DATA HANDBOOK

8048-based 8-bit Microcontrollers

BOOK IC14 1991

Philips Components

PHILIPS

8048-BASED 8-BIT MICROCONTROLLERS

CONTENTS

8048-based 8-bit microcontrollers

IC14

Preface

Philips Components offer a wide range and innovative line of 8048, 8051 and 68000 compatible microcontrollers and microprocessors based on industry standard architectures for embedded control applications.

Detailed information on the Philips microcontrollers and microprocessors product line is provided within a suite of three 1991 print-dated data handbooks, IC14, IC20 and IC21 as follows:

microcontrollers and microprocessors based on the:

- 8048 industry standard architecture are detailed in Philips Data Handbook IC14.
- 8051 industry standard architecture are detailed in Philips Data Handbook IC20.
- 68000 compatible industry standard architecture are detailed in the Philips Data Handbook IC21.

The IC14 data handbook deals specifically with the 8-bit microcontrollers that use the 8048 instruction set and derivative microcontrollers based on a modified/enhanced 8048 architecture and instruction set.

IC14 section 1 — Introduction, provides examples on ordering information, type numbering and tables detailing the Philips range of 8048, 8051 and 68000 compatible microcontrollers and microprocessors for embedded control applications.

IC14 sections: 2, 3, 4, 5, 6, 7, 8, and 9 provide detailed information on 8048, 84XX, 84CXXX and 33XX Families and derivatives, the 8048-based instruction set, the 84XX/84CXX/33XX serail I/O and Software Examples, Development Support and Package Outlines.

DEFINITIONS

Data sheet status	
Objective specification	This data sheet contains target or goal specifications for product development.
Preliminary specification	This data sheet contains preliminary data; supplementary data may be published later.
Product specification	This data sheet contains final product specifications.
Limiting values	
Limiting values given are in accordance with the Absolute Maximum Rating System (IEC 134). Stress above one or more of the limiting values may cause permanent damage to the device. These are stress ratings only and operation of the device at these or at any other conditions above those given in the Characteristics sections of this specification is not implied. Exposure to limiting values for extended periods may affect device reliability.	

Printed in The Netherlands

Pro electron type designation code for integrated circuits

TYPE DESIGNATION

Basic type number

This type designation applies to semiconductor monolithic, semiconductor multi-chip, thin film, thick- film and hybrid integrated circuits.

A basic type number consists of three letters followed by a serial number.

FIRST AND SECOND LETTER

Digital family circuits

The first two letters identify the family (see note 1).

Solitary circuits

The first letter divides the solitary circuits into:

S : solitary digital circuits
T : analog circuits
U : mixed analog/digital circuits

The second letter is a serial letter without any further significance except 'H' which stands for hybrid circuits (see note 2).

Microprocessors

The first two letters identify microprocessors and correlated circuits as follows:

MA: microcomputer central processing unit
MB: slice processor (see note 3)
MD: correlated memories
ME: other correlated circuits (interface, clock, peripheral controller, etc.)

Charge-transfer devices and switched capacitors.

The first two letters identify the following:

NH: hybrid circuits
NL: logic circuits
NM: memories
NS: analog signal processing, using switched capacitors
NT: analog signal processing, using change-transfer device
NX: imaging devices
NY: other correlated circuits

THIRD LETTER

The third letter indicates the operating ambient temperature range. The letters A to G give information about the temperature:

A : temperature range not specified below (see note 4)
B : 0 to + 70 °C
C : −55 to +125 °C
D : −25 to + 70 °C
E : −25 to + 85 °C
F : −40 to + 85 °C
G : −55 to + 85 °C

If a circuit is published for another temperature range, the letter indicating a narrower temperature range may be used or the letter 'A'.

Example : the range 0 to +75 °C can be indicated by 'B' or 'A'.

SERIAL NUMBER

This may be either a 4-digit number assigned by Pro Electron, or the serial number (which may be a combination of figures and letters) of an existing company type designation of the manufacturer.

To the basic type number may be added:

Version letter(s)

A single version letter may be added to the basic type number. This indicates a minor variant of the basic type or the package. Except for 'Z', which means customized wiring, the letter has no fixed meaning. The following letters are recommended for package variants:

C : for cylindrical
D : for ceramic DIL
F : for flat pack (2 leads)
G : for flat pack (4 leads)
H : for quadrature flat pack (OFP)
L : for chip on tape (foil)
P : for plastic DIL
Q : for QIL
T : for miniature plastic (mini-pack)
U : for uncased chip

Pro electron type designation code for integrated circuits

Alternatively a TWO LETTER SUFFIX may be used instead of a single package version letter, if the manufacturer (sponsor) wishes to give more information.

FIRST LETTER: General shape

C : cylindrical
D : dual-in-line (DIL)
E : power DIL (with external heatsink)
F : flat (leads on 2 sides)
G : flat (leads on 4 sides)
H : quadrature flat pack (QFP)
K : diamond (TO-3 family)
M : multiple-in-line (except dual-, triple-, quadruple-in-line)
Q : quadruple-in-line (QIL)
R : power QIL (with external heatsink)
S : single-in-line
T : triple-in-line
W : lead chip-carrier (LCC)
X : leadless chip-carrier (LLCC)
Y : pin grid array (PGA)

SECOND LETTER: Material

C : metal-ceramic
G : glass-ceramic (cerdip)
M : metal
P : plastic

To avoid confusion when the serial number ends with a letter, a hyphen is used preceding the suffix.

Examples (see note 5)

PCF1105WP : Digital IC, PC family, operational temperature range –40 to +85 °C, serial number 1105, plastic leaded chip-carrier.

GMB74LS00A-DC: Digital IC, GM family, operational temperature range 0 to +70 °C, company number 74LSS00A, ceramic DIL package.

TDA1000P : Analog circuit, no standard temperature range, serial number 1000, plastic DIL package.

SAC2000 : Solitary digital circuit, operational temperature range –55 to +125 °C.

Notes

1. A logic family is an assembly of digital circuits designed to be interconnected and defined by its basic electrical characteristics (such as: supply voltage, power consumption, propagation delay, noise immunity).

2. The first letter 'S' should be used for all solitary memories, to which, in the event of hybrids, the second letter 'H' should be added (e.g. SH for Bubble-memories).

3. By 'slice processor' is meant: a functional slice of microprocessor.

4. In the case of two same types with two different temperature ranges not specified below, one type should use the letter 'A' as the third letter and the other, the letter 'X'.

5. Some companies have been using version letters and/or two letter-suffix, which differ from the Pro Electron definitions. In case of confusion Pro Electron may be contacted.

RATING SYSTEMS

The rating systems described are those recommended by the International Electrotechnical Commission (IEC) in its Publication 134.

DEFINITIONS OF TERMS USED

Electronic device. An electronic tube or valve, transistor or other semiconductor device.

Note
This definition excludes inductors, capacitors, resistors and similar components.

Characteristic. A characteristic is an inherent and measurable property of a device. Such a property may be electrical, mechanical, thermal, hydraulic, electro-magnetic, or nuclear, and can be expressed as a value for stated or recognized conditions. A characteristic may also be a set of related values, usually shown in graphical form.

Bogey electronic device. An electronic device whose characteristics have the published nominal values for the type. A bogey electronic device for any particular application can be obtained by considering only those characteristics which are directly related to the application.

Rating. A value which establishes either a limiting capability or a limiting condition for an electronic device. It is determined for specified values of environment and operation, and may be stated in any suitable terms.

Note
Limiting conditions may be either maxima or minima.

Rating system. The set of principles upon which ratings are established and which determine their interpretation.

Note
The rating system indicates the division of responsibility between the device manufacturer and the circuit designer, with the object of ensuring that the working conditions do not exceed the ratings.

ABSOLUTE MAXIMUM RATING SYSTEM

Absolute maximum ratings are limiting values of operating and environmental conditions applicable to any electronic device of a specified type as defined by its published data, which should not be exceeded under the worst probable conditions.

These values are chosen by the device manufacturer to provide acceptable serviceability of the device, taking no responsibility for equipment variations, environmental variations, and the effects of changes in operating conditions due to variations in the characteristics of the device under consideration and of all other electronic devices in the equipment.

The equipment manufacturer should design so that, initially and throughout life, no absolute maximum value for the intended service is exceeded with any device under the worst probable operating conditions with respect to supply voltage variation, equipment component variation, equipment control adjustment, load variations, signal variation, environmental conditions, and variations in characteristics of the device under consideration and of all other electronic devices in the equipment.

DESIGN MAXIMUM RATING SYSTEM

Design maximum ratings are limiting values of operating and environmental conditions applicable to a bogey electronic device of a specified type as defined by its published data, and should not be exceeded under the worst probable conditions.

These values are chosen by the device manufacturer to provide acceptable serviceability of the device, taking responsibility for the effects of changes in operating conditions due to variations in the characteristics of the electronic device under consideration.

The equipment manufacturer should design so that, initially and throughout life, no design maximum value for the intended service is exceeded with a bogey device under the worst probable operating conditions with respect to supply voltage variation, equipment component variation, variation in characteristics of all other devices in the equipment, equipment control adjustment, load variation, signal variation and environmental conditions.

DESIGN CENTRE RATING SYSTEM

Design centre ratings are limiting values of operating and environmental conditions applicable to a bogey electronic device of a specified type as defined by its published data, and should not be exceeded under normal conditions.

These values are chosen by the device manufacturer to provide acceptable serviceability of the device in average applications, taking responsibility for normal changes in operating conditions due to rated supply voltage variation, equipment component variation, equipment control adjustment, load variation, signal variation, environmental conditions, and variations in the characteristics of all electronic devices.

