DRSZ    LASTR            ; ENABLE LAST TONE
0120  01                  JP      LST
0121  06                  JP      NLST
0122  D709      LST:      LD      EXITR,#09        ; SET LAST TONE LENGTH
0124  BD0078              SBIT    0,TEMP           ; SET LAST TONE FLAG
0127  0F                  JP      NF2
0128  9DFD      NLST:     LD      A,SP             ; *** RESTORE STACK POINTER   ***
012A  9402                ADD     A,#002           ; *** FROM TIMER INTERRUPT    ***
012C  9CFD                X       A,SP             ; *** AND RETURN TO MAIN      ***
012E  8E                  RET
012F  BD0070    NF:       IFBIT   0,TEMP           ; LAST TONE ?
0132  04                  JP      NF2              ; YES
0133  AE                  LD      A,[B]            ; NEW TONE
0134  9404      NF4:      ADD     A,#04            ; INCR EXTRACTION VALUE
0136  A6                  X       A,[B]
0137  D80A      NF2:      LD      TCNTR,#TCNT      ; REINITIALIZE TONE TIME
0139  2110                JMP     NXT
013B  D820      NXT1:     LD      TCNTR,#TCNT1     ; ADJUST FIRST TONE LENGTH
013D  2112                JMP     NXT2
                          .END
```

Bomb



TL/DD/10716-5

Bomb (Continued)

```
                        ;
                        ;
                        ; THE SERIAL INPUT (SI) AND TIMER I/O (TIO) PINS
                        ; MUST BE TIED TO THE SERIAL OUTPUT (SO) PIN.
                        ; OUTPUT IS ON SO.
                        ; USE 20 MHz XTAL, 1 µs INSTR CYCLE FOR THIS DEMO.
                        ;
                        ; WRITTEN BY: JERRY LEVENTER
                        ; DATE:       OCTOBER 4, 1989
                        ;
                        ;
                              .TITLE BOMB8
                              .CHIP  820
                        ;
        00D5                  PORTGC  =  0D5          ; PORT G CONFIGURATION
        00E9                  SIOR    =  0E9          ; SIO SHIFT REGISTER
        00EA                  TMRLO   =  0EA          ; TIMER LOW BYTE
        00EB                  TMRHI   =  0EB          ; TIMER HIGH BYTE
        00EC                  TAULO   =  0EC          ; TIMER REGISTER LOW BYTE
        00ED                  TAUHI   =  0ED          ; TIMER REGISTER HIGH BYTE
        00EE                  CNTRL   =  0EE          ; CONTROL REGISTER
        00EF                  PSW     =  0EF          ; PSW REGISTER
        0004                  TRUN    =  4
        0005                  TPND    =  5
        0002                  BUSY    =  2
        0000                  GIE     =  0
                        ;
                        ; ****  EXPLOSION REGISTERS AND CONSTANTS ****
                        ;
                        ; SOME OF THE FOLLOWING REGISTERS USE THE DRSZ
                        ; TEST AND MUST THEREFORE BE INITIALIED TO AT
                        ; LEAST "1".
                        ;
        00F6                  LASTR   =  0F6          ; CONTROL LAST TONE
        0002                   LAST   =  002          ; LAST TONE CONSTANT
        0004                   LAST2  =  004          ; EXIT CONSTANT
        00F7                  EXITR   =  0F7          ; TOTAL TIME TILL EXIT
        0010                   EXIT   =  010          ; EXIT CONSTANT
        00F3                  RNGVAL  =  0F3          ; HOLDS CURRENT RING VALUE
        00F8                  TCNTR   =  0F8          ; TIME FOR EACH TONE FREQ
        000A                   TCNT   =  0A           ; CONSTANT VALUE
        00F9                  LUPREG  =  0F9          ; TONE COUNT INSIDE RING
        00FA                  LUPCNT  =  0FA          ; TONE COUNT OUTSIDE RING (VARIABLE)
        0000                  FLAG    =  000          ; FLAG REGISTER FOR SUBROUTINES
                        ;
        00FF                  TVALO   =  0FF
        001A                  TVALHI  =  01A
                        ;
                        ;**** WHISTLE REGISTERS AND CONSTANTS ****
                        ;
        002F                  WSLO    =  02F          ; TIMER VALUES
        0000                  WSLHI   =  000
        00FF                  MINFQ   =  0FF          ; FINAL (LOW FREQ) TIMER VALUE
                        ;
        00F0                  FCNTR   =  0F0          ; FREQUENCY COUNT REGISTER
        0000                   FCNT   =  000
```

Bomb (Continued)

```
                ;
                ; *********************************
                MAIN:
0000 DD2F               LD      SP,#02F          ; DEFAULT INITIALIZATION OF SP
0002 BD0078             SBIT    0,FLAG           ; SET SUBROUTINE FLAG
                                                 ; 1 = WHISTLE
                                                 ; 0 = EXPLOSION
0005 3157               JSR     WHISTLE
0007 BD0068     MAIN2:  RBIT    0,FLAG
000A 300D               JSR     BOMB
000C FF                 JP      .                ; *** STOP HERE OR REPEAT ***
                ; *********************************
                ;
000D BCD530     BOMB:   LD      PORTGC,#030      ; CONFIGURE "SO" AS OUTPUT
0010 BCEE8A             LD      CNTRL,#08A       ; SK = DIV BY 8, PWM ON
0013 BCEF11             LD      PSW,#011         ; ENABLE TIMER INTERRUPT
0016 BCEAFF             LD      TMRLO,#TVALO     ; INITIALIZE TIMER
0019 BCEB1A             LD      TMRHI,#TVALHI
001C BCECFF             LD      TAULO,#TVALO
001F BCED1A             LD      TAUHI,#TVALHI
0022 D602               LD      LASTR,#LAST      ; INITIALIZE LAST TONE FLAG
0024 D710               LD      EXITR,#EXIT      ; INITIALIZE EXIT COUNT
0026 D80A               LD      TCNTR,#TCNT      ; INITIALIZE TONE COUNT
0028 DA0A               LD      LUPCNT,#10       ; INITIALIZE FIRST TONE FREQUENCY
002A BD0069             RBIT    1,FLAG           ; RESET LAST TONE FLAG BIT
                ;
002D A1         NOISE:  SC                       ;
002E DEF3               LD      B,#RNGVAL        ; POINT TO RING VALUE
0030 9AFF               LD      [B+],#0FF        ; INIT TO ALL ONE'S
0032 9EFF               LD      [B],#0FF
0034 BDEE7C             SBIT    TRUN,CNTRL       ; START THE TIMER
0037 BEF6A      SHIFT:  RBIT    BUSY,PSW
003A 9CE9               X       A,SIOR           ; RANDOM # TO SIO
003C BDEF7A             SBIT    BUSY,PSW
003F 9DFA               LD      A,LUPCNT         ; RESTORE EXTRACTION COUNT
0041 9CF9               X       A,LUPREG
                ;
                ; *********************************
                ; RING COUNTER (17 STAGE)
                ;
                ; THIS IS A SEVENTEEN STAGE RING COUNTER (LINEAR
                ; FEEDBACK SHIFT REGISTER) WITH THE RRC COMMAND.
                ; THE COUNTER'S 14th AND 17th STAGES THROUGH AN
                ; EXCLUSIVE-OR SERVE AS THE FEEDBACK FUNCTION.
                ; THIS 14, 17 RING COUNTER BREAKS DOWN INTO
                ; 1 CYCLE OF [(2 ** 17) - 1] COUNTS. SINCE THE EXCLUSIVE OR
                ; OCCURS AFTER THE ROTATE, IT IS THE 15th AND CARRY
                ; STAGES THAT ARE XOR'D (BIT 2 AND CARRY).
                ;
                ; BEFORE ROTATE: 14 17
                ; AFTER ROTATE:  15 CARRY
                ;
                ;  CARRY BIT = STAGE ONE
                ;  LOW ORDER BIT = STAGE 17
```

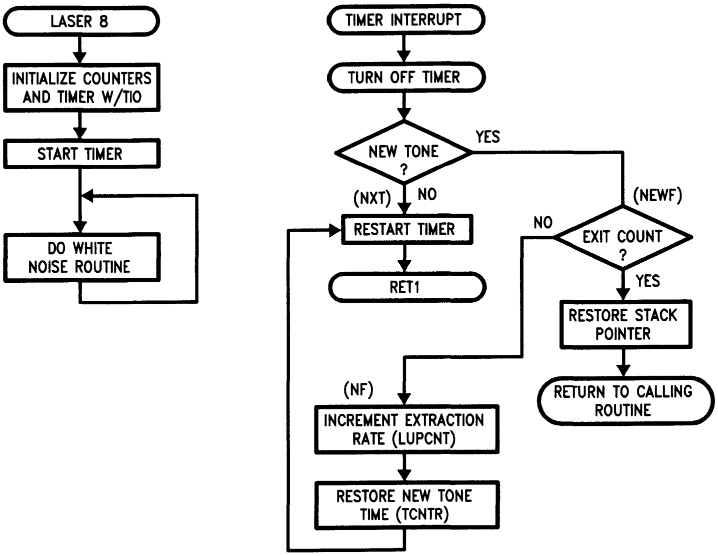
Explosion (Continued)

```
                     ; ***************************************
0043  AE             RING:  LD     A,[B]              ;  GET RANDOM #
0044  B0                    RRC    A                  ;  ROTATE UPPER BYTE
0045  A3                    X      A,[B-]
0046  AE                    LD     A,[B]
0047  B0                    RRC    A                  ;  ROTATE LOWER BYTE
0048  A2                    X      A,[B+]
0049  9804                  LD     A,#004             ;  PERFORM XOR
004B  85                    AND    A,[B]
004C  9200                  IFEQ   A,#000
004E  05                    JP     TSTLUP
004F  88                    IFC
0050  02                    JP     RC
0051  A1                    SC
0052  01                    JP     TSTLUP
0053  A0             RC:    RC
0054  C9             TSLUP: DRSZ   LUPREG             ;  POINT TO UPPER BYTE
0055  ED                    JP     RING               ;  EXTRACT THIS # ?
0056  AE                    LD     A,[B]              ;  NO, KEEP ROTATING
0057  2037                  JMP    SHIFT              ;  YES
                     ;
                     ; **** INTERRUPT ROUTINE ****
                     ;
      00FF                  .=     OFF
00FF  BDEF75                IFBIT  TPND,PSW           ;  TEST FOR EXIT
0102  01                    JP     TMOUT
0103  FF                    JP     .                  ;  ERROR
                     ;
0104 BDEE6C          TMOUT  RBIT   TRUN,CNTRL         ;  STOP TIMER
0107 BD0070                 IFBIT  0,FLAG             ;  BRANCH TO ROUTINE
010A  213B                  JMP    WSINT              ;  SET = WHISTLE, RESET = EXPLOSION
                     ;
010C DEFA                   LD     B,#LUPCNT
010E C8                     DRSZ   TCNTR              ;  TEST FOR NEW TONE
010F 01                     JP     NXT                ;  NO, DON'T INCREMENT LUPCNT
0110 0C                     JP     NEWF               ;  YES
0111 BDEF7C          NXT:   SBIT   4,PSW              ;  ENABLE TIMER INTRRUPT
0114 BDEF6D                 RBIT   5,PSW              ;  RESET TIMER PENDING FLAG
0117 DEF3                   LD     B,#RNGVAL          ;  POINT TO RANDOM #
0119 BDEE7C                 SBIT   TRUN,CNTRL         ;  RESTART TIMER
011C 8F                     RETI                      ;  RETURN TO RING COUNTER
011D C7              NEWF:  DRSZ   EXITR              ;  DO LAST TONE ?
011E 10                     JP     NF                 ;  NO
011F  C6                    DRSZ   LASTR              ;  IS LAST TONE DONE?
0120  01                    JP     LST                ;  NO
0121  06                    JP     NLST               ;  YES, RETURN TO MAIN
0122  D704           LST:   LD     EXITR,#LAST2       ;  LENGTHEN THE LAST TONE
0124  BD0079                SBIT   1,FLAG             ;  SET LAST TONE FLAG
0127  0F                    JP     NF2
0128  9DFD           NLST:  LD     A,SP               ;  ** RESTORE STACK POINTER **
012A  9402                  ADD    A,#002             ;  ** AND RETURN TO MAIN    **
012C  9CFD                  X      A,SP
012E  8E                    RET
                     ;
012F  BD0071         NF:    IFBIT  1,FLAG             ;  LAST TONE ?
0132  04                    JP     NF2                ;  YES, DON'T INCREMENT LUPCNT
0133  AE                    LD     A,[B]              ;  NEW TONE
0134  9404                  ADD    A,#04              ;  INCR EXTRACT COUNT (LUPCNT)
0136  A6                    X      A,[B]
0137  D80A           NF2:   LD     TCNTR,#TCNT        ;  REINITIALIZE TONE TIME
0139  2111                  JMP    NXT
```

Explosion (Continued)

```
                          ; *************************************
013B  BDF075    WSINT:     IFBIT   5,FCNTR          ; READY FOR NEW FREQUENCY ?
013E  06                   JP      TM               ; YES
013F  9DF0                 LD      A,FCNTR          ; NO, INCREMENT COUNT
0141  8A                   INC     A
0142  9CF0                 X       A,FCNTR
0144  8D                   RETSK                    ; NO, RETURN TO WHISTLE
0145  D000      TM:        LD      FCNTR,#FCNT      ; RESET NEW FREQUENCY COUNT
0147  DEEC                 LD      B,#TAULO         ; POINT TO AUTORELOAD REG
0149  AE                   LD      A,[B]            ; CHANGE FREQUENCY
014A  92FF                 IFEQ    A,#MINFQ         ; TIMER = MIN FREQ ?
014C  03                   JP      DONE
014D  8A                   INC     A
014E  A6                   X       A,[B]            ; STORE FREQ IN AUTO RELOAD
014F  8D                   RETSK
0150  9DFD      DONE:      LD      A,SP             ; ** RESTORE STACK POINTER **
0152  9402                 ADD     A,#002           ; ** AND RETURN TO MAIN    **
0154  9CFD                 X       A,SP
0156  8E                   RET
                          ; *********************************
0157  BCD508    WHISTLE:   LD      PORTGC,#008      ; TIO PIN (G3) AS OUTPUT
015A  BCEEA2               LD      CNTRL,#0A2       ; PWM WITH TIO TIGGLE, 8Tc
015D  D000                 LD      FCNTR,#FCNT      ; INIT FREQ COUNTER
015F  BCEA2F               LD      TMRLO,#WSLO      ; WHISTLE VALUE FOR TIMER
0162  BCEB00               LD      TMRHI,#WSLHI
0165  BCEC2F               LD      TAULO,#WSLO
0168  BCED00               LD      TAUHI,#WSLHI
                          ;
016B  BCEF11    BEGIN      LD      PSW,#011         ; ENTI, GIE = 1, TPND = 0
016E  BDEE7C               SBIT    TRUN,CNTRL       ; START TIMER
0171  FF                   JP      .                ; LOOP UNTIL TIMER INTERRUPT
0172  F8                   JP      BEGIN            ; RETURN HERE FROM INTERRUPT
                           .END
```

Laser Gun



TL/DD/10716–6

Laser Gun (Continued)

```
                        ;
                        ; TIMER INTERRUPT IS USED.
                        ; THE SERIAL OUTPUT PIN (SO) AND THE TIO PIN MUST BE
                        ; TIED TOGETHER.
                        ; OUTPUT IS ON SO AND TIO.
                        ;
                        ; TO ALTER THE DURATION OF THE LASER SHOT CHANGE THE
                        ; "EXIT" VALUE, HOWEVER, DO NOT EXCEED 03F HEX.
                        ; THE TIMER VALUES (TVALO, TVALHI) COMBINED WITH THE
                        ; TONE COUNT (TNCTR) CAN BE ADJUSTED TO ACHIEVE A
                        ; VARIETY OF SOUNDS.
                        ;
                        ; USE 20 MHz XTAL, 1 µs INSTR CYCLE TIME FOR THIS DEMO.
                        ;
                        ;
                        ; WRITTEN BY: JERRY LEVENTER
                        ; DATE:       OCTOBER 4, 1989
                        ;
                        ;
                                .TITLE LASER8
                                .CHIP   820
                        ;
          00D5                  PORTGC  =   0D5          ; PORT G CONFIGURATION
          00E9                  SIOR    =   0E9          ; SIO SHIFT REGISTER
          00EA                  TMRLO   =   0EA          ; TIMER LOW BYTE
          00EB                  TMRHI   =   0EB          ; TIMER HIGH BYTE
          00EC                  TAULO   =   0EC          ; TIMER REGISTER LOW BYTE
          00ED                  TAUHI   =   0ED          ; TIMER REGISTER HIGH BYTE
          00EE                  CNTRL   =   0EE          ; CONTROL REGISTER
          00EF                  PSW     =   0EF          ; PSW REGISTER
          0004                  TRUN    =   4
          0005                  TPND    =   5
          0002                  BUSY    =   2
                        ;
                        ; ****  SPECIAL REGISTERS AND COUNTERS ****
                        ; ANY REGISTER THAT IS USED FOR THE DRSZ TEST,
                        ; MUST BE INITIALIZED TO AT LEAST "1".
                        ;
          00F7                  EXITR   =   0F7          ; ROUTINE DURATION REGISTER
          003F                   EXIT   =   03F          ;  EXIT CONSTANT
          0002                  RNGVAL  =   002          ; HOLDS CURRENT RANDOM #
          00F8                  TCNTR   =   0F8          ; TONE DURATION REGISTER
          0020                   TCNT   =   020          ;  TONE CONSTANT
          00F9                  LUPREG  =   0F9          ; EXTRACTION RATE REGISTER
          0003                   XTRCT  =   003          ;  EXTRACT CONSTANT
          00FA                  LUPCNT  =   0FA          ; EXTRACTON VARIABLE REGISTER
          00FF                  TVALO   =   0FF          ; TIMER VALUES
          0000                  TVALHI  =   000
                        ;
                        ;********************************
                        ;****    BEGIN PROGRAM HERE    ****
                        ;********************************
                        ;
0000                    MAIN:   LD      SP,#02F      DD2F; DEFAULT INITIALIZATION OF SP
0002                    LUP:    LD      EXITR,#EXIT  D73F; INITIALIZE SHOT DURATION
0004                            JSR     LASER8       3018; *** LASER CALLING ROUTINE ***
0006                            LD      EXITR,#EXIT  D73F
0008                            JSR     LASER8       3018
```

Laser Gun (Continued)

```
000A  D73F                LD      EXITR,#EXIT
000C  3018                JSR     LASER8
000E  D715                LD      EXITR,#015          ; EXIT COUNT CAN BE INITIALIZED
0010  3018                JSR     LASER8              ; INSIDE PROGRAM IF SHOT RATE
0012  D715                LD      EXITR,#015          ; DOES NOT CHANGE.
0014  3018                JSR     LASER8
0016  B8                  NOP
0017  EA                  JP      LUP                 ; **** LOOP FOR DEMO ****
                ;
0018  BCD530    LASER8:   LD      PORTGC,#030
001B  BCEEAA              LD      CNTRL,#0AA          ; SK = DIV BY 8, PWM/TIO TIMER
001E  BCEF11              LD      PSW,#011            ; ENABLE TIMER INTERRUPT
0021  BCEAFF              LD      TMRLO,#TVALO        ; INITIALIZE TIMER
0024  BCEB00              LD      TMRHI,#TVALHI
0027  BCECFF              LD      TAULO,#TVALO
002A  BCED00              LD      TAUHI,#TVALHI
                ;         LD      EXITR,#EXIT         ; INITIALIZE EXIT COUNT
002D  D820                LD      TCNTR,#TCNT         ; INITIALIZE TONE COUNT
002F  DA03                LD      LUPCNT,#XTRCT       ; INITIALIZE EXTRACTION RATE
0031  BDEE7C              SBIT    TRUN,CNTRL          ; START TIMER
0034  A1        NOISE:    SC                          ; INIT. STAGE 1
0035  5D                  LD      B,#RNGVAL           ; POINT TO RANDOM NUMBER
0036  9EFF                LD      [B],#0FF            ; INIT RANDOM #
0038  9CE9      SHIFT:    X       A,SIOR              ; LOAD AND START SIOR
003A  BDEF7A              SBIT    BUSY,PSW
003D  9DFA                LD      A,LUPCNT            ; RESTORE EXTRACTION COUNT
003F  9CF9                LD      A,LUPREG
                ;
                ; **********************************
                ; RING COUNTER
                ;
                ; THIS IS A NINE STAGE RING COUNTER (LINEAR
                ; FEEDBACK SHIFT REGISTER) WITH THE RRC COMMAND.
                ; THE COUNTER'S 8th AND 9th STAGES, THROUGH AN
                ; EXCLUSIVE-OR SERVE AS THE FEEDBACK FUNCTION.
                ; SINCE THE EXCLUSIVE OR OCCURS AFTER THE ROTATE,
                ; IT IS THE 1st AND 9th STAGES THAT ARE XOR'D,
                ; (THE CARRY FLAG AND BIT 0).
                ;
                ;  CARRY BIT = STAGE 1
                ;  LOW ORDER BIT = STAGE 9
                ; ***********************************
0041  AE        RING:     LD      A,[B]               ; GET RANDOM #
0042  B0                  RRC     A                   ; ROTATE UPPER BYTE
0043  A6                  X       A,[B]               ;
0044  9800                LD      A,#000              ; PERFORM XOR
0046  85                  AND     A,[B]
0047  9200                IFEQ    A,#000
0049  05                  JP      TSTLUP
004A  88                  IFC
004B  02                  JP      RC
004C  A1                  SC
004D  01                  JP      TSTLUP
004E  A0        RC:       RC
```

Laser Gun (Continued)

```
004F  C9        TSTLUP: DRSZ    LUPREG             ; EXTRACT THIS # ?
0050  F0                JP      RING               ; NO, KEEP ROTATING
0051  AE                LD      A,[B]              ; YES
0052  E5                JP      SHIFT
                ;
                ; **** TIMER INTERRUPT ROUTINE ****
                ;
      00FF              .=      0FF
00FF  BDEF75            IFBIT   TPND,PSW           ; TEST TIMER PND FLAG
0102  01                JP      TMOUT
0103  FF                JP      .                  ; ERROR
                ;
0104  BDEE6C    TMOUT:  RBIT    TRUN,CNTRL         ; STOP TIMER
0107  DEFA              LD      B,#LUPCNT
0109  C8                DRSZ    TCNTR              ; TEST FOR NEW TONE
010A  01                JP      NXT                ; NO
010B  0B                JP      NEWF
010C  BDEF7C    NXT:    SBIT    4,PSW              ; ENABLE TIMER INTERRUPT
010F  BDEF6D            RBIT    5,PSW              ; RESET TPND FLAG
0112  5D                LD      B,#RNGVAL          ; POINT TO RANDOM #
0113  BDEE7C            SBIT    TRUN,CNTRL         ; RESTART TIMER
0116  8F                RETI                       ; RETURN
0117  C7        NEWF:   DRSZ    EXITR              ; EXIT COUNT = 0 ?
0118  07                JP      NF                 ; NO
0119  9DFD      NLST:   LD      A,SP               ; *** RESTORE STACK POINTER ***
011B  9402              ADD     A,#002             ; *** FROM TIMER INTERRUPT   ***
011D  9CFD              X       A,SP               ; *** AND RETURN TO MAIN     ***
011F  8E                RET
0120  AE        NF:     LD      A,[B]              ; NEW TONE
0121  9404              ADD     A,#04              ; INCR EXTRACTION VALUE
0123  A6                X       A,[B]
0124  D820              LD      TCNTR,#TCNT        ; REINITIALIZE TONE TIME
0126  E5                JP      NXT
                        .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 pushbutton 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

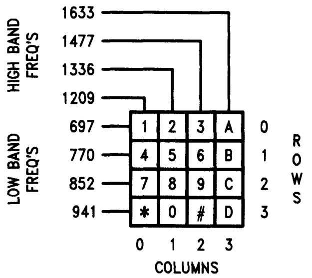
The matrix for selecting the high and low band frequencies associated with each key is shown in *Figure 1*. Each key is uniquely referenced by selecting one of the four low band frequencies associated with the matrix rows, coupled with selecting one of the four high band frequencies associated with the 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:

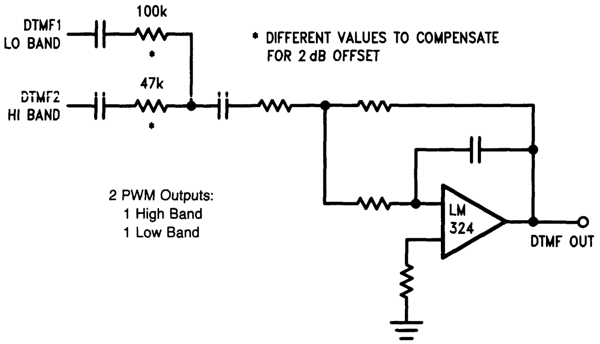
```
        LD      B,#R1
LUP:    X       A,[B]
        LD      A,[B,]
        LAID
        X       A,[B+]
        DEC     A
        IFBNE   #4
        JP      LUP
```

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.

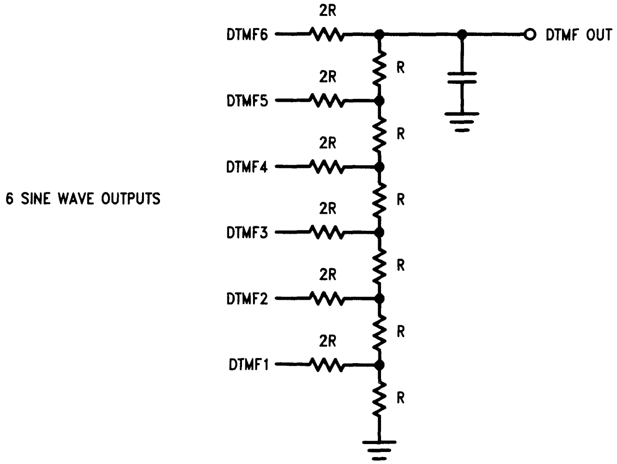
The KBRDEC subroutine provides a low true decoding of the DTMF keyboard input and assumes that the keyboard input has been encoded in a low true column/row format, with the columns of the keyboard being sequentially strobed.

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

```
;
; DTMF GENERATION WITH A 3.58 MHZ      VERNE H. WILSON
;        CRYSTAL FOR COP820C/840C         10/28/89
;
; DTMF - DUAL TONE MULTIPLE FREQUENCY
;
; PROGRAM NAME: DTMF35.MAC
;
         .TITLE DTMF35
         .CHIP 840
;
;
; THIS DTMF PROGRAM IS BASED ON A COP820C/840C RUNNING
; WITH A CKI CLOCK OF 3.579545 MHZ (TV COLOR CRYSTAL
; FREQUENCY) IN DIVIDE BY 10 MODE, FOR AN INSTRUCTION
; CYCLE TIME OF 2.7936511 MICROSECONDS.
;
; THIS PROGRAM CONTAINS THREE SUBROUTINES, ONE FOR A
; LOW TRUE ROW/COLUMN DTMF KEYBOARD DECODING (KBRDEC),
; AND THE OTHER TWO (DTMFGP, DTMFLP) FOR ALTERNATE
; METHODS OF DTMF GENERATION.
;
; KEYBOARD INPUT DATA IS IN ACCUMULATOR WITH A
;    LOW TRUE FORMAT AS FOLLOWS:
;        BITS 7 TO 4 : LOW TRUE COLUMN VALUE (E,D,B,7)
;        BITS 3 TO 0 : LOW TRUE ROW VALUE (E,D,B,7)
;
; ASSUMPTION MADE THAT COLUMN STROBES (LOW TRUE) ARE
;    OUTPUT, WHILE ROW VALUES (LOW TRUE) ARE INPUT.
;
; THE FIRST METHOD OF DTMF GENERATION CONSISTS OF
; GENERATING TWO PWM OUTPUTS ON THE G PORT G2 AND G3
; OUTPUT PINS. THESE TWO OUTPUTS NEED TO BE MIXED
; EXTERNALLY WITH AN APPROPIATE LM324 OP AMP FEEDBACK
; CIRCUIT TO GENERATE THE DTMF.
;
; THE SECOND METHOD OF DTMF GENERATION USES ROM LOOKUP
; TABLES TO SIMULATE THE TWO DTMF SINE WAVES AND
; COMBINES THEM ARITHMETICALLY. THE RESULT IS OUTPUT ON
; THE LOWER SIX BITS OF THE L PORT (L0 - L5). THESE SIX
; OUTPUTS ARE COMBINED EXTERNALLY WITH A LADDER NETWORK
; TO GENERATE THE DTMF.
;
; THE SECOND DTMF GENERATION METHOD USES APPROXIMATELY
; THREE TIMES AS MUCH ROM CODE (INCLUDING PROGRAM CODE
; AND ROM TABLES) AS THE FIRST METHOD, BUT HAS THE
; ADVANTAGE OF ELIMINATING THE COST OF THE EXTERNAL
; ACTIVE COMPONENT (LM324 OR EQUIVALENT).
;
; BOTH OF THE DTMF SUBROUTINES GENERATE THEIR OUTPUTS
; FOR A PERIOD OF 100 MILLISECONDS.
```

TL/DD/10740-1

```
                    ;
                    ; DECLARATIONS:
                    ;
     0000                     KDATA  = 0          ; *** KEYBOARD DATA ***
     00D0                     PORTLD = 0D0        ; PORTL DATA REG
     00D1                     PORTLC = 0D1        ; PORTL CONFIG REG
     00D4                     PORTGD = 0D4        ; PORTG DATA REG
     00D5                     PORTGC = 0D5        ; PORTG CONFIG REG
     00D7                      PORTI = 0D7        ; PORTI INPUT PINS
     00DC                      PORTD = 0DC        ; PORTD REG
     00EA                      TMRLO = 0EA        ; TIMER LOW COUNTER
     00EB                      TMRHI = 0EB        ; TIMER HIGH COUNTER
     00EC                      TAULO = 0EC        ; TMR AUTORELOAD REG LO
     00ED                      TAUHI = 0ED        ; TMR AUTORELOAD REG HI
     00EE                      CNTRL = 0EE        ; CONTROL REG
     00EF                        PSW = 0EF        ; PROC STATUS WORD
     00F0                         R0 = 0F0        ; LB FREQ LOOP COUNTER
     00F1                         R1 = 0F1        ; LB FREQ LOOP COUNT
     00F2                         R2 = 0F2        ; LB FREQ Q COUNT
     00F3                         R3 = 0F3        ; LB FREQ R COUNT
                    ;
0000 DD2F           START:    LD        SP,#02F       ; INITIALIZE STACK PTR
                    ;
                    ;                                 KEYBOARD HEX DIGIT MATRIX
                    ;                                          1  2  3  A
0002 DEDC                     LD        B,#PORTD    ;          4  5  6  B
0004 9E00                     LD        [B],#0      ;          7  8  9  C
0006 A0             LOOP:     RC                    ;          *  0  #  D
0007 AE                       LD        A,[B]       ; DTMF TEST LOOP
0008 9405                     ADD       A,#5        ; SEQUENCE IS 1,5,9,D,4,
000A A6                       X         A,[B]       ; 8,#,A,7,0,3,B,*,2,6,C
000B 6C                       RBIT      4,[B]       ; HEX MATRIX TO LOOKUP
000C 9420                     ADD       A,#020      ;    TABLE FOR LOW TRUE
000E A4                       LAID                  ;    COLUMN/ROW INPUT TO
000F 3210                     JSR       KBRDEC      ;    KBRDEC SUBROUTINE
0011 A1                       SC                    ; SET C IF NOT SINGLE KEY
0012 DED7                     LD        B,#PORTI    ; TEST BIT 0 OF PORTI TO
0014 70                       IFBIT     0,[B]       ;    DETERMINE WHICH
0015 03                       JP        BYPA        ;    DTMF SUBROUTIINE
0016 3040                     JSR       DTMFGP      ; TWO PWM OUTPUTS ON
0018 02                       JP        BYPB        ;    G PORT PINS G2,G3
0019 308E           BYPA:     JSR       DTMFLP      ; SIX LADDER OUTPUTS ON
                                                    ;    L PORT PINS L0 - L5
001B DEDC           BYPB:     LD        B,#PORTD    ; D0 WILL TOGGLE FOR EACH
001D E8                       JP        LOOP        ;    CALL OF SUBROUTINE
                    ;
                    ;
                    ;
```

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

```
;
;
; HALF PERIODS FOR THE 8 DTMF FREQUENCIES (697,770,852,
;     941,1209,1336,1477, AND 1633 KHZ) ARE 257,232,
;     210,190,148,134,121, AND 110 Tc's RESPECTIVELY
;
; THE 100 MSEC DIVIDED BY HALF PERIOD QUOTIENTS ARE
;     139,154,170,188,241,267,295, AND 325 RESPECTIVELY
;
; THE 100 MSEC DIVIDED BY HALF PERIOD REMAINDERS ARE
;     72,67,95,75,127,17,100, AND 45 RESPECTIVELY
;
;
;
;
; BINARY FORMAT FOR THE HEX DIGIT KEY VALUE FROM THE
;    KBRDEC SUBROUTINE IS 0000RRCC,
;       WHERE  -  RR IS ROW SELECT (LB FREQUENCIES)
;              -  CC IS COLUMN SELECT (HB FREQUENCIES)
;
; FREQUENCY VECTORS (HB & LB) FOR FREQ PARAMETER TABLE
;    MADE FROM KEY VALUE
;
; HB FREQ VECTORS (4) END WITH 00 FOR TIMER COUNTS,
;    WHERE VECTOR FORMAT IS 0011CC00
;
; LB FREQUENCY VECTORS (12) END WITH:
;    11 FOR HALF PERIOD LOOP COUNTS,
;         WHERE VECTOR FORMAT IS 0011RR11
;    10 FOR 100 MSEC DIVIDED BY HALF PERIOD QUOTIENTS,
;         WHERE VECTOR FORMAT IS 0011RR10
;    01 FOR 100 MSEC DIVIDED BY HALF PERIOD REMAINDERS,
;         WHERE VECTOR FORMAT IS 0011RR01
;
; FREQ PARAMETER TABLE AT HEX 003*  (REQUIRED LOCATION)
;
;
;
;                KEY VALUE
;                 0000RRCC
;
;   TIMER        T    CC00
;   R1           F    RR11
;   R2           Q    RR10
;   R3           R    RR01
;
;
;
```

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

```
0063 A2                       X        A,[B+]      ; HB FREQ PARAMETER TO
0064 9A00                     LD       [B+],#0     ;   AUTORELOAD REGISTER
0066 9EB0                     LD       [B],#0B0    ; START TIMER PWM
0068 DED4                     LD       B,#PORTGD
006A DCF1                     LD       X,#R1
006C BB             LUP1:     LD       A,[X-]
006D 72                       IFBIT    2,[B]       ; TEST LB OUTPUT
006E 03                       JP       BYP1
006F B2                       X        A,[X+]
0070 7A                       SBIT     2,[B]       ; SET LB OUTPUT
0071 03                       JP       BYP2
0072 B8             BYP1:     NOP
0073 6A                       RBIT     2,[B]       ; RESET LB OUTPUT
0074 B2                       X        A,[X+]
0075 C2             BYP2:     DRSZ     R2          ; DECR. QUOT. COUNT
0076 01                       JP       LUP2
0077 0E                       JP       FINI        ; Q COUNT FINISHED
0078 C0             LUP2:     DRSZ     R0          ; DECR. F COUNT
0079 FE                       JP       LUP2        ; LB (HALF PERIOD)
                    ;
007A BE                       LD       A,[X]       ; ******************
007B 921F                     IFEQ     A,#31       ; BALANCE        ***
007D EE                       JP       LUP1        ; LOW BAND       ***
007E B8                       NOP                  ; FREQUENCY      ***
007F B8                       NOP                  ; RESIDUE        ***
0080 9226                     IFEQ     A,#38       ; DELAY FOR      ***
0082 E9                       JP       LUP1        ; EACH OF THE    ***
0083 A4                       LAID                 ; FOUR LOW BAND  ***
0084 B8                       NOP                  ; FREQUENCIES    ***
0085 E6                       JP       LUP1        ; ******************
0086 C3             FINI:     DRSZ     R3        ; DECR. REMAINDER COUNT
0087 FE                       JP       FINI      ; REM. COUNT NOT FINISHED
0088 BDEE6C                   RBIT     4,CNTRL   ; STOP TIMER
008B 6B                       RBIT     3,[B]     ; OPTIONAL CLR HB OUTPUT
008C 6A                       RBIT     2,[B]     ; OPTIONAL CLR LB OUTPUT
008D 8E                       RET                ; RETURN FROM SUBROUTINE
                    ;
                    ;
                    ;
```

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

```
                          ; KEY 3
    00C0 31                             .BYTE     49,02D,23,096
    00C1 2D
    00C2 17
    00C3 96
                          ;
                          ; KEY A
    00C4 31                             .BYTE     49,02D,21,0AD
    00C5 2D
    00C6 15
    00C7 AD
                          ;
                          ; KEY 4
    00C8 0B                             .BYTE     11,05E,7,07C
    00C9 5E
    00CA 07
    00CB 7C
                          ;
                          ; KEY 5
    00CC 0B                             .BYTE     11,05E,19,083
    00CD 5E
    00CE 13
    00CF 83
                          ;
                          ; KEY 6
    00D0 0B                             .BYTE     11,05E,23,096
    00D1 5E
    00D2 17
    00D3 96
                          ;
                          ; KEY B
    00D4 0B                             .BYTE     11,05E,21,0AD
    00D5 5E
    00D6 15
    00D7 AD
                          ;
                          ; KEY 7
    00D8 0A                             .BYTE     10,069,7,07C
    00D9 69
    00DA 07
    00DB 7C
                          ;
                          ; KEY 8
    00DC 0A                             .BYTE     10,069,19,083
    00DD 69
    00DE 13
    00DF 83
                          ;
                          ; KEY 9
    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
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
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
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
                  ;
                  ;
                  ; LF770:   1 CYCLE OF SINE WAVE SPREAD
                  ;                  ACROSS 11 TIMING LOOP (LUP42) CYCLES
                  ;
                  ;          FREQ. = 1 / (11 X 117 1/3) = 774.79 HZ
                  ;          ERROR = (774.79 - 770) / 770  = + 0.62 %
                  ;
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
0168 06               .BYTE     6
                  ;
```