The equipment manufacturer should design so that, initially, no design centre value for the intended service is exceeded with a bogey electronic device in equipment operating at the stated normal supply voltage.

HANDLING MOS DEVICES

Though all our MOS integrated circuits incorporate protection against electrostatic discharges, they can nevertheless be damaged by accidental over-voltages. In storing and handling them, the following precautions are recommended.

Caution

Testing or handling and mounting call for special attention to personal safety. Personnel handling MOS devices should normally be connected to ground via a resistor.

Storage and transport

Store and transport the circuits in their original packing. Alternatively, use may be made of a conductive material or special IC carrier that either short-circuits all leads or insulates them from external contact.

Testing or handling

Work on a conductive surface (e.g. metal table top) when testing the circuits or transferring them from one carrier to another. Electrically connect the person doing the testing or handling to the conductive surface, for example by a metal bracelet and a conductive cord or chain. Connect all testing and handling equipment to the same surface.
Signals should not be applied to the inputs while the device power supply is off. All unused input leads should be connected to either the supply voltage or ground.

Mounting

Mount MOS integrated circuits on printed circuit boards *after* all other components have been mounted. Take care that the circuits themselves, metal parts of the board, mounting tools, and the person doing the mounting are kept at the same electric (ground) potential. If it is impossible to ground the printed-circuit board the person mounting the circuits should touch the board before bringing MOS circuits into contact with it.

Soldering

Soldering iron tips, including those of low-voltage irons, or soldering baths should also be kept at the same potential as the MOS circuits and the board.

Static charges

Dress personnel in clothing of non-electrostatic material (no wool, silk or synthetic fibres). After the MOS circuits have been mounted on the board proper handling precautions should still be observed. Until the sub-assemblies are inserted into a complete system in which the proper voltages are supplied, the board is no more than an extension of the leads of the devices mounted on the board. To prevent static charges from being transmitted through the board wiring to the device it is recommended that conductive clips or conductive tape be put on the circuit board terminals.

Transient voltages

To prevent permanent damage due to transient voltages, do not insert or remove MOS devices, or printed-circuit boards with MOS devices, from test sockets or systems with power on.

Voltage surges

Beware of voltage surges due to switching electrical equipment on or off, relays and d.c. lines.

Section 1 - Introduction

Introduction

CONTENTS

Section

1.0 INTRODUCTION

IC14 is one of three books detailing the Philips product line of microcontrollers and microprocessors. This book deals only with the 8-bit microcontrollers that use the 8048 instruction set and derivative microcontrollers based on the 8048 architecture and instruction set. Microcontrollers based on the 8051 and 68000 instruction sets are dealt with in other books.

1.1 Single-chip 8-bit microcontrollers with 8048-based instruction set

The microcontrollers presented in this book are based on the 8048 industry standard 8-bit microcontroller. Section 2 provides full technical details on the 8048 and derivatives.

The NMOS 84XX family of derivatives are optimized for cost-effective applications. The I/O and interrupt capabilities have been enhanced, and the pin count has been reduced in comparison with the 8048. This family is detailed in Section 3.

Section 4 deals with the CMOS 84CXXX family of derivatives. The 84CXXX microcontroller family is essentially a CMOS enhancement of the NMOS 84XX family and contains a wide variety of more application specific derivatives; most of the derivatives feature a supply voltage range of 2.5 to 5.5 V. These devices are ideally suited for low-power, low-voltage applications, especially in the telecommunication and consumer areas.

The CMOS 33XX family is detailed in Section 5. These devices are optimized for telephony applications. Several members of the 33XX family feature a supply voltage range of 1.8 to 6.0 V. The wide supply voltage range plus on-chip application-specific hardware, based on many years of Philips experience in electronics, make the 84XX, 84CXXX and 33XX microcontroller families unique in the microcontroller market.

1.2 Manufacturing and support

Philips single-chip microcontrollers are produced in several locations throughout the world, using state-of-the-art manufacturing facilities. Philips microcontrollers conform to the highest quality standards in the industry. Philips has a commitment to offer the best support, documentation, and development tools available. This includes in-circuit emulators with debugging and real-time tracing software as well as assemblers and compilers. The presence of Philips sales offices all over the world facilitates fast and efficient communication with the manufacturing sites. Furthermore, application support centres in several locations around the world provide quick solutions to customer application queries.

1.3 Type numbering

Fig.1 gives detailed examples of the type numbering system used throughout this book.

1.4 A range of microcontrollers and microprocessors for embedded control applications

Philips Components offers the industry's most complete and innovative line of microcontrollers and microprocessors for embedded control applications. Based on industry standard architectures, over sixty microcontrollers derivatives have been developed around the 8-bit 8048 and 80C51, and the 16-bit 68000 compatible CPU. Philips has a long history of experience in developing state-of-the-art technology for use in sophisticated cost-sensitive consumer and telecommunication applications to sophisticated highly-reliable medical instrumentation and automobile control systems.

Many derivatives integrate application-specific functions such as EEPROM for applications where security or real estate is critical.

To meet the growing demand for hand-held and portable equipment, a full line of low-voltage, low-power devices is available. These devices offer increased performance without increased voltage/power consumption.

To reduce interconnect complexity, the two wire I^2C serial bus interface hardware has been incorporated in many derivatives (see Fig.2). More than 85 microcontrollers and other devices support the I^2C-bus.

The following pages give a brief overview of the type numbers and the most prominent features of the complete range of microcontrollers and microprocessors. Full technical details are available in data sheets, data handbooks and user manuals. Since documentation is

continuously updated and enhanced, the nearest Philips Components sales office should be contacted for the most recent information.

Table 1 deals with the 8-bit microcontrollers based on the 8048 architecture. These microcontrollers are optimized for low-cost applications and are particularly suited to consumer and telephony applications. The low supply voltage (specified down to 1.8 V) and low power features of most of these ICs make them ideally suited for battery operated applications.

Table 2 details the industry standard 8051 microcontrollers and derivatives. These derivatives contain a wide variety of peripheral functions integrated on-chip: expansion of I/O-lines, A/D conversion, PWM outputs, I²C-bus interface, EEPROM, expanded program ROM and data RAM, powerful capture/compare timer logic etc. Several of the derivatives have the following additional features: higher operating frequencies, super small packaging (e.g. 87C751 in skinny DIP-24 package), and enhanced supply voltage range. In addition to the derivatives mentioned here, 8051-derivatives based on a customer specification can also be produced, facilitating an even faster and more economic solution to all system designer's needs.

Table 3 deals with 16/32-bit microprocessor and microcontroller derivatives based on the powerful 68000 architecture and instruction set. Derivative bus interfaces are compatible with the 68000 bus. These products are most effectively used in applications where high performance and/or a wide addressing range are required. The high level of integration makes cost-effective 16/32-bit systems a reality. The 68070 is a highly integrated microprocessor featuring on-chip DMA channels, I²C interface, MMU, UART, timers and extra decoding logic. This IC practically forms a complete

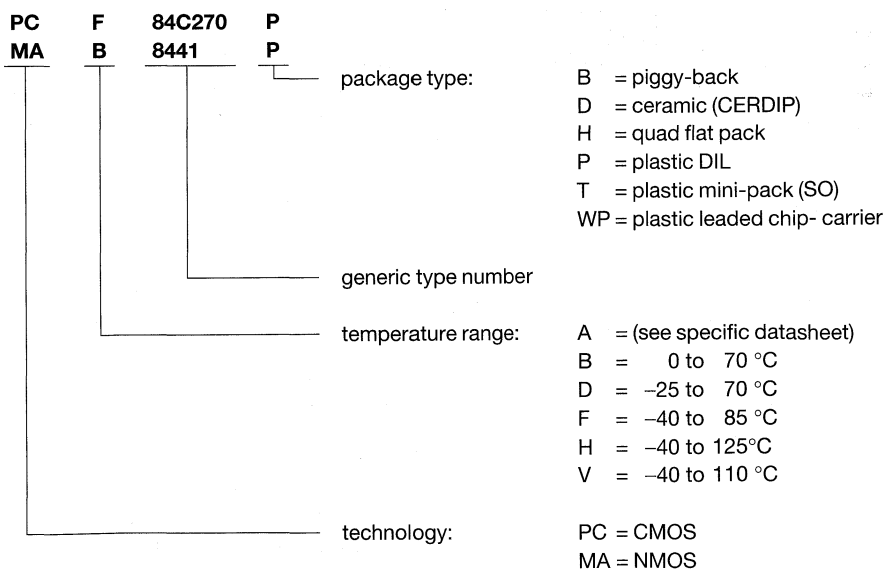


Fig.1 Type numbering.

16/32-bit system on a single chip. The 66470, Video and System Controller, can be used as a companion chip of the 68070 and provides a complete display, bit manipulation logic, DRAM and System controller on one chip.

The 9XC1XX products are 68K-compatible microcontrollers with on-chip ROM (or EPROM), RAM, EEPROM, I²C interface, UART, 8051 compatible bus (in addition to, or instead of the 68 K bus), etc.
Low power versions are in preparation.

In addition to the above devices, 68000 architecture 16/32-bit derivatives based on customer specifications can also be produced.