TL/DD/10740-14

```
                    ;
                    ; LF852:   1 CYCLE OF SINE WAVE SPREAD
                    ;                  ACROSS 10 TIMING LOOP (LUP42) CYCLES
                    ;
                    ;          FREQ. = 1 / (10 X 117 1/3) = 852.27 HZ
                    ;          ERROR = (852.27 - 852) / 852  = + 0.03 %
                    ;
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
                    ;
                    ;
                    ; LF941:   1 CYCLE OF SINE WAVE SPREAD
                    ;                  ACROSS 9 TIMING LOOP (LUP42) CYCLES
                    ;
                    ;          FREQ. = 1 / (9 X 117 1/3) = 946.97 HZ
                    ;          ERROR = (946.97 - 941) / 941  = + 0.63 %
                    ;
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
                    ;
                    ;
                    ;
                    ; HF1209:  1 CYCLE OF SINE WAVE SPREAD
                    ;                  ACROSS 7 TIMING LOOP (LUP42) CYCLES
                    ;
                    ;          FREQ. = 1 / (7 X 117 1/3) = 1217.53 HZ
                    ;          ERROR = (1217.53 - 1209) / 1209  = + 0.71 %
                    ;
017C 10                       .BYTE     16,29,32,23,9,0,3
017D 1D
017E 20
017F 17
0180 09
0181 00
0182 03
                    ;
```

TL/DD/10740-15

```
                          ;
                          ; HF1336:  3 CYCLES OF SINE WAVE SPREAD
                          ;                  ACROSS 19 TIMING LOOP (LUP42) CYCLES
                          ;
                          ;          FREQ. = 3 / (19 X 117 1/3) = 1345.69 HZ
                          ;          ERROR = (1345.69 - 1336) / 1336  = + 0.73 %
                          ;
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
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
                          ;
                          ;
                          ; HF1477:  4 CYCLES OF SINE WAVE SPREAD
                          ;                  ACROSS 23 TIMING LOOP (LUP42) CYCLES
                          ;
                          ;          FREQ. = 4 / (23 X 117 1/3) = 1482.21 HZ
                          ;          ERROR = (1482.21 - 1477) / 1477  = + 0.35 %
                          ;
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
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
01AA 12                         .BYTE     18,3,2
01AB 03
01AC 02
                  ;
                  ;
                  ; HF1633:  4 CYCLES OF SINE WAVE SPREAD
                  ;                   ACROSS 21 TIMING LOOP (LUP42) CYCLES
                  ;
                  ;          FREQ. = 4 / (21 X 117 1/3) = 1623.38 HZ
                  ;          ERROR = (1633 - 1623.38) / 1633  = - 0.59 %
                  ;
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
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
01C1 01                         .BYTE     1
                  ;
                  ;
                  ;
```

TL/DD/10740-17

```
                       .FORM
                 ;
                 ; DTMF KEYBOARD DECODE SUBROUTINE (KBRDEC)
                 ;
                 ; KEYBOARD INPUT DATA IS IN ACCUMULATOR WITH A
                 ;    LOW TRUE FORMAT AS FOLLOWS:
                 ;        BITS 7 TO 4 : LOW TRUE COLUMN VALUE (E,D,B,7)
                 ;        BITS 3 TO 0 : LOW TRUE ROW VALUE (E,D,B,7)
                 ;
                 ; ASSUMPTION MADE THAT COLUMN STROBES (LOW TRUE) ARE
                 ;    OUTPUT, WHILE ROW VALUES (LOW TRUE) ARE INPUT.
                 ;
                 ; LOW TRUE COLUMN/ROW INPUT DIGIT IN ACCUMULATOR IS
                 ;     TRANSFORMED INTO A DTMF HEX DIGIT KEY VALUE
                 ;
                 ; TABLE LOOKUP TRANSFORMATION CHECKS FOR MULTIPLE KEYS,
                 ;     NO KEY, OR NO COLUMN SELECT, AND THEN PRODUCES
                 ;     A DTMF HEX DIGIT KEY VALUE WITH A BINARY FORMAT
                 ;     OF 0000RRCC FOR A SINGLE KEY INPUT,
                 ;     WHERE   -  RR IS LOW BAND (LB) FREQUENCY SELECT
                 ;                - CC IS HIGH BAND (HB) FREQUENCY SELECT
                 ;
                 ; KBRDEC SUBROUTINE IS EXITED WITH A RETURN (RET)
                 ;     COMMAND TO INDICATE MULTIPLE KEYS, NO KEY,
                 ;     OR NO COLUMN SELECT
                 ;
                 ; KBRDEC SUBROUTINE IS EXITED WITH A RETURN AND SKIP
                 ;     (RETSK) COMMAND TO INDICATE A SINGLE KEY ENTRY
                 ;
                 ;
    0200                .=0200
                 ;
                 ; LOW TRUE TRANSLATION TABLE  - ONLY E,D,B,7 ACCEPTABLE
                 ;
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
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
                 ;
```

TL/DD/10740-18

```
                   ;
0210 5F            KBRDEC:  LD     B,#KDATA
0211 A6                     X      A,[B]      ; STORE LOW TRUE
0212 AE                     LD     A,[B]      ;   COLUMN/ROW VALUE
0213 95F0                   AND    A,#0F0     ; EXTRACT LOW TRUE COLUMN
0215 65                     SWAP   A          ;   & PUT IN LOWER NIBBLE
0216 A4                     LAID              ; 0000CC00 FROM TABLE
0217 A0                     RC                ; SHIFT TABLE VALUE DOWN
0218 B0                     RRC    A          ;   TWO BITS TO PRODUCE
0219 B0                     RRC    A          ;   000000CC
021A A6                     X      A,[B]      ; STORE RESULT
021B 950F                   AND    A,#0F      ; EXTRACT LOW TRUE ROW
021D A4                     LAID              ; 0000RR00 FROM TABLE
021E 84                     ADD    A,[B]      ; ADD TO PRODUCE 0000RRCC
021F 930F                   IFGT   A,#0F      ; RETURN IF MULTIPLE KEYS,
0221 8E                     RET               ;    NO KEYS, OR NO COLUMN
0222 8D                     RETSK             ; RETURN AND SKIP
                                              ;    IF SINGLE KEY

                   ;
                   ;
                   ;
                            .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

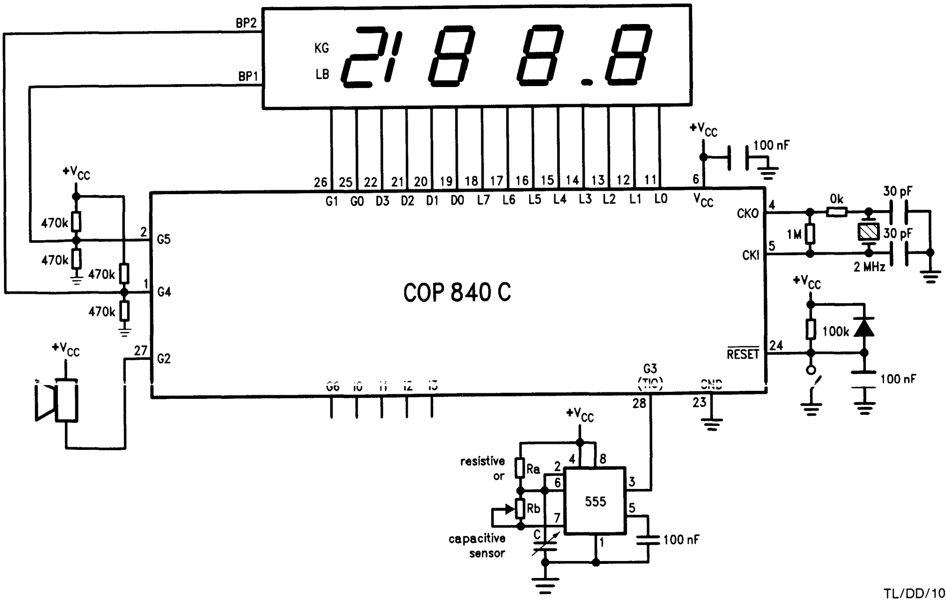
Today's most popular digital scales all have the following characteristics:

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.*



TL/DD/10788–1

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).

$$T = (startvalue - endvalue) *tc/N$$

```
;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

```
        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.

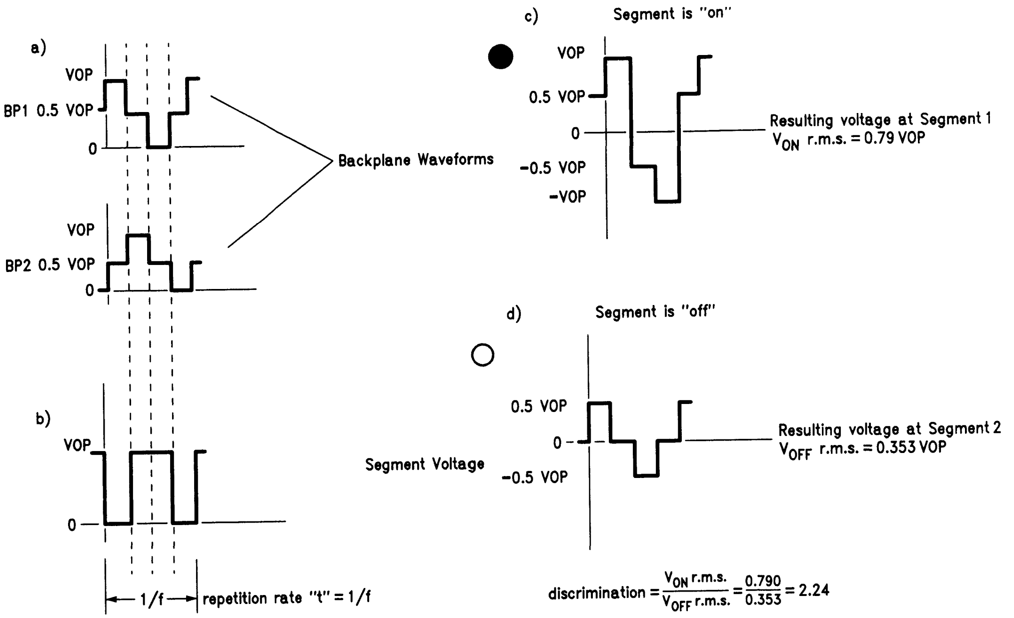
The timing is done using the COP800's 16-bit timer in the PWM autoload mode. The timer and the assigned 16-bit autoload register are preset with proper values. By setting the TRUN-flag in the CNTRL-register the timer is decremented each instruction cycle. A flag (TPND) is set at underflow and the timer 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).

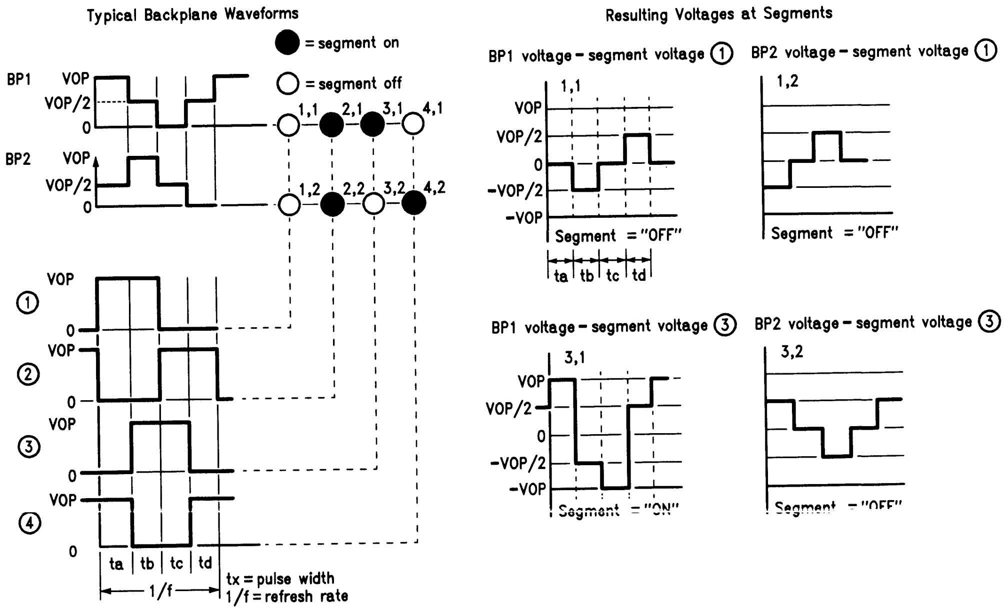
Figure 3 shows the four different waveforms which must be generated to meet all possible combinations of two segments connected to the same driving terminal (off-off, on-off, off-on, on-on).

Figure 4 shows the internal segment and backplane connections for a typical 2-way mux LCD.



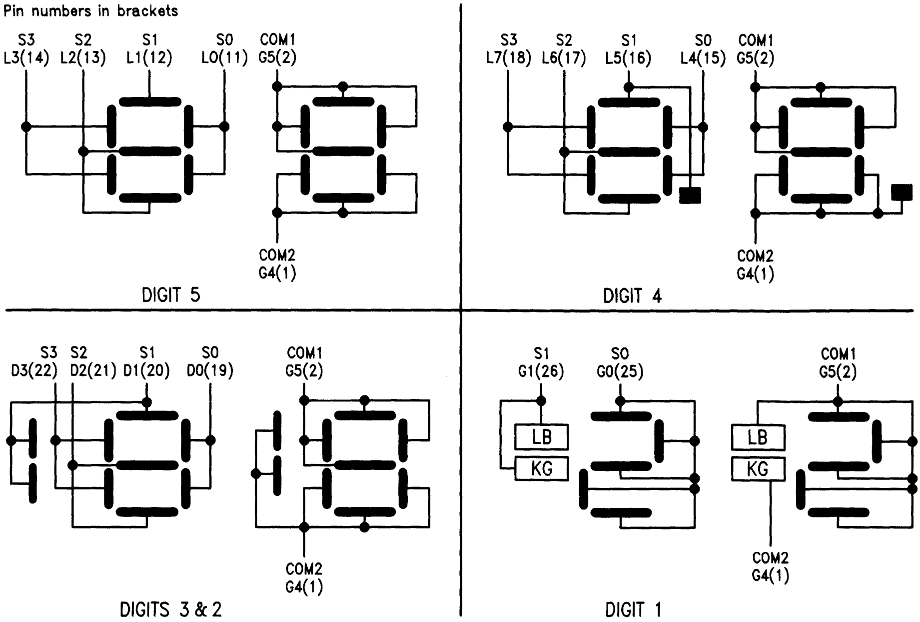
TL/DD/10788-7

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.

```
;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

```
$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

```
        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

```
          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.

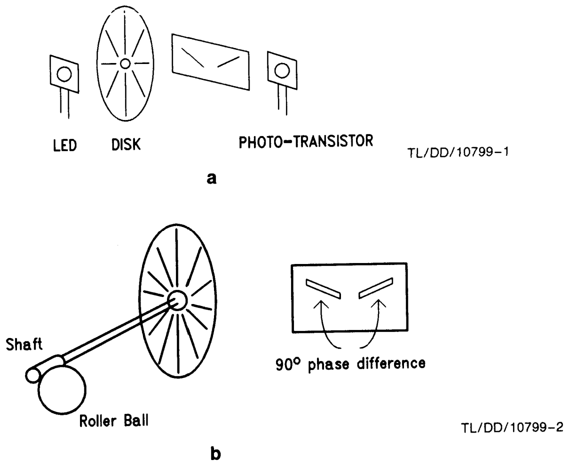
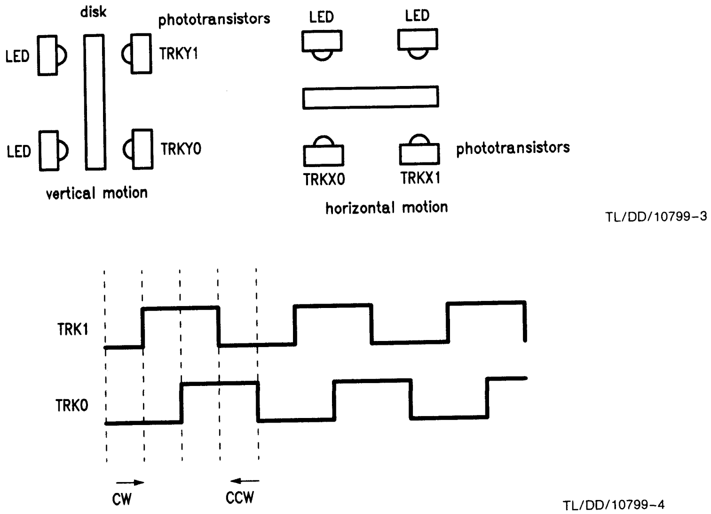


FIGURE 1



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

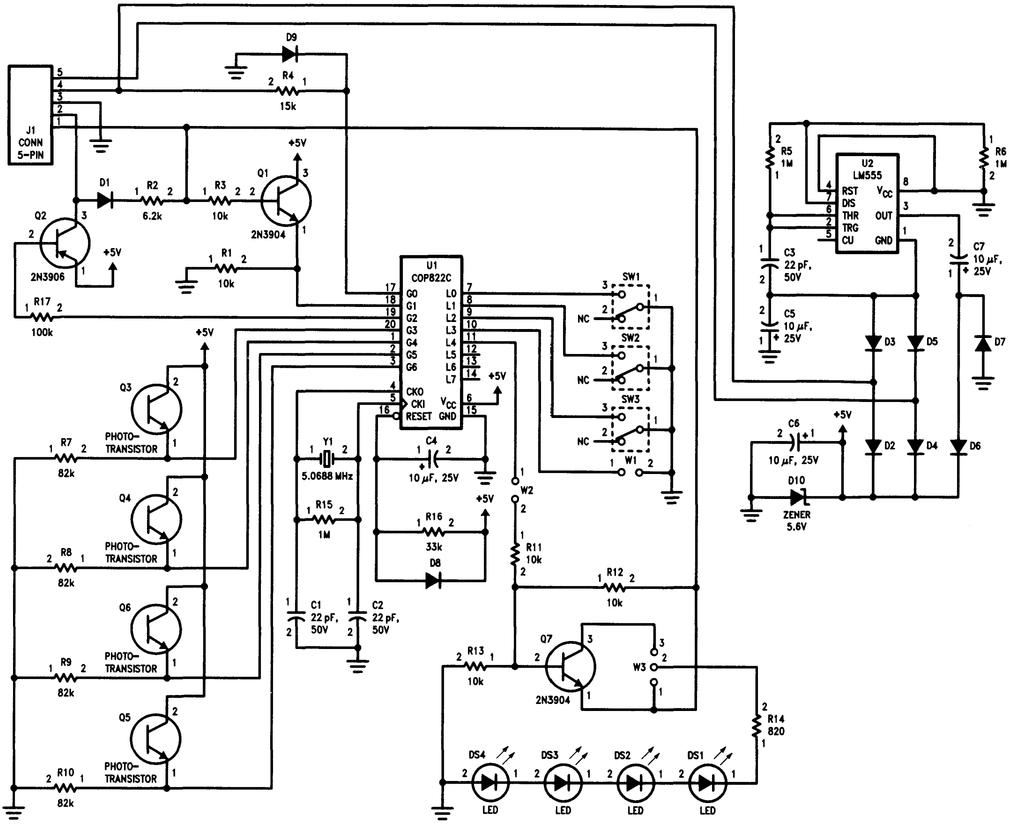
TEMP	=	0F1	Work Space
ASAVE	=	0F4	Save A Register
PSSAVE	=	0F6	Save PSW Register
WORDPT	=	000	Word Pointer
WORD1	=	001	Buffer to Store Report
WORD2	=	002	Buffer
WORD3	=	003	Buffer
CHANGE	=	004	Movement or Button Change
XINC	=	005	X Direction Counter
YINC	=	006	Y Direction Counter
NUMWORD	=	007	Number 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

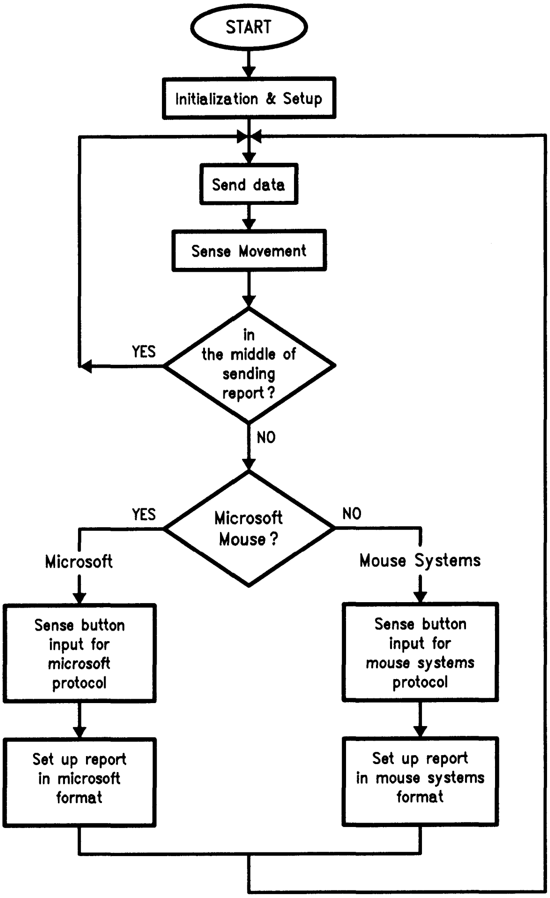
Note 1: All diodes are 1N4148.

Note 2: All resistor values are in ohms, 5%, ⅛W.

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

                ;
                ;       MICROSOFT AND MOUSE SYSTEM COMPATIBLE MOUSE
                ;       02/14/89
                ;       NAME : AMOUSE.MAC
                ;
                        .TITLE AMOUSE
                        .CHIP 820
                ;
                ;
00D0            PORTLD  =       0D0     ; PORT L DATA
00D1            PORTLC  =       0D1     ; PORT L CONFIG
00D2            PORTLP  =       0D2     ; PORT L PIN
                ;
00D4            PORTGD  =       0D4     ; PORT G DATA
00D5            PORTGC  =       0D5     ; PORT G CONFIG
00D6            PORTGP  =       0D6     ; PORT G PIN
                ;
00EA            TMRLO   =       0EA     ; TIMER LOW BYTE
00EB            TMRHI   =       0EB     ; TIMER HIGH BYTE
00EC            TAULO   =       0EC     ; TIMER REGISTER LOW BYTE
00ED            TAUHI   =       0ED     ; TIMER REGISTER HIGH BYTE
                ;
00EE            CNTRL   =       0EE     ; CONTROL REGISTER
00EF            PSW     =       0EF     ; PSW REGISTER
                ;
                ;       CONSTANT DECLARE
                ;
0000            INTR    =       0
0003            TIO     =       3
0004            SO      =       4
0005            SK      =       5
0006            SI      =       6
0007            CKO     =       7
                ;
0007            TSEL    =       7
0006            CSEL    =       6
0005            TEDG    =       5
0004            TRUN    =       4
0003            MSEL    =       3
0002            IEDG    =       2
0001            S1      =       1
0000            S0      =       0
                ;
0007            HCARRY  =       7
0006            CARRY   =       6
0005            TPND    =       5
0004            ENTI    =       4
0003            IPND    =       3
0002            BUSY    =       2
0001            ENI     =       1
0000            GIE     =       0
```

TL/DD/10799-7

```
               ;
               ;       TSTATUS BITS
               ;
0002           TBAUB   =       2        ;BAUD RATE TIMER BIT
               ;
0000           RPT     =       0        ;REPORT BIT OF CHANGE (CHANGE)
0001           SYRPT   =       1        ;SET UP MOUSE SYSTEM LAST 2 WORDS (CHANGE)
0007           USOFT   =       7        ;MICROSOFT (MTYPE)
0002           XMT     =       2        ;G2 AS XMT BIT (PORTGD)
               ;
0003           SW      =       3        ;SLIDE SWITCH (PORTLP,MTYPE)
               ;
               ;       REGISTER ASSIGNMENTS
               ;
00F0                   RSVD    =        0F0
00F1                   TEMP    =        0F1
00F3                   TBAU    =        0F3     ;BAUD RATE TIMER
00F4                   ASAVE   =        0F4     ;SAVE A
00F5                   BSAVE   =        0F5     ;SAVE B
00F6                   PSSAVE  =        0F6     ;SAVE PSW
               ;
               ;       VARIABLES
               ;
0000                   WORDPT  =        000     ;WORD POINTER
0001                   WORD1   =        001     ;BUFFER TO STORE REPORTS
0002                   WORD2   =        002
0003                   WORD3   =        003
               ;
0004                   CHANGE  =        004     ;MOVEMENT CHANGE OR BUTTON PRESSED
0005                   XINC    =        005     ;X DIRECTION COUNTER
0006                   YINC    =        006     ;Y DIRECTION COUNTER
0007                   NUMWORD =        007     ;NUMBER OF BYTES TO SEND
0008                   SENDST  =        008     ;SERIAL PROTOCOL STATE
               ;
0009                   TBAUR   =        009     ;BAUD RATE TIMER RELOAD
000A                   TSTATUS =        00A     ;COUNTER STATUS
000B                   MTYPE   =        00B     ;MOUSE TYPE
               ;
000C                   GTEMP   =        00C     ;TRACK INPUT FROM G
000D                   TRACKS  =        00D     ;PREVIOUS TRACK STATUS
               ;
000E                   BTEMP   =        00E     ;BUTTON INPUT
000F                   BUTSTAT =        00F     ;PREVIOUS BUTTON STATUS
               ;
               ;
               ; MOST POSITIVE = SPACE = HI = ON = 0 = START BIT = RBIT
               ; MOST NEGATIVE = MARK = LO = OFF = 1 = STOP BIT = SBIT
               ;
               ;       MICROSOFT FORMAT
               ;
               ;       1  L  R  Y7  Y6  X7  X6
```

TL/DD/10799–8

```
                    ;         0  X5 .............  X0
                    ;         0  Y5 .............  Y0
                    ;
                    ;         1200 BAUD 7 BIT NO PARITY 2 STOP BITS
                    ;
                    ;         MOUSE SYSTEMS FORMAT (FIVE BYTE PACKED BINARY)
                    ;
                    ;         1  0   0   0   0   L*  M*  R*
                    ;         X7 ......................  X0
                    ;         Y7 ......................  Y0
                    ;         X7 ......................  X0
                    ;         Y7 ......................  Y0
                    ;
                    ;         1200 BAUD 7 BIT NO PARITY 2 STOP BITS
                    ;
                    ;         G6,G5,G4,G3 ARE SENSOR INPUTS
                    ;
                    ;         L0, L1 AND L2 ARE BUTTON INPUTS
                    ;
                    ;         G0 IS INTERRUPT INPUT FOR DETECTING RTS TOGGLE
                    ;
                    ;         USE G2 AS TRANSMIT
                    ;
                    ;         G1 USED FOR RECEIVING COMMANDS FROM HOST (RESERVED)
                    ;
                    START:
0000 DD2F                     LD       SP,#02F
0002 BCEF00                   LD       PSW,#0            ;DISABLE INTR
0005 BCEE80                   LD       CNTRL,#080        ;10000000 - AUTORELOAD
                                                         ;RISING EDGE EXT INT
0008 BCD504                   LD       PORTGC,#004       ;G2 AS OUTPUT, OTHERS AS HI-Z
000B BCD404                   LD       PORTGD,#004       ;G2 DATA 1 "MARK"
                    ;
000E BCD130                   LD       PORTLC,#030       ;HI-Z INPUTS FOR L6-7,OUTPUT L4,5
0011 BCD00F                   LD       PORTLD,#0F        ;WEAK PULL UP FOR L0-3
                    ;
                    ;         INIT RAM
                    ;
0014 5B                       LD       B,#CHANGE
0015 9A00                     LD       [B+],#0           ;(CHANGE)
0017 9A00                     LD       [B+],#0           ;(XINC)
0019 9A00                     LD       [B+],#0           ;(YINC)
001B BC0A00                   LD       TSTATUS,#0
                    ;
001E 9DD6                     LD       A,PORTGP
0020 B0                       RRC      A
0021 953C                     AND      A,#03C            ;NOW IN 6,5,4,3
0023 9C0D                     X        A,TRACKS          ;GET INITIAL VALUE OF SENSORS
                    ;
0025 3067                     JSR      SELECT            ;SELECT MOUSE TYPE
                    ;
```

TL/DD/10799-9

```
                   ;****************************************************
                   ;
                   ;       CRYSTAL FREQ = 4.96 MHZ  2.016 US INST CYCLE
                   ;       FOR 1200 BAUD - TIMER = 413 COUNT
                   ;
                   ;****************************************************
                   ;
                   LTIMER:
0027 DEEA                  LD      B,#TMRLO
0029 9A9D                  LD      [B+],#09D       ;FOR 2.016 US CYCLE
002B 9A01                  LD      [B+],#01
002D 9A9D                  LD      [B+],#09D
002F 9E01                  LD      [B],#01
                   ;
0031 BC0800                LD      SENDST,#0       ;SET TO IDLE STATE
0034 9DEF                  LD      A,PSW
0036 9713                  OR      A,#013          ;ENABLE INTRS SET GIE
0038 9CEF                  X       A,PSW
003A BDEE7C                SBIT    TRUN,CNTRL      ;START TIMER
                   ;
                   MLOOP:
003D BCD03F                LD      PORTLD,#03F     ;TURN ON LED (NOT USED)
0040 3191                  JSR     SDATA
0042 3077                  JSR     SENSOR
0044 9D08                  LD      A,SENDST        ;IF SENDING REPORT
0046 9300                  IFGT    A,#0            ;JUST DO SENSOR
0048 F4                    JP      MLOOP
                   ;
0049 9DD2                  LD      A,PORTLP        ;GET INPUT FROM BUTTONS (L0,L1,L2)
004B B0                    RRC     A               ;PUT IN CARRY FOR CHECKING
004C 51                    LD      B,#BTEMP        ;PREPARATION TO SEE WHAT BUTTON IS PRESSED
                   ;
004D BD0B77                IFBIT   USOFT,MTYPE
0050 0B                    JP      LPUS
                   ;
0051 3210                  JSR     BUTMS           ;MOUSE SYSTEMS
0053 316C                  JSR     SRPTMS
                   ;
0055 BDD273                IFBIT   SW,PORTLP
0058 E4                    JP      MLOOP           ;CONTINUE IF NO CHANGE IN SWITCH
0059 306B                  JSR     USM             ;ELSE NEW SET UP
005B E1                    JP      MLOOP
                   LPUS:
005C 3200                  JSR     BUTUS           ;MICROSOFT
005E 3136                  JSR     SRPTUS
                   ;
0060 BDD273                IFBIT   SW,PORTLP
0063 3071                  JSR     SYM             ;IF CHANGED IN SWITCH, NEW SET UP
0065 203D         +        JP      MLOOP
                   ;
                   ;****************************************************
```

TL/DD/10799-10

```
                    ;       SELECT MOUSE TYPE
                    ;**************************************************
                    ;
                    SELECT:
0067 BDD273                 IFBIT   SW,PORTLP       ;CHECK JUMPER
006A 06                     JP      SYM
                    ;
                    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

```
0086 87                OR     A,[B]      ;OVERLAY WITH PREVIOUS (TRACKS)
0087 97B0              OR     A,#0B0     ;X MOVEMENT TABLE
0089 A5                JID
            ;
008A 0F     NOISEX:    JP     YDIR
            ;
            INCX:
008B 9D05              LD     A,XINC
008D 8A                INC    A
008E 03                JP     COMX       ;CHECK IF LIMIT IS REACHED
            DECX:
008F 9D05              LD     A,XINC
0091 8B                DEC    A
            COMX:                        ;CHECK FOR LIMIT
0092 9250              IFEQ   A,#80
0094 05                JP     YDIR       ;YES DO NOTHING
0095 9C05              X      A,XINC     ;ELSE NEW POSITION
0097 5B                LD     B,#CHANGE
0098 78                SBIT   RPT,[B]    ;(CHANGE)
0099 52                LD     B,#TRACKS
            ;
            YDIR:
009A 52                LD     B,#TRACKS
009B AB                LD     A,[B-]     ;(TRACKS) Y IN 5,4
009C 65                SWAP   A
009D B0                RRC    A
009E B0                RRC    A
009F B0                RRC    A
00A0 95C0              AND    A,#0C0
00A2 87                OR     A,[B]      ;(GTEMP)
00A3 65                SWAP   A
00A4 97C0              OR     A,#0C0     ;Y MOVEMENT TABLE
00A6 A5                JID
            ;
     00B0              .=0B0
            MOVEMX:
00B0 8A                .ADDR  NOISEX     ;0
00B1 8F                .ADDR  DECX       ;1
00B2 8B                .ADDR  INCX       ;2
00B3 8A                .ADDR  NOISEX     ;3
00B4 8B                .ADDR  INCX       ;4
00B5 8A                .ADDR  NOISEX     ;5
00B6 8A                .ADDR  NOISEX     ;6
00B7 8F                .ADDR  DECX       ;7
00B8 8F                .ADDR  DECX       ;8
00B9 8A                .ADDR  NOISEX     ;9
00BA 8A                .ADDR  NOISEX     ;A
00BB 8B                .ADDR  INCX       ;B
00BC 8A                .ADDR  NOISEX     ;C
00BD 8B                .ADDR  INCX       ;D
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

```
                    ;
0101 BDEF75                 IFBIT   TPND,PSW
0104 07                     JP      TINTR
0105 BDEF73                 IFBIT   IPND,PSW
0108 0A                     JP      XINTR
                    ;
                    INTRET:                         ;INTERRUPT RETURN
0109 9DF4                   LD      A,ASAVE
010B 8F                     RETI
                    ;
                    ;**************************************************
                    ;       TIMER INTERRUPT
                    ;       UPDATE ALL THE COUNTERS
                    ;**************************************************
                    ;
                    TINTR:
010C BDEF6D                 RBIT    TPND,PSW
010F BD0A7A                 SBIT    TBAUB,TSTATUS   ;SET BIT IN TSTATUS
0112 F6                     JP      INTRET
                    ;
                    ;********************************************************
                    ;       EXTERNAL INTERRUPT
                    ;       RESPONSE TO RTS TOGGLING
                    ;       BY SENDING AN 'M' 4DH
                    ;********************************************************
                    ;
0113 BDEF6B         XINTR:  RBIT    IPND,PSW
0116 BD0B77                 IFBIT   USOFT,MTYPE     ;ONLY IF MICROSOFT PROTOCOL
0119 01                     JP      XINTR1          ;CONTINUE
011A EE                     JP      INTRET          ;ELSE DO NOTHING
                    XINTR1:
011B BC01FF                 LD      WORD1,#0FF      ;ALL MARK
011E BC024D                 LD      WORD2,#'M'
0121 BC0702                 LD      NUMWORD,#02
                    ;
0124 9D08                   LD      A,SENDST
0126 9200                   IFEQ    A,#0            ;IF IDLE, SEND 'M'
0128 05                     JP      RTSR2
                    ;
0129 BC0001                 LD      WORDPT,#WORD1   ;FAKE CONTINUE LAST CHAR
012C 2109         +         JP      INTRET
                    ;
                    RTSR2:
012E BC0002                 LD      WORDPT,#WORD2   ;'M' ONLY
0131 BC0801                 LD      SENDST,#01
0134 2109         +         JP      INTRET
                    ;
                    ;********************************************************
                    ;       SUBROUTINE SET UP REPORT 'SRPT' FOR MICROSOFT
                    ;                                       -------------
                    ;       CHANGE OF STATUS DETECTED
```

TL/DD/10799-14

```
                     ;       SET UP THE 3 WORDS FOR REPORTING IF IN IDLE STATE
                     ;*********************************************************
                     ;
                     SRPTUS:
0136 5B                      LD      B,#CHANGE
0137 70                      IFBIT   RPT,[B]
0138 01                      JP      SRUS1
0139 8E                      RET                     ;EXIT IF NOT CHANGE
                     ;
                     SRUS1:
013A BDEF68                  RBIT    GIE,PSW         ;DISABLE INTERRUPT
013D 5F                      LD      B,#WORDPT
013E 9A01                    LD      [B+],#WORD1     ;(WORDPT)SET WORD POINTER
0140 9D05                    LD      A,XINC
0142 65                      SWAP    A
0143 B0                      RRC     A
0144 B0                      RRC     A
0145 9503                    AND     A,#03           ;X7,X6
0147 A6                      X       A,[B]           ;(WORD1)
                     ;
0148 9D06                    LD      A,YINC
014A 65                      SWAP    A
014B 950C                    AND     A,#0C           ;Y7,Y6
014D 87                      OR      A,[B]           ;(WORD1)
014E 9740                    OR      A,#040          ;SET BIT 6
0150 BD0F87                  OR      A,BUTSTAT       ;GET BUTTON STATUS
0153 A2                      X       A,[B+]          ;(WORD1)
                     ;
0154 9D05                    LD      A,XINC
0156 953F                    AND     A,#03F          ;X0-X5
0158 A2                      X       A,[B+]          ;(WORD2)
                     ;
0159 9D06                    LD      A,YINC
015B 953F                    AND     A,#03F          ;Y0-Y5
015D A2                      X       A,[B+]          ;(WORD3)
015E 68                      RBIT    RPT,[B]         ;(CHANGE)RESET CHANGE OF STATUS
015F AA                      LD      A,[B+]          ;INC B
0160 9A00                    LD      [B+],#0         ;(XINC)
0162 9A00                    LD      [B+],#0         ;(YINC)
                     ;
0164 9A03                    LD      [B+],#03        ;(NUMWORD)SEND 3 BYTES
0166 9E01                    LD      [B],#01         ;(SENDST)SET TO START BIT STATE
                     ;
0168 BDEF78                  SBIT    GIE,PSW         ;ENABLE INTERRUPT
016B 8E                      RET
                     ;
                     ;*********************************************************
                     ;       SUBROUTINE SET UP REPORT 'SRPT' FOR MOUSE SYSTEMS
                     ;                                       -----------------
                     ;       CHANGE OF STATUS DETECTED
                     ;       SET UP THE FIRST 3 WORDS FOR REPORTING
```