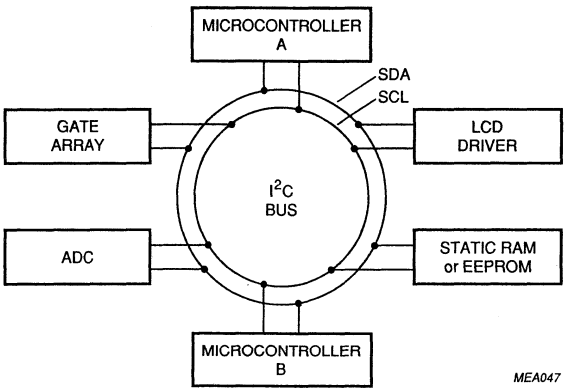


Fig.2 Typical I²C-bus configuration.

Table 1a CMOS microcontrollers with an enhanced instruction set based on the 8048

GENERIC TYPE NUMBER	ROM (bytes)	RAM (bytes)	EEPROM (bytes)	I/O	I^2C	V_{DD} (V)	I_{DD} (MAX.) (mA)[1]	f_{XTAL} (MAX.) (MHz)[9]	PIGGY-BACK/ ROM-LESS TYPE
3315	1.5 K	160	-	20	N	1.8 - 6.0	1.6	10	3301B
3343	3 K	224	-	20	Y	1.8 - 6.0	1.6	10	3301B
3344	2 K	224	-	20	N	2.5 - 6.0	1.6	3.58	3344B
3346	4 K	128	256	20	Y	2.5 - 6.0	1.6	10	3346B
3347	1.5 K	64	-	12	N	2.5 - 6.0	1.6	3.58	3344B [3]
3348	8 K	256	-	20	Y	1.8 - 6.0	1.6	10	3301B
3349	4 K	224	-	20	N	2.5 - 6.0	1.6	3.58	3344B
80C49	2 K	128	-	27	N	5 ± 10%	15	11	80C39
84C12	1 K	64	-	13	N	2.5 - 5.5	3.2	10	84C00B/T
84C22	2 K	64	-	13	N	2.5 - 5.5	3.2	10	84C00B/T
84C42	4 K	64	-	13	N	2.5 - 5.5	3.2	10	84C00B/T
84C121	1 K	64	8	13	N	2.5 - 5.5	3.2	10	84C121B
84C00T	-	256	-	20	Y	2.5 - 5.5	3.2	10	-
84C21	2 K	64	-	20	Y	2.5 - 5.5	3.2	10	84C00B/T
84C41	4 K	128	-	20	Y	2.5 - 5.5	3.2	10	84C00B/T
84C81	8 K	256	-	20	Y	2.5 - 5.5	3.2	10	84C00B/T
84C85	8 K	256	-	32	Y	2.5 - 5.5	3.2	10	84C85B
84C230	2 K	64	-	12	N	2.5 - 5.5	3.2	10	84C430B [4]
84C270P	2 K	128	-	8	N	2.5 - 5.5	3.2	10	84C270B
84C430	4 K	128	-	25	Y	2.5 - 5.5	3.2	10	84C430B
84C440	4 K	128	-	18	Y	4.5 - 5.5	10	10	84C640B
84C441	4 K	128	-	17	Y	4.5 - 5.5	10	7	84C641B
84C443	4 K	128	-	18	N	4.5 - 5.5	10	10	84C640B
84C444	4 K	128	-	17	N	4.5 - 5.5	10	7	84C641B

INSTRUCTION SET	PACKAGES	TEMPERATURE RANGES	FEATURES	A[5]	C[6]	G[7]	T[8]
8048 [2]	DIL-28, SO-28	D	specified for telecom			•	•
8048 [2]	DIL-28, SO-28	D	specified for telecom, I²C-bus interface			•	•
8048 [2]	DIL-28, SO-28	D	specified for telecom, tone generator (DTMF/modem/musical)			•	•
8048 [2]	DIL-28, SO-28	D	specified for telecom, 256-bytes EEPROM (for redial)			•	•
8048 [2]	DIL-20, SO-20	D	specified for telecom, tone generator (DTMF/modem/musical)			•	•
8048 [2]	DIL-28, SO-28	D	specified for telecom, I²C-bus interface			•	•
8048 [2]	DIL-28, SO-28	D	specified for telecom, tone generator (DTMF/modem/musical)			•	•
8048	DIL-40, PLCC-44	H and F	CMOS 8049 with idle mode			•	
8048 [2]	DIL-20, SO-20	F	low-cost		•	•	
8048 [2]	DIL-20, SO-20	F	low-cost		•	•	
8048 [2]	DIL-20, SO-20	F	low-cost		•	•	
8048 [2]	DIL-20, SO-20	F	8-byte EEPROM		•	•	•
8048 [2]	VSO-56 (T-type)	F	ROM-less versions of this family. The B-type is a 28-pin piggy-back version for prototyping		•	•	
8048 [2]	DIL-28, SO-28	F	I²C-bus interface, 8 x 10 mA drive lines		•	•	
8048 [2]	DIL-28, SO-28	F	I²C-bus interface, 8 x 10 mA drive lines		•	•	
8048 [2]	DIL-28, SO-28	F	I²C-bus interface, 8 x 10 mA drive lines		•	•	
8048 [2]	DIL-40, VSO-40	F	I²C-bus interface, 32 I/O lines		•	•	
8048 [2]	DIL-40, SO-40	F	64-segment LCD driver	•	•	•	
8048 [2]	DIL-40, SO-40	F	additional hardware for keyboard with capacitive keys			•	
8048 [2]	QFP-64	F	96-segment LCD driver (PCA84C430B is a DIL-64 package)			•	
8048 [2]	SDIL-42	D	14-bit PWM output(s) for VST, 3-bit software A/D input for AFC, 5 x 6-bit PWM outputs, OSD 2 lines of 16 characters, 64 character fonts, RC osc.		•		
8048 [2]	SDIL-42	D	14-bit PWM output(s) for VST, 3-bit software A/D input for AFC, 5 x 6-bit PWM outputs, OSD 2 lines of 16 characters, 64 character fonts, LC osc.		•		
8048 [2]	SDIL-42	D	14-bit PWM output(s) for VST, 3-bit software A/D input for AFC, 5 x 6-bit PWM outputs, OSD 2 lines of 16 characters, 64 character fonts, RC osc.		•		
8048 [2]	SDIL-42	D	14-bit PWM output(s) for VST, 3-bit software A/D input for AFC, 5 x 6-bit PWM outputs, OSD 2 lines of 16 characters, 64 character fonts, LC osc.		•		

GENERIC TYPE NUMBER	ROM (bytes)	RAM (bytes)	EEPROM (bytes)	I/O	I²C	V_{DD} (V)	I_{DD} (MAX.) (mA)[1]	f_{XTAL} (MAX.) (MHz)[9]	PIGGY-BACK/ ROM-LESS TYPE
84C470	4 K	128	-	8	N	2.5 - 5.5	3.2	10	84C470B
84C640	6 K	128	-	18	Y	4.5 - 5.5	10	10	84C640B
84C641	6 K	128	-	17	Y	4.5 - 5.5	10	7	84C641B
84C643	6 K	128	-	18	N	4.5 - 5.5	10	10	84C640B
84C644	6 K	128	-	17	N	4.5 - 5.5	10	7	84C641B
84C840	8 K	192	-	18	Y	4.5 - 5.5	10	10	84C640B
84C841	8 K	192	-	17	Y	4.5 - 5.5	10	10	84C641B
84C843	8 K	192	-	18	N	4.5 - 5.5	10	10	84C640B
84C844	8 K	192	-	17	N	4.5 - 5.5	10	7	84C641B
84C633A	6 K	256	-	28	N	2.5 - 5.5	5.2	16	84C633B
84C853A	8 K	256	-	33	N	2.5 - 5.5	5.2	16	84C853B

Notes to Table 1a

1. Supply current for f_{XTAL} max.
2. Modified 8048 instruction set due to I²C-bus, CMOS, and derivative functions.
3. Adapter available to convert 3344B piggy-back to 3347 piggy-back.
4. Adapter available to convert 84C430B piggy-back to 84C230 piggy-back.

INSTRUCTION SET	PACKAGES	TEMPERATURE RANGES	FEATURES	A[5]	C[6]	G[7]	T[8]
8048 [2]	DIL-40	F	additional hardware for keyboard with capacitive or mechanical keys			•	
8048 [2]	SDIL-42	D	14-bit PWM output(s) for VST, 3-bit software A/D input for AFC, 5 x 6-bit PWM outputs, OSD 2 lines of 16 characters, 64 character fonts, RC osc.		•		
8048 [2]	SDIL-42	D	14-bit PWM output(s) for VST, 3-bit software A/D input for AFC, 5 x 6-bit PWM outputs, OSD 2 lines of 16 characters, 64 character fonts, LC osc.		•		
8048 [2]	SDIL-42	D	14-bit PWM output(s) for VST, 3-bit software A/D input for AFC, 5 x 6-bit PWM outputs, OSD 2 lines of 16 characters, 64 character fonts, RC osc.		•		
8048 [2]	SDIL-42	D	14-bit PWM output(s) for VST, 3-bit software A/D input for AFC, 5 x 6-bit PWM outputs, OSD 2 lines of 16 characters, 64 character fonts, LC osc.		•		
8048 [2]	SDIL-42	D	14-bit PWM output(s) for VST, 3-bit software A/D input for AFC, 5 x 6-bit PWM outputs, OSD 2 lines of 16 characters, 64 character fonts, RC osc.		•		
8048 [2]	SDIL-42	D	14-bit PWM output(s) for VST, 3-bit software A/D input for AFC, 5 x 6-bit PWM outputs, OSD 2 lines of 16 characters, 64 character fonts, LC osc.		•		
8048 [2]	SDIL-42	D	14-bit PWM output(s) for VST, 3-bit software A/D input for AFC, 5 x 6-bit PWM outputs, OSD 2 lines of 16 characters, 64 character fonts, RC osc.		•		
8048 [2]	SDIL-42	D	14-bit PWM output(s) for VST, 3-bit software A/D input for AFC, 5 x 6-bit PWM outputs, OSD 2 lines of 16 characters, 64 character fonts, LC osc.		•		
8048 [2]	VSO-56	F	64-segment LCD drive (PCA84C633B is a DIL-64 package)	•	•	•	
8048 [2]	DIL-40, VSO-40	F	16-bit timer and 16-bit counter with capture and compare register		•	•	

5. A = Automotive.
6. C = Consumer.
7. G = General/EDP.
8. T = Telecom.
9. In 1991 33XXA and 84CXXXA versions will be available for most devices with an f_{XTAL} max. of 16 MHz.

Table 1b NMOS microcontrollers with an enhanced instruction set based on the 8048

TYPE NUMBER	ROM (bytes)	RAM (bytes)	I/O	I²C	V_{DD} (V)	I_{DD} (MAX.) (mA)[1]	I_{DD} (MAX.) (mA)[2]	f_{XTAL}(MAX.) (MHz)	10 mA DRIVE LINES
8048	1 K	64	25	N	5 ± 10%	80	90	11	-
8049	2 K	128	25	N	5 ± 10%	90	100	11	-
8050	4 K	256	25	N	5 ± 10%	100	120	11	-
8411	1 K	64	20	Y	5 ± 10%	85	100	6	8
8421	2 K	64	20	Y	5 ± 10%	85	100	6	8
8441	4 K	128	20	Y	5 ± 10%	85	100	6	8
8461	6 K	128	20	Y	5 ± 10%	85	100	6	8
8422	2 K	64	13	Y	5 ± 10%	70	90	6	8
8442	4 K	128	13	Y	5 ± 10%	70	90	6	8

Notes to Table 1b

1. Supply current given for f_{XTAL} max. and 5 V supply, 0 to +70 °C.
2. Supply current given for f_{XTAL} max. and 5 V supply, –40 to +85 °C and –40 to +110 °C.
3. Modified 8048 instruction set due to I²C-bus, CMOS, and derivative functions.
4. A = Automotive.
5. C = Consumer.
6. G = General/EDP.
7. T = Telecom.

ROM-LESS VERSION	INSTRUCTION SET	PACKAGES	TEMPERATURE RANGES	FEATURES	A[4]	C[5]	G[6]	T[7]
8035	8048	DIL-40, PLCC-44	V,B and F	industry standard, code-efficient 8048 instruction set			•	
8039	8048	DIL-40, PLCC-44	V,B and F	industry standard, code-efficient 8048 instruction set			•	
8040	8048	DIL-40, PLCC-44	V,B and F	industry standard, code-efficient 8048 instruction set			•	
8401	8048 [3]	DIL-28, SO-28	V,B and F	zero-crossing detector, byte-level I²C-bus interface		•	•	
8401	8048 [3]	DIL-28, SO-28	V,B and F	zero-crossing detector, byte-level I²C-bus interface		•	•	
8401	8048 [3]	DIL-28, SO-28	V,B and F	zero-crossing detector, byte-level I²C-bus interface		•	•	
8401	8048 [3]	DIL-28, SO-28	V,B and F	zero-crossing detector, byte-level I²C-bus interface		•	•	
8041	8048 [3]	DIL-20, PLCC-68	V,B and F	zero-crossing detector, bit-level I²C-bus interface		•	•	
8041	8048 [3]	DIL-20, PLCC-68	V,B and F	zero-crossing detector, bit-level I²C-bus interface		•	•	

Table 2a CMOS 80C51 family and derivatives

GENERIC TYPE NUMBER	ROM (bytes)	RAM (bytes)	EPROM	EEPROM	I/O	I^2C	INTERRUPTS		COUNTER TIMERS	V_{DD} (V)
							INT.	EXT.		
80C51	4 K	128	87C51	-	32	N	3	2	2 x 16-bit	5 ± 20%
80C52	8 K	256	87C52	-	32	N	4	2	3 x 16-bit	5 ± 20%
83C053	8 K	192	-	-	28	N	3	2	2 x 16-bit	5 ± 20%
83C054	16 K	192	87C054	-	28	N	3	2	2 x 16-bit	5 ± 20%
83C451	4 K	128	87C451	-	52/56	N	3	2	2 x 16-bit	5 ± 20%
83C528	32 K	512	87C528	-	32	Y	5	2	3 x 16-bit	5 ± 20%
83C550	4 K	128	87C550	-	32	N	4	2	2 x 16-bit	5 ± 20%
83C552	8 K	256	87C552	-	48	Y	9	6	3 x 16-bit [1]	5 ± 20%
83C562	8 K	256	-	-	48	N	8	6	3 x 16-bit [1]	5 ± 10%
83C652	8 K	256	(87C654)	-	32	Y	4	2	2 x 16-bit	5 ± 10%
83C654	16 K	256	87C654	-	32	Y	4	2	2 x 16-bit	5 ± 20%
83C751	2 K	64	87C751	-	19	Y	3	2	1 x 16-bit	5 ± 20%
83C752	2 K	64	87C752	-	21	Y	4	2	1 x 16-bit	5 ± 20%
83C851	4 K	128	-	256	32	N	3	2	2 x 16-bit	5 ± 10%
83CL410	4 K	128	-	-	32	Y	3	10	2 x 16-bit	2.5 ± 5.5 [3]

Notes to Table 2a

1. One timer (T2) is coupled to 4 capture and 3 compare registers.
2. Supply current given for f_{XTAL} max. and maximum supply voltage.
3. 1.5 V - 5.5 V available Q4 1990.
4. For 30 MHz: 0 - 70°C, 5 V ± 10%, I_{DD}(max) = 44 mA.
5. 16 MHz version in preparation.
6. Quad-flat pack in preparation.
7. A = Automotive.
8. C = Consumer.
9. G = General/EDP.
10. T = Telecom.

I_{DD} (MAX.) (mA)[2]	f_{XTAL} (MAX.) (MHz)	ROM-LESS TYPE	PACKAGES	TEMPERATURE RANGES	FEATURES	A[7]	C[8]	G[9]	T[10]
20	16, 24, 30[4]	80C31	DIL-40, PLCC-44, QFP44	B, F and H	standard 80C51	•		•	
30	16	80C32	DIL-40, PLCC-44	B, F and H	standard 80C52			•	
30	12	n/a	SDIL-42	B and F	on screen display, 9 PWM outputs, 3 software A/D inputs		•		
30	12	n/a	SDIL-42	B and F	as 83C053		•		
25	16	80C451	DIL-64, PLCC-68	B and F	six or seven 8-bit I/O ports			•	
35	16	80C528	DIL-40, PLCC-44	B, F and H	Watch-Dog Timer, 512 bytes of internal RAM, timer T2, bit level I^2C bus	•	•	•	•
35	12, [5]	80C550	DIL-40, PLCC-44	B and F	8 x 8-bit A/D, Watch-Dog Timer, timer T2	•		•	
45	16	80C552	PLCC-68, QPF80	B, F and H	Watch-Dog Timer, 2 PWM outputs, 8-channel multiplexed 10-bit ADC, 16-bit timer with 4 capture and 3 compare registers, byte level I^2C bus	•		•	•
24	16	80C562	PLCC-68, QPF80	B, F and H	Watch-Dog Timer, 2 PWM outputs, 8-channel multiplexed 8-bit ADC, 16-bit timer with 4 capture and 3 compare registers	•		•	
24	12	80C652	DIL-40, PLCC-44, QFP44	B, F and H	byte level I^2C bus	•		•	
27	16	80C654	DIL-40, PLCC-44, QFP44	B, F and H	byte level I^2C bus	•		•	
22	16	N/A	DIL-24, PLCC-28	B, F and H	bit level I^2C bus			•	
22	16	N/A	DIL-24, PLCC-28	B, F and H	bit level I^2C bus			•	
32	12	80C851	DIL-40, PLCC-44, QFP44	B and F	256 bytes EEPROM			•	•
22	20	80CL410	DIL-40, PLCC-44 [6]	B and F	low-voltage low-power 8051 with I^2C-bus and wake-up facility (No UART)			•	•

Table 2b NMOS 8051 family and derivatives

TYPE NUMBER	ROM (bytes)	RAM (bytes)	I/O	UART	INTERRUPTS		COUNTER TIMERS	V_{CC} (V)	I_{CC} (MAX.) (mA)[1]	f_{XTAL} (MAX.) (MHz)
					INT.	EXT.				
8051AH-2	4 K	128	32	Y	3	2	2 x 16-bit	5 ± 10%	125	12
8052AH	8 K	256	32	Y	4	2	3 x 16-bit	5 ± 10%	175	12

Notes to Table 2b

1. Supply current given for f_{XTAL} max. and 5 V supply.
2. A = Automotive.
3. C = Consumer.
4. G = General/EDP.
5. T = Telecom.