TL/DD/10799-15

```
                        ;       IF IN IDLE STATE
                        ;*********************************************************
                        ;
                        SRPTMS:
    016C 5B                     LD      B,#CHANGE
    016D 70                     IFBIT   RPT,[B]
    016E 01                     JP      SRMS1
    016F 8E                     RET                     ;EXIT IF NO CHANGE
                        ;
                        SRMS1:
    0170 BDEF68                 RBIT    GIE,PSW         ;DISABLE INTERRUPT
    0173 5F                     LD      B,#WORDPT
    0174 9A01                   LD      [B+],#WORD1     ;(WORDPT)SET WORD POINTER
    0176 9D0F                   LD      A,BUTSTAT
    0178 A2                     X       A,[B+]          ;(WORD1)
                        ;
    0179 9D05                   LD      A,XINC
    017B A2                     X       A,[B+]          ;(WORD2)
                        ;
    017C A1                     SC
    017D 64                     CLR     A
    017E BD0681                 SUBC    A,YINC          ;FOR MOUSE SYSTEM NEG Y
    0181 A2                     X       A,[B+]          ;(WORD3)
                        ;
    0182 68                     RBIT    RPT,[B]         ;(CHANGE)RESET CHANGE OF STATUS
    0183 79                     SBIT    SYRPT,[B]       ;(CHANGE)
    0184 AA                     LD      A,[B+]          ;INC B
    0185 9A00                   LD      [B+],#0         ;(XINC)
    0187 9A00                   LD      [B+],#0         ;(YINC)
                        ;
    0189 9A03                   LD      [B+],#03        ;(NUMWORD)SEND 3 BYTES
    018B 9E01                   LD      [B],#01         ;(SENDST)SET TO START BIT STATE
                        ;
    018D BDEF78                 SBIT    GIE,PSW         ;ENABLE INTERRUPT
    0190 8E                     RET
                        ;
                        ;
                        ;*********************************************************
                        ;       SUBROUTINE TO SEND DATA 'SDATA'
                        ;       CHECK THE BIT TO SEND AND DRIVE THE OUTPUT TO THE
                        ;       DESIRED VALUE
                        ;
                        ;       SENDST          STATE
                        ;         0             IDLE
                        ;         1             START BIT
                        ;         2-8           DATA
                        ;         2-9           DATA    (FOR MOUSE SYSTEMS)
                        ;         9-10          STOP BIT
                        ;         10-11         STOP BIT   (FOR MOUSE SYSTEMS)
                        ;         11            NEXT WORD
                        ;         12            NEXT WORD  (FOR MOUSE SYSTEMS)
```

TL/DD/10799-16

```
                        ;
                        ;**********************************************************
                        ;
0191 55                 SDATA:  LD      B,#TSTATUS
0192 72                         IFBIT   TBAUB,[B]       ;(TSTATUS)CHECK IF BAUD RATE TIMER ENDS
0193 01                         JP      SDATA1
0194 8E                         RET
                        ;
                        SDATA1:
0195 6A                         RBIT    TBAUB,[B]       ;(TSTATUS)
0196 AA                         LD      A,[B+]          ;INC B TO (MTYPE)
0197 9D08                       LD      A,SENDST
0199 97F0                       OR      A,#0F0
019B A5                         JID
                        ;
019C 8E                 IDLE:   RET                     ;EXIT IF IDLE
                        ;
019D 77                 STAT9:  IFBIT   USOFT,[B]       ;(MTYPE)
019E 16                         JP      STOPB
                        DATAB:
019F 9D00                       LD      A,WORDPT
01A1 9CFE                       X       A,B             ;B POINTS TO THE WORD
                        ;
01A3 A0                         RC
01A4 AE                         LD      A,[B]
01A5 B0                         RRC     A               ;XMIT LEAST SIG BIT
01A6 A6                         X       A,[B]
01A7 DED4                       LD      B,#PORTGD
01A9 88                         IFC
01AA 7A                         SBIT    XMT,[B]
01AB 89                         IFNC
01AC 6A                         RBIT    XMT,[B]
                        ;
01AD 9D08               NEXT:   LD      A,SENDST
01AF 8A                         INC     A
01B0 9C08                       X       A,SENDST
01B2 8E                         RET                     ;EXIT
                        ;
01B3 77                 STAT11: IFBIT   USOFT,[B]       ;(MTYPE)
01B4 04                         JP      NXWORD
                        ;
01B5 BDD47A             STOPB:  SBIT    XMT,PORTGD
01B8 F4                         JP      NEXT
                        ;
01B9 9D00               NXWORD: LD      A,WORDPT
01BB 8A                         INC     A
01BC BD0783                     IFGT    A,NUMWORD       ;NUMBER OF WORDS TO SEND
01BF 09                         JP      ENDRPT          ;END OF REPORT
01C0 9C00                       X       A,WORDPT
01C2 BC0801                     LD      SENDST,#01      ;SEND START BIT
                        ;
```

TL/DD/10799-17

```
01C5 BDD46A         STARTB: RBIT   XMT.PORTGD    ;SEND START BIT
01C8 E4                     JP     NEXT
                    ;
01C9 BD0471         ENDRPT: IFBIT  SYRPT,CHANGE
01CC 04                     JP     SY2RPT
                    ;
01CD BC0800                 LD     SENDST,#0
01D0 8E                     RET
                    ;
                    ;**************************************************
                    ;       SET UP LAST 2 WORDS IN MOUSE SYSTEM FORMAT
                    ;**************************************************
                    ;
                    SY2RPT:
01D1 BDEF68                 RBIT   GIE,PSW       ;DISABLE INTERRUPT
                    ;
01D4 5F                     LD     B,#WORDPT
01D5 9A01                   LD     [B+],#WORD1   ;(WORDPT)SET WORD POINTER
01D7 9D05                   LD     A,XINC
01D9 A2                     X      A,[B+]        ;(WORD1)
                    ;
01DA A1                     SC
01DB 64                     CLR    A
01DC BD0681                 SUBC   A,YINC        ;FOR MOUSE SYSTEM NEG Y
01DF A2                     X      A,[B+]        ;(WORD2)
                    ;
01E0 AA                     LD     A,[B+]        ;INC B
01E1 69                     RBIT   SYRPT,[B]     ;(CHANGE)RESET CHANGE OF STATUS
01E2 AA                     LD     A,[B+]        ;INC B
01E3 9A00                   LD     [B+],#0       ;XINC
01E5 9A00                   LD     [B+],#0       ;YINC
                    ;
01E7 9A02                   LD     [B+],#02      ;(NUMWORD)SEND 2 BYTES
01E9 9E01                   LD     [B],#01       ;(SENDST)SET TO START BIT STATE
                    ;
01EB BDEF78                 SBIT   GIE,PSW       ;ENABLE INTERRUPT
01EE 21C5         +         JP     STARTB
                    ;
     01F0                   .=01F0
                    ;
01F0 9C                     .ADDR  IDLE          ;0
01F1 C5                     .ADDR  STARTB        ;1
01F2 9F                     .ADDR  DATAB         ;2
01F3 9F                     .ADDR  DATAB         ;3
01F4 9F                     .ADDR  DATAB         ;4
01F5 9F                     .ADDR  DATAB         ;5
01F6 9F                     .ADDR  DATAB         ;6
01F7 9F                     .ADDR  DATAB         ;7
01F8 9F                     .ADDR  DATAB         ;8
01F9 9D                     .ADDR  STAT9         ;9
01FA B5                     .ADDR  STOPB         ;10
```

TL/DD/10799-18

```
01FB B3                 .ADDR   STAT11          ;11
01FC B9                 .ADDR   NXWORD          ;12
01FD 9C                 .ADDR   IDLE            ;13
01FE 9C                 .ADDR   IDLE            ;14
01FF 9C                 .ADDR   IDLE            ;15
                ;
                ;
                ;************************************************
                ;       SAMPLE BUTTON INPUT     FOR MICROSOFT
                ;                               -------------
                ;       INDICATE BUTTON STATUS
                ;************************************************
                ;
                BUTUS:
0200 9E00               LD      [B],#0          ;(BTEMP), (A=PORTLP, CARRY ROTATED)
0202 89                 IFNC                    ;MICROSOFT: 1=KEY DEPRESSED
0203 7D                 SBIT    5,[B]           ;(BTEMP)
                ;
0204 B0                 RRC     A
0205 B0                 RRC     A
0206 89                 IFNC
0207 7C                 SBIT    4,[B]           ;(BTEMP)
                ;
0208 AA                 LD      A,[B+]          ;(BTEMP)
0209 82                 IFEQ    A,[B]           ;(BUTSTAT)
020A 8E                 RET                     ;NO CHANGE
                ;
020B A6                 X       A,[B]           ;(BUTSTAT)
020C BD0478             SBIT    RPT,CHANGE      ;INDICATE TO SEND DATA
020F 8E                 RET
                ;
                ;
                ;************************************************
                ;       SAMPLE BUTTON INPUT     FOR MOUSE SYSTEMS
                ;                               -----------------
                ;       INDICATE BUTTON STATUS
                ;************************************************
                ;
                BUTMS:
0210 9E87               LD      [B],#087        ;(BTEMP)
                ;
0212 89                 IFNC                    ;MOUSE SYSTEM: 0=KEY DEPRESSED
0213 6A                 RBIT    2,[B]           ;(BTEMP)
                ;
0214 B0                 RRC     A
0215 89                 IFNC
0216 69                 RBIT    1,[B]           ;(BTEMP)
                ;
0217 B0                 RRC     A
0218 89                 IFNC
0219 68                 RBIT    0,[B]           ;(BTEMP)
```

TL/DD/10799-19

```
                   ;
    021A AA                LD      A,[B+]          ;(BTEMP)
    021B 82                IFEQ    A,[B]           ;(BUTSTAT)
    021C 8E                RET                     ;NO CHANGE
                   ;
    021D A6                X       A,[B]           ;(BUTSTAT)
    021E BD0478            SBIT    RPT,CHANGE      ;INDICATE TO SEND DATA
    0221 8E                RET
                   ;
                   ;************************************************************
                   ;
         03D0              .=03D0
    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
```

TL/DD/10799-20

```
    03F7 56
    03F8 45
    03F9 52
    03FA 20
    03FB 31
    03FC 2E
    03FD 30
                   ;
                           .END
```

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  000D         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.

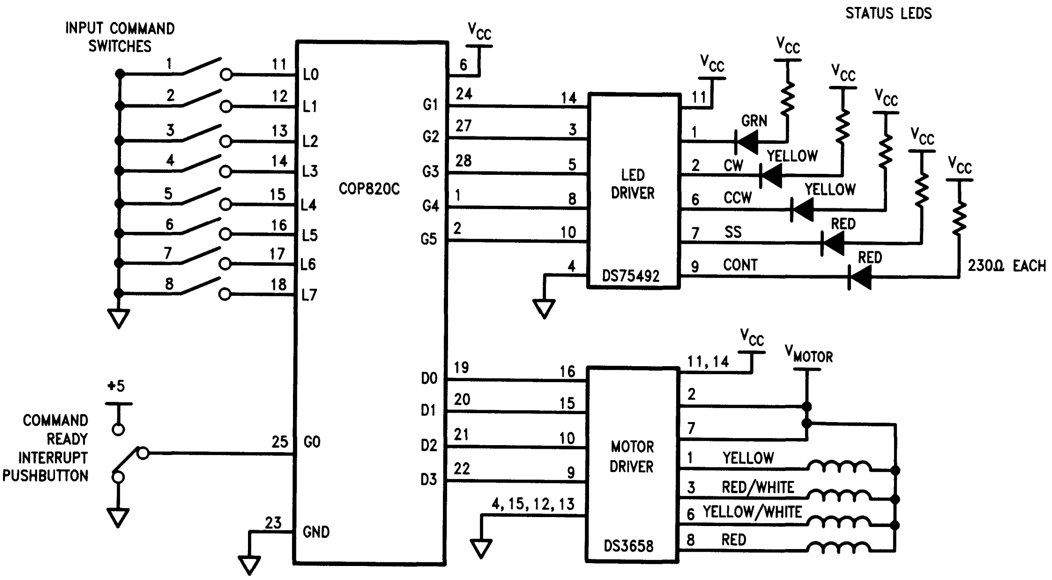
The determination of which windings to activate and deactivate to step the motor is performed by a single subroutine in this example. A block of memory is allocated to store a step pointer and the four possible stepper drive values are shown in Table I (9,C,6,3). Consecutive memory locations are used 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

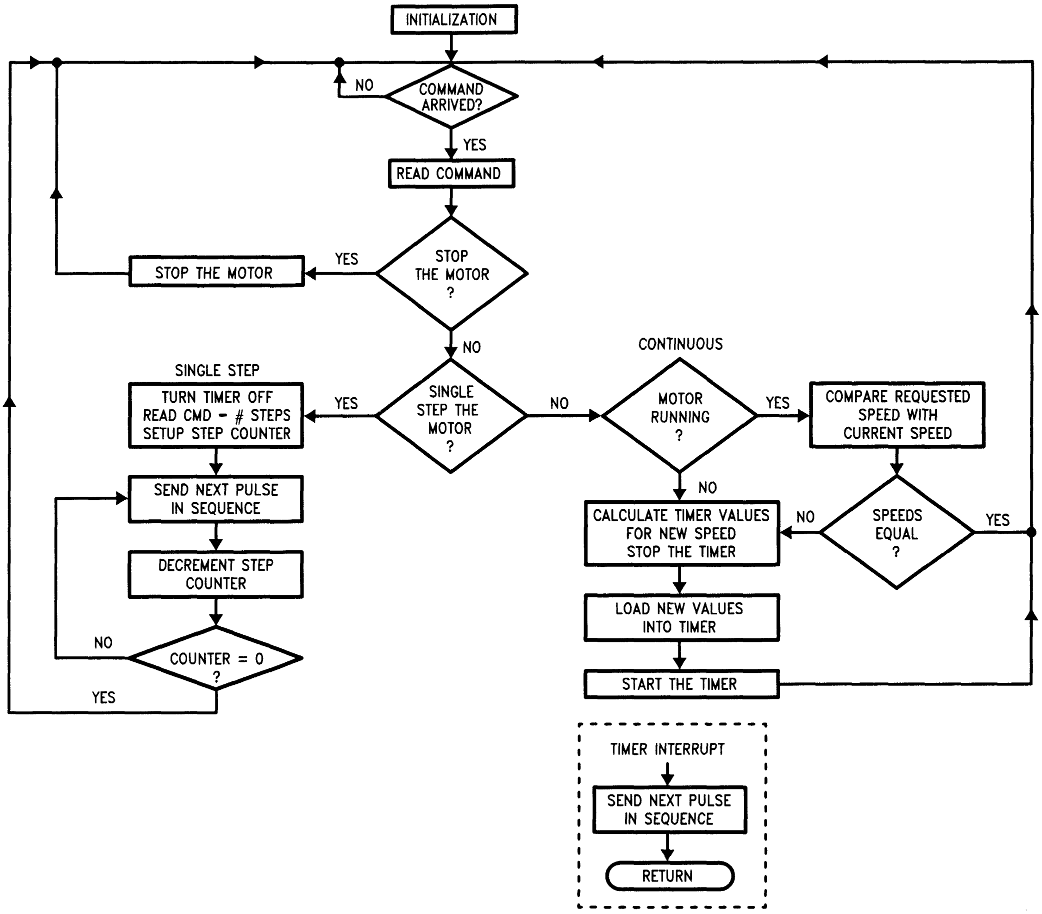


TL/DD/11044-1

FIGURE 1. Schematic Diagram

2

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

```
0005 BCEF03            LD      PSW,#003          ;GLOBAL INT ENABLE/EXTINT ENABLE
0008 BCD401            LD      PORTGD,#01
000B BCD53E            LD      PORTGC,#03E       ;CONFIG PORTG FOR OUTPUTS
000E BCDC09            LD      PORTD,#09         ;START MOTOR DRIVE VALUE
0011 BCD100            LD      PORTLC,#00        ;CONFIG PORTL FOR INPUTS
0014 BCD0FF            LD      PORTLD,#0FF       ;CONFIG PORTL FOR WEAK PULL-UPS
0017 5F                LD      B,#MS0            ;SETUP MOTOR DRIVE VALUES
0018 9A09              LD      [B+],#09
001A 9A0C              LD      [B+],#0C
001C 9A06              LD      [B+],#06
001E 9E03              LD      [B],#03
0020 D200              LD      STPPTR,#00        ;INIT STEP POINTER
0022 BC0700            LD      FLGREG,#00        ;INIT FLAG REGISTER


               ;READ, DECODE, AND EXECUTE COMMAND
                                                 ;******************************
0025 BDD479    TOP:    SBIT    READY,PORTGD      ;TURN ON READY FOR NEXT CMD LED
0028 3081              JSR     WAIT              ;WAIT FOR CMD AND READ CMD
002A BDD469            RBIT    READY,PORTGD      ;TURN OFF READY FOR NEXT CMD LED
002D 9C04              X       A,CMD             ;STORE IN CMD REGISTER
002F BD0470            IFBIT   GO,CMD            ;IF STOP BIT SET
0032 08                JP      STOP              ;THEN STOP MOTOR
0033 BD0472            IFBIT   MODE,CMD          ;ELSE CHEK MODE
0036 3041              JSR     SSTEP             ;IF MODE SET THEN GO SINGLE STEP
0038 305F              JSR     CONT              ;ELSE GO CONTINUOUS
003A EA                JP      TOP               ;GO WAIT FOR NEXT COMMAND
               STOP:                             ;STOP THE MOTOR
003B 308E              JSR     TMRSET            ;STOP THE TIMER
003D BCD401            LD      PORTGD,#01        ;TURN OFF ALL LEDS
0040 E4                JP      TOP               ;GO WAIT FOR NEXT CMD


               ;SINGLE STEP THE MOTOR  (SS)

               SSTEP:                            ;******************************
0041 308E              JSR     TMRSET            ;STOP TIMER
0043 BCD410            LD      PORTGD,#010       ;TURN ON SS LED (RST ALL OTHER LEDS)
0046 3081              JSR     WAIT              ;WAIT FOR CMD BYTE 2 (# STEPS)
0048 8A                INC     A                 ;ADD 1 TO CORRECT FOR LOOP
0049 9CF0              X       A,CREG0           ;STORE #STEPS IN LOBYTE COUNT REG
004B 9D04              LD      A,CMD             ;LOAD HIBYTE # STEPS
004D 65                SWAP    A                 ;MOVE TO LOWER NIBBLE
004E 950F              AND     A,#0F             ;GET RID OF UPPER BITS
0050 8A                INC     A                 ;ADD 1 TO CORRECT FOR LOOP
0051 9CF1              X       A,CREG1           ;MOVE TO HIBYTE OF COUNT REG
0053 C0        TP2:    DRSZ    CREG0             ;DECR LOBYTE AND IF NOT ZERO
0054 05                JP      DO                ;THEN GO DO A STEP
0055 C1        MID:    DRSZ    CREG1             ;ELSE DECR HIBYTE AND IF NOT ZERO
0056 01                JP      DO2               ;THEN GO DO A STEP AND RST LO COUNT
0057 8D                RETSK                     ;ELSE END OF LOOP    RETURN
```

TL/DD/11044-5

```
NATIONAL SEMICONDUCTOR CORPORATION
COP800 CROSS ASSEMBLER, REV: D1, 12 OCT 88

0058 D0FF          DO2:    LD      CREG0,#0FF              ;RESET LOBYTE OF COUNTER
005A 3098          DO:     JSR     NXTVAL                  ;STEP THE MOTOR
005C 3158                  JSR     DELAY                   ;SLOW THE STEPPING
005E F4                    JP      TP2                     ;GO TO TOP OF LOOP


                   ;RUN THE MOTOR CONTINUOUSLY  (NS = NON-STOP = CONTINUOUSLY)

                   CONT:                                   ;*****************************
005F BDEE74                IFBIT   TRUN,CNTRL              ;IF MOTOR ALREADY RUNNING NS
0062 01                    JP      CHKSPD                  ;THEN CHECK THE CURRENT SPEED
0063 03                    JP      SETGO                   ;ELSE GO START THE MOTOR
0064 3148          CHKSPD: JSR     SPEED                   ;COMPARE INPUT WITH ACTUAL SPD
0066 8E                    RET                             ;IF EQUAL RET ELSE RESTART MOTOR
0067 308E          SETGO:  JSR     TMRSET                  ;STOP THE TIMER
0069 BCD420                LD      PORTGD,#020             ;TURN ON CONTINUOUS LED
006C 3126                  JSR     TIMVAL                  ;CALCULATE TIMER (SPEED) VALUE
006E AE                    LD      A,[B]                   ;LOAD A WITH TVAL1
006F 9CEB                  X       A,TMRHI                 ;MOVE SPEED VAL INTO TIMER
0071 AB                    LD      A,[B-]                  ;LOAD A WITH TVAL1 POINT TO TVAL0
0072 9CED                  X       A,TAUHI                 ;MOVE SPEED VAL INTO AUTORELOAD REG
0074 AE                    LD      A,[B]                   ;LOAD A WITH TVAL0
0075 9CEA                  X       A,TMRLO                 ;MOVE SPEED VAL INTO TIMER
0077 AE                    LD      A,[B]                   ;LOAD A WITH TVAL0
0078 9CEC                  X       A,TAULO
007A BDEF7C                SBIT    ENTI,PSW                ;ENABLE TIMER INTERRUPT
007D BDEE7C                SBIT    TRUN,CNTRL              ;START THE TIMER
0080 8E                    RET                             ;RET TO MAIN AND WAIT FOR TMRINT

                   ;SUPPORT ROUTINES ****************************************************

                   WAIT:                                   ;*****************************
                           ;WAIT FOR AN EXTERNAL INTERRUPT TO SIGNAL AN INCOMMING COMMAND
                           ;READ THE INCOMMING COMMAND FROM PORT L
0081 BD0770                IFBIT   INT,FLGREG              ;IF EXTERNAL INTERRUPT OCCURED
0084 01                    JP      OUT                     ;THEN JUMP OUT OF LOOP
0085 FB                    JP      WAIT                    ;ELSE CONTINUE TO WAIT
0086 BD0768        OUT:    RBIT    INT,FLGREG              ;RESET EXTERNAL INTERRUPT FLAG
0089 9DD2                  LD      A,PORTLP                ;READ INCOMMING COMMAND
008B 96FF                  XOR     A,#0FF                  ;COMPLEMENT INCOMMING COMMAND
008D 8E                    RET                             ;RETURN COMMAND IN ACC


                   TMRSET:                                 ;*****************************
                           ;RESET THE TIMER
008E BDEE6C                RBIT    TRUN,CNTRL              ;STOP THE TIMER
0091 BDEF6D                RBIT    TPND,PSW                ;RESET THE TIMER PENDING BIT
0094 BDEF6C                RBIT    ENTI,PSW                ;DISABLE TIMER INTERRUPT
0097 8E                    RET


```

TL/DD/11044-6

```
                NXTVAL:                                 ;******************************
                          ;SEND THE NEXT DRIVE VALUE TO STEP THE MOTOR ONE STEP IN THE
                          ;APPROPRIATE DIRECTION (CW OR CCW)
0098 9DF2                 LD      A,STPPTR              ;LOAD STEP VALUE POINTER
009A DED4                 LD      B,#PORTGD             ;POINT TO PORT G
009C BD0471               IFBIT   DIR,CMD               ;IF CLOCKWISE
009F 11                   JP      IPTR                  ;THEN GO INCREMENT POINTER
00A0 6A         DPTR:     RBIT    CW,[B]                ;ELSE RST CW LED
00A1 7B                   SBIT    CCW,[B]               ;TURN ON CCW LED
00A2 8B                   DEC     A                     ;AND DECREMENT POINTER
00A3 92FF                 IFEQ    A,#0FF                ;IF OFF BOTTOM OF STEPS
00A5 9803                 LD      A,#03                 ;THEN LOOP TO TOP OF STEPS
00A7 9CF2       WRVAL:    X       A,STPPTR              ;A -> STPPTR (SAVE NEW STPPTR)
00A9 9DF2                 LD      A,STPPTR              ;[STPPTR] -> PORTD (LOOKUP VAL)
00AB 9CFE                 X       A,B
00AD AE                   LD      A,[B]
00AE 9CDC                 X       A,PORTD               ;WRITE STEP VALUE TO MOTOR
00B0 8E                   RET
00B1 6B         IPTR:     RBIT    CCW,[B]               ;TURN OFF CCW LED
00B2 7A                   SBIT    CW,[B]                ;TURN ON CW LED
00B3 8A                   INC     A                     ;INCREMENT THE STEP POINTER
00B4 9204                 IFEQ    A,#04                 ;IF OFF TOP OF STEPS
00B6 64                   CLR     A                     ;THEN LOOP TO BOTTOM OF STEPS
00B7 EF                   JP      WRVAL                 ;GO WRITE VALUE TO MOTOR


                ;INTERRUPT HANDLERS
     00FF                 . = OFF                       ;******************************
                          ;BRANCH TO THE APPROPRIATE INTERRUPT HANDLER
00FF BDEF75               IFBIT   TPND,PSW              ;TIMER UNDERFLOW
0102 08                   JP      TMRINT
0103 BDEF73               IFBIT   IPND,PSW              ;EXTERNAL INTERRUPT
0106 16                   JP      EXTINT
0107 BDEF78               SBIT    GIE,PSW               ;SOFTWARE TRAP
010A 8D                   RETSK

                TMRINT:                                 ;******************************
                          ;RESET THE TIMER INTERRUPT PENDING BIT AND STEP THE MOTOR
010B 9CF9                 X       A,0F9                 ;CONTEXT SAVE ROUTINE
010D 9DFE                 LD      A,B
010F 9CFA                 X       A,0FA
0111 BDEF6D               RBIT    TPND,PSW              ;RESET PENDING BIT
0114 3098                 JSR     NXTVAL                ;STEP THE MOTOR
0116 9DFA                 LD      A,0FA                 ;CONTEXT RESTORE ROUTINE
0118 9CFE                 X       A,B
011A 9DF9                 LD      A,0F9
011C 8F                   RETI

                EXTINT:                                 ;******************************
011D BD0778               SBIT    INT,FLGREG            ;SET INTERRUPT OCCURED FLAG
0120 3158                 JSR     DELAY                 ;WAIT
```