Table 3 The CMOS 68000 family and derivatives

TYPE NUMBER	ROM (BYTES)	RAM (BYTES)	EPROM	EEPROM	I^2C	V_{DD} (V)	CLOCK (MAX.) (MHz)	ROM-LESS TYPE	PACKAGES	TEMPERATURE RANGE
68070	-	-	-	-	Y	-	10, 12.5, 15	-	PLCC84, QFP120	B and F
66470	-	-	-	-	-	-	-	-	PLCC84, QFP120	B and F
93C100	34 K	512	97C100	-	Y	5 ± 20%	15	90C100	PLCC84, QFP80	B and F
93C110	34 K	512	-	256	Y	5 ± 20%	15	-	PLCC84, QFP80	B and F

ROM-LESS VERSION	INSTRUCTION SET	PACKAGES	TEMPERATURE RANGES	FEATURES	A[2]	C[3]	G[4]	T[5]
8031AH-2	8051	DIL-40, PLCC-44	V, B and F	standard 8051	•		•	
8032AH	8051	DIL-40, PLCC-44	V, B and F	standard 8052	•		•	

Section 2 - 8048 Family and Derivatives

8048/80C49
Single-chip 8-bit microcontroller family specification

CONTENTS

Single-chip 8-bit microcontroller family specification

8048/80C49

1 INTRODUCTION

This section describes the characteristics of the 8048/80C49 family of microcontrollers in detail. The 8048/80C49 family consists of the devices shown in Table 1.

Table 1 8048/80C49 family

DEVICE	TEMPERATURE RANGE	RAM	ROM	I/O LINES
NMOS				
MAB8048H	0 to + 70 °C	64 x 8	1 K x 8	27
MAF8048H	–40 to + 85 °C	64 x 8	1 K x 8	27
MAF80A48H	–40 to + 110 °C	64 x 8	1 K x 8	27
MAB8035HL	0 to + 70 °C	64 x 8	-	27
MAF8035HL	–40 to + 85 °C	64 x 8	-	27
MAF80A35HL	–40 to + 110 °C	64 x 8	-	27
MAB8049H	0 to + 70 °C	128 x 8	2 K x 8	27
MAF8049H	–40 to + 85 °C	128 x 8	2 K x 8	27
MAF80A49H	–40 to + 110 °C	128 x 8	2 K x 8	27
MAB8039HL	0 to + 70 °C	128 x 8	-	27
MAF8039HL	–40 to + 85 °C	128 x 8	-	27
MAF80A39HL	–40 to + 110 °C	128 x 8	-	27
MAB8050H	0 to + 70 °C	256 x 8	4 K x 8	27
MAF8050H	–40 to + 85 °C	256 x 8	4 K x 8	27
MAF80A50H	–40 to + 110 °C	256 x 8	4 K x 8	27
MAB8040HL	0 to + 70 °C	256 x 8	-	27
MAF8040HL	–40 to + 85 °C	256 x 8	-	27
MAF80A40HL	–40 to + 110 °C	256 x 8	-	27
CMOS				
PCB80C49	0 to + 70 °C	128 x 8	2 K x 8	27
PCF80C49	–40 to + 85 °C	128 x 8	2 K x 8	27
PCA80C49	–40 to + 110 °C	128 x 8	2 K x 8	27
PCB80C39	0 to + 70 °C	128 x 8	-	27
PCF80C39	–40 to + 85 °C	128 x 8	-	27
PCA80C39	–40 to + 110 °C	128 x 8	-	27

All versions are available in a 40-lead DIL plastic package (suffix P); the NMOS versions are also available in a 44-lead PLCC package (suffix WP).

The following generic terms are used throughout this section:

8048/80C49	refers to all family members
8048	refers to the NMOS microcontrollers
80C49	refers to the CMOS microcontrollers.

The 8048/80C49 instructions set enables individual I/O lines to be set and reset directly; individual accumulator bits can be tested. A large variety of branch and table look-up instructions enable efficient implementation of standard logic functions. Code efficiency is high; over 70 % of the instructions are single-byte, and the remainder are two bytes.

The on-chip 8-bit timer/event counter counts either machine cycles ÷32, or external events. The counter can be configured to generate interrupt requests upon overflow.

Program memory, data memory, and I/O can easily be expanded using standard memory and I/O devices.

2 FEATURES

- 8-bit CPU, ROM, RAM, I/O in a single package
- On-chip RAM, ROM, I/O (see Table 1)
- 8-bit timer/event counter
- Internal oscillator circuit
- Single-level vectored interrupts: external and timer/counter
- 17 internal registers: accumulator and 16 addressable registers
- Over 90 instructions: 70 % single-byte; all instructions 1 or 2 cycles
- Easily expandable memory and I/O
- TTL compatible inputs and outputs
- Single 5 V supply
- Reduced power consumption in the Power-down mode

Additional features of the PCB80C39/PCB80C49

- Low power consumption
- Power reduction mode: Idle mode
- Termination of Idle mode by an external or timer/counter interrupt
- Very low power consumption in the Power-down mode (battery back-up)

3 GENERAL DESCRIPTION

A block diagram of the microcontroller is shown in Fig.1. The following sections describe the various functional blocks in detail.

3.1 Arithmetic section

The arithmetic section of the microcontroller contains the basic data manipulation logic and can be divided into the following blocks:

- Accumulator
- Arithmetic Logic Unit (ALU)
- Carry Flag
- Instruction Decoder.

In a typical operation, data stored in the accumulator is combined in the ALU with data from another source on the internal bus (such as a register or I/O port) and the result is stored in the accumulator or another register.

3.1.1 Instruction decoder

The operation code (op code) portion of each instruction is stored in the instruction decoder and converted to outputs which control the function of each of the blocks of the arithmetic section. These lines control the source of data and the destination register as well as the operation performed in the ALU.

3.1.2 Arithmetic logic unit

The ALU accepts 8-bit data bytes from one or two sources and generates an 8-bit result under the control of the instruction decoder. The ALU can perform the following functions:

- Add With or Without Carry
- AND, OR, Exclusive OR
- Increment/Decrement
- Bit Complement
- Rotate Left, Right
- Swap Nibbles
- BCD Decimal Adjust.

If the operation performed by the ALU results in a value represented by more than 8 bits (overflow of most significant bit), a Carry Flag is set in the Program Status Word.

3.1.3 Accumulator

The accumulator is the single most important data register in the microcontroller, being one of the sources of input to the ALU and often the destination of the result of operations performed in the ALU. Also, data to and from I/O ports and memory normally passes through the accumulator.

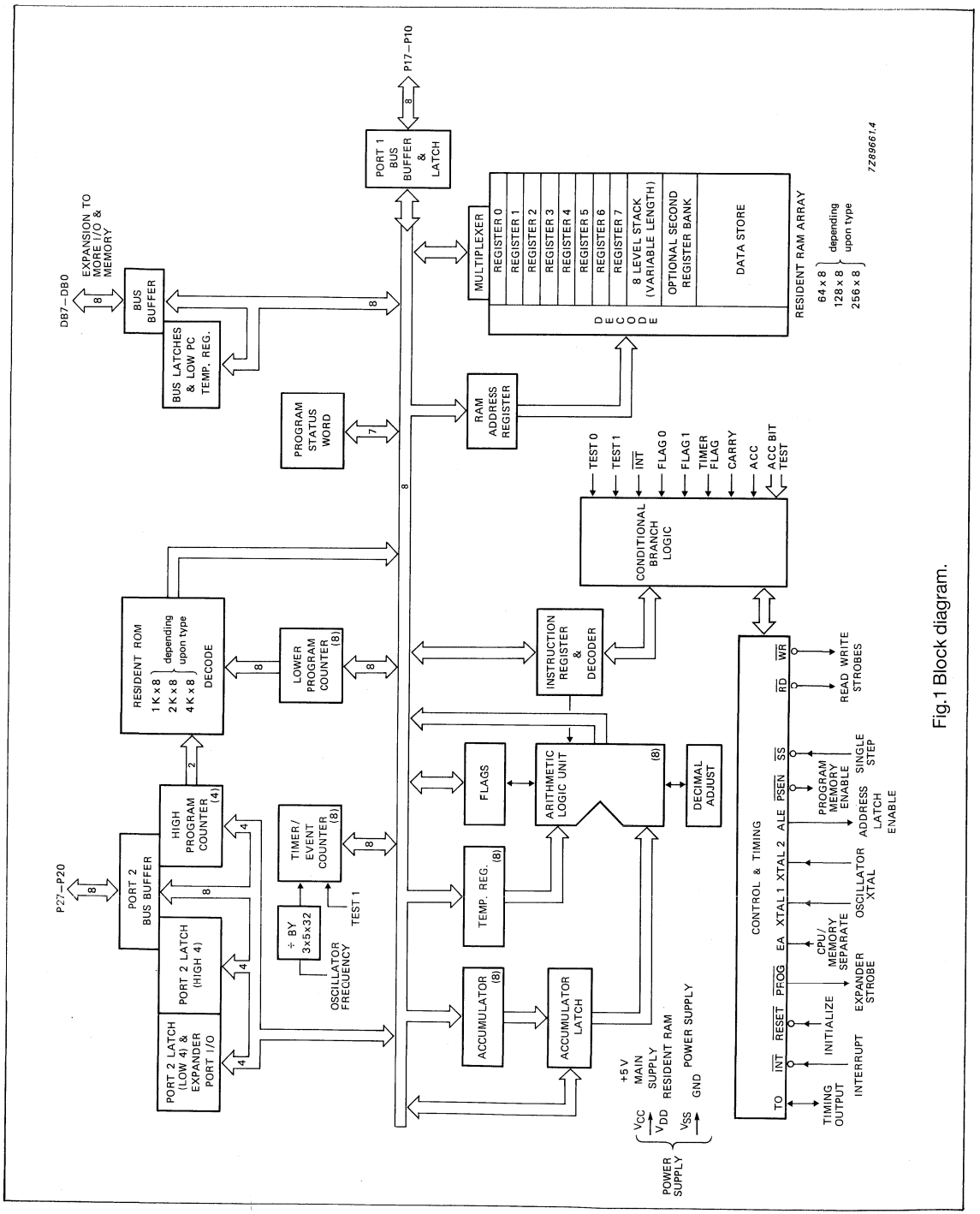


Fig.1 Block diagram.