TL/DD/11044-7

NATIONAL SEMICONDUCTOR CORPORATION
COP800 CROSS ASSEMBLER, REV: D1, 12 OCT 88

```
0122 BDEF6B                 RBIT    IPND,PSW                ;RESET PENDING BIT
0125 8F                     RETI

                    ;SUPPORT ROUTINES CONTINUED

                    TIMVAL:                                 ;*****************************
                            ;During continuous operation, the motor is stepped once every
                            ;timer underflow. Therefore, a timer value is calculated that will
                            ;produce timer underflows every X microseconds causing the motor
                            ;to step Xsteps/second.
                            ;For example:  To step 100 times per second.
                            ;        microseconds/step = 1000000uS/sec x 1sec/100steps = 10000
                            ;        10000uS/step = 02718Hex uS/step
                            ;        1uS = one count down of the timer
                            ; Therefore, load the timer with 02718H for 100 steps/sec.

0126 DE14                   LD      B,#TVAL0                ;POINT TO STORAGE FOR TIMVAL
0128 BD0474                 IFBIT   4,CMD                   ;IF LOWEST SPEED BIT SET
012B 17                     JP      SLOWER                  ;THEN USE SLOWEST SPEED
012C BD0475                 IFBIT   5,CMD                   ;IF SECOND LOWEST SPD BIT SET
012F 0E                     JP      SLOW                    ;THEN USE SLOW SPEED
0130 BD0476                 IFBIT   6,CMD                   ;IF SECOND HIGHEST SPD BIT SET
0133 05                     JP      FAST                    ;THEN USE FAST SPEED
0134 9A02           FASTER: LD      [B+],#02                ;ELSE USE FASTEST SPEED
0136 9E08                   LD      [B],#08                 ;400steps/sec = 2rev/sec
0138 8E                     RET
0139 9A88           FAST:   LD      [B+],#088               ;200steps/sec = 1rev/sec
013B 9E13                   LD      [B],#013
013D 8E                     RET
013E 9A18           SLOW:   LD      [B+],#018               ;100steps/sec = .5rev/sec
0140 9E27                   LD      [B],#027
0142 8E                     RET
0143 9A54           SLOWER: LD      [B+],#054               ;25steps/sec = .125rev/sec
0145 9E9C                   LD      [B],#09C
0147 8E                     RET


                    SPEED:                                  ;*****************************
                            ;COMPARE CURRENT MOTOR SPEED WITH DESIRED MOTOR SPEED
0148 3126                   JSR     TIMVAL                  ;CALCULATED DESIRED SPEED VAL
014A 9D14                   LD      A,TVAL0
014C BDEC82                 IFEQ    A,TAULO                 ;IF DESIRED LBYTE EQUALS CURRENT LBYTE
014F 01                     JP      TSTHI                   ;THEN GO TEST HI-BYTE
0150 8D                     RETSK                           ;ELSE NOT EQUAL RETURN AND SKIP
0151 9D15           TSTHI:  LD      A,TVAL1
0153 BDED82                 IFEQ    A,TAUHI                 ;IF HI-BYTE EQUALS CURRENT HI-BYTE
0156 8E                     RET                             ;THEN DESIRED = CURRENT RETURN
0157 8D                     RETSK                           ;ELSE DESIRED != CURRENT RET & SKIP

                    DELAY:                                  ;*****************************
                            ;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     0056  *     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.

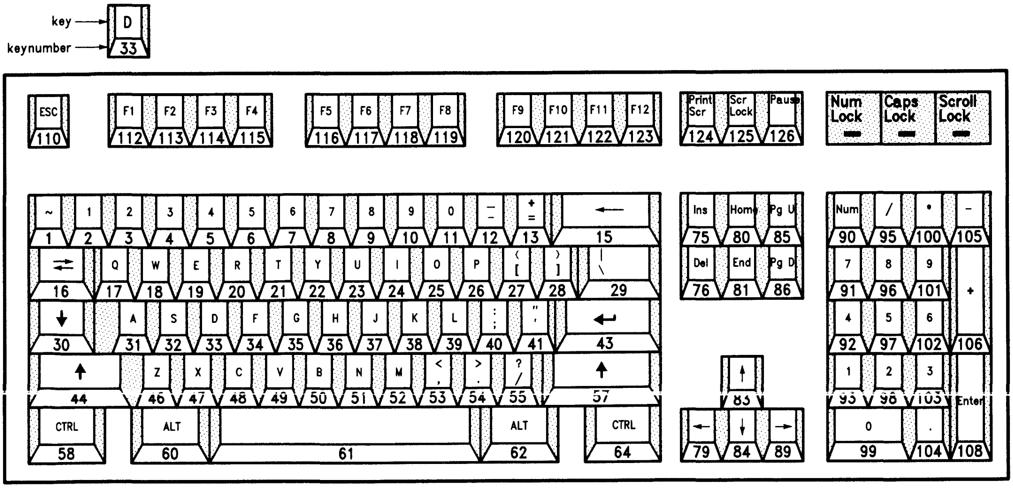
MF2 KEYBOARD KEY-LAYOUT

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

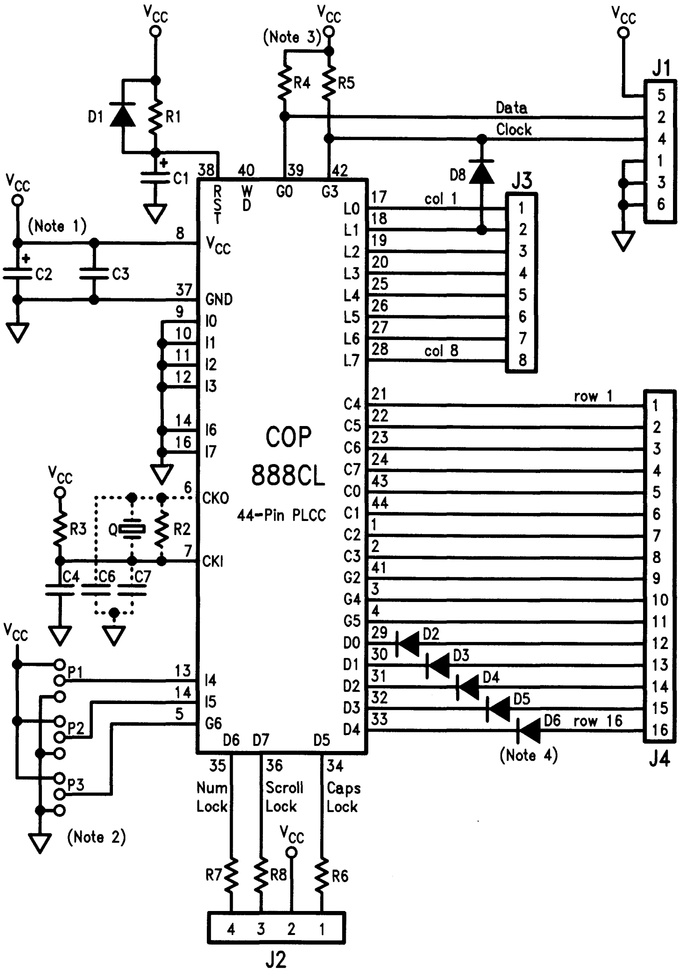
Laptop/Notebook Keyboard With COP888CL

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 ≤ 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.

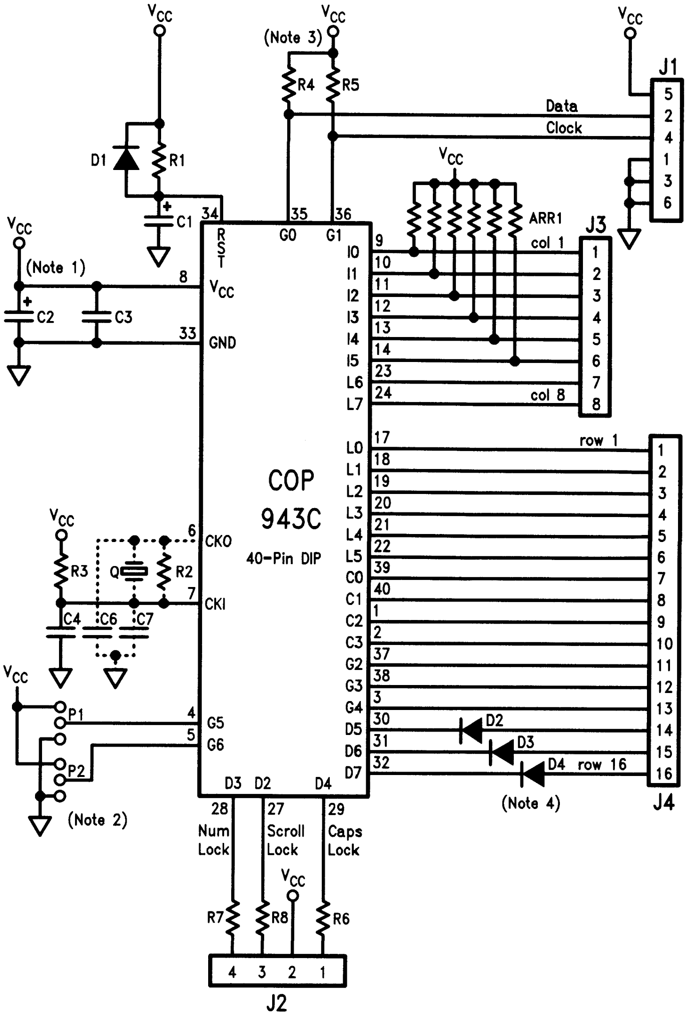
The XT or AT/PS-2 operation mode can be selected via a hardware switch. Additional inputs for customer specific settings are available.

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.

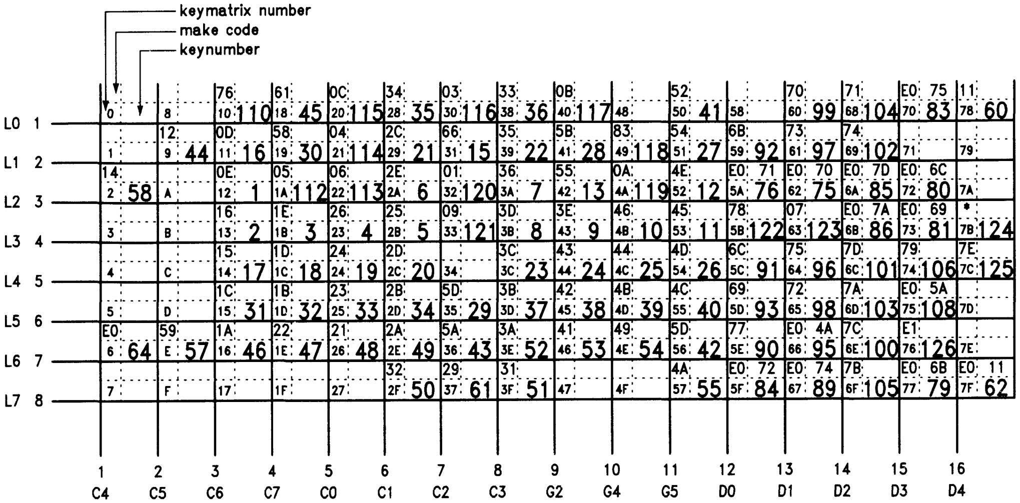
Note 4: Diodes D2–D4 should be removed if keyboard has hardware keyrollover (diodes in matrix).

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".



	1 C4	2 C5	3 C6	4 C7	5 C0	6 C1	7 C2	8 C3	9 G2	10 G4	11 G5	12 D0	13 D1	14 D2	15 D3	16 D4
L0 1	0	8	76 10 110	61 18 45	0C 20 115	34 28 35	03 30 116	33 38 36	0B 40 117	48	52 50 41	58	70 60 99	71 68 104	E0 75 70 83	11 78 60
L1 2	1	12 9 44	0D 11 16	58 19 30	04 21 114	2C 29 21	66 31 15	35 39 22	5B 41 28	83 49 118	54 51 27	6B 59 92	73 61 97	74 69 102	71	79
L2 3	14 2 58	A	0E 12 1	05 1A 112	06 22 113	2E 2A 6	01 32 120	36 3A 7	55 42 13	0A 4A 119	4E 52 12	E0 71 5A 76	E0 70 62 75	E0 7D 6A 85	E0 6C 72 80	7A
L3 4	3	B	16 13 2	1E 1B 3	26 23 4	25 2B 5	09 33 121	3D 3B 8	3E 43 9	46 4B 10	45 53 11	78 5B 122	07 63 123	E0 7A 6B 86	E0 69 73 81	* 7B 124
L4 5	4	C	15 14 17	1D 1C 18	24 24 19	2D 2C 20	34	3C 3C 23	43 44 24	44 4C 25	4D 54 26	6C 5C 91	75 64 96	7D 6C 101	79 74 106	7E 7C 125
L5 6	5	D	1C 15 31	1B 1D 32	23 25 33	2B 2D 34	5D 35 29	3B 3D 37	42 45 38	4B 4D 39	4C 55 40	69 5D 93	72 65 98	7A 6D 103	E0 5A 75 108	7D
L6 7	E0 6 64	59 E 57	1A 16 46	22 1E 47	21 26 48	2A 2E 49	5A 36 43	3A 3E 52	41 46 53	49 4E 54	5D 56 42	77 5E 90	E0 4A 66 95	7C 6E 100	E1 76 126	7E
L7 8	7	F	17	1F	27	32 2F 50	29 37 61	31 3F 51	47	4F	4A 57 55	E0 72 5F 84	E0 74 67 89	7B 6F 105	E0 6B 77 79	E0 11 7F 62

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:

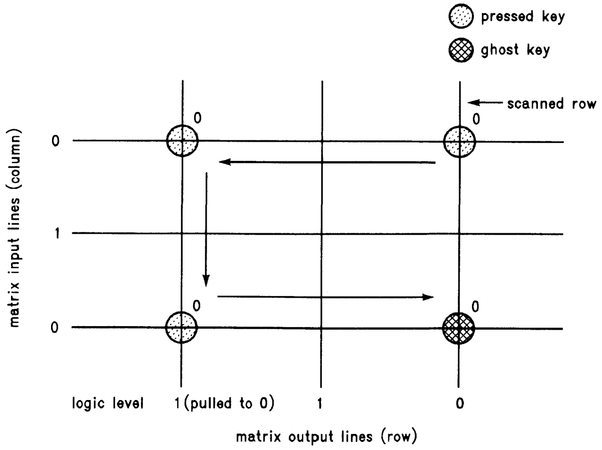
- key detection
- software key rollover
- key decoding and encoding
- keycode transmission
- keyboard command set

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,TYPC01            ;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:

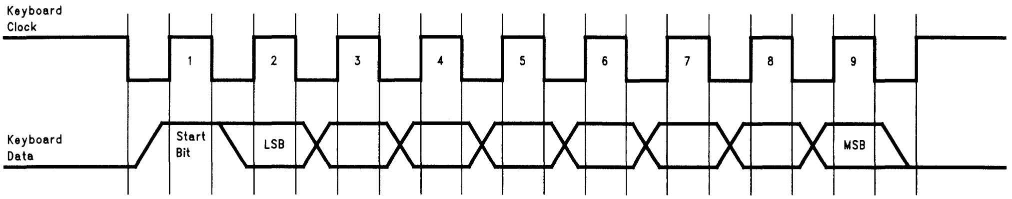
9 bits in length
1 start bit (high)
8 data bits (LSB first)

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)

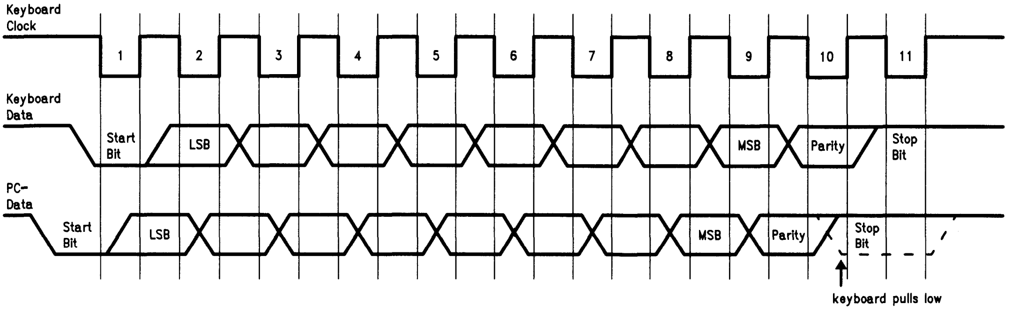
If no data is transmitted, both data and clock lines are in the high state. The clock signal is always provided by the keyboard. *Figure 6* shows the XT and the AT protocol timings.

XT-Protocol



TL/DD/11091-15

AT-Protocol



TL/DD/11091-16

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

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"

AN-739

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

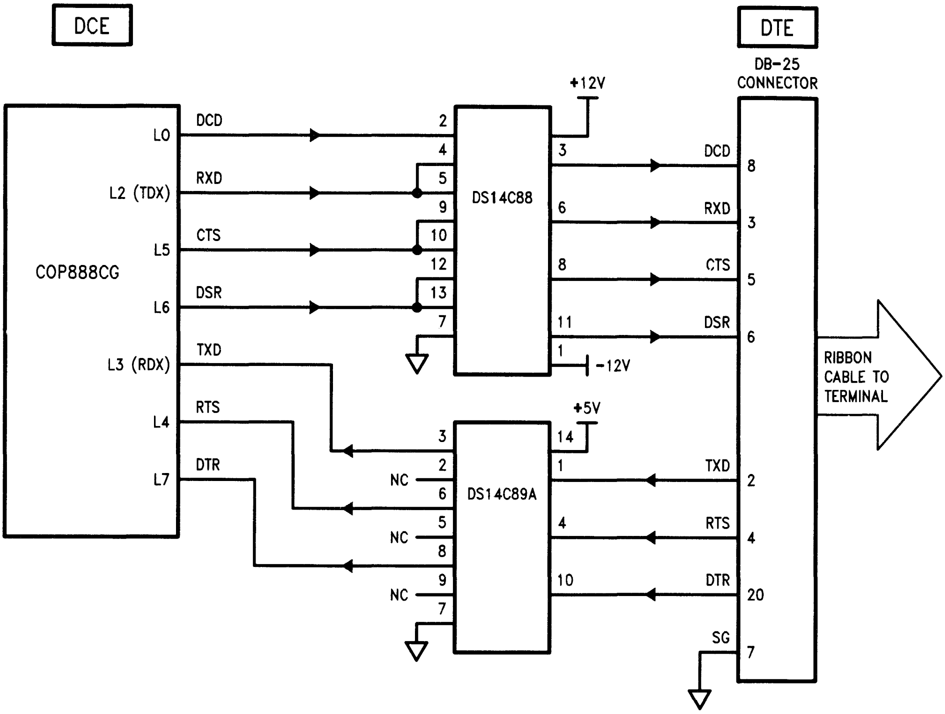
UART receive buffer and checks for errors. If no errors are detected, the incoming data is placed in the data buffer for echoing. If errors are detected, an error message is queued for transmission.

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.

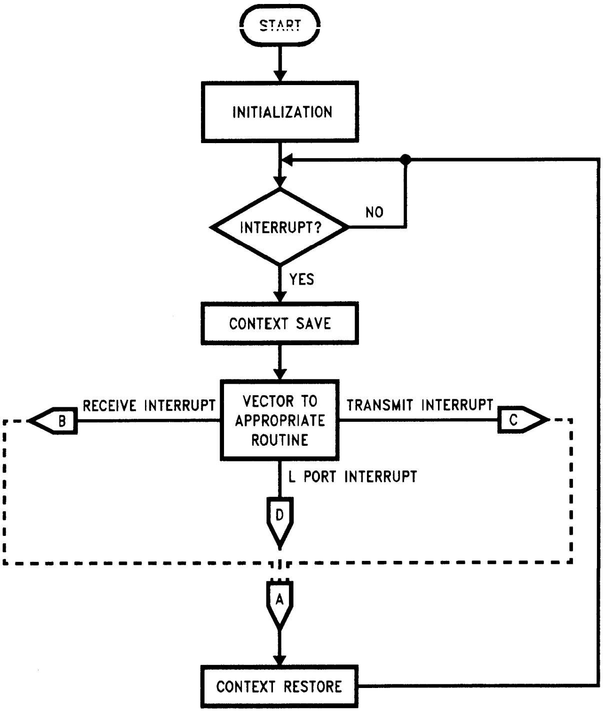
CONCLUSION

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.



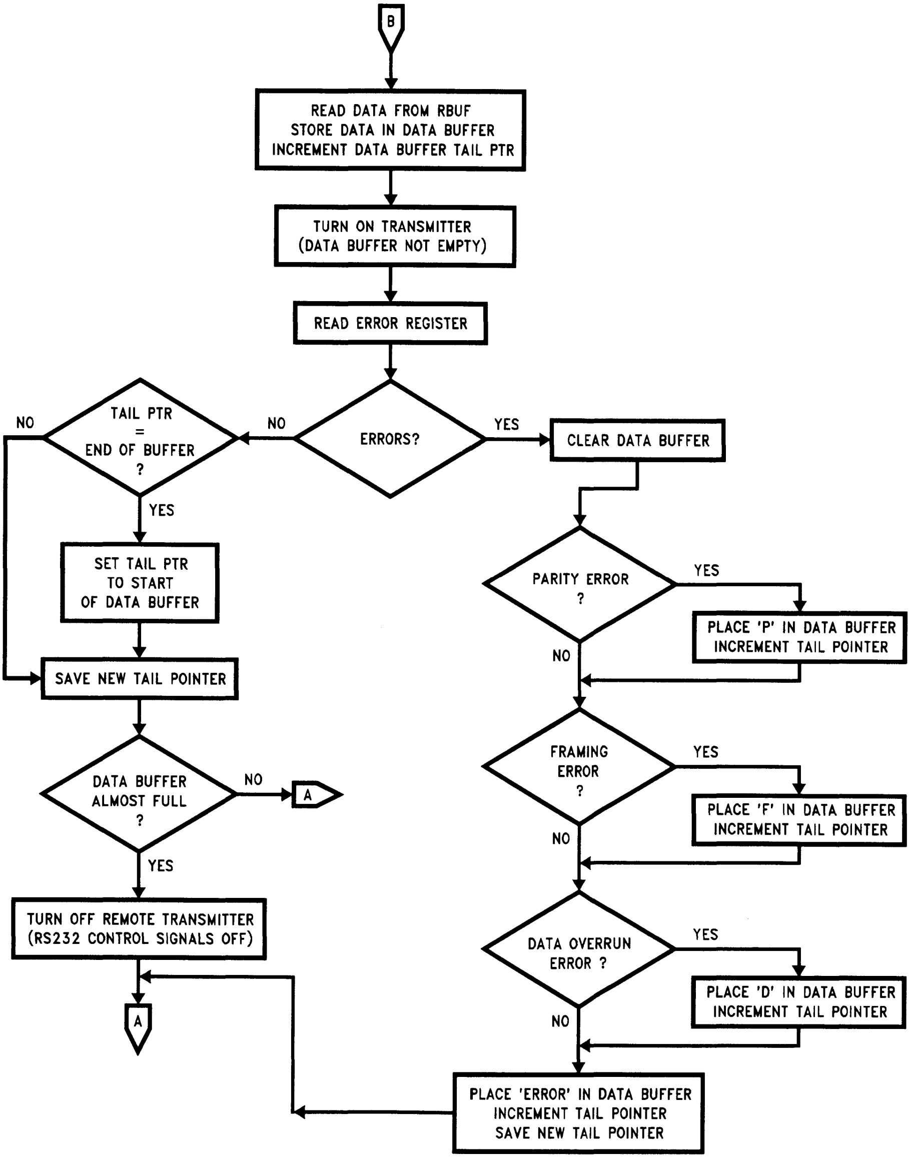
TL/DD/11110-1

FIGURE 1. Interface Diagram



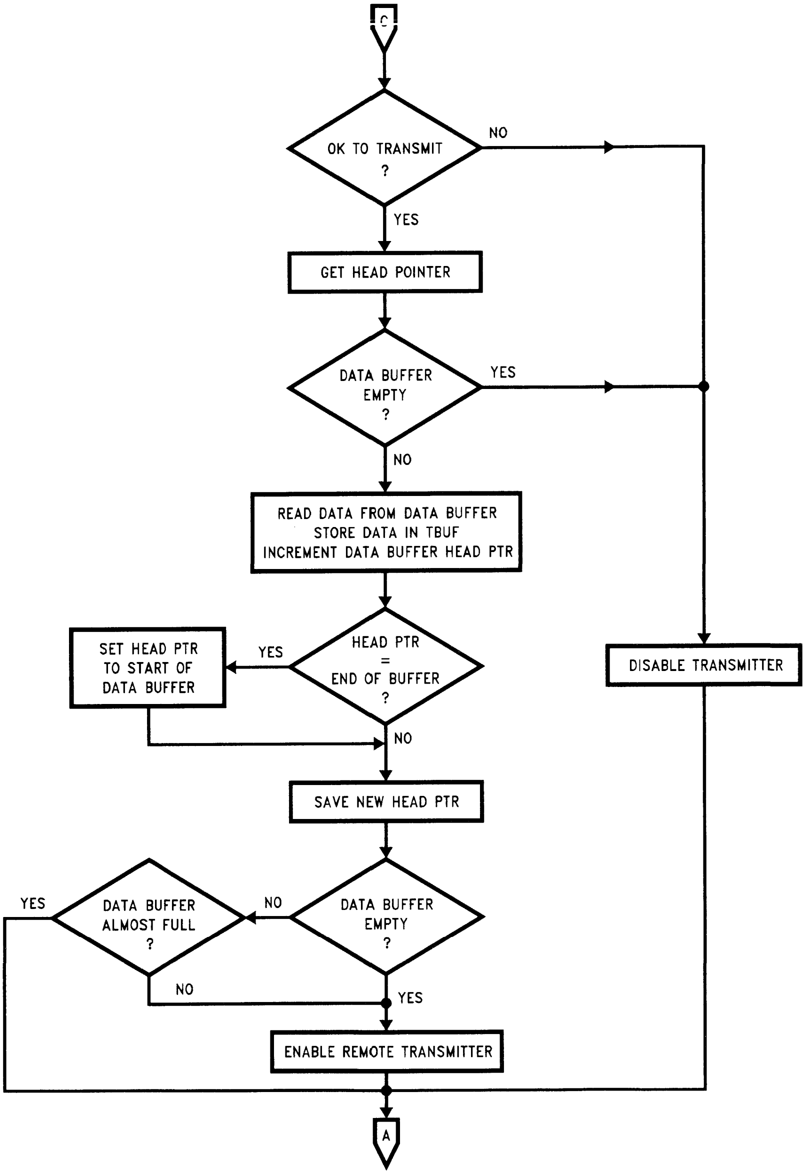
TL/DD/11110-2

FIGURE 2. Main Program Flow



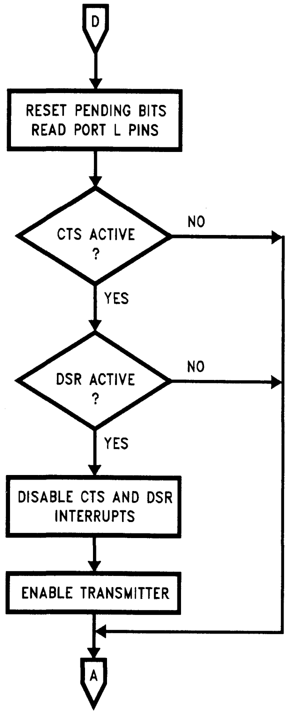
TL/DD/11110-3

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


     0000                   DCD     =  00     ;Bit position of DCD signal on L port pins
     0005                   CTS     =  05     ;Bit position of CTS signal on L port pins
     0007                   DTR     =  07     ;Bit position of DTR signal on L port pins
     0004                   RTS     =  04     ;Bit position of RTS signal on L port pins
     0006                   DSR     =  06     ;Bit position of DSR signal on L port pins
     0005                   ETDX    =  05     ;Bit position of TDX enable pin in ENUI
     0000                   TIE     =  00     ;Bit position of TX interrupt enable bit
     0001                   RIE     =  01     ;Bit position of RX interrupt enable bit
     0005                   PE      =  05     ;Bit position of parity error in ENUR
     0006                   FE      =  06     ;Bit position of framing error in ENUR
     0007                   DOE     =  07     ;Bit position of data overrun error in ENUR



                           .INCLD  COP888.INC

0002 3008           MAIN:   JSR     INIT               ;INITIALIZE UART
0004 FF                     JP      .                  ;DO OTHER TASKS
0005 3044                   JSR     DISABLE            ;DISABLE UART
0007 FF                     JP      .                  ;DO OTHER TASKS

                    INIT:
0008 9FEF                   LD      B,#PSW
000A 68                     RBIT    GIE,[B]            ;DISABLE ALL INTERRUPTS
000B BCBE00                 LD      PSR,#00            ;UART OFF (POWERDOWN)
000E BCD165                 LD      PORTLC,#065        ;SET I/O
0011 9FD0                   LD      B,#PORTLD          ;NOT READY TO RECEIVE
0013 7E                     SBIT    DSR,[B]            ;   TURN OFF DATA SET READY
0014 7D                     SBIT    CTS,[B]            ;   TURN OFF CLEAR TO SEND
0015 68                     RBIT    DCD,[B]            ;   TURN ON DATA CARRIER DETECT
0016 BC1E10                 LD      HEAD,#START        ;INIT HEAD POINTER
0019 BC1F10                 LD      TAIL,#START        ;INIT TAIL POINTER
001C 9FE8                   LD      B,#ICNTRL          ;CONFIGURE PORTL INTERRUPTS
001E 6E                     RBIT    LPEN,[B]           ;  DISABLE PORTL INTERRUPTS
001F BCC890                 LD      WKEDG,#090         ;  SELECT FALLING EDGE FOR RTS AND DTR
0022 BCC990                 LD      WKEN,#090          ;  ENABLE RTS AND DTR INTERRUPT
0025 BCCA00                 LD      WKPND,#00          ;  CLEAR PORTL INTERRUPT PENDING FLAGS
0028 7E                     SBIT    LPEN,[B]           ;  ENABLE PORT L INTERRUPTS
0029 BCBA89                 LD      ENU,#ENUCMD        ;SELECT BITS/CHAR AND PARITY OPTION
002C BCBB00                 LD      ENUR,#00           ;CLEAR ERROR BITS
002F BCBC20                 LD      ENUI,#ENUICMD      ;SELECT CLOCK,INTERRUPTS,STOPBITS
0032 BCBD04                 LD      BAUD,#BAUDVAL      ;SETUP BRG
0035 9FBC                   LD      B,#ENUI
0037 78                     SBIT    TIE,[B]            ;ENABLE TRANSMITTER INTERRUPT
0038 79                     SBIT    RIE,[B]            ;ENABLE RECEIVER INTERRUPT
0039 BCBEC8                 LD      PSR,#PSRVAL        ;UART ON
003C 9FD0                   LD      B,#PORTLD          ;READY TO RECEIVE
003E 6E                     RBIT    DSR,[B]            ;  TURN ON DATA SET READY
003F 6D                     RBIT    CTS,[B]            ;  TURN ON CLEAR TO SEND
0040 9FEF                   LD      B,#PSW
0042 78                     SBIT    GIE,[B]            ;ENABLE ALL INTERRUPTS
```

TL/DD/11110-7

```
NATIONAL SEMICONDUCTOR CORPORATION
COP800 CROSS ASSEMBLER,REV:D1,12 OCT 88


0043 8E                   RET

                DISABLE:
0044 BDEF68               RBIT    GIE,PSW         ;DISABLE INTERRUPTS
0047 BCD061               LD      PORTLD,#061     ;TURN OFF HANDSHAKING SIGNALS
004A BCBE00               LD      PSR,#00         ;UART POWERDOWN
004D BCBA00               LD      ENU,#00         ;CLEAR UART CONTROL REGISTERS
0050 BCBC00               LD      ENUI,#00
0053 BCBB00               LD      ENUR,#00
0056 9FC9                 LD      B,#WKEN         ;DISABLE RTS AND DTR INTERRUPTS
0058 6C                   RBIT    RTS,[B]
0059 6F                   RBIT    DTR,[B]
005A BDEF78               SBIT    GIE,PSW         ;ENABLE INTERRUPTS
005D 8E                   RET


                ;INTERRUPT ROUTINES

     00FF                 . = 0FF                 ;INTERRUPT START ADDRESS
00FF 67                   PUSH    A               ;CONTEXT SAVE
0100 9DFE                 LD      A,B
0102 67                   PUSH    A
0103 9DEF                 LD      A,PSW
0105 67                   PUSH    A
0106 B4                   VIS
0107 8C         REST:     POP     A               ;CONTEXT RESTORE
0108 9CEF                 X       A,PSW
010A 8C                   POP     A
010B 9CFE                 X       A,B
010D 8C                   POP     A
010E 8F                   RETI


                ;PORT L INTERRUPTS
                ;  The port L interrupts are used to indicate a return to active
                ; state of the DTR and RTS signals from the remote receiver.
                ; If both DTR and RTS are active, the remote receiver is ready
                ; to accept data and the COP transmitter is enabled.

                LINT:                             ;PORT L INTERRUPT
010F BCCA00               LD      WKPND,#00       ;RESET PENDING BITS
0112 9DD2                 LD      A,PORTLP        ;READ PORT L PINS
0114 6010                 IFBIT   #RTS,A          ;IF RTS (ACTIVE LOW) NOT PRESENT
0116 06                   JP      NOTRDY          ;THEN REMOTE NOT READY TO RECEIVE
0117 6080                 IFBIT   #DTR,A          ;IF DTR (ACTIVE LOW) NOT PRESENT
0119 03                   JP      NOTRDY          ;THEN REMOTE NOT READY TO RECEIVE
011A 9FBC       READY:    LD      B,#ENUI
011C 78                   SBIT    TIE,[B]         ;RE-ENABLE TRANSMITTER INTERRUPT
011D E9         NOTRDY:   JP      REST            ;EXIT INTERRUPT


```

TL/DD/11110-8

```
NATIONAL SEMICONDUCTOR CORPORATION
COP800 CROSS ASSEMBLER,REV:D1,12 OCT 88


                    ;UART RECEIVE INTERRUPT
                    ;  The UART receive interrupt does the following:
                    ;      1. Reads the received data
                    ;      2. Checks for receiver errors
                    ;      3. If no errors detected, places the received data in
                    ;          the transmit/receive buffer and enables the transmitter.
                    ;      4. If errors detected, the transmit/receive buffer is cleared
                    ;          of ALL data and an error message is placed in the data buffer.
                    RCVINT:                             ;RECEIVER INTERRUPT
011E 9D1F                   LD      A,TAIL
0120 9CFE                   X       A,B                 ;GET TAIL POINTER
0122 9DB9                   LD      A,RBUF              ;READ RECEIVED DATA
0124 A2                     X       A,[B+]              ;STORE RECEIVED DATA
0125 9DBB                   LD      A,ENUR              ;READ ERROR REGISTER
0127 BDBC78                 SBIT    TIE,ENUI            ;ENABLE TRANSMITTER INTERRUPT
012A 60E0                   ANDSZ   A,#0E0              ;CHECK FOR PE,DOE,FE
012C 1A                     JP      ERROR               ;THROW DATA AWAY IN BUFFER
012D 9DFE                   LD      A,B                 ;LOAD ACC WITH NEW TAIL PTR
012F 921E                   IFEQ    A,#END+1            ;IF END OF DATA BUFFER
0131 9810                   LD      A,#START            ; SET TAIL PTR TO START OF BUFFER
0133 9C1F                   X       A,TAIL              ;SAVE TAIL PTR
0135 9D1E                   LD      A,HEAD              ;IS DATA BUFFER FULL?
0137 A1                     SC
0138 BD1F81                 SUBC    A,TAIL              ;A = HEAD - TAIL
013B 89                     IFNC                        ;IF BORROWED (TAIL > HEAD)
013C 940E                   ADD     A,#SIZE             ;THEN ADD BUFFER SIZE TO RESULT
013E 9303                   IFGT    A,#03               ;IF DATA BUFFER NOT FULL
0140 2107                   JMP     REST                ;  THEN EXIT INTERRUPT
0142 BDD07D         RXOFF:  SBIT    CTS,PORTLD          ;  ELSE TURN OFF REMOTE TRANSMITTER
0145 2107                   JMP     REST                ;EXIT INTERRUPT
                    ERROR:
0147 BC1E10                 LD      HEAD,#START         ;CLEAR BUFFER
014A 9F10                   LD      B,#START            ;POINT TO START OF BUFFER
014C 6020                   IFBIT   PE,A
014E 9A50                   LD      [B+],#'P'           ;P = PARITY
0150 6040                   IFBIT   FE,A
0152 9A46                   LD      [B+],#'F'           ;F = FRAMING
0154 6080                   IFBIT   DOE,A
0156 9A44                   LD      [B+],#'D'           ;D = DATA OVERRUN
0158 9A20                   LD      [B+],#020           ;BLANK SPACE
015A 9A45                   LD      [B+],#'E'
015C 9A52                   LD      [B+],#'R'
015E 9A52                   LD      [B+],#'R'
0160 9A4F                   LD      [B+],#'O'
0162 9A52                   LD      [B+],#'R'
0164 9A0A                   LD      [B+],#0A            ;LINE FEED
0166 9A0D                   LD      [B+],#0D            ;CARRIAGE RETURN
0168 9DFE           OUTERR: LD      A,B                 ;SAVE NEW TAIL PTR
016A 9C1F                   X       A,TAIL
016C 2107                   JMP     REST

```

TL/DD/11110-9

```
NATIONAL SEMICONDUCTOR CORPORATION
COP800 CROSS ASSEMBLER,REV:D1,12 OCT 88


                   ;UART TRANSMIT INTERRUPT
                   ;  The UART transmit interrupt does the following:
                   ;       1. Checks for RTS and DTR signals (OK to transmit?)
                   ;       3. If OK to transmit and buffer not empty, transmits data.
                   ;       4. If not OK to transmit or buffer empty, disables transmitter.

                   XMITINT:                            ;TRANSMITTER INTERRUPT
016E 9DD2                  LD      A,PORTLP
0170 6090                  ANDSZ   A,#090              ;IS IT OK TO TRANSMITT?
0172 219C                  JMP     IDLE                ;NO:  GO TURN OFF TRANSMITTER
0174 9D1E                  LD      A,HEAD              ;YES: GET PTR TO DATA
0176 BD1F82                IFEQ    A,TAIL              ;IF DATA BUFFER EMPTY
0179 219C                  JMP     IDLE                ;THEN TURN OFF TRANSMITTER
017B 9CFE                  X       A,B                 ;ELSE
017D AA                    LD      A,[B+]              ;GET TRANSMIT DATA
017E 9CB8                  X       A,TBUF              ;SEND TRANSMIT DATA
0180 9DFE                  LD      A,B                 ;LOAD ACC WITH NEW HEAD PTR
0182 921E                  IFEQ    A,#END+1            ;IF END OF DATA BUFFER
0184 9810                  LD      A,#START            ;  SET HEAD PTR TO START OF BUFFER
0186 9C1E                  X       A,HEAD              ;SAVE HEAD PTR
0188 9D1E                  LD      A,HEAD              ;IS DATA BUFFER FULL?
018A BD1F82                IFEQ    A,TAIL              ;IF BUFFER EMPTY
018D 09                    JP      NFULL               ;    THEN NOT FULL
018E A1                    SC                          ;    ELSE CHECK HOW FULL
018F BD1F81                SUBC    A,TAIL              ;A = HEAD - TAIL
0192 89                    IFNC                        ;IF BORROWED (TAIL > HEAD)
0193 940E                  ADD     A,#SIZE             ;THEN ADD BUFFER SIZE TO RESULT
0195 9303                  IFGT    A,#03               ;IF DATA BUFFER NOT FULL
0197 BDD06D        NFULL:  RBIT    CTS,PORTLD          ;    THEN TURN ON REMOTE TRANSMITTER
019A 2107                  JMP     REST                ;    ELSE EXIT INTERRUPT
019C 9FBC          IDLE:   LD      B,#ENUI
019E 68                    RBIT    TIE,[B]             ;DISABLE TRANSMITTER INTERRUPT
019F 2107                  JMP     REST                ;EXIT INTERRUPT

                   ;Software Trap
                   ;
01A1 B5            SFTINT: RPND
01A2 2000                  JMP     00                  ;RESTART

                   ;VECTOR INTERRUPT TABLE

     01E2                  .=01E2
01E2 010F                  .ADDRW  LINT                ;L PORT INTERRUPT
     01EC                  .=01EC
01EC 016E                  .ADDRW  XMITINT             ;TRANSMITTER INTERRUPT
01EE 011E                  .ADDRW  RCVINT              ;RECEIVER INTERRUPT
     01FE                  .=01FE
01FE 01A1                  .ADDRW  SFTINT              ;SOFTWARE INTERRUPT/TRAP
                           .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

AN-952

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

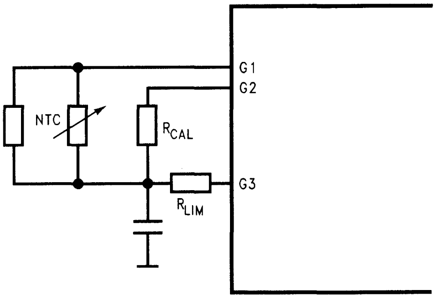
The linearization reduces the code, improves the accuracy and the tolerance of the NTC-R network (e.g. NTC = 100 kΩ ±10%, R = 12 kΩ ±1%, NTC//R ±2%). Using that method the useful range does not cover the whole operating temperature range of the NTC.

GENERAL ACCURACY CONSIDERATIONS

Using a single slope A/D conversion the accuracy is dependent on the following parameters:

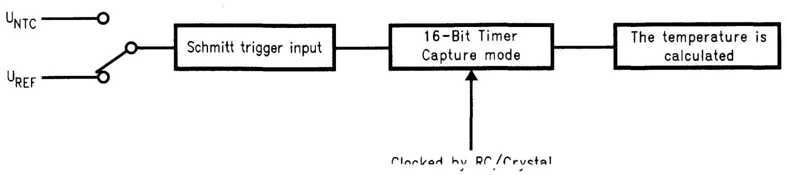
- Stability of the Clock frequency
- Time constant of the RC network
- Accuracy of the Schmitt Trigger level
- Non-linearity of the RC-network

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.



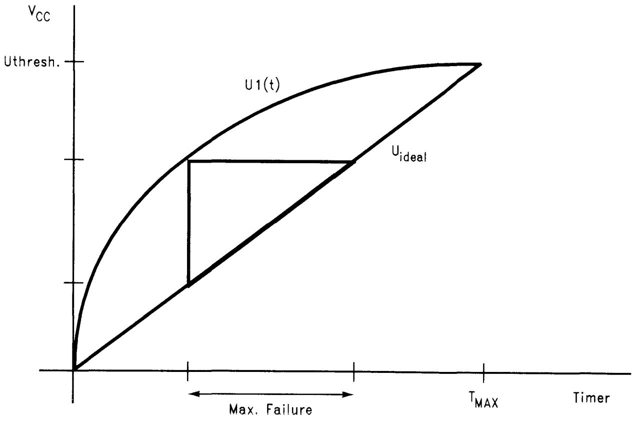
TL/DD12075-2

FIGURE 2. Basic Circuit Implementation



TL/DD12075-1

FIGURE 1. Simple A/D Conversion



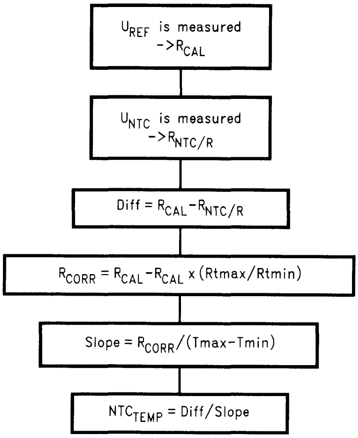
TL/DD12075-3

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.



TL/DD12075-4

FIGURE 4. Accuracy Improvement

Measurement:

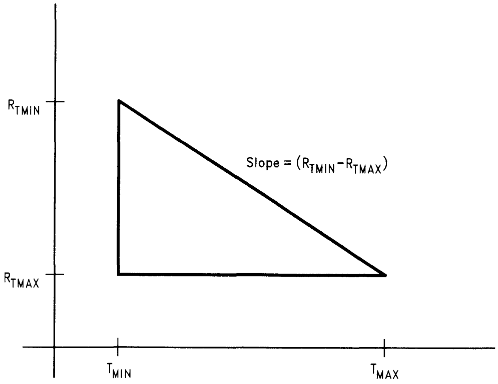
- Timer Capture mode: R_{CAL} * C is measured
- Timer Capture mode: $R_{NTC//R}$ * C is measured

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.



TL/DD12075-5

FIGURE 5. Resistance vs Temperature Characteristics

APPLICATION EXAMPLE

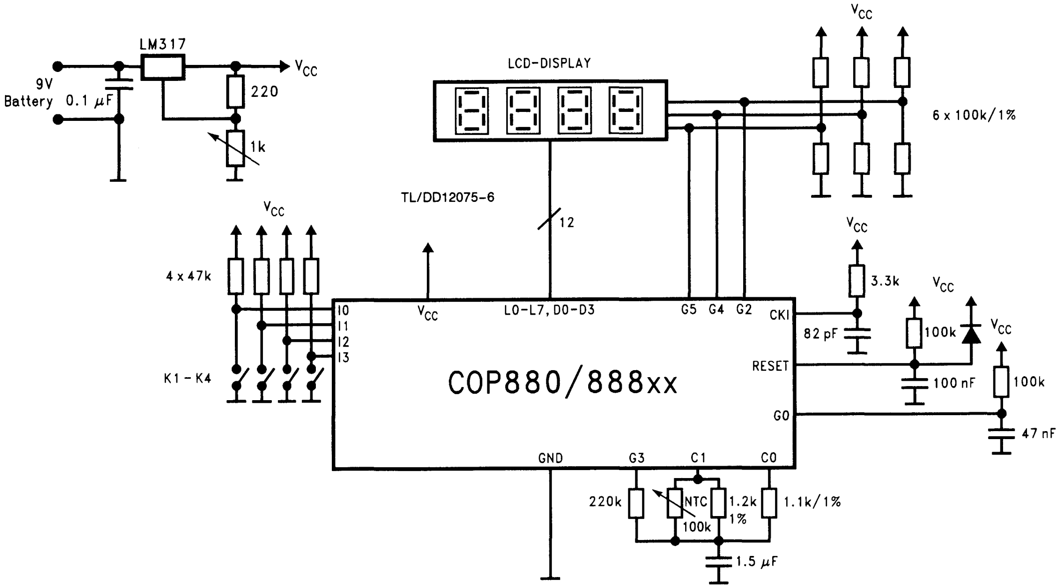
The following application example for temperature measurement demonstrates the procedure. The temperature is measured from 20° to 100° and is displayed on a Triplex LCD display.

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.



TL/DD12075-6

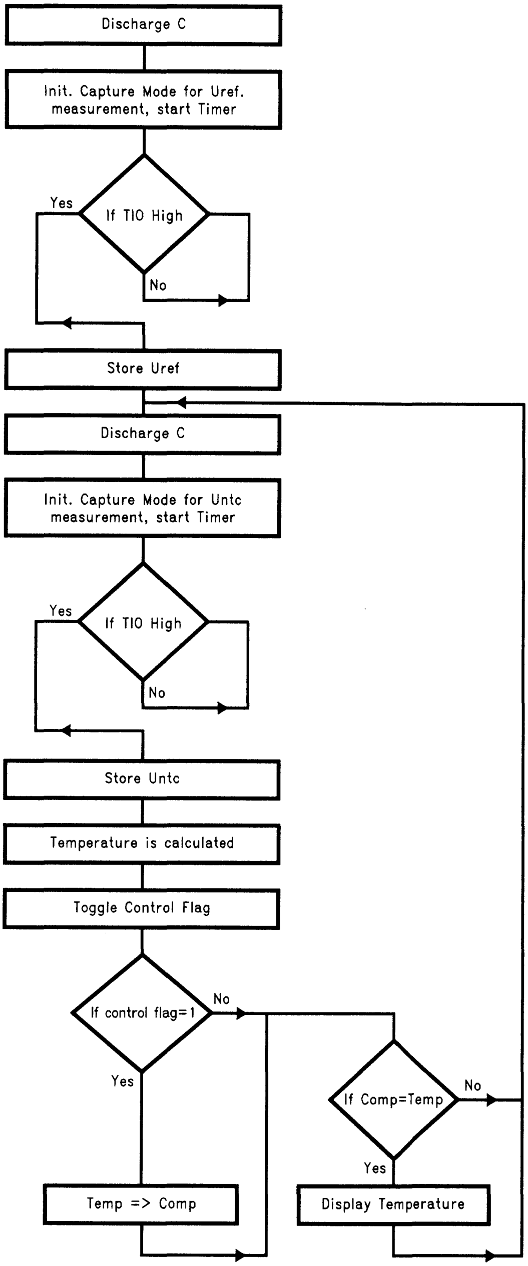
TL/DD12075-7

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

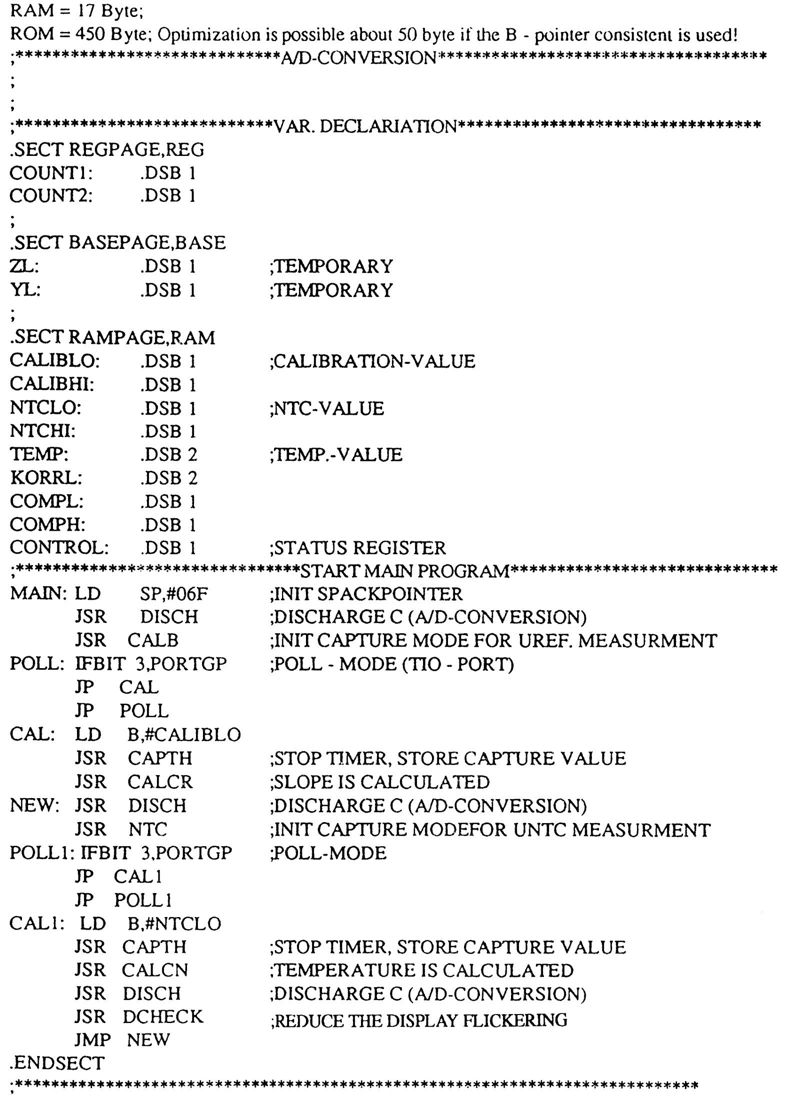
Figure 7 shows the flow chart of the program.



TL/DD12075-8

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/DD12075-9