3.2 Program memory

Members of the 8048/80C49 family contain 1024, 2048 or 4096 bytes of resident program memory which is addressed by the program counter. In the 8048, 8049, 8050 and 80C49, this memory is ROM which is mask programmable at the factory; in the 8035, 8039, 8040 and 80C39, there is no internal program memory and these devices must be used with external memory. Program code is completely interchangeable among the various versions.

The program memory address space is divided into two 2048-byte memory banks: MB0 and MB1 (see Fig.2). Each memory bank is further subdivided into pages of 256 bytes; this latter subdivision applies only to conditional branches. To access the other 2 K bytes in devices with 4 K bytes of program memory, a select memory bank (SEL MB) instruction followed by a JUMP or CALL instruction must be executed to cross the 2 K boundary.

There are three program memory locations of special importance:

Location 0 activating the $\overline{\text{RESET}}$ input of the microcontroller causes the first instruction to be fetched from location 0.

Location 3 activating the interrupt input ($\overline{\text{INT}}$) of the microcontroller causes a jump to subroutine at location 3 (if the interrupt is enabled).

Location 7 a timer/counter interrupt resulting from timer counter overflow causes a jump to subroutine at location 7 (if the interrupt is enabled).

Therefore, the first instruction to be executed after initialization is stored in location 0, the first byte of an external interrupt service routine is stored in location 3, and the first word of a timer/counter service routine is stored in location 7. Program memory can be used to store constants as well as executable code. Instructions such as MOVP and MOVP3 allow easy access to data 'look-up' tables.

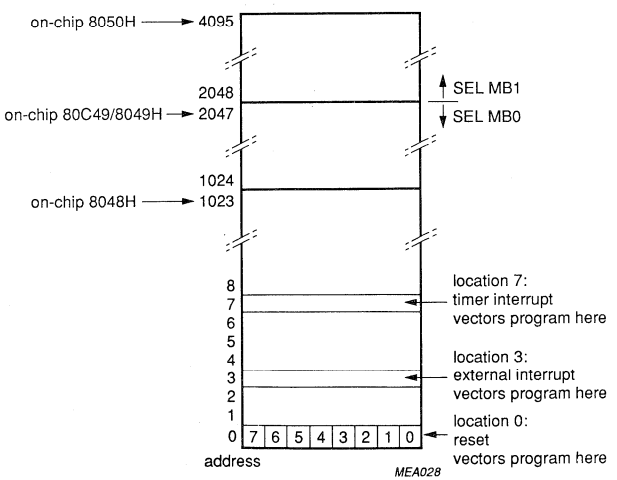


Fig.2 Program memory map.

3.3 Data memory

Resident data memory (RAM) is organized as 64, 128, or 256 x 8-bit bytes in the 8048, 8049/80C49, and 8050 respectively. All locations are indirectly addressable through either of two RAM Pointer Registers which reside at locations 0 and 1 of the register array. In addition, the first 8 locations (0-7) of the array are designated as working registers and are directly addressable by several instructions (see Fig.3). Since these registers are more easily addressed, they are usually used to store frequently accessed intermediate results. The DJNZ instruction makes very efficient use of the working registers as program loop counters by allowing the programmer to decrement and test the register in a single instruction.

By executing a Register Bank Switch instruction (SEL RB), RAM locations 24-31 are designated as the working registers in place of locations 0-7 and are then directly addressable. This second bank of working registers may be used as an extension of the first bank or reserved for use during interrupt service routines allowing the registers of Bank 0 used in the main program to be instantly 'saved' by a Bank Switch. Note that if this second bank is not used, locations 24-31 are still addressable as general purpose RAM. Since the two RAM Pointer Registers (R0 and R1) are part of the working register array, bank switching effectively creates two more pointer registers (R0' and R1') which can be used with R0 and R1 to easily access up to four separate working areas in RAM at one time.

RAM locations (8-23) also serve a dual role in that they contain the program counter stack. These locations are addressed by the Stack Pointer. Two locations (bytes) are used per CALL, allowing up to eight levels of subroutine nesting. If the level of subroutine nesting is less than 8, unused stack registers may be used as general purpose RAM locations. Each level of subroutine nesting not used provides the user with two additional RAM locations.

If additional RAM is required, up to 256 bytes may be added externally and indirectly addressed using the MOVX A,@Rr and MOVX @Rr,A instructions. If more RAM (more than 256 bytes extra) is required, I/O port lines may be used to select banks (256 bytes) of external memory.

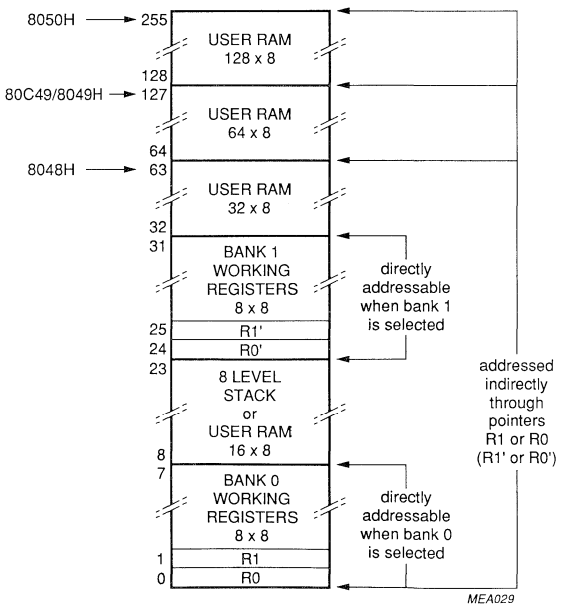


Fig.3 Data memory map.

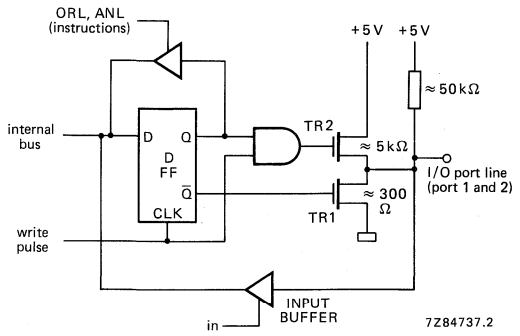


Fig.4. Quasi-bidirectional port structure of the 8048.

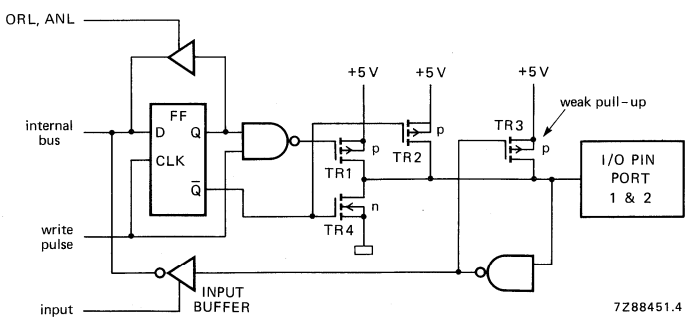


Fig.5 Quasi-bidirectional port structure of the 80C49.

3.4 Input/Output

Members of the 8048/80C49 family have 27 I/O port lines. These lines are arranged as:

- three 8-bit ports; each port line may be configured as an input, output or bidirectional port.
- three 'test' inputs which can alter program sequences when tested by conditional jump instructions.

3.4.1 PORTS 1 AND 2

Ports 1 and 2 are each 8-bits wide and have identical characteristics. Data written to these ports is latched and remains unchanged until rewritten. As input ports these lines are non-latching, i.e. input data must be present until read by an input instruction. Inputs are fully TTL compatible and outputs will drive one standard TTL load.

These ports are called quasi-bidirectional because of a special output circuit configuration which allows each port line to serve as an input, and output, or both even though outputs are latched. Figure 4 shows the circuit configuration of the 8048 in detail. Each port line is pulled up to V_{CC} through a resistive device of relatively high impedance (50 kΩ).

This pull-up is sufficient to provide the source current for a TTL HIGH level, yet can be pulled LOW by a standard TTL gate thus allowing the same pin to be used for both input and output.

When a logic 1 is written to a port, TR1 is turned off and, to provide fast switching during a '0' to '1' transition, TR2 is temporarily switched on for the duration of the internal write pulse (1/5 of a machine cycle), driving the output rapidly to V_{CC} (see Fig.4). When a logic 0 is written to the port, TR1 is turned on and provides TTL current sinking capability.

The port structure of the 80C49 is shown in Fig.5. Each port line has a high impedance pull-up transistor (TR3) which is turned on when the port line is pulled above 2 V by an external source or by writing a logic 1 to the port. This pull-up is sufficient to provide the source current for a TTL HIGH level, but can be pulled LOW by a standard TTL gate enabling the port to be used for both input and output. When a logic 1 is written to the port, a second high impedance pull-up transistor (TR2) pulls the port line HIGH and, to provide fast switching during a '0' to '1' transition, a relatively low impedance transistor (TR1) is temporarily switched on for the duration of the internal write pulse (1/5 of a machine cycle), driving the output rapidly to V_{DD}. When a logic 0 is written to the port, a low impedance transistor (TR4) is turned on and provides TTL current sinking capability.

Since the pull-down transistor is a low impedance device, a logic 1 must first be written to any port which is to be used as an input. Reset initializes all port lines to the high impedance (logic 1) state.