```
;*****************************************************************************
.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/DD12075-10

```
;*****************************************************************************
;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/DD12075-11

```
        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/DD12075-12

```
;*************************************************************************
;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/DD12075-13

LCD Triplex Drive with COP820CJ

National Semiconductor
Application Note 953
Klaus Jaensch and
Siegfried Rueth

INTRODUCTION

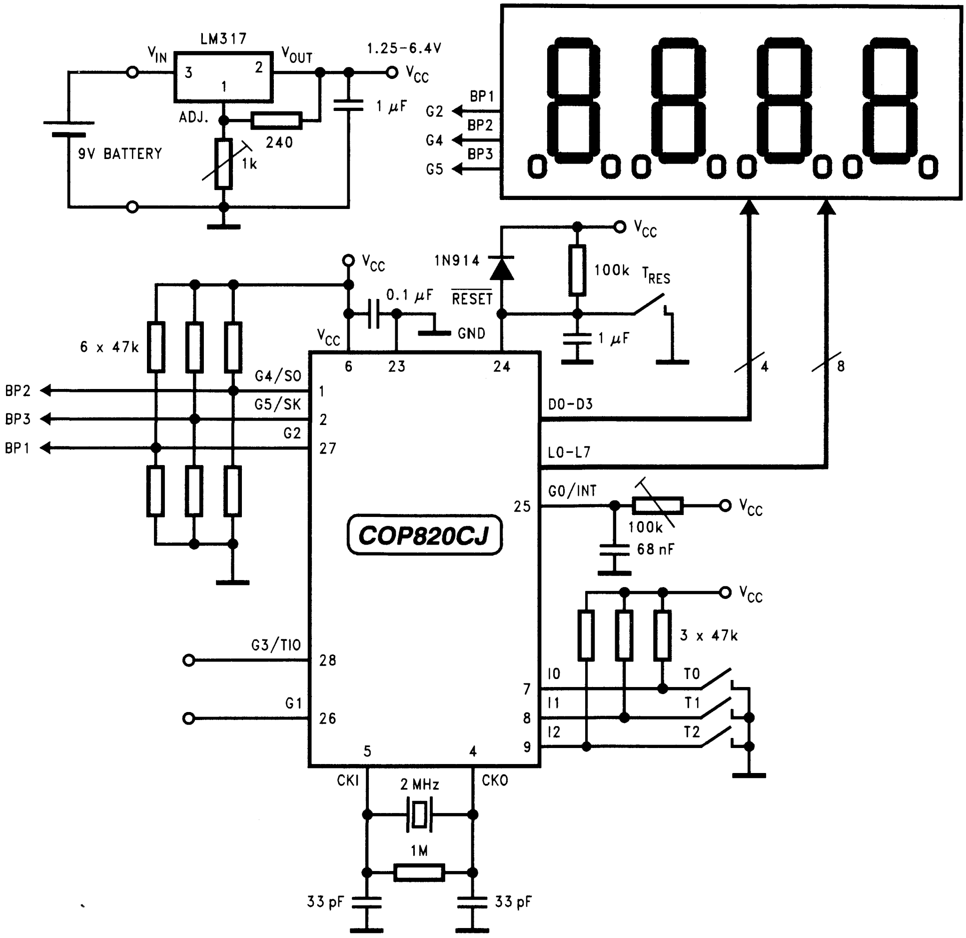
There are many applications which use a microcontroller in combination with a Liquid Crystal Display. The normal method to control a LCD panel is to connect it to a special LCD driver device, which receives the display data from a microcontroller. A cheaper solution is to drive the LCD directly from 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.

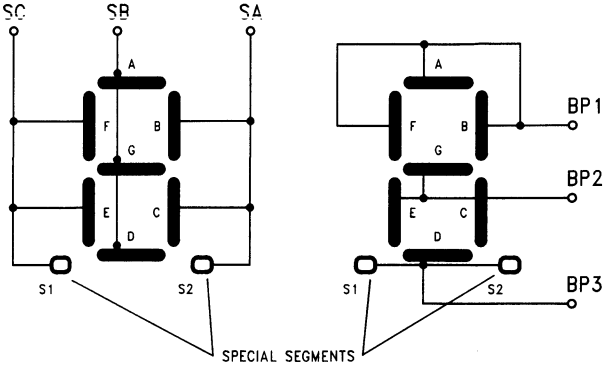
The multiplex rate of a LCD is determined by the number of its backplanes (segment-common planes). The number of segments controlled by one line (with one segment pin) is equal to the number of backplanes on the LCD. So, a three way multiplexed LCD has three backplanes and three segments are controlled 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/DD12076–1

FIGURE 1. Schematic for LCD Triplex Driver



TL/DD12076-2

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.

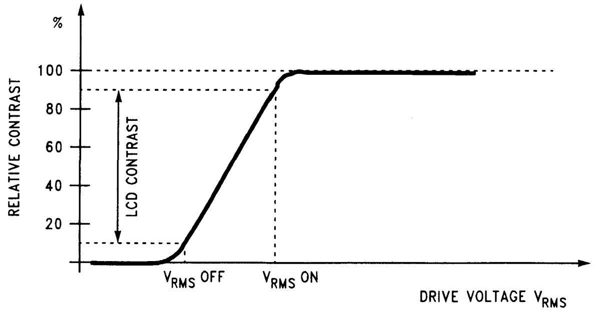
Common to all LCD's is that the voltage across backplane(s) and segment(s) has to be an AC-voltage. This is to avoid electrochemical degradation of the liquid crystal layer. A segment being "off" or "on" depends on the **r.m.s.** voltage across a segment.

The maximum attainable ratio of "on" to "off" r.m.s. voltage (discrimination) is determined by the multiplex ratio. It is given by:

$(V_{ON}/V_{OFF})\text{max} = \text{SQR}((\text{SQR}(N) + 1)/(\text{SQR}(N) - 1))$
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.



TL/DD12076-3

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.

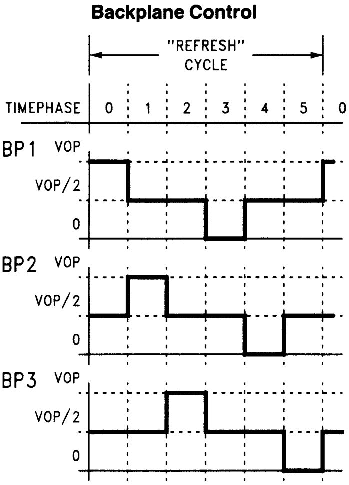
The backplane connection scheme is shown in *Figure 1*.

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.



TL/DD12076-4

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

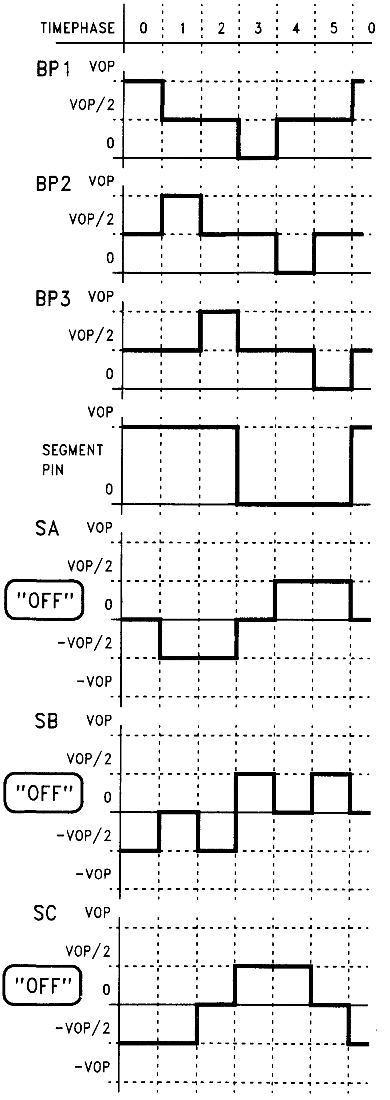
Figure 5 through *Figure 12* below show all possible combinations of controlling a **"Segment Triple"** with help of the 3 backplane connections and one segment pin. The segment switching has to be done according to the ON/OFF combination required (see also Table I).

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).

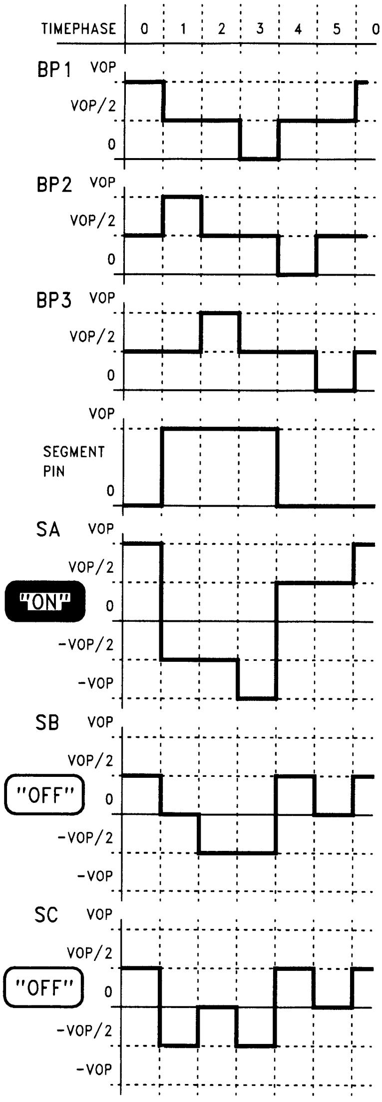
Segment/Backplane Control-Timing



TL/DD12076-5

tiphtab address = 0

FIGURE 5

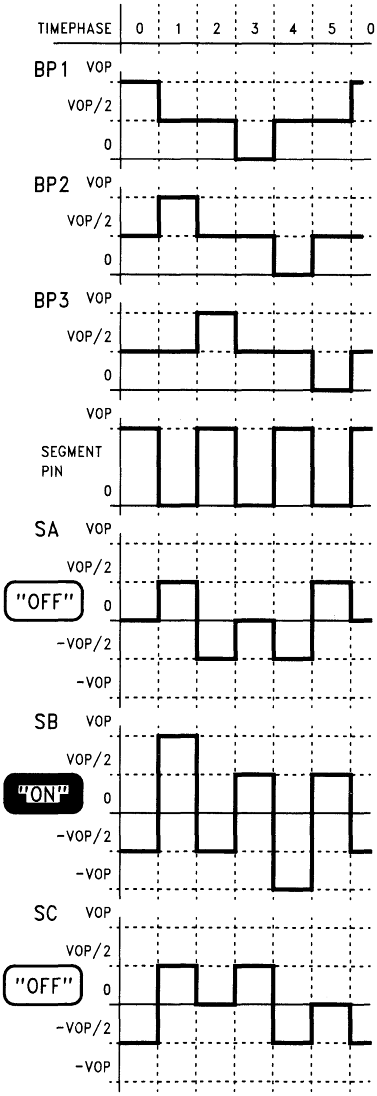


TL/DD12076-6

tiphtab address = 1

FIGURE 6

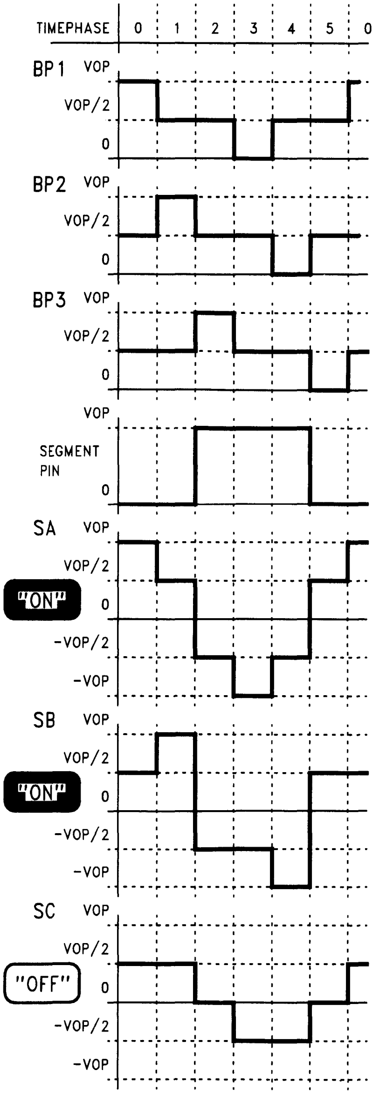
Segment/Backplane Control-Timing



TL/DD12076-7

tiphtab address = 2

FIGURE 7

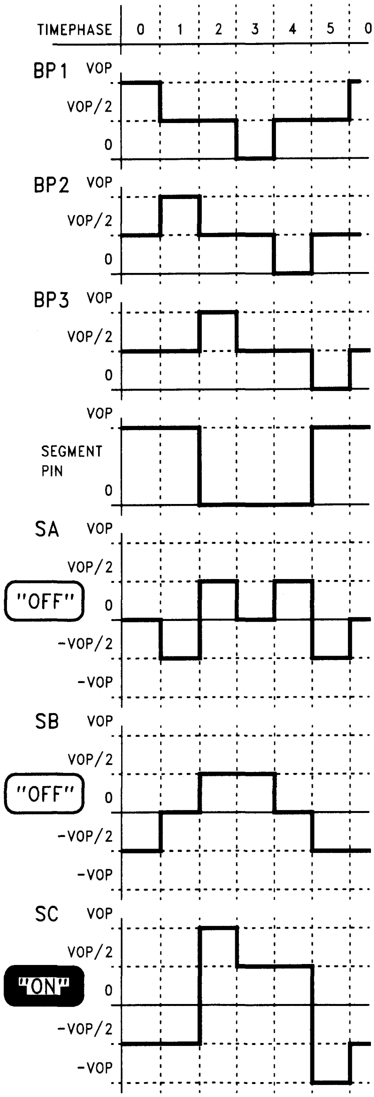


TL/DD12076-8

tiphtab address = 3

FIGURE 8

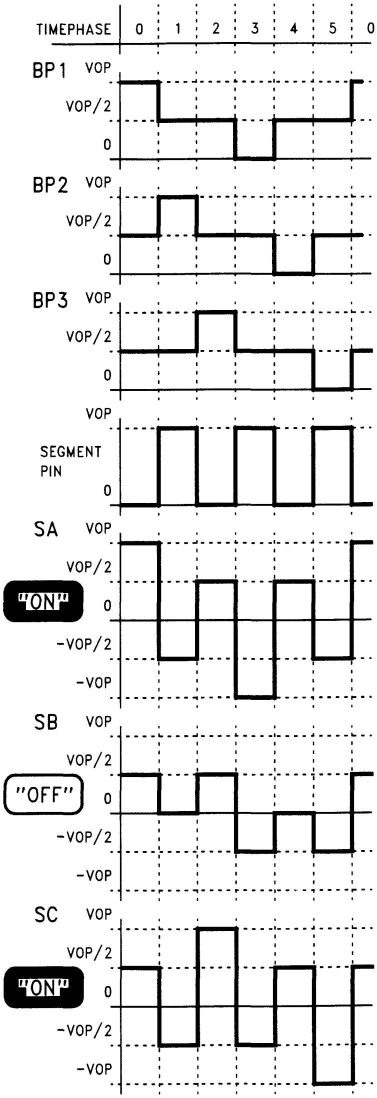
Segment/Backplane Control-Timing



TL/DD12076-9

tiphtab address = 4

FIGURE 9

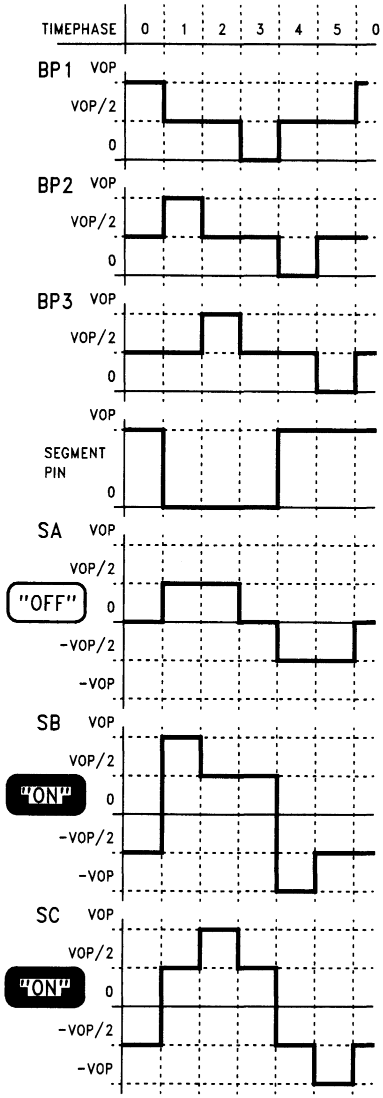


TL/DD12076-10

tiphtab address = 5

FIGURE 10

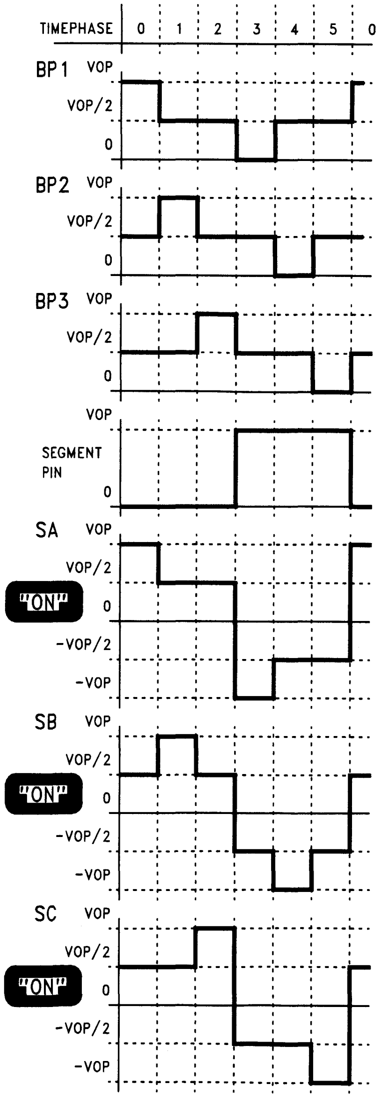
Segment/Backplane Control-Timing



TL/DD12076-11

tiphtab address = 6

FIGURE 11



TL/DD12076-12

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:

$$\mathbf{(V_{TH}/V_{CC})\ min = 0.376}$$

$$\mathbf{(V_{TH}/V_{CC})\ max = 0.572}$$

at $V_{CC} = 5V$

Charge Time:

$$\mathbf{T = -(ln(1\text{-}V_{TH}/V_{CC})^*RC)}$$

To prevent a flickering display one should aim at a minimum refresh frequency of $f_{refr} = 30$ Hz. This means an interrupt frequency of $f_{int} = 6 \times 30$ 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:

$$\mathbf{RC_{max} = T_{max}/(-ln(1-(V_{TH}/V_{CC})max)) = 5.5\ ms \times 0.849}$$

$$\mathbf{RC_{max} = 6.48\ ms}$$

$$\mathbf{(RC_{min} = 5.98\ ms)}$$

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:

$$\mathbf{T_{min} = RC_{min}{}^*(-ln(1-(V_{TH}/V_{CC})min = RC_{min}{}^*0.472}$$

and

$$\mathbf{f_{refr,max} = f_{int,max}/6 = 1/(T_{min}{}^*6)}$$

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.

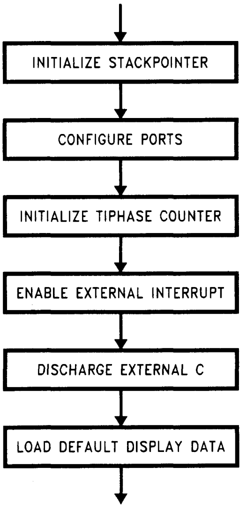
SOFTWARE

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/DD12076-13

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 timephase from the **seg1st**.

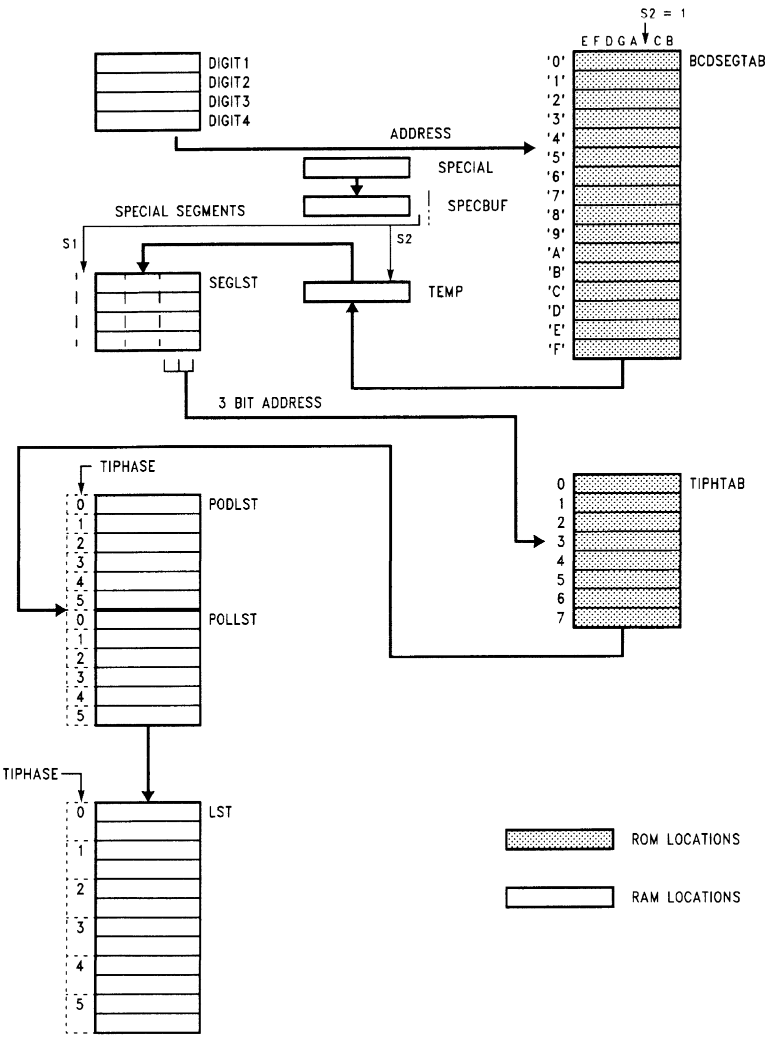
These 3 bits address the 8 bytes of the **tiphtab** table in ROM. Each byte of this table contains the **time curve** for a segment pin (only 6 bits out of 8 are used).Using this information, the program creates the lists **for port D and port L (pod1st, pol1st)**. Every byte of this list contains the **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

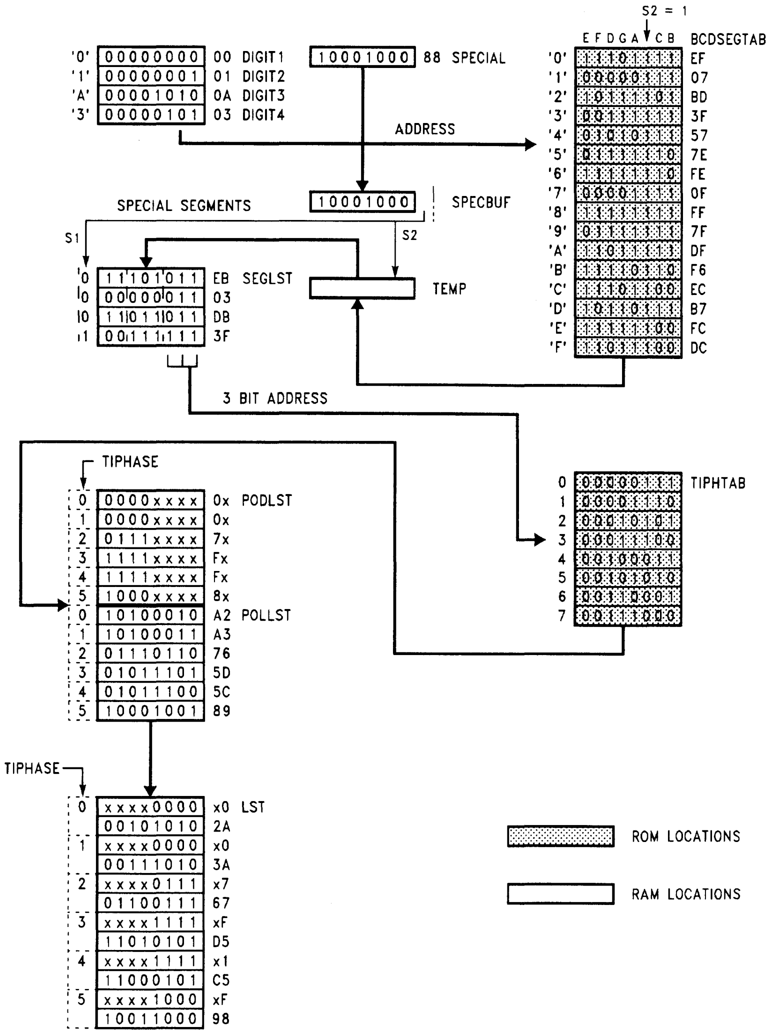
ROM: 152 bytes incl. look up tables

RAM: 43 bytes (*Figure 15* illustrates the RAM locations)



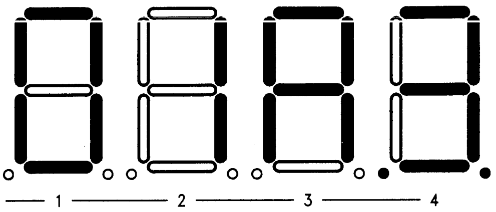
TL/DD12076-14

FIGURE 14. Data Flow Chart for Update Routine



TL/DD12076-15

FIGURE 15. Data Flow Chart for Update Routine



TL/DD12076-16

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

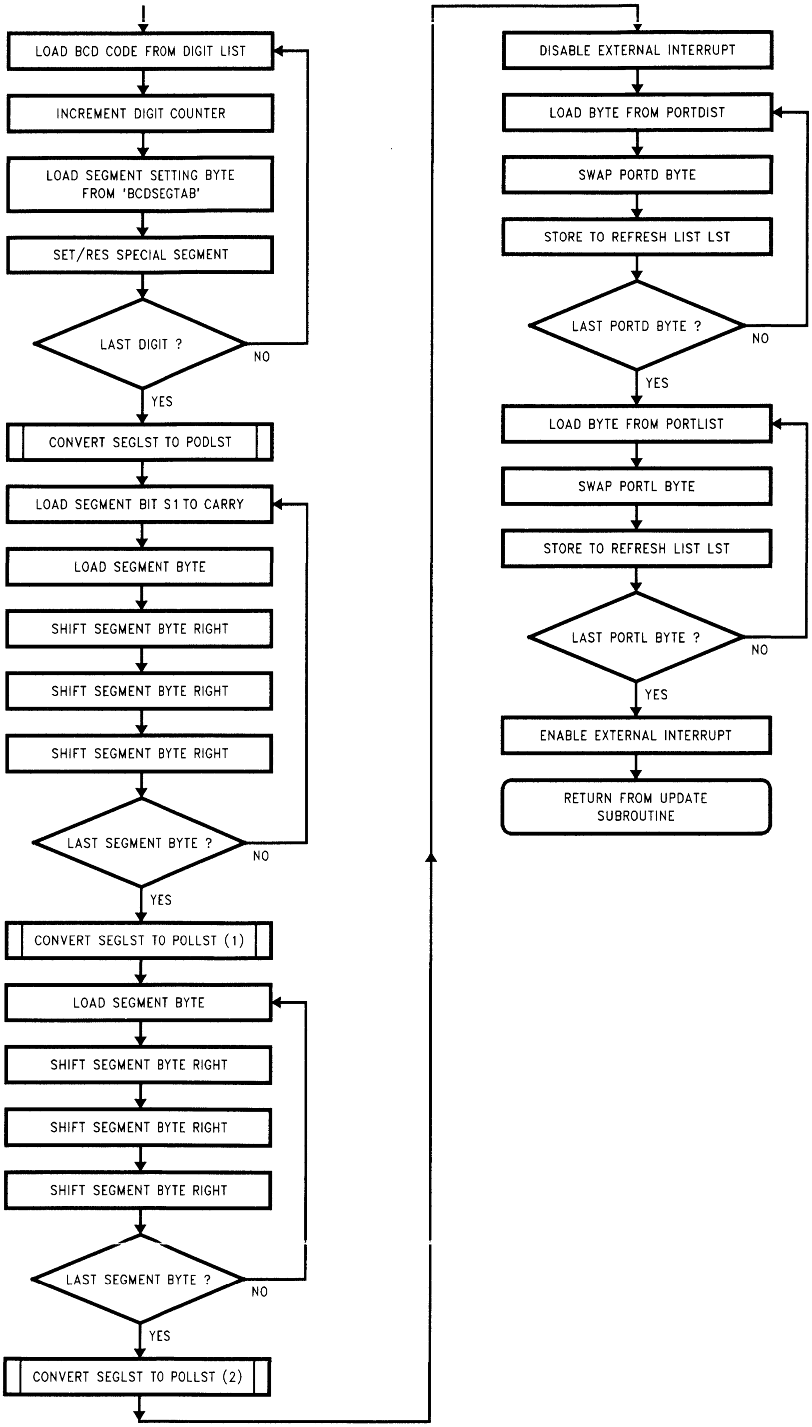
Tiphase	G5	G4	G2	Portg Data	Hex	Portg Config.	Hex
0	0/0	0/0	1/1	XX00X1XX	04	XX00X1XX	04
1	0/0	1/1	0/0	XX01X0XX	10	XX01X0XX	10
2	1/1	0/0	0/0	XX10X0XX	20	XX10X0XX	20
3	0/0	0/0	0/1	XX00X0XX	00	XX00X1XX	04
4	0/0	0/1	0/0	XX00X0XX	00	XX01X0XX	10
5	0/1	0/0	0/0	XX00X0XX	00	XX10X0XX	20

data/configuration register of portg

0/0 : Hi-Z input
0/1 : output low
1/1 : output high

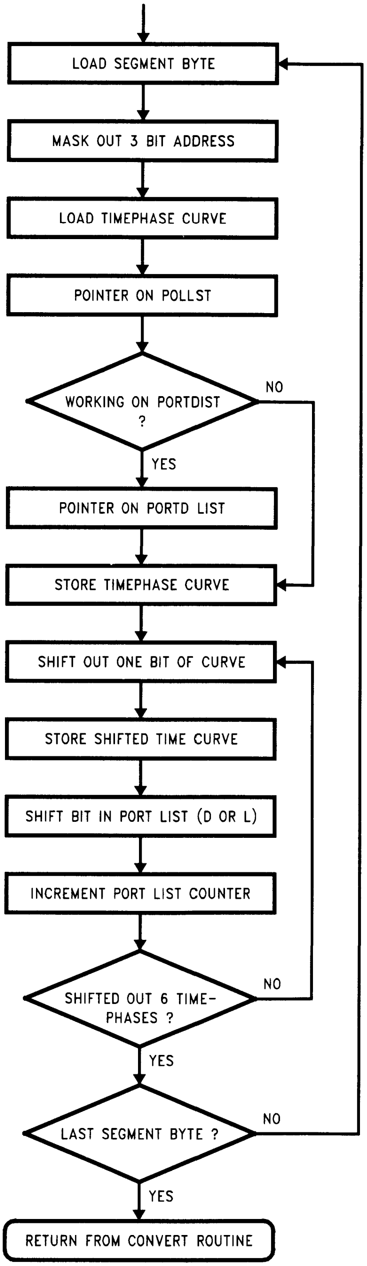
SUMMARY OF IMPORTANT DATA

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/DD12076-17

FIGURE 17. Flowchart for Update Routine



TL/DD12076-18

FIGURE 18. Flowchart for Convert Subroutine

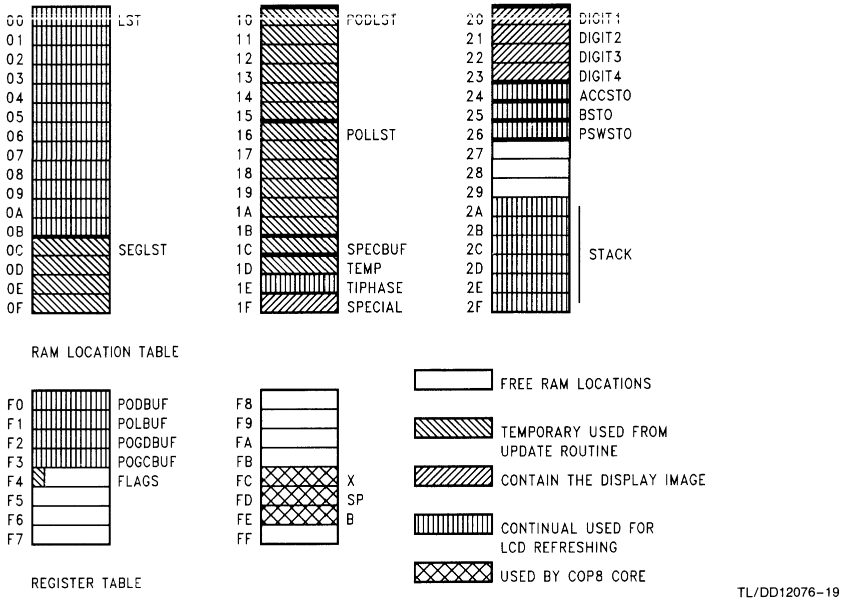
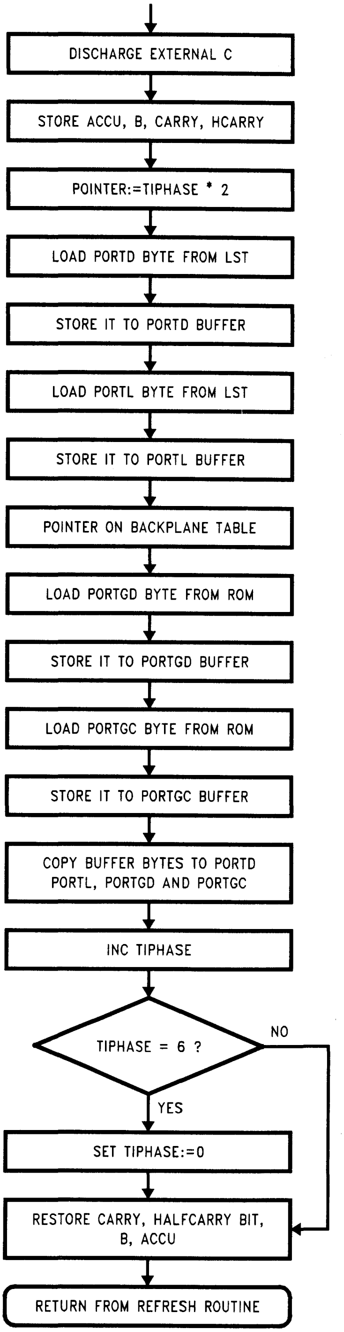


FIGURE 19. RAM Assignment



TL/DD12076-20

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/DD12076-21