It is important to note that the ORL and ANL instructions are read/write operations. When executed, the microcontroller 'reads' the port, modifies the data, and then 'writes' the data back to the port. The 'writing' operation (essentially an OUTL instruction) momentarily enables the low impedance pull-up transistor again even if the data was unchanged from a logic 1. This specifically applies to configurations that have inputs and outputs mixed together on the same port.

3.4.2 BUS

BUS is a true bidirectional port with associated input and output strobes.

If the bidirectional feature is not needed, BUS can serve as either a latched output port or a non-latching input port. Input and output lines on this port can not be mixed. Data is written and latched using the OUTL instruction, and read (input) using the INS instruction. The INS and OUTL instructions generate pulses on the corresponding $\overline{RD}$ and $\overline{WR}$ output strobe lines; however, in this mode they are generally not used.

When operating as a bidirectional port, the MOVX instructions are used to read from and write to the port. A write to the port generates a pulse on the $\overline{WR}$ output line and output data is valid at the falling edge of $\overline{WR}$. When the port is read, a pulse is generated on the $\overline{RD}$ output line and input data must be valid at the falling edge of $\overline{RD}$. When not being written or read, the BUS lines are in a high impedance state.

3.5 Test (T0, T1) and $\overline{INT}$ inputs

These pins may be used as inputs and are testable with the conditional jump instruction, enabling inputs to cause program branches without the necessity to read an input port into the accumulator. The T0, T1 and $\overline{INT}$ pins may also be used for other purposes (see the following sections).

3.6 Program status word

The program status word (PSW) is an 8-bit byte which can be loaded to and from the accumulator. Figure 6 shows the information available in the PSW. The PSW is actually a collection of flip-flops throughout the machine which can be read or written as a whole. The ability to write to the PSW permits easy restoration of machine status after a power-down sequence.

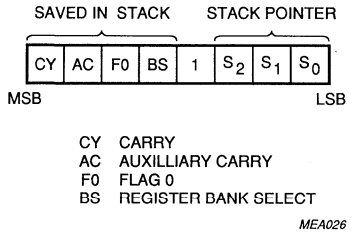


Fig.6 Program status word.

During an interrupt or CALL, the upper four bits of PSW are stored in the program counter stack and are restored upon return with the RETR instruction. The RET return instruction does not restore the PSW.

The PSW bit definitions are as follows:

Bits 0-2 stack pointer bits (S_0, S_1, S_2)

Bit 3 not used (logic 1 when read)

Bit 4 working register bank switch bit (BS):
0 = Bank 0
1 = Bank 1

Bit 5 Flag 0 bit (F0); user controlled flag which can be complemented or cleared, and tested with the conditional jump instruction JF0.

Bit 6 Auxiliary Carry (AC); carry bit affected by the ADD instruction and used by the decimal adjust instruction DA A.

Bit 7 Carry (CY); carry flag which indicates that the previous operation has resulted in an overflow of the accumulator.

3.7 Program Counter and Stack

The program counter (PC) is a 12-bit counter/register that points to the location from which the next instruction is to be fetched (see Fig.7). When the EA input is LOW, the PC addresses locations 0 to 1023 of internal program memory; at the 1 K boundary, an automatic switch-over to external memory is made (for the 8049/80C49, this occurs at the 2 K boundary; for the 8050, there is no automatic switch-over to external program memory). When the EA input is HIGH, all instruction fetches are from external memory. The total address range is 4 K bytes. The PC is initialized to zero by activating the Reset line.

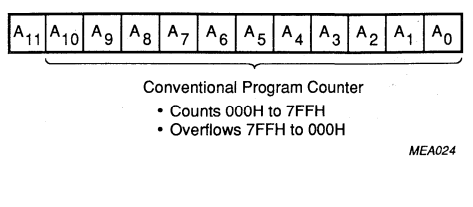


Fig.7 Program counter.

An interrupt or CALL to a subroutine causes the contents of the program counter to be stored in one of the 8 register pairs of the program counter stack (see Fig.8). The register pair to be used is determined by a 3-bit stack pointer which is part of the PSW.

Data RAM locations 8-23 are available as stack registers and are used to store the program counter and 4 bits of the PSW. When initialized to 000, the stack pointer points to RAM locations 8 and 9. The first subroutine jump or interrupt results in the program counter contents being transferred to locations 8 and 9 of the RAM array. The stack pointer is then incremented by one to point to locations 10 and 11 in anticipation of another CALL.

Nesting of subroutines within subroutines can continue up to 8 times without overflowing the stack. If overflow does occur the deepest address stored (locations 8 and 9) will be overwritten and lost since the stack pointer overflows from 111 to 000; the stack pointer also underflows from 000 to 111.

The end of a subroutine, which is signalled by a return instruction (RET or RETR), causes the stack pointer to be decremented and the contents of the resulting register pair to be transferred to the program counter.

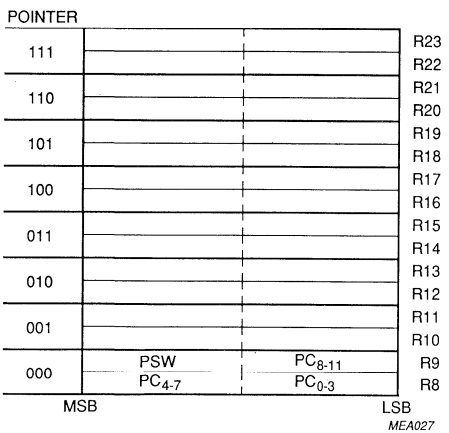


Fig.8 Program counter stack.

3.8 Central processing unit

The CPU can perform the following functions:

- ADD with or without carry
- Logical AND, OR, Exclusive OR
- Increment, decrement
- Bit complement
- Rotate left, rotate right
- Swap nibbles
- BCD decimal adjust.

If the operation performed by the arithmetic logic unit (ALU) results in value represented by more than 8 bits (8-bit value plus overflow), the Carry flag in set in the PSW.

Conditional branch logic enables several conditions (internal and external) to be tested by the user's program. By using the conditional jump instruction the conditions that are listed in Table 2 can effect a change in the program execution sequence.

Table 2 Jump conditions

TEST	JUMP ON	
Accumulator	All zeros	Not all zeros
Accumulator Bit	-	1
Carry Flag	0	1
User Flags (F0, F1)	-	1
Timer Overflow Flag	-	1
Test Inputs (T0, T1)	0	1
Interrupt Input ($\overline{INT}$)	0	-

3.9 Interrupts

Figure 9 shows the interrupt logic. An interrupt sequence may be initiated by:

- applying a LOW level to the $\overline{INT}$ pin, or
- a counter overflow.

Before an interrupt is serviced, the microcontroller completes execution of the current instruction and then performs a CALL to the interrupt service routine. The RETR instruction restores the microcontroller to the state it was in before the interrupt occurred. The external interrupt ($\overline{INT}$) has priority over the internal (counter) interrupt.

$\overline{INT}$ is level triggered and active LOW to permit 'WIRED ORing' of several interrupt sources to the input pin. $\overline{INT}$ is sampled every instruction cycle; when a LOW level is detected, a 'jump to subroutine' at location 3 in program memory is performed as soon as all cycles of the current instruction are complete. For 2-cycle instructions, $\overline{INT}$ is sampled during the latter cycle only. $\overline{INT}$ must remain LOW for at least 3 machine cycles to ensure correct interrupt operation. As for any CALL to subroutine, the program counter and program status word are saved in the stack. Program memory location 3 usually contains an unconditional jump to an interrupt service subroutine elsewhere in program memory.

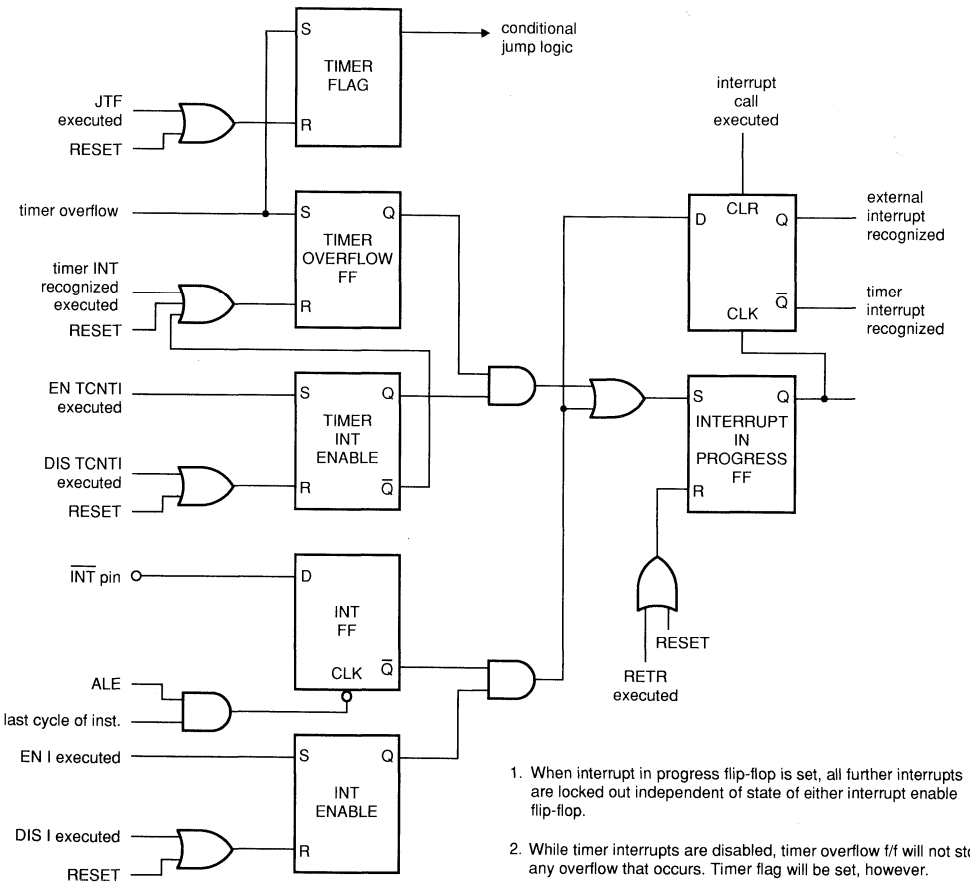
The end of an interrupt service subroutine is signalled by the execution of the Return and Restore Status instruction (RETR). The interrupt structure is single level; once an interrupt is detected, all further interrupt requests are ignored until execution of an RETR re-enables the