```
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/DD12076-22

```
                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/DD12076–23

```
                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/DD12076-24

```
                                  ;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 nxtsgl          ;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/DD12076-25

```
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/DD12076-26

```
                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/DD12076-27

Section 3

MICROWIRE/PLUS™ Peripherals

Section 3 Contents

National Semiconductor

MICROWIRE™ and MICROWIRE/PLUS™: 3-Wire Serial Interface

National's MICROWIRE and MICROWIRE/PLUS provide for high-speed, serial communications in a simple 3-wire implementation.

Originally designed to interface COP400 microcontrollers to peripheral devices, the MICROWIRE protocol has been extended to both the COP800 and HPC™ families with the enhanced version, MICROWIRE/PLUS.

Because 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.

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).

Because the COP800 and HPC families have 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.

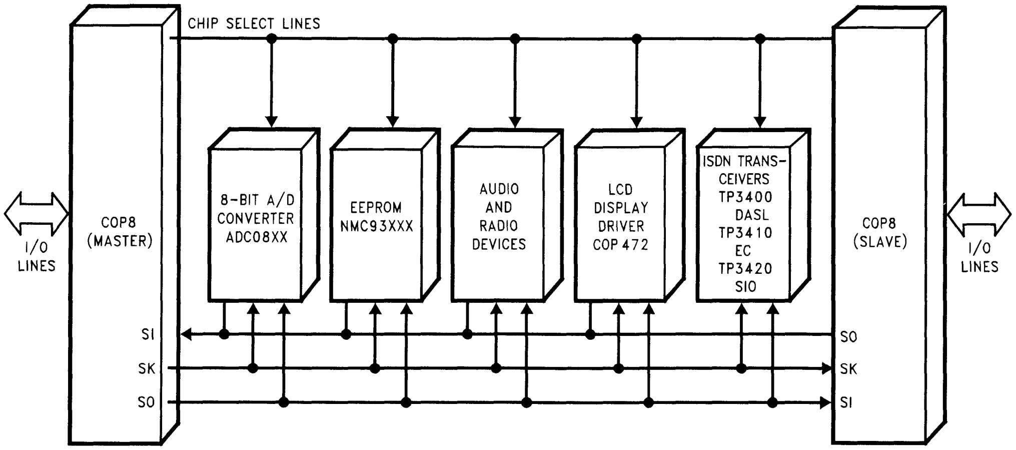
MICROWIRE/PLUS can also provide additional I/O capability for COP800 and HPC microcontrollers by connecting, for example, external 8-bit parallel-to-serial shift registers to 8-bit serial-to-parallel shift registers.

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

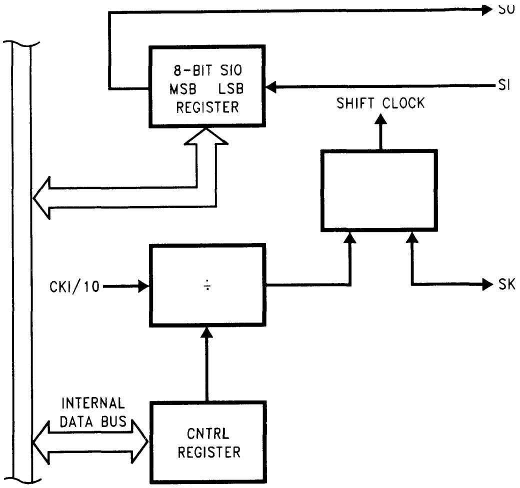
Both MICROWIRE and MICROWIRE/PLUS give all the members of National's microcontroller families the flexibility and design-ease to implement a solution quickly, simply, and cost-effectively.

MICROWIRE/PLUS System Block



TL/XX/0074-1

MICROWIRE/PLUS Block Diagram



TL/XX/0074-2

MICROWIRE and MICROWIRE/PLUS Peripherals

Part Number	Description	Databook
A/D CONVERTERS AND COMPARATORS		
ADC0811	11 Channel 8-Bit A/D Converter with Multiplexer	Linear
ADC0819	19 Channel 8-Bit A/D Converter with Multiplexer	Linear
ADC0831	1 Channel 8-Bit A/D Converter with Multiplexer	Linear
ADC0838	8 Channel 8-Bit A/D Converter with Multiplexer	Linear
ADC0832	2 Channel 8-Bit A/D Converter with Multiplexer	Linear
ADC0833	4 Channel 8-Bit A/D Converter with Multiplexer	Linear
ADC0834	4 Channel 8-Bit A/D Converter with Multiplexer	Linear
ADC0852	Multiplexed Comparator with 8-Bit Reference Divider	Linear
ADC0854	Multiplexed Comparator with 8-Bit Reference Divider	Linear
DISPLAY DRIVERS		
COP472-3	3 x 12 Multiplexed Expandable LCD Display Driver	Microcontroller
MM5450	35 Output LED Display Driver	Interface
MM5451	34 Output LED Display Driver	Interface
MM5483	31 Segment LCD Display Driver	Interface
MM5484	16 Segment LED Display Driver	Interface
MM5486	33 Output LED Display Driver	Interface
MM58201	8 Backplane and 24 Segment Multiplexed LCD Driver	Interface
MM58241	32 Output High Voltage Display Driver	Interface
MM58242	20 Output High Voltage Display Driver	Interface
MM58248	35 Output High Voltage Display Driver	Interface
MM58341	32 Output High Voltage Display Driver	Interface
MM58342	20 Output High Voltage Display Driver	Interface
MM58348	35 Output High Voltage Display Driver	Interface
MEMORY DEVICES		
NM93C06	16 x 16 CMOS EEPROM	Memory
NM93C13	16 x 16 CMOS EEPROM	Memory
NM93C14	64 x 16 CMOS EEPROM	Memory
NM93C46	64 x 16 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
NM93C56	128 x 16 CMOS EEPROM	Memory
NM93CS66	256 x 16 CMOS EEPROM with Write Protect	Memory
NM93C66	256 x 16 CMOS EEPROM	Memory

Note: The low voltage (2V–6V) versions of the NM93C06, NM93C46, NM93C56 and NM93C66 are also available.

MICROWIRE and MICROWIRE/PLUS Peripherals (Continued)

Part Number	Description	Databook
TELECOM DEVICES		
TP3420	S Interface Device (SID)	Telecom
AUDIO AND RADIO DEVICES		
DS8906	AM/FM Digital PLL Synthesizer	Interface
DS8907	AM/FM Digital PLL Frequency Synthesizer	Interface
DS8908	AM/FM Digital PLL Frequency Synthesizer	Interface
DS8911	AM/FM/TV Sound Up-Conversion Frequency Synthesizer	Interface
LMC1992	Stereo Volume/Tone/Fade with Source Select	Linear
LMC1993	Stereo Volume/Tone/Fade/Loudness with Source Select	Linear
LMC835	7 Band Graphic Equalizer	Linear

National Semiconductor

COP472-3 Liquid Crystal Display Controller

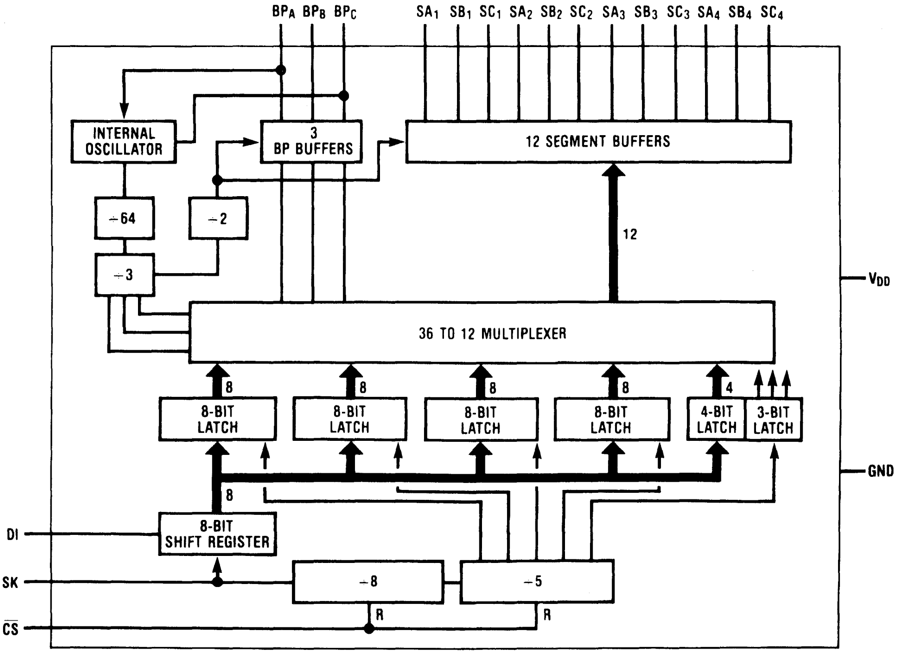
General Description

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} V_{IH}		 0.7 V_{DD}	 0.8 9.5	 Volts Volts
BPA (as Osc. in) V_{IL} V_{IH}		 V_{DD}−0.6	 0.6 V_{DD}	 Volts Volts
Output Levels, BPC (as Osc. Out) V_{OL} V_{OH}		 V_{DD}−0.4	 0.4 V_{DD}	 Volts Volts
Backplane Outputs (BPA, BPB, BPC) $V_{BPA, BPB, BPC}$ ON $V_{BPA, BPB, BPC}$ OFF	 During BP+ Time	 $V_{DD}-\Delta V$ ⅓ $V_{DD}-\Delta V$	 V_{DD} ⅓ $V_{DD}+\Delta V$	 Volts Volts
$V_{BPA, BPB, BPC}$ ON $V_{BPA, BPB, BPC}$ OFF	During BP− Time	0 ⅔ $V_{DD}-\Delta V$	ΔV ⅔ $V_{DD}+\Delta V$	Volts Volts
Segment Outputs (SA_1 ~ SA_4) V_{SEG} ON V_{SEG} OFF	 During BP+ Time	 0 ⅔ $V_{DD}-\Delta V$	 ΔV ⅔ $V_{DD}+\Delta V$	 Volts Volts
V_{SEG} ON V_{SEG} OFF	During BP− Time	$V_{DD}-\Delta V$ ⅓ $V_{DD}-\Delta V$	V_{DD} ⅓ $V_{DD}+\Delta V$	Volts 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} Data Hold, t_{HOLD}		 1.0 100		 μs ns
$\overline{CS}$ t_{SETUP} t_{HOLD}		 1.0 1.0		 μs 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: $\Delta V = 0.05V_{DD}$.

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

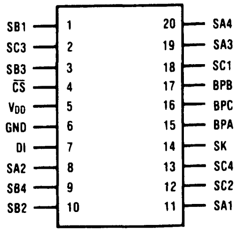
GND = 0V, V_{DD} = 3.0V to 5.5V, T_A = −40°C to +85°C (depends on display characteristics)

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: ΔV = 0.05 V_{DD}.

Dual-In-Line Package



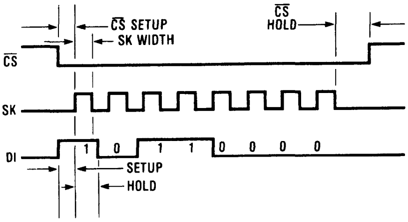
Top View

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

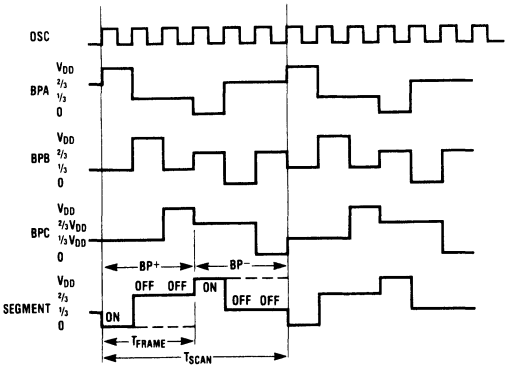
Order Number COP472MW-3 or COP472N-3
See NS Package Number M20A or N20A

FIGURE 2. Connection Diagram



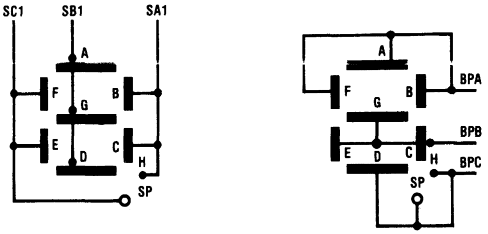
TL/DD/6932-3

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	Digit 1
2	SB1, BPB	SG	
3	SC1, BPA	SF	
4	SC1, BPB	SE	
5	SB1, BPC	SD	
6	SA1, BPB	SC	
7	SA1, BPA	SB	
8	SB1, BPA	SA	
9	SA2, BPC	SH	Digit 2
10	SB2, BPB	SG	
11	SC2, BPA	SF	
12	SC2, BPB	SE	
13	SB2, BPC	SD	
14	SA2, BPB	SC	
15	SA2, BPA	SB	
16	SB2, BPA	SA	
17	SA3, BPC	SH	Digit 3
18	SB3, BPB	SG	
19	SC3, BPA	SF	
20	SC3, BPB	SE	
21	SB3, BPC	SD	
22	SA3, BPB	SC	
23	SA3, BPA	SB	
24	SB3, BPA	SA	
25	SA4, BPC	SH	Digit 4
26	SB4, BPB	SG	
27	SC4, BPA	SF	
28	SC4, BPB	SE	
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 $\overline{\text{CE}}$ 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 $\overline{\text{CS}}$ high.

Note: $\overline{\text{CS}}$ may be turned high after any step. For example to load only 2 digits of data, do steps 1, 2, 3, and 7.

$\overline{\text{CS}}$ 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 $\overline{\text{CS}}$ 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.

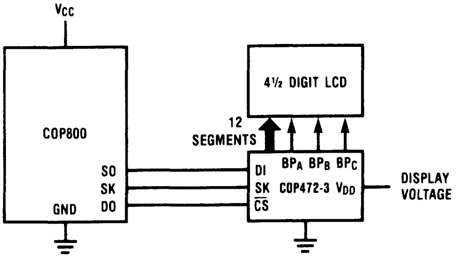
4. Turn CS high to both chips.
5. Turn CS low to master COP472-3.
6. Shift in 32 bits of data for the master's 4 digits.
7. Shift in four bits of special segment data, a one and three zeros.

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.

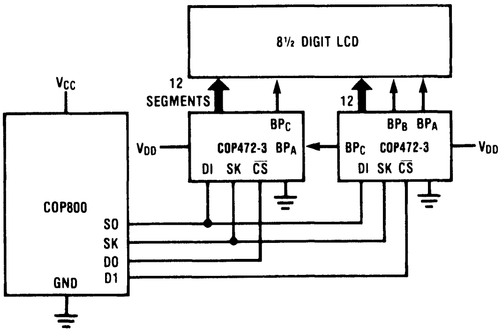
8. Turn $\overline{\text{CS}}$ high.

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

Section 4

COP8 Development Support

Section 4 Contents

National Semiconductor

Development Support

Our job doesn't end when you buy a National microcontroller, it only begins.

The next step is to help you put that microcontroller to work—delivering real-world performance in a real-world application.

That's why we offer you such a comprehensive, powerful, easy-to-use package of development tools.

Microcontroller Development Support
COP400 Family

The COPS™ Microcontroller Development system is a complete, inexpensive system, designed to support both hardware and software development of the COP400 family of microcontrollers.

Using a standard IBM® PC® platform as a host, this system provides the tools to write, assemble, debug and emulate software for user target design.

The development system itself consists of two circuit boards that interface with each other and to the host computer using a software package. The first board is called the Brain Board. It provides the major functional features of the system, linking the various elements of the host system. The other board is called the Personality Board and it is common for all members of the COP400 family of microcontrollers.

Microcontroller Development Support
COP800 Family

MetaLink Corporation's iceMASTER™ COP8 Model 400 In-Circuit Emulator provides complete real-time full speed emulation of all COP8 family devices. It consists of a base unit and interchangeable probe cards, which support various configurations and packages. The source symbolic debugger with a window based user interface is a powerful tool to accomplish software and hardware debug and integration tasks.

COP800 code development is supported by a macro cross-assembler running DOS on the IBM compatible PC.

COP800 development is also supported with a low cost Designer's Kit. The Designer's Kit includes a simulator with a window based menu driven user interface and the COP8 cross-assembler. It is a tool designed for product evaluation and code development and debug. It comes equipped with complete debug capability and full assembler. The host for the designer kit is an IBM PC/XT/AT or compatible running DOS.

Microcontroller Development Support
HPC™ Family

HPC-MDS is a complete packaged system for all members of the HPC family except for HPC46100. The host system is IBM PC/AT® (PC-DOS, MS-DOS) and Sun® SPARCstation (SunOS™). It provides true real time in-system emulation with support tools such as ANSI compatible C-Compiler, assembler, Linker and Source/Symbolic debugger. The debugger interface is based on MS-Windows 3.0 for IBM PC/AT and a line debugger for Sun SPARCstation users.

HPC-MDS gives the user the flexibility to symbolically debug his code and download it to the target hardware. The user can set breakpoints and traces, can execute time measurements and examine and modify internal registers and I/O.

A low cost HPC designer's kit is also available. The kit has complete in-system emulation capability and is packaged with an evaluation version of C compiler and full package of Assembler/Linker.

The HPC46100 DSP-Microcontroller, is supported by a development kit for ROM emulation, logic and timing analysis, code debug with inverse assembly and PC based debug monitor. The kit consists of a Logic Analyzer Interface Board, a Target Board, Assembler/Linker/Librarian software, an inverse assembler to run on Hewlett-Packard 1650 and 16500A/B logic analyzers and PC based debug monitor, "The Serial Hook".

Third Party development support is also available for various sources for the HPC family.

Hewlett Packard offers HP64775 emulator/analyzer for 30 MHz HPC 16083/16064 and 20 MHz 16400E emulation. The stand alone HP system provides a very fast serial link to the host system and offers complete emulation and timing and logic analysis capability. The software tools for HP emulator are provided by National Semiconductor®.

Signum System offers a USP-HPC in-circuit emulator for the HPC46100 with 40 MHz 1 wait state real time emulation. This system is supported with 256 kbyte overlay emulation memory, 32k frames deep trace buffer memory, complex breakpoints, high level language source/symbolic debugger, fast serial download and a window based menu driven user interface.

The language tools hosted on the IBM PC/AT and compatibles and Sun SPARCstation are available from National Semiconductor to support third party emulation systems.

Emulation Technology offers a passive preprocessor and inverse assembler package for HP1650 and 16500A series of Logic analyzers. The preprocessor provides a low cost and convenient way of doing timing and state analysis of the HPC based design.

Emulation Technology also offers debug tool accessories for 68-pin PLCC and 80-pin (QFP) Quad Flat Packages. This includes PLCC to QFP adapter, QFP test clip and a QFP surface mount replacement base.

Programming support for the HPC emulator devices is available from Data I/O on their Unisite models.

For more details on the third party support tools for NSC's microcontroller products, please contact the third party office in your area or the National Semiconductor sales office.

Dial-A-Helper On-Line Applications Support

Dial-A-Helper lets you communicate directly with the Microcontroller Applications Engineers at National.

Using standard computer communications software, you can dial into the automated Dial-A-Helper Information System 24 hours a day.

You can leave messages on the electronic bulletin board for the Applications Engineers, then retrieve their responses.

You can select and then download specific applications data.

Dial-A-Helper

Voice:	(408) 721-5582 (8 a.m.–5 p.m. PST)
Modem:	(408) 739-1162 (24 Hrs./day)
Setup:	Baud rate 300 bps or 1200 bps 8 bits, no parity, 1 stop

Dedicated Applications Engineers

We've assembled a dedicated team of highly trained, highly experienced engineering professionals to help you implement your solution quickly, effectively, efficiently and to ensure that it's the best solution for your specific application.

At National, we believe that the best technology is also the most usable technology. That's why our microcontrollers provide such practical solutions to such real design problems. And that's why our microcontroller development support includes such comprehensive tools and such powerful engineering resources.

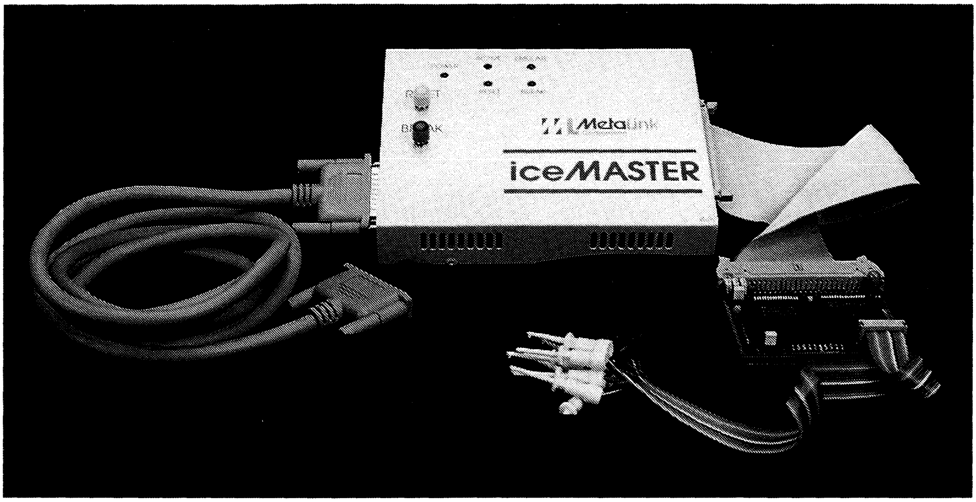
No one makes more microcontrollers than National and no one does more to help you put those microcontrollers to work.

NOTES

National Semiconductor

COP8 Development System

iceMASTER™ COP8/400



TL/DD/11386-1

Product Overview

The iceMASTER COP8/400 in-circuit emulator manufactured by MetaLink Corporation and marketed by National Semiconductor provides complete real-time emulation support for all members of the COP8 family. This stand-alone system is designed to provide maximum flexibility to the user through the interchangeable probe cards to support the various configurations and packages of the COP8 family. The interchangeable probe card connects to a common base unit which is linked with an IBM® PC® host through the RS-232 serial communications channel. Full assembly-level symbolic debugging is supported.

MetaLink COP8 iceMASTER Feature List

- Flexible, easy-to-use windowed interface, with window size, position, contents and color being completely configurable.
- Fast serial download with 115.2 kBaud using a standard PC COMM port.
- Context-sensitive hypertext on-line help system.
- Commands can be accessed via pull-down menus and/or redefinable hot keys.
- Dynamically annotated code feature displays contents of all accessed (read and write) memory locations and registers, as well as flow-of-control direction change markers next to each instruction executed when single-stepping.
- 4k-frame trace buffer captures data in real-time. Trace information consists of address and data bus values and user-selectable probe clips (external event lines). Trace buffer data can be viewed as raw hex or disassembled instructions. The probe clip bit values can be displayed in binary, hex or digital waveform formats.
- Performance analyzer with a resolution better than 6 μs. Up to 15 independent memory areas based on code address, line number or label ranges can be defined. Analysis results can be viewed in bar graph format or as actual frequency count.
- 32k of break and trace triggers. Triggers can be enabled, disabled, set or cleared. They can be simple triggers based on code or address ranges or complex triggers based on code address, direct address, opcode value, opcode class or immediate operand. Complex breakpoints can be ANDed and ORed together.
- Memory operations for program memory include single-line assembler, disassembler, view, change, and write to file.
- Memory operations for data memory include fill, move, change, compare, dump to file and examine, modify for registers and program variables.
- Complete status of debugger including breakpoints, trace triggers, etc. can be saved to file for later resumption of debugging process.

Specifications

EMULATOR SYSTEM REQUIREMENTS

Basic Emulator System Model 400
Interchangeable Probe Card
+5V, 1.5A Power Source

MODELS

400 Emulator with:
- 4k Trace Buffer
- 2 Performance Analyzers
- Full WATCHDOG™ Timer Support

FILE FORMATS

Intel HEX and National Semiconductor

MACRO

Repetitive Routines
User-created and callable

MEMORY OPERATIONS

Program Memory:
- Single Line Assembler
- Disassemble
- Disassemble to File
- View/Change
- Mapping

Data/Code Memory:
- Dump
- Dump to File
- Fill
- Move
- Change
- Compare

Registers:
- Examine/Modify

Program Variables:
- Examine/Modify

OPERATING CHARACTERISTICS

Electrically Transparent
Operationally Transparent

USER INTERFACE

Keyboard or Mouse Control
Pull-Down and Pop-Up Menus
Main Screen Windows:
- Registers/SFRs/PSW Bits
- Stack
- Up to 5 Internal Data Memory
- Up to 5 Code Memory
- Source Program
- Watch
- System Status

User Window Controls:
- Selectable (On/Off)
- Movable
- Resizable
- Scrollable
- Color Selection
- Highlighting

Function/Hot Key Access:
- User-Assignable

EMULATION CONTROLS

Reset from Emulator
Reset from Target
Reset Processor
Go
Go From
Go Until
Slow Motion
Step
Step Line
Step Over
Step To
Repetition Counter

PERFORMANCE ANALYZER

Real-Time Program Profiling
5.4 μs Sampling Period
7 Year Duration
Display Options:
- Bar Graph
- Frequency Count

Display Modes:
- Raw
- Symbolic

Up to 15 Bin Capacity:
- Multiple Ranges per Bin

User-Controlled Bin Setup:
- By Address
- By Symbol
- Automatic

TRACE

Trace Triggers:
- Start
- Center
- End
- Variable

4k-Frame Trace Buffer

Specifications (Continued)

Trace Contents:
- Address
- Data
- External Clips

Trace Display Modes:
- Raw Hex
- Symbolic
- Binary (Clips)
- Digital Waveform (Clips)

Trace Buffer Operations:
- Write Buffer to File
- Search Trace Buffer

HELP

On-Line
Context Sensitive
Hypertext/Hyperlinked

SOURCE/SYMBOL SUPPORT

Source-Level Debug

ELECTRICAL SPECIFICATIONS

Input Power (Maximum):
- 1.5A @ +5 V_{DC} ±5%

MECHANICAL SPECIFICATIONS

Emulator Dimensions:
- 1.0″ x 7.0″ x 5.5″
  (2.5cm x 17.8cm x 14cm)

Probe Card Cable Length:
- 14.0″ (35.6cm)

Emulator Weight:
- 2.0 lbs. (0.9 kg)

WARRANTY

One (1) year limited warranty, parts and labor, for registered users.

iceMASTER COP8	
Emulation Memory	
Program	32k
Real Time:	DC - 10 MHz
Breakpoints:	32k
Trace On:	32k
Trace Off:	32k
Pass Count	32K
Trigger Conditions:	
PC Address and Range	X
Opcode Value	X
Opcode Class	X
SFRs/Registers	X
Direct Byte Address and Range	X
Direct Bit Address and Range	X
Immediate Operand Value	X
Read/Write to Bit Address	X
Register Address Modes	X
Read/Write to Register Address	X
Logical AND/OR of Any of the Above	X
External Input	X
Operating Modes	
Single-Chip/ROM	X

HOST SYSTEM REQUIREMENTS

IBM PC-XT/PC-AT or compatibles, 640 kbytes of Memory with 5.25″ Double Density Floppy Drive.

RS-232 Serial Port

MS-DOS or PC-DOS Operating System

Ordering Information

Emulator Ordering Information

Part Number	Description
IM-COP8/400	MetaLink base unit in-circuit emulator for all COP8 devices, symbolic debugger software and RS-232 serial interface cable
MHW-PS3	Power Supply: 110V/60 Hz
MHW-PS4	Power Supply: 220V/50 Hz

Probe Card Ordering Information

Device	Package	Voltage Range	Probe Card
COP880C, 8780C	44 PLCC	4.5V–5.5V	MHW-880C44D5PC
		2.5V–6.0V	MHW-880C44DWPC
COP880C, 8780C	40 DIP	4.5V–5.5V	MHW-880C40D5PC
		2.5V–6.0V	MHW-880C40DWPC
COP881C, 8781C, 840C, 820C	28 DIP	4.5V–5.5V	MHW-880C28D5PC
		2.5V–6.0V	MHW-880C28DWPC
COP842C, 822C, 8742C	20 DIP	4.5V–5.5V	MHW-880C20D5PC
		2.5V–6.0V	MHW-880C20DWPC
COP820CJ	28 DIP	4.5V–5.5V	MHW-820CJ28D5PC
		2.3V–6.0V	MHW-820CJ28DWPC
COP822CJ	20 DIP	4.5V–5.5V	MHW-820CJ20D5PC
		2.3V–6.0V	MHW-820CJ20DWPC
COP8640C, 8620C	28 DIP	4.5V–5.5V	MHW-8640C28D5PC
		2.5V–6.0V	MHW-8640C28DWPC
COP8642C, 8622C	20 DIP	4.5V–5.5V	MHW-8640C20D5PC
		2.5V–6.0V	MHW-8640C20DWPC
COP888CF	44 PLCC	4.5V–5.5V	MHW-888CF44D5PC
		2.5V–6.0V	MHW-888CF44DWPC
COP888CF	40 DIP	4.5V–5.5V	MHW-888CF40D5PC
		2.5V–6.0V	MHW-888CF40DWPC
COP884CF	28 DIP	4.5V–5.5V	MHW-884CF28D5PC
		2.5V–6.0V	MHW-884CF28DWPC
COP888CL	44 PLCC	4.5V–5.5V	MHW-888CL44D5PC
		2.5V–6.0V	MHW-888CL44DWPC
	40 DIP	4.5V–5.5V	MHW-888CL40D5PC
		2.5V–6.0V	MHW-888CL40DWPC

Ordering Information (Continued)

Probe Card Ordering Information (Continued)

Device	Package	Voltage Range	Probe Card
COP884CL	28 DIP	4.5V–5.5V	MHW-884CL28D5PC
		2.5V–6.0V	MHW-884CL28DWPC
COP888CG, 888CS	44 PLCC	4.5V–5.5V	MHW-888CG44D5PC
		2.5V–6.0V	MHW-888CG44DWPC
	40 DIP	4.5V–5.5V	MHW-888CG40D5PC
		2.5V–6.0V	MHW-888CG40DWPC
COP884CG, 884CS	28 DIP	4.5V–5.5V	MHW-884CG28D5PC
		2.5V–6.0V	MHW-884CG28DWPC

LANGUAGE TOOLS

Product	NSID	Description	Includes	Number
COP800 Family	MOLE-COP8-IBM	Assembly Language Software for the COP800 Family	COP800 System Software User's Manual	424410527

Single-Chip Emulator

Form, Fit, Function Emulator Ordering Information

Part Number	Emulator		Clock Option	Description
	Part Number	Package		
COP880C	COP880CMHEL-X	44 LDCC	X = 1: Crystal X = 2: External X = 3: R/C	Multi-Chip Module, UV Erasable
	COP8780CV	44 PLCC	Programmable	One-Time Programmable
	COP8780CEL	44 LDCC		UV Erasable
	COP880CMHD-X	40 DIP	X = 1: Crystal X = 2: External X = 3: R/C	Multi-Chip Module, UV Erasable
	COP8780CN		Programmable	One-Time Programmable
	COP8780CJ			UV Erasable
COP881C, COP840C, COP820C	COP881CMHD-X	28 DIP	X = 1: Crystal X = 2: External X = 3: R/C	Multi-Chip Module, UV Erasable
	COP8780CN		Programmable	One-Time Programmable
	COP8780CJ			UV Erasable

Single-Chip Emulator (Continued)

Form, Fit, Function Emulator Ordering Information (Continued)

Part Number	Emulator		Clock Option	Description
	Part Number	Package		
COP881C, COP840C, COP820C	COP881CMHEA-X	28 LCC (Shoebox)	X = 1: Crystal X = 2: External X = 3: R/C	Multi-Chip Module, Same Footprint as 28 SO, UV Erasable
	COP8781CWN	28 SO	Programmable	One-Time Programmable
	COP8781CMC			UV Erasable
COP842C	COP842CMHD-X	20 DIP	X = 1: Crystal X = 2: External X = 3: R/C	Multi-Chip Module, UV Erasable
COP822C	COP822CMHD-X			
COP842C, COP822C	COP8742CN	20 DIP	Programmable	One-Time Programmable
	COP8742CJ			UV Erasable
	COP8742CWM	20 SO	Programmable	One-Time Programmable
	COP8742CMC			UV Erasable
COP8640C, COP8620C	COP8640CMHD-X	28 DIP	X = 1: Crystal X = 2: External X = 3: R/C	Multi-Chip Module, UV Erasable
	COP8640CMHEA-X	28 LCC (Shoebox)		Multi-Chip Module, Same Footprint as 28 SO, UV Erasable
COP8642C, COP8622C	COP8642CMHD-X	20 DIP	X = 1: Crystal X = 2: External X = 3: R/C	Multi-Chip Module, UV Erasable
COP820CJ	COP820CJMHD-X	28 DIP	X = 1: Crystal X = 2: External X = 3: R/C	Multi-Chip Module, UV Erasable
	COP820CJMHEA-X	28 LCC (Shoebox)		Multi-Chip Module, Same Footprint as 28 SO, UV Erasable
COP822CJ	COP822CJMHD-X	20 DIP	X = 1: Crystal X = 2: External X = 3: R/C	Multi-Chip Module, UV Erasable
COP888CL	COP888CLMHEL-X	44 LDCC	X = 1: Crystal X = 3: R/C	Multi-Chip Module, UV Erasable
	COP888CLMHD-X	40 DIP		
COP884CL	COP884CLMHD-X	28 DIP	X = 1: Crystal X = 3: R/C	Multi-Chip Module, UV Erasable
	COP884CLMHEA-X	28 LCC (Shoebox)		Multi-Chip Module, Same Footprint as 28 SO, UV Erasable
COP888CF	COP888CFMHEL-X	44 LDCC	X = 1: Crystal X = 3: R/C	Multi-Chip Module, UV Erasable
	COP888CFMHD-X	40 DIP		
COP884CF	COP884CFMHD-X	28 DIP	X = 1: Crystal X = 3: R/C	Multi-Chip Module, UV Erasable
	COP884CFMHEA-X	28 LCC (Shoebox)		Multi-Chip Module, Same Footprint as 28 SO, UV Erasable
COP888CG	COP888CGMHEL-X	44 LDCC	X = 1: Crystal X = 3: R/C	Multi-Chip Module, UV Erasable
	COP888CGMHD-X	40 DIP		

Single-Chip Emulator (Continued)

Form, Fit, Function Emulator Ordering Information (Continued)

Part Number	Emulator		Clock Option	Description
	Part Number	Package		
COP884CG	COP884CGMHD-X	28 DIP	X = 1: Crystal X = 3: R/C	Multi-Chip Module, UV Erasable
	COP884CGMHEA-X	28 LCC (Shoebox)		Multi-Chip Module, Same Footprint as 28 SO, UV Erasable
COP888EG	COP888EGMHEL-X	44 LDCC	X = 1: Crystal X = 3: R/C	Multi-Chip Module, UV Erasable
	COP888EGMHD-X	40 DIP		
COP884EG	COP884EGMHD-X	28 DIP	X = 1: Crystal X = 3: R/C	Multi-Chip Module, UV Erasable
	COP884EGMHEA-X	28 LCC (Shoebox)		Multi-Chip Module, Same Footprint as 28 SO, UV Erasable
COP888CS	COP888CSMHEL-X	44 LDCC	X = 1: Crystal X = 3: R/C	Multi-Chip Module, UV Erasable
	COP888CSMHD-X	40 DIP		
COP884CS	COP884CSMHD-X	28 DIP	X = 1: Crystal X = 3: R/C	Multi-Chip Module, UV Erasable
	COP884CSMHEA-X	28 LCC (Shoebox)		Multi-Chip Module, Same Footprint as 28 SO, UV Erasable

Programming Support

The main board and scrambler boards can be purchased separately or as a set. The table below lists the product identification numbers of the Duplicator Board products.

Product ID	Description
COP8-PRGM-28D	COP8 Duplicator Board for 28-pin DIP Multi-Chip Module (MCM) and for use with Scrambler Boards
COP8-SCRM-DIP	MCM-Scrambler Board for 20-pin DIP and 40-pin DIP
COP8-SCRM-PCC	MCM-Scrambler Board for 44-pin PLCC/LDCC
COP8-PRGM-DIP	COP8 Duplicator Board with DIP MCM Scrambler Board (PRGM-28D and SCRM-DIP)
COP8-PRGM-PCC	COP8 Duplicator Board with PLCC/LDCC MCM Scrambler Board (PRGM-28D and SCRM-PCC)
COP8-SCRM-87A	Scrambler Board for COP8780 devices, 28-pin DIP, 40-pin DIP, 28-pin SO
COP8-SCRM-87B	Scrambler Board for COP8780 devices, 20-pin DIP, 20-pin SO, 44-pin PLCC/LDCC
COP8-PRGM-87A	COP8 Duplicator Board with COP8-SCRM-87A Scrambler Board
COP8-PRGM-87B	COP8 Duplicator Board with COP8-SCRM-87B Scrambler Board
COP8-PRGM-SBX	COP8 Duplicator Board with COP8-SCRM-SBX Scrambler Board
COP8-SCRM-SBX	Scrambler Board for 28-pin LCC MCM Package (Shoebox)

Programming Support (Continued)

The COP device pin/package types, COP device numbers, and the Duplicator Board product identification number for each package type are listed in the table below.

Package Type	COP Devices	COP Duplicator Product ID #
20-Pin DIP	842CMH, 8642CMH, 822CJMH	COP8-PRGM-DIP
28-Pin DIP	884CLMH/CFMH/CGMH/EGMH/CSMH, 881CMH, 8640CMH, 820CJMH	COP8-PRGM-28D
28-Pin LCC (Shoebox)	881CMH, 820CJMH, 8640CMH, 884CFMH/CLMH/CGMH/EGMH/CSMH	COP8-PRGM-SBX
40-Pin DIP	888CLMH/CFMH/CGMH/EGMH/CSMH, 880CMH, 943CMH	COP8-PRGM-DIP
44-Pin PLCC/LDCC	888CLMH/CFMH/CGMH/EGMH/CSMH, 880CMH	COP8-PRGM-PCC
28-Pin DIP or SO, 40-Pin DIP	8780C, 8781C	COP8-PRGM-87A
20-Pin DIP or SO, 44-Pin PLCC/LDCC	8780C, 8742C	COP8-PRGM-87B

COP800 DESIGNER'S TOOL KIT

TL/DD/11386-2

General Description

The COP800 Designer's Tool Kit is available today to help you evaluate National's COP800 microcontroller family. The Kit contains programmer's manuals, device data sheets, application notes, and pocket reference guides for immediate in-circuit evaluation. The Designer Kit includes an assembler and simulator, which allow you to write, test and debug COP800 code before your target system is finalized.

The simulator can handle script files that simulate hardware inputs and interrupts to the device being simulated. Any simulator command and comments may be included in a script file. The simulator also supports an additional command called WAIT, used to simulate machine cycles to delay before continuing with the script file.

A capture file feature enables you to record current cycle count and changes to an output port which are caused by the program under test. When used in combination with script files, this feature provides powerful software testing and debug capability.

Features

- Software simulator
- Assembler
- Programmer's manuals
- Device data sheets
- Application notes
- Assembler manual
- Tool kit user's guide
- Pocket reference guides
- COP8 SIM user's guide

Features (Continued)

Simulator Commands

@RAM [ramadd]	Causes a break in execution to occur when a write to the specified RAM location is attempted.
ASM [add]	Assembles directly to ROM at specified address or starting at last address used by command.
BR [add]	Set breakpoint at the indicated ROM address.
CAPTURE fname	Saves all hardware outputs in the file specified.
CAPTUREOFF	Stops capture and closes capture file.
CY n	Sets cycle counter.
DASM [add]	Disassembles memory to screen starting at specified address or last location disassembled.
EVAL n [op] [n]	Evaluates input in decimal, hex, and binary. Can do simple calculations where op may be +, −, /,or *.
GO [add] [add]	Sets breakpoint at second address. Go from first address.
GOTIL add	Go from the current PC until the PC = add.

LISTON	Turns on screen listing during stepping.
LISTOFF	Turns off screen listing.
LOAD filename	Loads Intel hex format file into simulator.
PRINTON	Sends all debug output to printer.
PRINTOFF	Stops sending debug output to printer.
RAM add [n]	Sets RAM location at indicated address to value specified.
REG	Shows register status in debug window.
RESET	Simulates a hardware reset.
RESTORE fname	Restores simulator state from a file created with the SAVE command.
ROM add [n]	Sets ROM location at indicated address to value specified.
SAVE filename	Saves the simulator state in the specified file.
SCRIPT fname	Executes a script file.
STEP [n]	Single step execution of n instructions.
STEPTIL add	Single step until the PC = add.
QUIT, EXIT	Return to DOS.

Ordering Information

NSID	Description	Includes:
COP8-TOOL-KIT	COP800 Designer's Tool Kit	Software Simulator Assembler Programmer's Manual Assembler Manual Tool Kit User's Guide

Section 5

Appendices/ Physical Dimensions

Section 5 Contents

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.

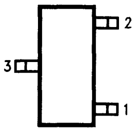
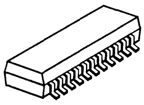
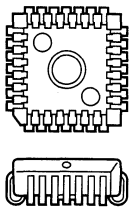
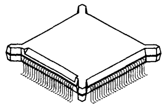
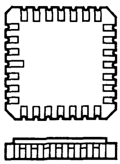
Production processes now permit both surface mount and insertion mount components to be assembled on the same PC board.

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.

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	LM392M
LF351M	LM393M
LF451CM	LM741CM
LF353M	LM1458M
LF355M	LM2901M
LF356M	LM2902M
LF357M	LM2903M
LF444CWM	LM2904M
LM10CWM	LM2924M
LM10CLWM	LM3403M
LM308M	LM4250M
LM308AM	LM324M
LM310M	LM339M
LM311M	LM365WM
LM318M	LM607CM
LM319M	LMC669BCWM
LM324M	LMC669CCWM
LM339M	LF441CM
LM346M	
LM348M	
LM358M	
LM359M	

Regulators and References

Part Number	Part Number
LM317LM	LM2931M-5.0
LF3334M	LM3524M
LM336M-2.5	LM78L05ACM
LF336BM-2.5	LM78L12ACM
LM336M-5.0	LM78L15ACM
LM336BM-5.0	LM79L05ACM
LM337LM	LM79L12ACM
LM385M	LM79L15ACM
LM385M-1.2	LP2951ACM
LM385BM-1.2	LP2951CM
LM385M-2.5	
LM385BM-2.5	
LM723CM	
LM2931CM	

Data Acquisition Circuits

Part Number	Part Number
ADC0802LCV	ADC1025BCV
ADC0802LCWM	ADC1025CCV
ADC0804LCV	DAC0800LCM
ADC0804LCWM	DAC0801LCM
ADC0808CCV	DAC0802LCM
ADC0809CCV	DAC0806LCM
ADC0811BCV	DAC0807LCM
ADC0811CCV	DAC0808LCM
ADC0819BCV	DAC0830LCWM
ADC0819CCV	DAC0830LCV
ADC0820BCV	DAC0832LCWM
ADC0820CCV	DAC0832LCV
ADC0838BCV	
ADC0838CCV	
ADC0841BCV	
ADC0841CCV	
ADC0848BCV	
ADC0848CCV	
ADC1005BCV	
ADC1005CCV	

Industrial Functions

Part Number	Part Number
AH5012CM	LM13600M
LF13331M	LM13700M
LF13509M	LMC555CM
LF13333M	LM567CM
LM555CM	MF4CWM-50
LM556CM	MF4CWM-100
LM567CM	MF6CWM-50
LM1496M	MF10CCWM
LM2917M	MF6CWM-100
LM3046M	MF5CWM
LM3086M	
LM3146M	

Commercial and Automotive

Part Number	Part Number
LM386M-1	LM1837M
LM592M	LM1851M
LM831M	LM1863M
LM832M	LM1865M
LM833M	LM1870M
LM837M	LM1894M
LM838M	LM1964V
LM1131CM	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

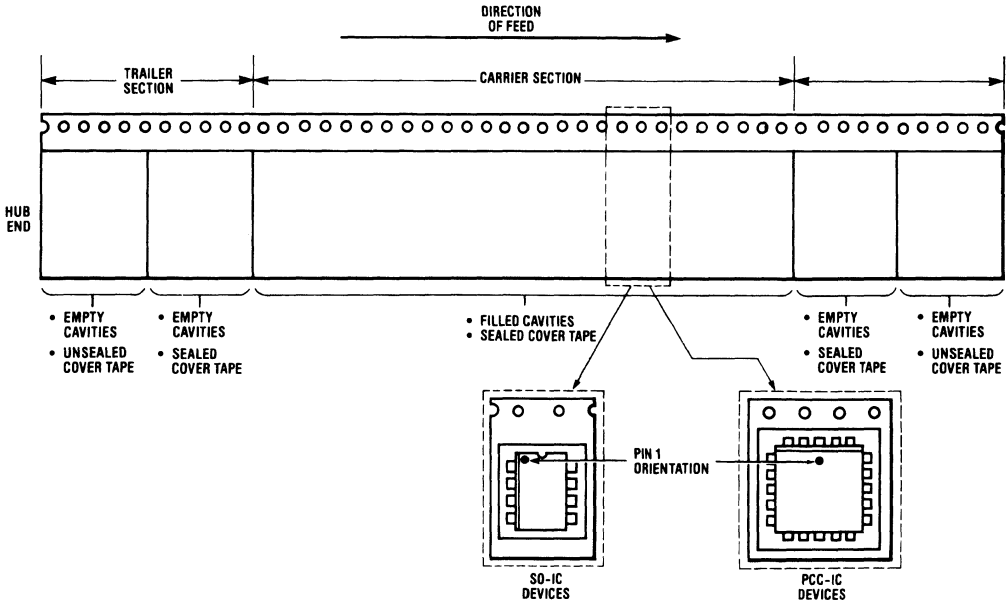
→ Direction of Feed

	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

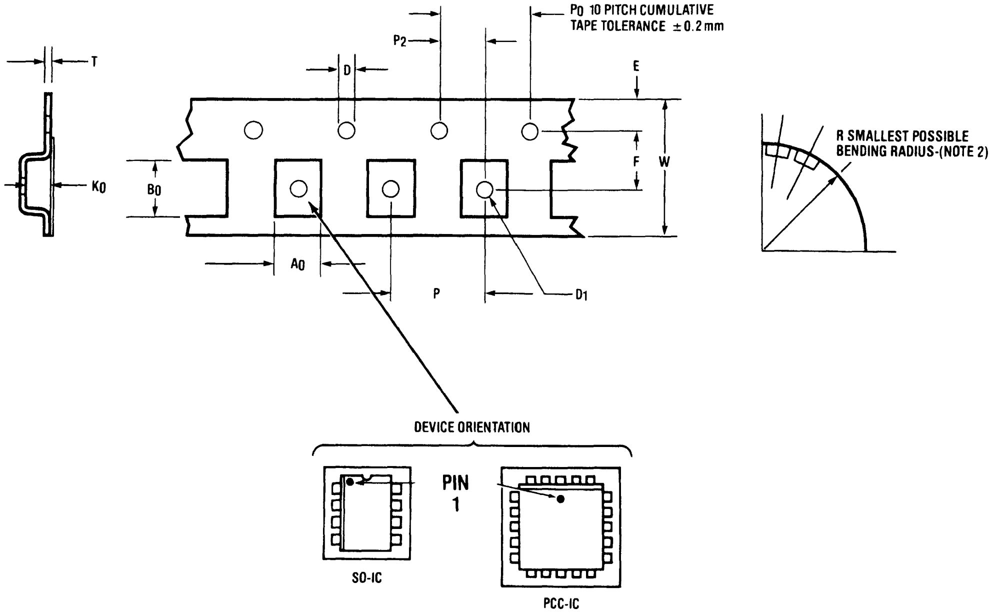


TL/DD/11325-7

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.

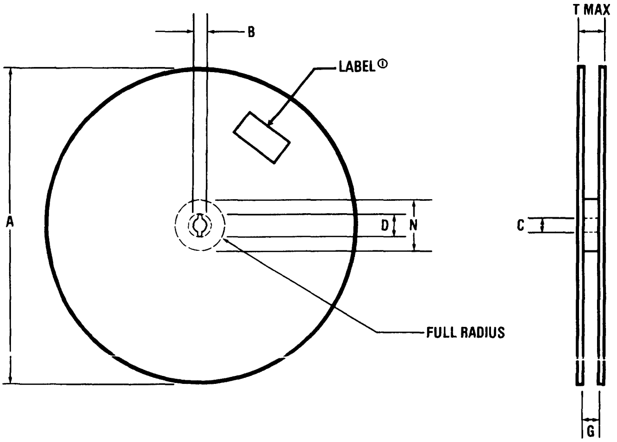
Note 3: Cavity tape material shall be PVC conductive (less than 10^5 Ohms/Sq).

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

STAR™* 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: $\frac{\textbf{Inches}}{\textbf{Millimeters}}$

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

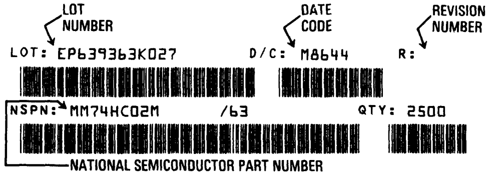
Date Code

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 bimetallic 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.

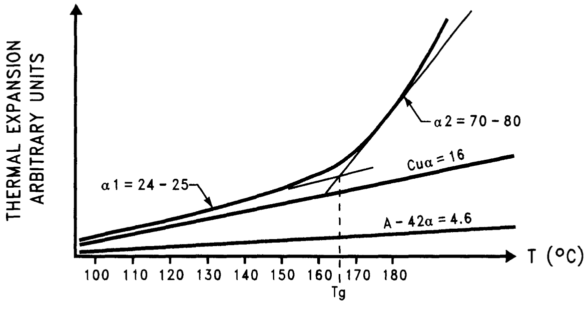
VAPOR PHASE/IR REFLOW SOLDERING

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

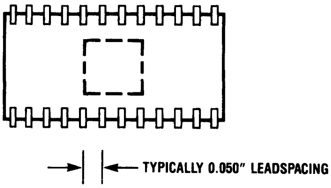
Based on the results presented, it is noted that surface-mounted components are as reliable as standard molded DIP packages. Whereas DIPs were never processed by being totally immersed in a hot solder wave during printed circuit board soldering, surface mounted components such as SOICs (Small Outline) 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.

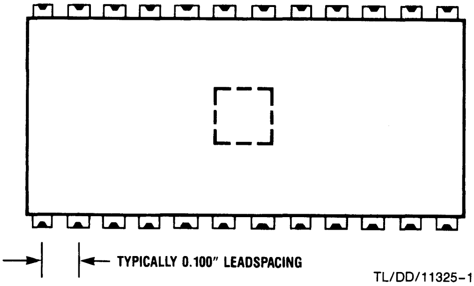
COMPONENT SIZE COMPARISON

S.O. Package



TL/DD/11325-12

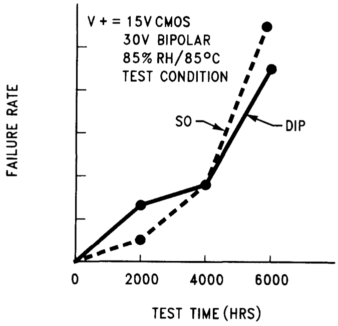
Standard DIP Package



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.

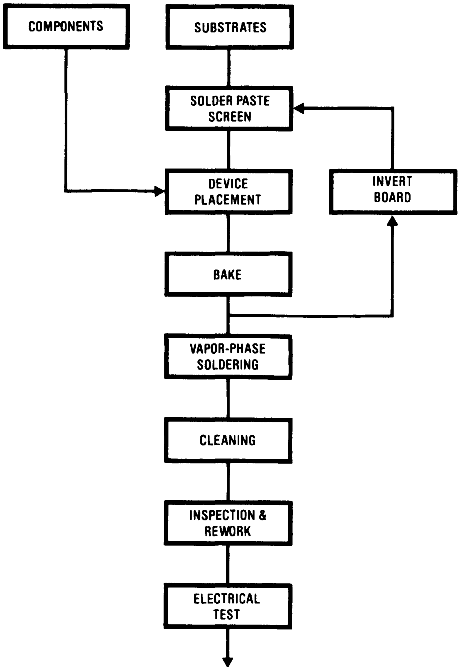
Usual variations encountered by users of SO packages are:

- 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.

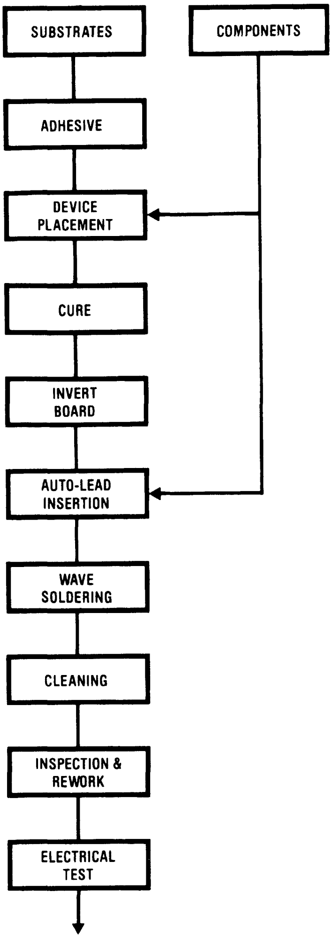
PRODUCTION FLOW

Basic Surface-Mount Production Flow



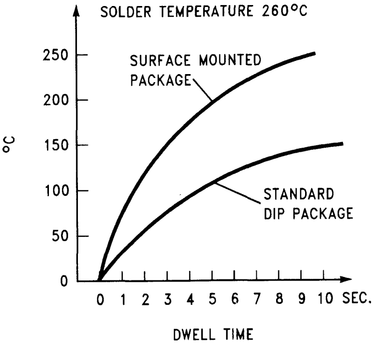
TL/DD/11325-15

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).

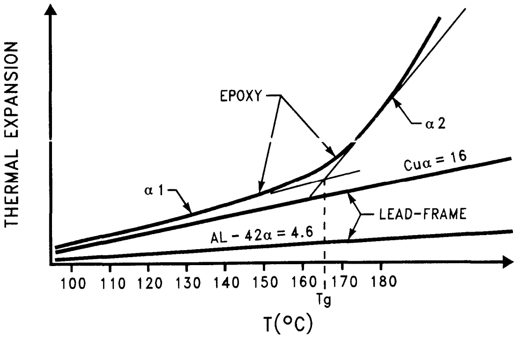


TL/DD/11325-17

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

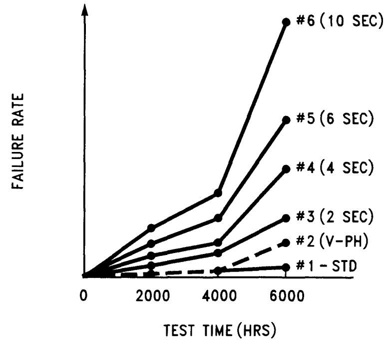
FIGURE C

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

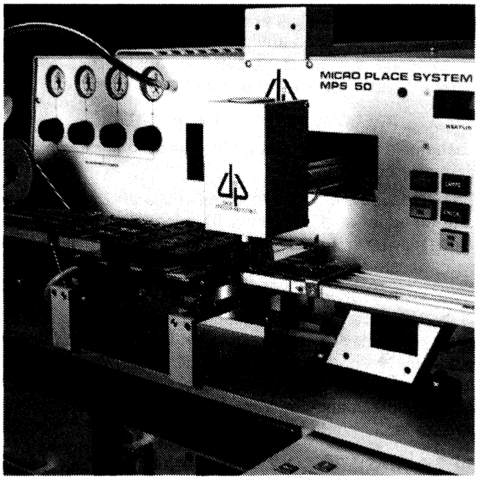
(b) Sequential placement
  - Either a X-Y moving table system or a θ, X-Y moving pickup system used
  - Individual components picked and placed onto boards

(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:

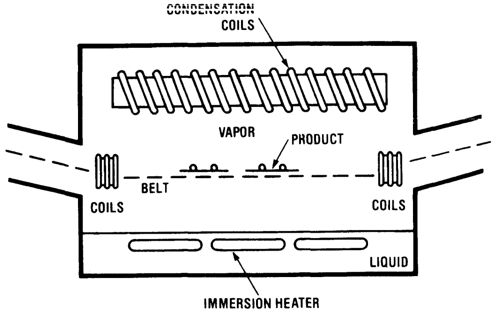
- FC-70, 215°C vapor (most applications) or FX-38
- FC-71, 253°C vapor (low lead or tin-plate)

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



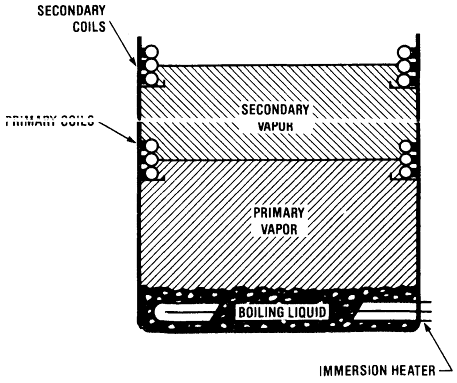
TL/DD/11325-21

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

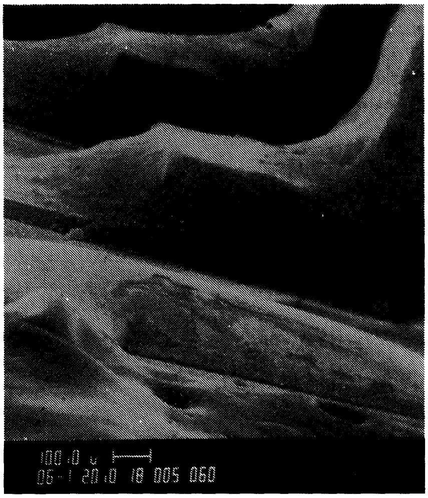
TL/DD/11325-22

Batch-Fed Production Vapor-Phase Soldering Unit



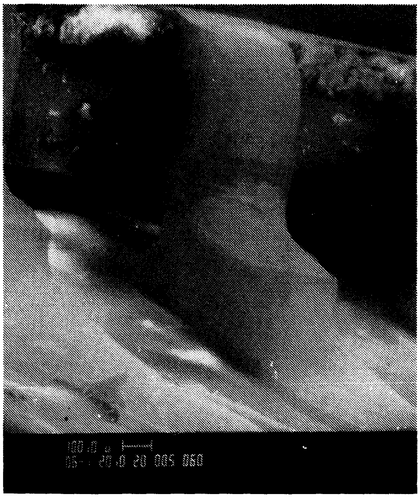
TL/DD/11325-23

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 ⅛″, to avoid damage to screens and minimize distortion.

SOLDER PASTE

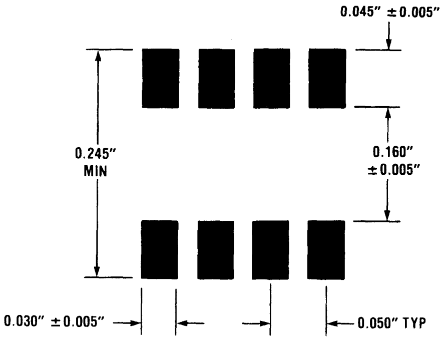
Selection of solder paste tends to be confusing, due to numerous formulations available from various manufacturers. In general, the following guidelines are sufficient to qualify a particular paste for production:

- 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.

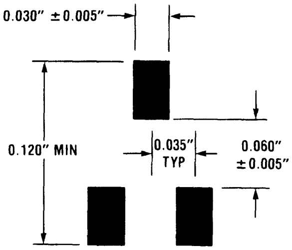
RECOMMENDED SOLDER PADS FOR SO PACKAGES

SO-8, SO-14, SO-16



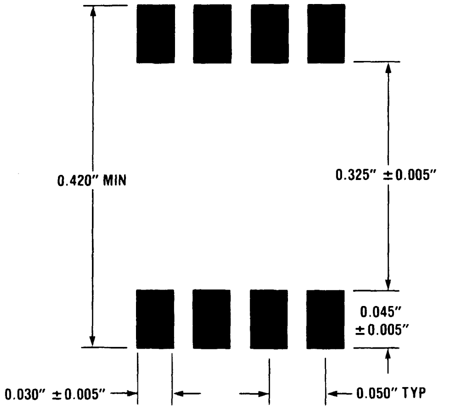
TL/DD/11325-26

SOT-23



TL/DD/11325-28

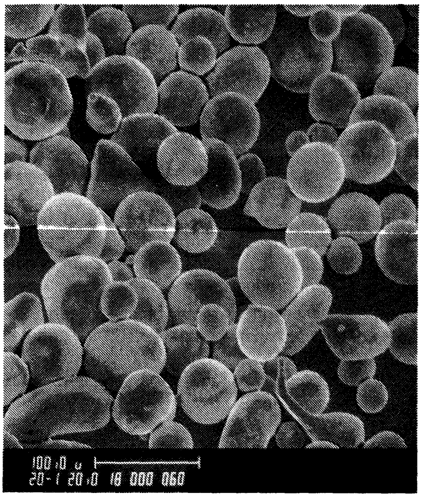
SO-16L, SO-20



TL/DD/11325-27

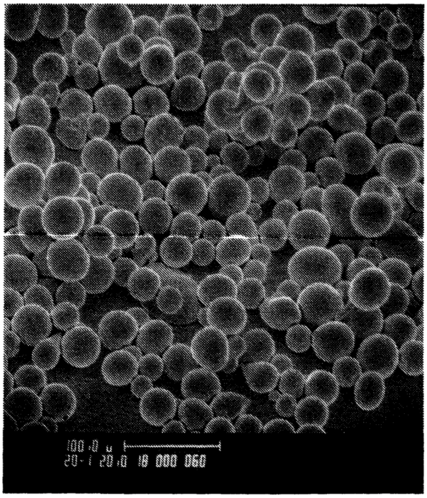
Comparison of Particle Size/Shape of Various Solder Pastes

200 × Alpha (62/36/2)



TL/DD/11325-29

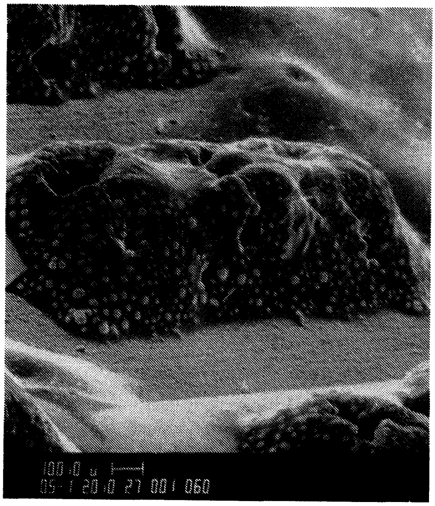
200 × Kester (63/37)



TL/DD/11325-30

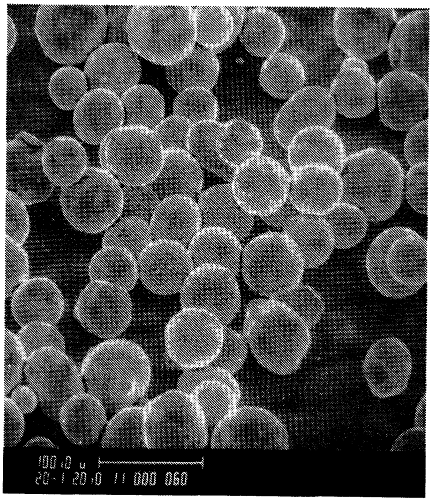
Comparison of Particle Size/Shape of Various Solder Pastes (Continued)

Solder Paste Screen on Pads



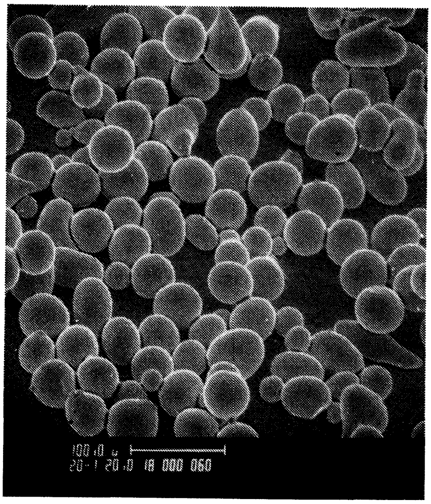
TL/DD/11325-31

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:

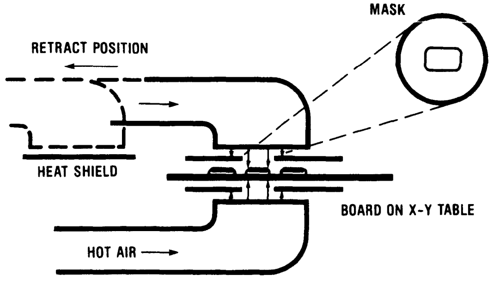
- Ion contamination, where ionic residue left on boards would cause corrosion to metallic components, affecting the performance of the board.
- Electro-migration, where ionic residue and moisture present on electrically-biased boards would cause dentritic growth between close spacing traces on the substrate, 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.

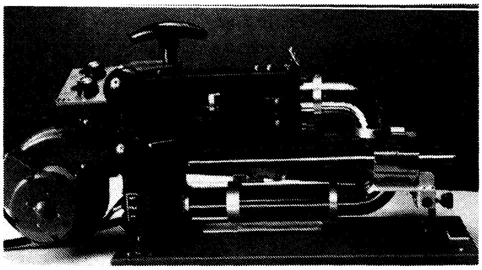
When rework is necessary in the field, specially-designed tweezers that thermally heat the component may be used to remove it from its site. The replacement can be fluxed at the lead tips or, if necessary, solder paste can be dispensed onto the pads using a varimeter. After being placed into position, 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.

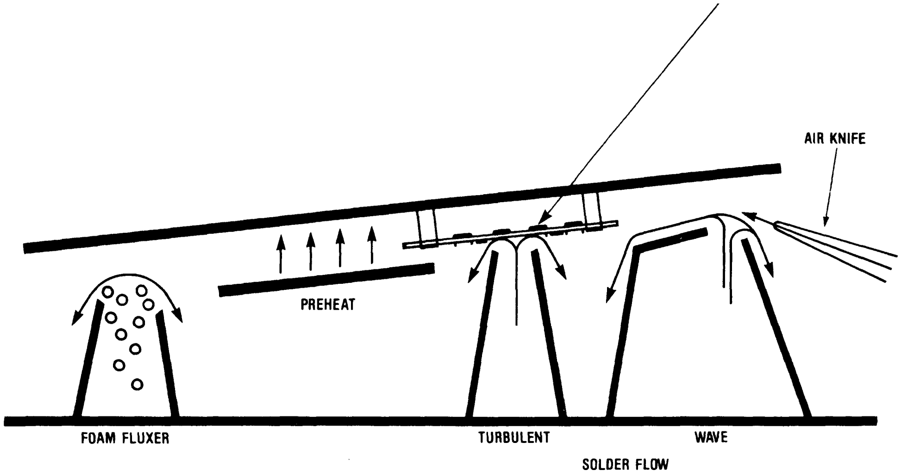
The controls required for wave soldering are:

- 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

(a) Same Side

(b) Opposite Sides



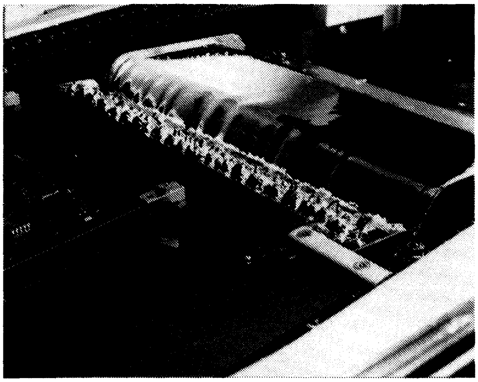
TL/DD/11325-36

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

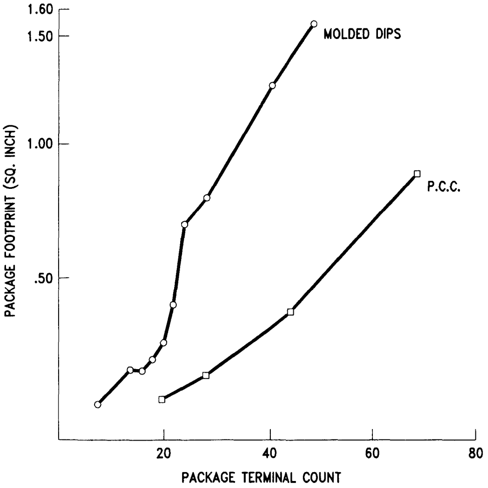
3) Reflowing of the solder or solder paste.

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

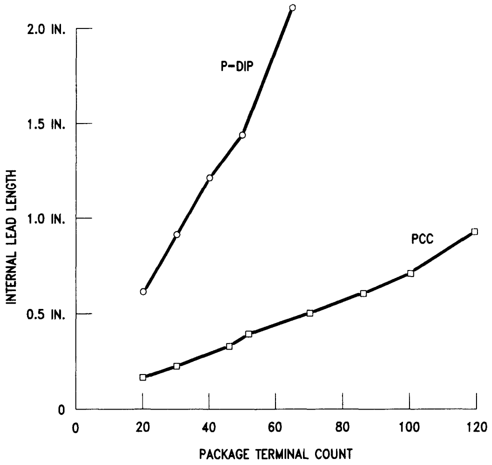
There are four principle advantages offered the user by switching from P-DIP to PLCC. These four advantages are outlined below as follows:

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.



TL/ZZ/0001-1

FIGURE 1. Footprint Area of PLCC vs. P-DIP

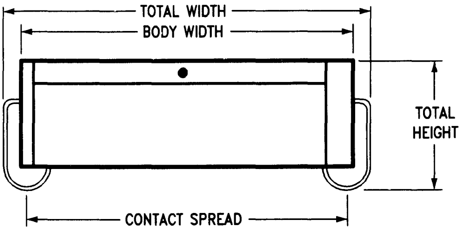


TL/ZZ/0001-2

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



TL/ZZ/0001-3

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

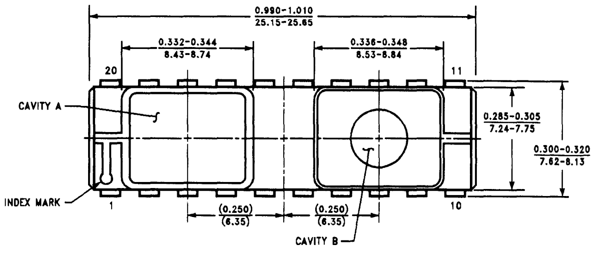
ADDITIONAL INFORMATION AND SERVICES

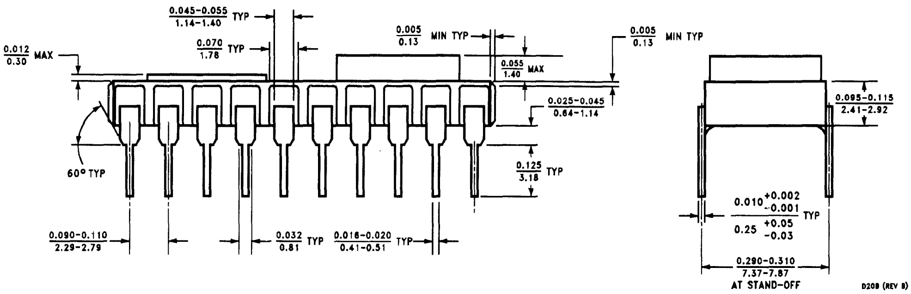
National Semiconductor offers additional Databooks which cover surface mount technology in much greater detail. We also have a surface mount laboratory to provide demonstrations and customer support, as well as technology development. Feel free to contact us about these additional resources.

National Semiconductor

20 Lead Ceramic Sidebrazed Dual-in-Line Package, Dual Cavity
NS Package Number D20B

All dimensions are in $\frac{\text{inches}}{\text{millimeters}}$

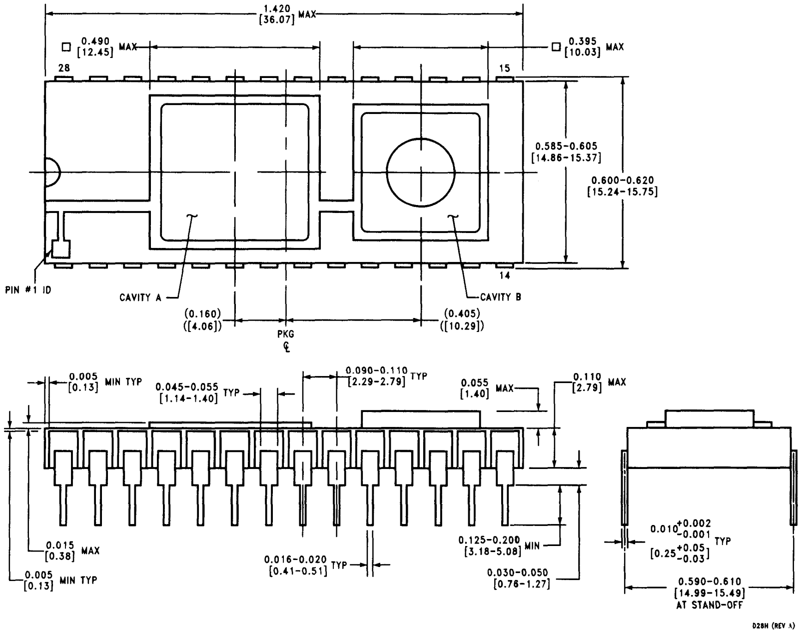




---

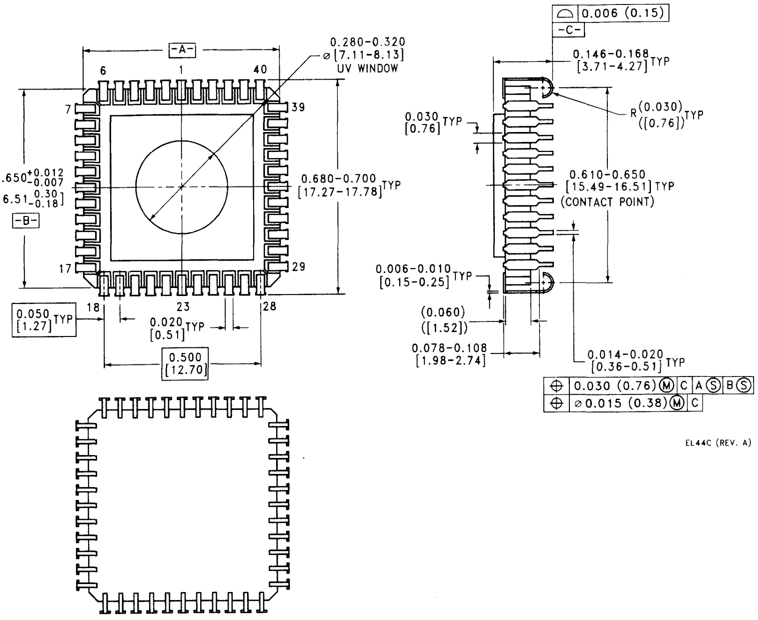
28 Lead Ceramic Sidebrazed Dual-in-Line Package, Dual Cavity
NS Package Number D28H

All dimensions are in inches [millimeters]



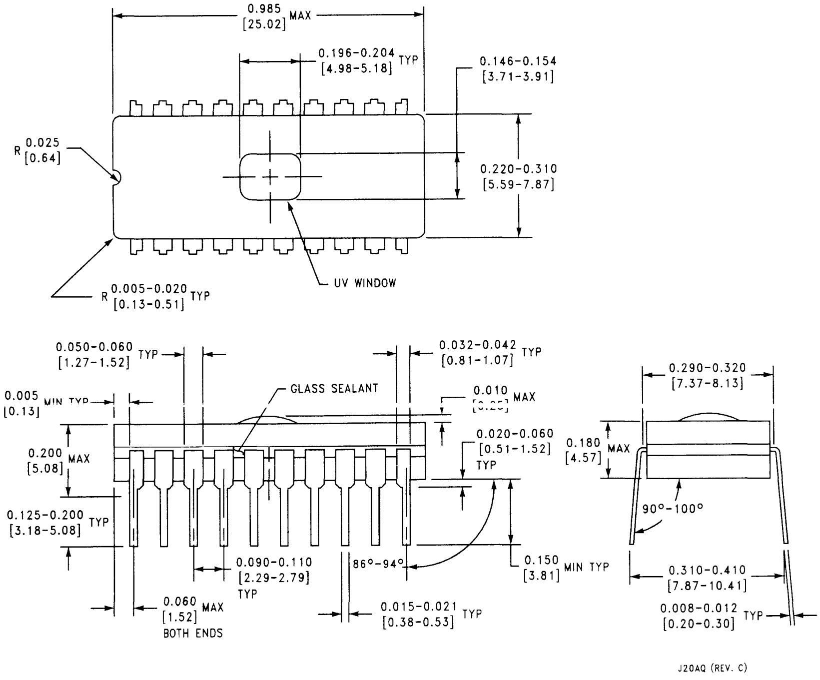
44 Lead Ceramic Quad J-Bend, EPROM
NS Package Number EL44C

All dimensions are in inches [millimeters]



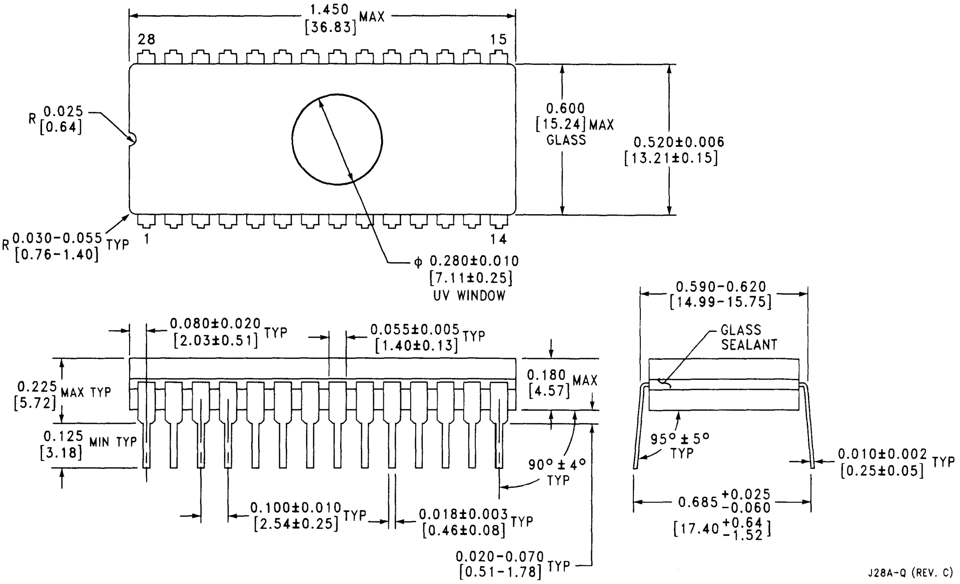
20 Lead Ceramic Dual-in-Line Package, EPROM
NS Package Number J20AQ

All dimensions are in inches [millimeters]



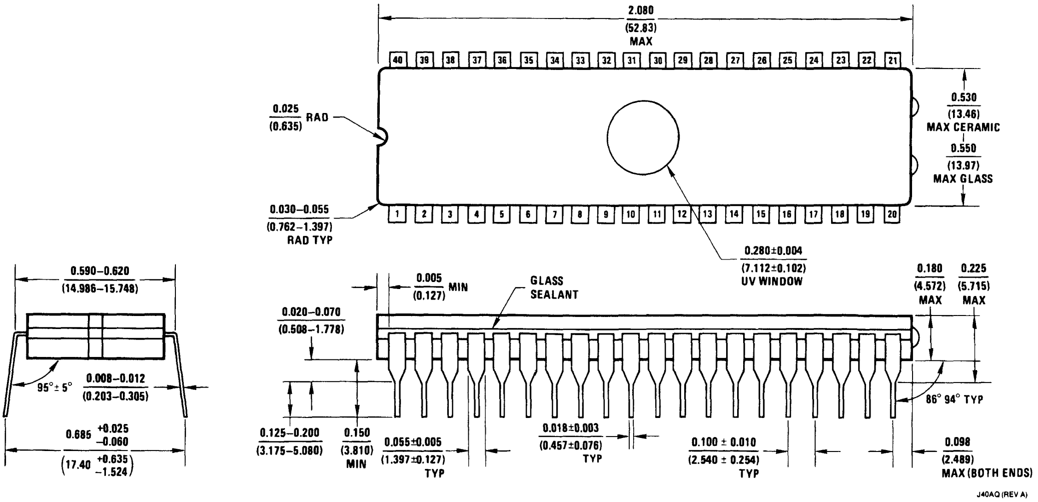
28 Lead Ceramic Dual-in-Line Package, EPROM
NS Package Number J28AQ

All dimensions are in inches [millimeters]



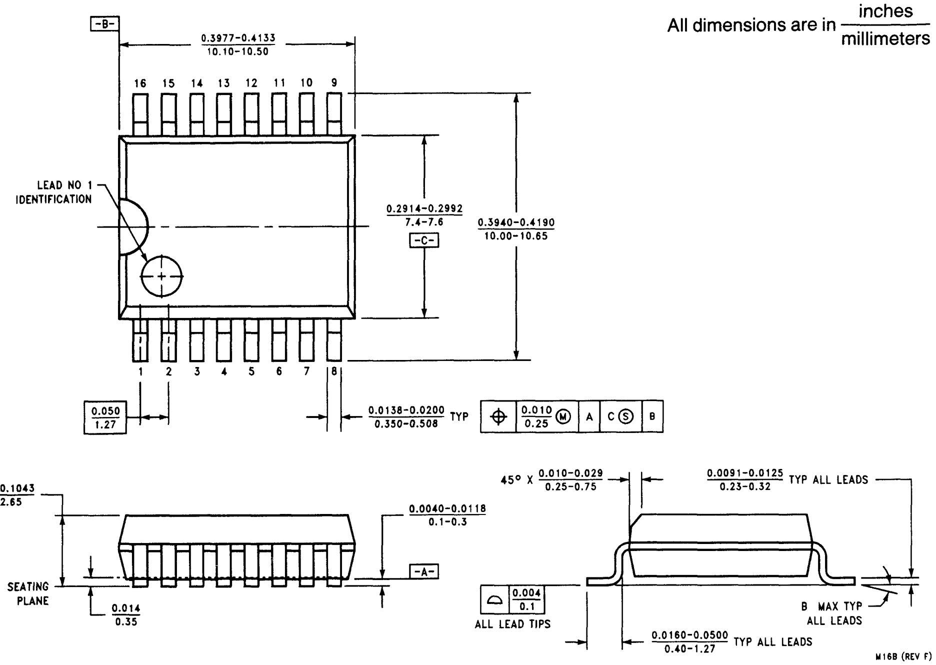
40 Lead Ceramic Dual-in-Line Package, EPROM
NS Package Number J40AQ

All dimensions are in inches (millimeters)



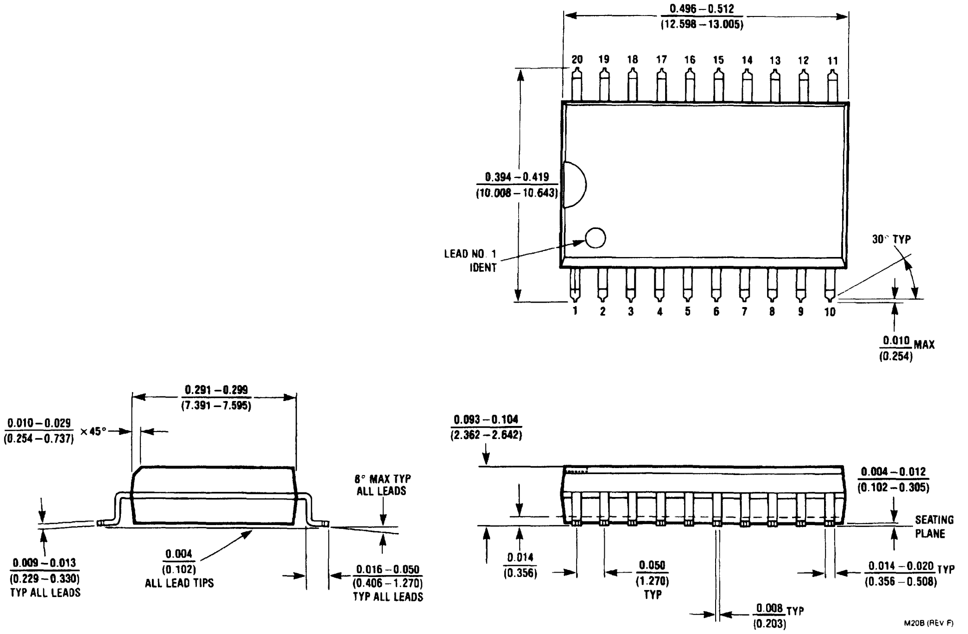
16 Lead (0.300″ Wide) Molded Small Outline Package, JEDEC NS Package Number M16B

All dimensions are in $\frac{\text{inches}}{\text{millimeters}}$



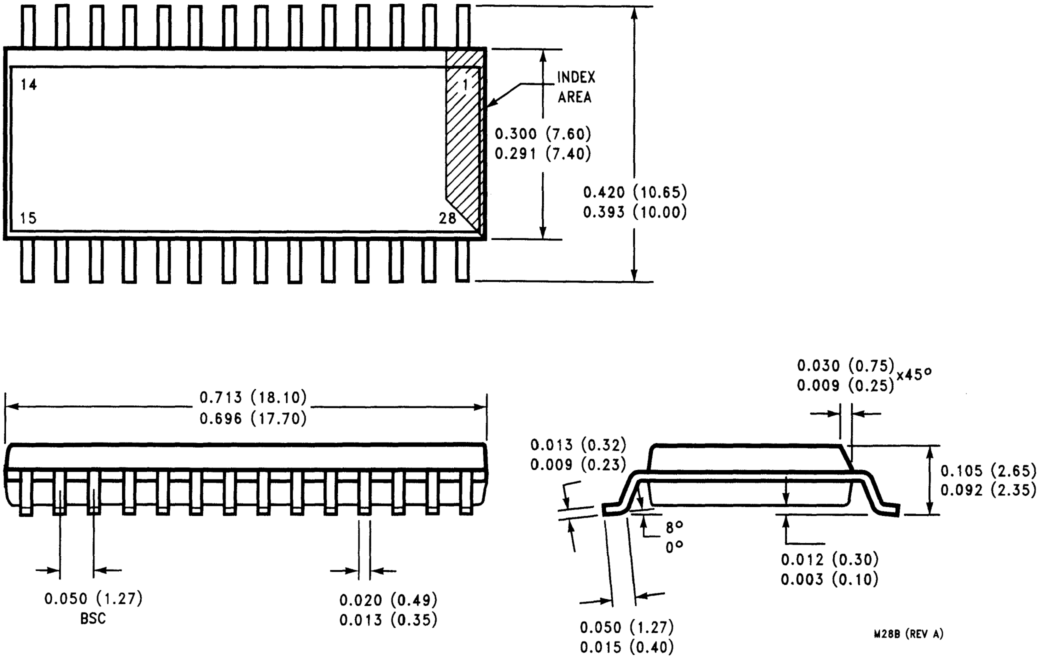
20 Lead (0.300″ Wide) Molded Small Outline Package, JEDEC NS Package Number M20B

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)

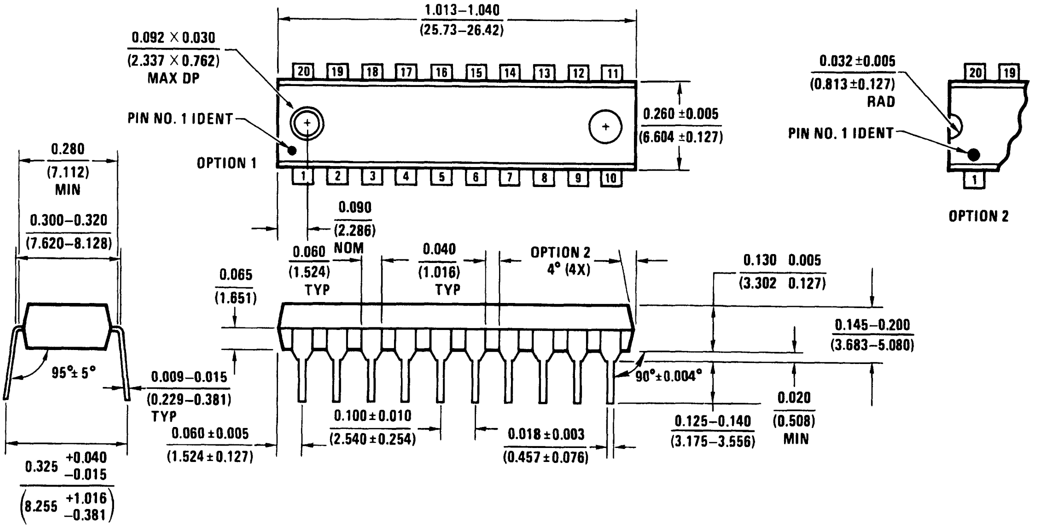


M28B (REV A)

---

20 Lead (0.300″ Wide) Molded Dual-in-Line Package
NS Package Number N20A

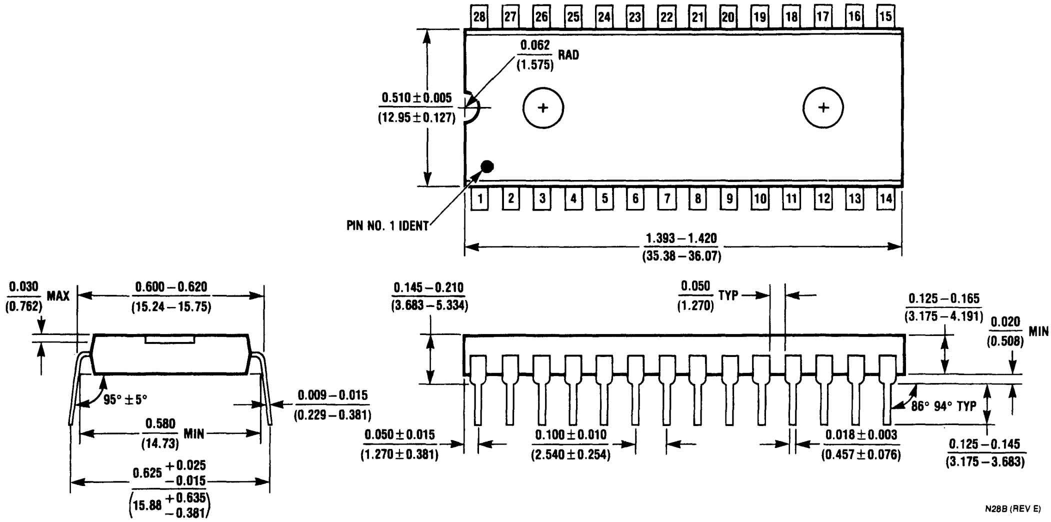
All dimensions are in inches (millimeters)



N20A (REV G)

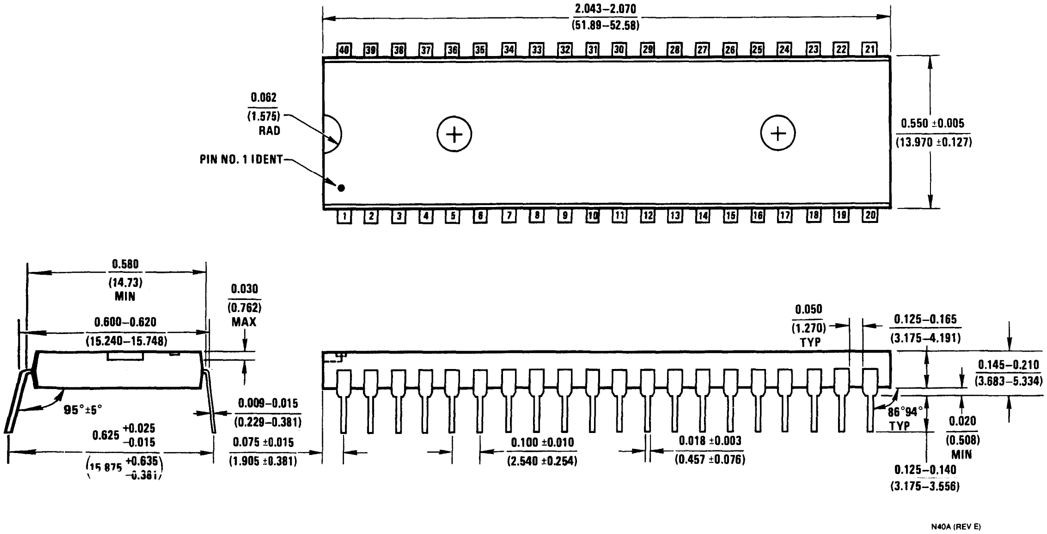
28 Lead (0.600" Wide) Molded Dual-in-Line Package
NS Package Number N28B

All dimensions are in inches (millimeters)



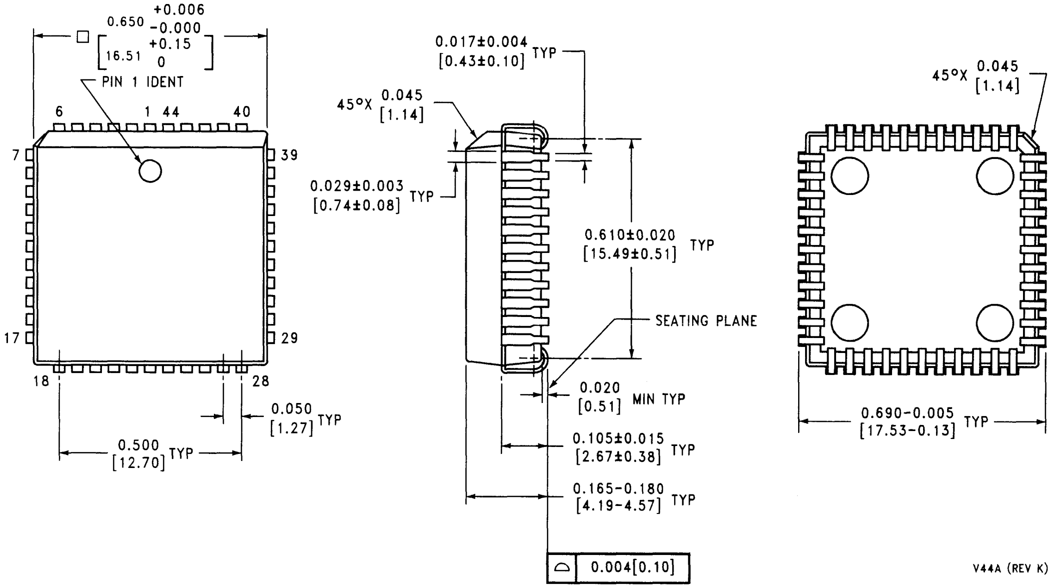
40 Lead (0.600" Wide) Molded Dual-in-Line Package
NS Package Number N40A

All dimensions are in inches (millimeters)



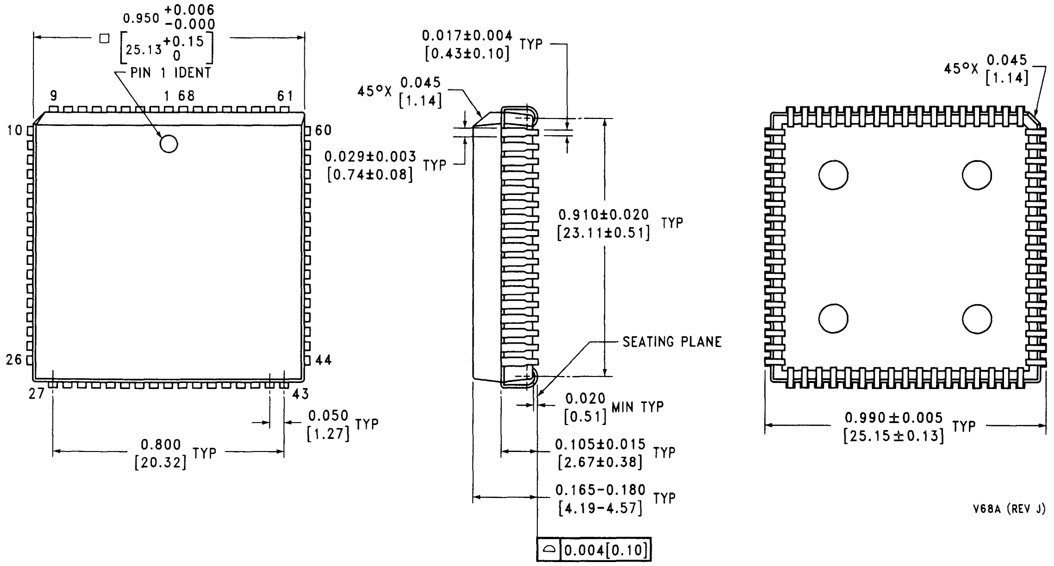
44 Lead Molded Plastic Leaded Chip Carrier
NS Package Number V44A

All dimensions are in inches [millimeters]



68 Lead Molded Plastic Leaded Chip Carrier
NS Package Number V68A

All dimensions are in inches [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 (ABTC, IBF, BiCMOS SCAN, LOW VOLTAGE BiCMOS, EXTENDED TTL TECHNOLOGY) DATABOOK—1994

ABTC/BCT Description and Family Characteristics • ABTC/BCT Ratings, Specifications and Waveforms
ABTC Applications and Design Considerations • Quality and Reliability • Integrated Bus Function (IBF) Introduction
54/74ABT3283 Synchronous Datapath Multiplexer • 74FR900/25900 9-Bit 3-Port Latchable Datapath Multiplexer
54/74ACTQ3283 32-Bit Latchable Transceiver with Parity Generator/Checker and Byte Multiplexing
SCAN18xxxA BiCMOS 5V Logic with Boundary Scan • 74LVT Low Voltage BiCMOS Logic
VME Extended TTL Technology for Backplanes

ALS/AS LOGIC DATABOOK—1990

Introduction to Advanced Bipolar Logic • Advanced Low Power Schottky • Advanced Schottky

ASIC DESIGN MANUAL/GATE ARRAYS & STANDARD CELLS—1987

SSI/MSI Functions • Peripheral Functions • LSI/VLSI Functions • Design Guidelines • Packaging

CMOS LOGIC DATABOOK—1988

CMOS AC Switching Test Circuits and Timing Waveforms • CMOS Application Notes • MM54HC/MM74HC
MM54HCT/MM74HCT • CD4XXX • MM54CXXX/MM74CXXX • Surface Mount

CLOCK GENERATION AND SUPPORT (CGS) DESIGN DATABOOK—1994

Low Skew Clock Buffers/Drivers • Video Clock Generators • Low Skew PLL Clock Generators
Crystal Clock Generators

COP8™ DATABOOK—1994

COP8 Family • COP8 Applications • MICROWIRE/PLUS Peripherals • COP8 Development Support

CROSSVOLT™ LOW VOLTAGE LOGIC SERIES DATABOOK—1994

LCX Family • LVX Translator Family • LVX Bus Switch Family • LVX Family • LVQ Family • LVT Family

DATA ACQUISITION DATABOOK—1993

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

EMBEDDED CONTROLLERS DATABOOK—1992

COP400 Family • COP800 Family • COPS Applications • HPC Family • HPC Applications
MICROWIRE and MICROWIRE/PLUS Peripherals • Microcontroller Development Tools

FDDI DATABOOK—1991

FDDI Overview • DP83200 FDDI Chip Set • Development Support • Application Notes and System Briefs

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® ADVANCED SCHOTTKY TTL LOGIC DATABOOK—1990

Circuit Characteristics • Ratings, Specifications and Waveforms • Design Considerations • 54F/74FXXX

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: DATA TRANSMISSION DATABOOK—1994

TIA/EIA-232 (RS-232) • TIA/EIA-422/423 • TIA/EIA-485 • Line Drivers • Receivers • Repeaters
Transceivers • Low Voltage Differential Signaling • Special Interface • Application Notes

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.

LINEAR APPLICATION SPECIFIC IC's DATABOOK—1993

Audio Circuits • Radio Circuits • Video Circuits • Display Drivers • Clock Drivers • Frequency Synthesis
Special Automotive • Special Functions • Surface Mount

LOCAL AREA NETWORKS DATABOOK—1993 SECOND EDITION

Integrated Ethernet Network Interface Controller Products • Ethernet Physical Layer Transceivers
Ethernet Repeater Interface Controller Products • Token-Ring Interface Controller (TROPIC)
Hardware and Software Support Products • FDDI Products • Glossary and Acronyms

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 DATABOOK—1994

FLASH • CMOS EPROMs • CMOS EEPROMs • PROMs • Application Notes

MEMORY APPLICATIONS HANDBOOK—1994

FLASH • EEPROMs • EPROMs • Application Notes

OPERATIONAL AMPLIFIERS DATABOOK—1993

Operational Amplifiers • Buffers • Voltage Comparators • Instrumentation Amplifiers • 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—1993

Linear Voltage Regulators • Low Dropout Voltage Regulators • Switching Voltage Regulators • Motion Control
Peripheral Drivers • High Current Switches • Surface Mount

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—1994

Evolution of IEEE 1149.1 Standard • SCAN BiCMOS Products • SCAN ACMOS Products • System Test Products
Other IEEE 1149.1 Devices

TELECOMMUNICATIONS—1994

COMBO and SLIC Devices • ISDN • Digital Loop Devices • Analog Telephone Components • Software
Application Notes

VHC/VHCT ADVANCED CMOS LOGIC DATABOOK—1993

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 and Product Datasheets. The topics discussed are the advantages of VHC/VHCT AC Performance, Low Noise Characteristics and Improved Interface Capabilities.

NOTES

NATIONAL SEMICONDUCTOR CORPORATION DISTRIBUTORS

ALABAMA
Huntsville
Anthem Electronics
(205) 890-0302
Hamilton/Hallmark
(205) 837-8700
Pioneer Technology
(205) 837-9300
Time Electronics
(800) 258-8513

ARIZONA
Phoenix
Hamilton/Hallmark
(602) 437-1200
Scottsdale
Alliance Electronics Inc.
(602) 483-9400
Tempe
Anthem Electronics
(602) 966-6600
Bell Industries
(602) 966-7800
Time Electronics
(602) 967-2000

CALIFORNIA
Agora Hills
Bell Industries
(818) 706-2608
Pioneer Standard
(818) 865-5800
Time Electronics
(818) 707-2890
Calabasas
F/X Electronics
(818) 591-9220
Chatsworth
Anthem Electronics
(818) 775-1333
Costa Mesa
Hamilton/Hallmark
(714) 641-4100
Irvine
Anthem Electronics
(714) 768-4444
Bell Industries
(714) 727-4500
Pioneer Standard
(714) 753-5090
Zeus Elect. an Arrow Co.
(714) 581-4622
Los Angeles
Bell Industries
(310) 447-6321
Rocklin
Anthem Electronics
(916) 624-9744
Bell Industries
(916) 652-0414
Roseville
Hamilton/Hallmark
(916) 624-9781
San Diego
Anthem Electronics
(619) 453-9005
Hamilton/Hallmark
(619) 571-7540
Pioneer Standard
(619) 546-4906
Time Electronics
(619) 674-2800
San Jose
Anthem Electronics
(408) 453-1200
Hamilton/Hallmark
(408) 743-3300
Pioneer Technology
(408) 954-9100
Zeus Elect. an Arrow Co.
(408) 629-4789
Sunnyvale
Bell Industries
(408) 734-8570
Time Electronics
(408) 734-9888
Tustin
Time Electronics
(714) 669-0100
Woodland Hills
Hamilton/Hallmark
(818) 594-0404
Time Electronics
(818) 593-8400

COLORADO
Denver
Bell Industries
(303) 691-9010
Englewood
Anthem Electronics
(303) 790-4500
Hamilton/Hallmark
(303) 790-1662
Time Electronics
(303) 721-8882

CONNECTICUT
Cheshire
Hamilton/Hallmark
(203) 271-2844
Shelton
Pioneer Standard
(203) 929-5600
Waterbury
Anthem Electronics
(203) 575-1575

FLORIDA
Altamonte Springs
Anthem Electronics
(407) 831-0007
Bell Industries
(407) 339-0078
Pioneer Technology
(407) 834-9090
Deerfield Beach
Pioneer Technology
(305) 428-8877
Fort Lauderdale
Hamilton/Hallmark
(305) 484-5482
Time Electronics
(305) 484-1778
Lake Mary
Zeus Elect. an Arrow Co.
(407) 333-9300
Largo
Hamilton/Hallmark
(813) 541-7440
Orlando
Chip Supply
Die Distributor
(407) 298-7100
Time Electronics
(407) 841-6565
Winter Park
Hamilton/Hallmark
(407) 657-3300

GEORGIA
Duluth
Hamilton/Hallmark
(404) 623-4400
Pioneer Technology
(404) 623-1003
Norcross
Bell Industries
(404) 662-0923
Time Electronics
(404) 368-0969

ILLINOIS
Addison
Pioneer Electronics
(708) 495-9680
Bensenville
Hamilton/Hallmark
(708) 860-7780
Elk Grove Village
Bell Industries
(708) 640-1910
Schaumburg
Anthem Electronics
(708) 884-0200
Time Electronics
(708) 303-3000

INDIANA
Fort Wayne
Bell Industries
(219) 423-3422
Indianapolis
Advent Electronics Inc.
(317) 872-4910
Bell Industries
(317) 875-8200
Hamilton/Hallmark
(317) 872-8875
Pioneer Standard
(317) 573-0880

IOWA
Cedar Rapids
Advent Electronics
(319) 363-0221

KANSAS
Lenexa
Hamilton/Hallmark
(913) 332-4375

KENTUCKY
Lexington
Hamilton/Hallmark
(602) 288-4911

MARYLAND
Columbia
Anthem Electronics
(410) 995-6840
Bell Industries
(410) 290-5100
Hamilton/Hallmark
(410) 988-9800
Time Electronics
(410) 720-3600
Gaithersburg
Pioneer Technology
(301) 921-0660

MASSACHUSETTS
Andover
Bell Industries
(508) 474-8880
Beverly
Sertech Laboratories
(508) 927-5820
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) 532-9900
Tyngsboro
Port Electronics
(508) 649-4880
Wilmington
Anthem Electronics
(508) 657-5170
Zeus Elect. an Arrow Co.
(508) 658-0900

MICHIGAN
Grand Rapids
Pioneer Standard
(616) 698-1800
Novi
Hamilton/Hallmark
(313) 347-4271
Plymouth
Pioneer Standard
(313) 416-2157
Wyoming
R. M. Electronics, Inc.
(616) 531-9300

MINNESOTA
Bloomington
Hamilton/Hallmark
(612) 881-2600
Eden Prairie
Anthem Electronics
(612) 944-5454
Pioneer Standard
(612) 944-3355
Edina
Time Electronics
(612) 943-2433
Thief River Falls
Digi-Key Corp.
"Catalog Sales Only"
(800) 344-4539

MISSOURI
Earth City
Hamilton/Hallmark
(314) 291-5350
Manchester
Time Electronics
(314) 391-6444

NEW JERSEY
Cherry Hill
Hamilton/Hallmark
(609) 424-0110
Fairfield
Bell Industries
(201) 227-6060
Hamilton/Hallmark
(201) 575-3390
Pioneer Standard
(201) 575-3510
Marlton
Time Electronics
(609) 596-6700
Mount Laurel
Seymour Electronics
(609) 235-7474
Parsippany
Hamilton/Hallmark
(201) 515-1641
Pine Brook
Anthem Electronics
(201) 227-7960
Wayne
Time Electronics
(201) 785-8250

NEW MEXICO
Albuquerque
Bell Industries
(505) 292-2700
Hamilton/Hallmark
(505) 828-1058

NATIONAL SEMICONDUCTOR CORPORATION DISTRIBUTORS (Continued)

NEW YORK
Binghamton
Pioneer
(607) 722-9300
Buffalo
Summit Electronics
(716) 887-2800
Commack
Anthem Electronics
(516) 864-6600
Fairport
Pioneer Standard
(716) 381-7070
Hauppauge
Hamilton/Hallmark
(516) 434-7490
Time Electronics
(516) 273-0100
Port Chester
Zeus Elect. an Arrow Co.
(914) 937-7400
Rochester
Hamilton/Hallmark
(800) 475-9130
Summit Electronics
(716) 334-8110
Ronkonkoma
Hamilton/Hallmark
(516) 737-0600
Syracuse
Time Electronics
(315) 432-0355
Westbury
Hamilton/Hallmark Export Div.
(516) 997-6868
Woodbury
Pioneer Electronics
(516) 921-8700
Seymour Electronics
(516) 496-7474

NORTH CAROLINA
Morrisville
Pioneer Technology
(919) 460-1530
Raleigh
Hamilton/Hallmark
(919) 872-0712

OHIO
Cleveland
Pioneer
(216) 587-3600
Columbus
Time Electronics
(614) 794-3301
Dayton
Bell Industries
(513) 435-8660
Bell Industries-Military
(513) 434-8231
Hamilton/Hallmark
(513) 439-6735
Pioneer Standard
(513) 236-9900

OKLAHOMA
Tulsa
Hamilton/Hallmark
(918) 254-6110
Pioneer Standard
(918) 665-7840
Radio Inc.
(918) 587-9123

OREGON
Beaverton
Anthem Electronics
(503) 643-1114
Bell Industries
(503) 644-3444
Hamilton/Hallmark
(503) 526-6200
Pioneer Technology
(503) 626-7300
Portland
Time Electronics
(503) 684-3780

PENNSYLVANIA
Horsham
Anthem Electronics
(215) 443-5150
Pioneer Technology
(215) 674-4000
Mars
Hamilton/Hallmark
(412) 281-4150
Pittsburgh
Pioneer Standard
(412) 782-2300
Trevose
Bell Industries
(215) 953-2800

TEXAS
Austin
Hamilton/Hallmark
(512) 258-8848
Minco Technology Labs.
(512) 834-2022
Pioneer Standard
(512) 835-4000
Time Electronics
(512) 346-7346
Carrollton
Zeus Elect. an Arrow Co.
(214) 380-6464
Dallas
Hamilton/Hallmark
(214) 553-4300
Pioneer Standard
(214) 386-7300
Houston
Hamilton/Hallmark
(713) 781-6100
Pioneer Standard
(713) 495-4700
Richardson
Anthem Electronics
(214) 238-7100
Time Electronics
(214) 644-4644

UTAH
Midvale
Bell Industries
(801) 255-9611
Salt Lake City
Anthem Electronics
(801) 973-8555
Hamilton/Hallmark
(801) 266-2022
West Valley
Time Electronics
(801) 973-8494

WASHINGTON
Bellevue
Pioneer Technology
(206) 644-7500
Bothell
Anthem Electronics
(206) 483-1700
Kirkland
Time Electronics
(206) 820-1525
Redmond
Bell Industries
(206) 867-5410
Hamilton/Hallmark
(206) 881-6697

WISCONSIN
Brookfield
Pioneer Electronics
(414) 784-3480
Mequon
Taylor Electric
(414) 241-4321
New Berlin
Hamilton/Hallmark
(414) 780-7200
Waukesha
Bell Industries
(414) 547-8879
Hamilton/Hallmark
(414) 784-8205

CANADA
WESTERN PROVINCES
Burnaby
Hamilton/Hallmark
(604) 420-4101
Semad Electronics
(604) 451-3444
Calgary
Electro Sonic Inc.
(403) 255-9550
Semad Electronics
(403) 252-5664
Zentronics
(403) 295-8838
Edmonton
Zentronics
(403) 482-3038
Markham
Semad Electronics Ltd.
(416) 475-6500
Richmond
Electro Sonic Inc.
(604) 273-2911
Zentronics
(604) 273-5575
Winnipeg
Zentronics
(204) 694-1957

EASTERN PROVINCES
Mississauga
Hamilton/Hallmark
(416) 564-6060
Time Electronics
(416) 672-5300
Zentronics
(416) 507-2600
Nepean
Hamilton/Hallmark
(613) 226-1700
Zentronics
(613) 226-8840
Ottawa
Electro Sonic Inc.
(613) 728-8333
Semad Electronics
(613) 526-4866
Pointe Claire
Semad Electronics
(514) 694-0860
Ville St. Laurent
Hamilton/Hallmark
(514) 335-1000
Zentronics
(514) 737-9700
Willowdale
ElectroSonic Inc.
(416) 494-1666
Winnipeg
Electro Sonic Inc.
(204) 783-3105