interrupt input logic. The interrupt logic is re-enabled at the beginning of the second cycle of the RETR instruction. This sequence is identical for an internal interrupt generated by timer overflow. If an internal timer/counter generated interrupt and an external interrupt are detected at the same time, the external source interrupt will be serviced. A second external interrupt can be simulated by enabling the timer/counter interrupt, loading FFH in the Counter (one less than the terminal count), and enabling the event counter mode. A '1' to '0' transition on the T1 input will then cause an interrupt vector to location 7.

3.9.1 Interrupt timing

The EN I and DIS I instructions may be used to enable/disable the external interrupt ($\overline{INT}$). Interrupts are disabled by RESET and remain disabled until enabled by software.

An interrupt request must be removed before the RETR instruction is executed to return from the service routine; otherwise the microcontroller will immediately re-service the interrupt request. Many peripheral devices prevent this situation by resetting the interrupt request line when the microcontroller accesses (reads or writes) the peripheral's data buffer register. If the interrupting device does not require access by the microcontroller, an output port line may be designated as an 'interrupt acknowledge' which is activated by the interrupt service subroutine to reset the interrupt request. The $\overline{INT}$ pin may also be tested using the conditional jump instruction JNI. This instruction may be used to detect the presence of a pending interrupt before interrupts are enabled. If interrupt is left disabled, $\overline{INT}$ may be used as another test input like T0 and T1.



1. When interrupt in progress flip-flop is set, all further interrupts are locked out independent of state of either interrupt enable flip-flop.
2. While timer interrupts are disabled, timer overflow f/f will not store any overflow that occurs. Timer flag will be set, however.

MEA016

Fig.9 Interrupt logic.

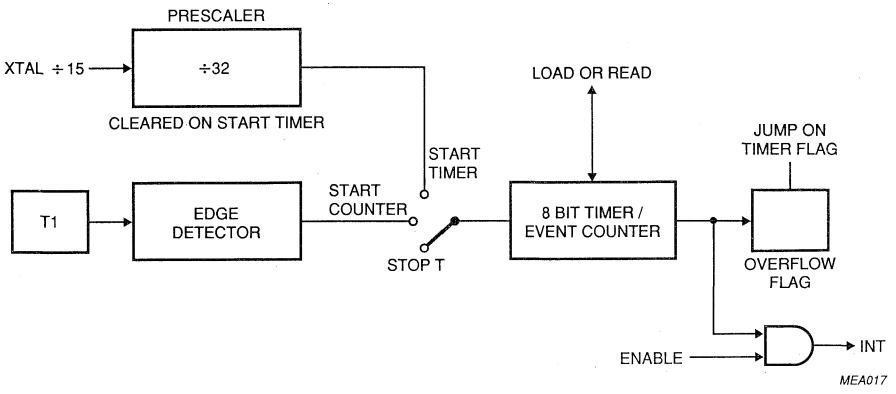


Fig.10 Timer/event counter.

3.10 Timer/event counter

The on-chip counter (see Fig.10) may be used to:
- count external events, or
- generate accurate time delays

freeing the CPU for other tasks (useful in real-time applications). In both modes the counter operation is the same, the only difference is the counter's input source.

3.10.1 Counter operation

The 8-bit binary counter is presettable and readable with two MOV instructions which transfer the contents of the accumulator to the counter and vice versa. The contents of the counter may be affected by RESET and should be initialized by software. The counter is stopped by a RESET or STOP TCNT instruction and remains stopped until started:
- as a timer by a START T instruction, or
- as an event counter by a START CNT instruction.

Once started the counter will increment to the maximum count (FF) and overflow to zero, continuing its count until stopped by a STOP TCNT instruction or RESET.

The increment from maximum count to zero (overflow) sets an overflow flag flip-flop and generates an interrupt request. The state of the overflow flag is testable with the conditional jump instruction JTF. The flag is reset by executing a JTF of by RESET. The interrupt request is stored in a latch and then ORed with the external interrupt input. The timer interrupt may be enabled or disabled independently of the external interrupt by the EN TCNTI and DIS TCNTI instructions. If enabled, the counter overflow will cause a subroutine call to location 7 where the timer or counter interrupt service routine may be located.

If both timer and external interrupts occur simultaneously, the external interrupt will be serviced and the CALL will be to location 3. Since the timer interrupt is latched, it will remain pending until the external interrupt is serviced and will immediately be recognized upon return from the external interrupt service routine. The pending timer interrupt is reset by the CALL to location 7 or may be removed by executing a DIS TCNTI instruction.

3.10.2 EVENT COUNTER OPERATION

Execution of a START CNT instruction connects the T1 input pin to the counter input and enables the counter. Subsequent HIGH-to-LOW transitions on T1 will cause the counter to increment. The maximum rate at which the counter may be incremented is once per three instruction cycles; there is no minimum frequency. T1 must be held LOW for at least 1 machine cycle to ensure that the LOW state is detected, and T1 input must remain HIGH for at least 1/5 of a machine cycle after each LOW-to-HIGH transition.

3.10.3 TIMER OPERATION

Execution of a START T instruction connects the internal clock to the counter input and enables the counter. The internal clock is derived by gating the machine cycle clock through a ÷32 prescaler. The prescaler is reset during the START T instruction. The prescaler output increments the counter every 32 machine cycles. Various delays from 1 to 256 counts can be obtained by presetting the counter and detecting overflow. Times longer than 256 counts may be achieved by accumulating multiple overflows in a register under software control. For time resolution of less than 1 count, an external clock should be applied to the T1 input and the counter should be configured in the event counter mode. ALE divided by 3 or more can serve as this external clock. Very small delays or 'fine tuning' of larger delays can easily be accomplished by software delay loops.

A serial link is often desirable in 8048 applications. Table 3 lists the timer counts and cycles needed for specific baud rates at various crystal frequencies.

3.11 Clock and timing circuits

All 8048 timing signals are generated on-chip; the only requirement is an external frequency reference which can be a crystal, a ceramic resonator, or an external clock source.

3.11.1 OSCILLATOR

The on-chip oscillator is a high gain parallel resonant circuit. The XTAL1 pin is the input to the amplifier stage; XTAL2 is the output. A crystal may be connected between XTAL1 and XTAL2 to provide the feedback and phase shift required for oscillation. If an accurate frequency reference is not required, a ceramic resonator may be used in place of the crystal. An externally generated clock may also be applied to XTAL1-XTAL2 to provide the frequency reference.

Table 3(a) Baud rate generation

FREQUENCY (MHz)	T_{cy} (µs)	T0 Prr ($1/5T_{cy}$) (ns)	TIMER PRESCALER ($32T_{cy}$) (µs)
4	3.75	750	120
6	2.50	500	80
8	1.88	375	60.2
11	1.36	275	43.5

Table 3(b) Baud Rate Generation

BAUD RATE	TIMER COUNTS + INSTRUCTION CYCLES AT VARIOUS OSCILLATOR FREQUENCIES			
	4 MHZ	6 MHz	8 MHz	11 MHz
110	75 + 24 Cycles .01% Error	113 + 20 Cycles .01% Error	151 + 3 Cycles .01% Error	208 + 28 Cycles .01% Error
300	27 + 24 Cycles .1% Error	41 + 21 Cycles .03% Error	55 + 13 Cycles .01% Error	76 + 18 Cycles .04% Error
1200	6 + 30 Cycles .1% Error	10 + 13 Cycles .1% Error	12 + 27 Cycles .06% Error	19 + 4 Cycles .12% Error
1800	4 + 20 Cycles .1% Error	6 + 30 Cycles .1% Error	9 + 7 Cycles .17% Error	12 + 24 Cycles .12% Error
2400	3 + 15 Cycles .1% Error	5 + 6 Cycles .4% Error	6 + 18 Cycles .12% Error	9 + 18 Cycles .12% Error
4800	1 + 23 Cycles 1.0% Error	2 + 19 Cycles .4% Error	3 + 14 Cycles .74% Error	4 + 25 Cycles .12% Error

3.11.2 STATE COUNTER

The output of the oscillator is divided by 3 in the State Counter to create a clock (CLK) which defines the state times of the machine (see Fig.11). CLK can be made available at an external pin (T0) by executing an ENT0 CLK instruction. The output of CLK on T0 is disabled by RESET.

3.11.3 CYCLE COUNTER

CLK is then divided by 5 by the cycle counter to provide a clock which defines a machine cycle consisting of 5 machine states (see Fig.11). This clock is called Address Latch Enable (ALE) and is always available on the ALE output pin. External memory/peripheral address data is valid and may be latched on the falling edge of ALE.

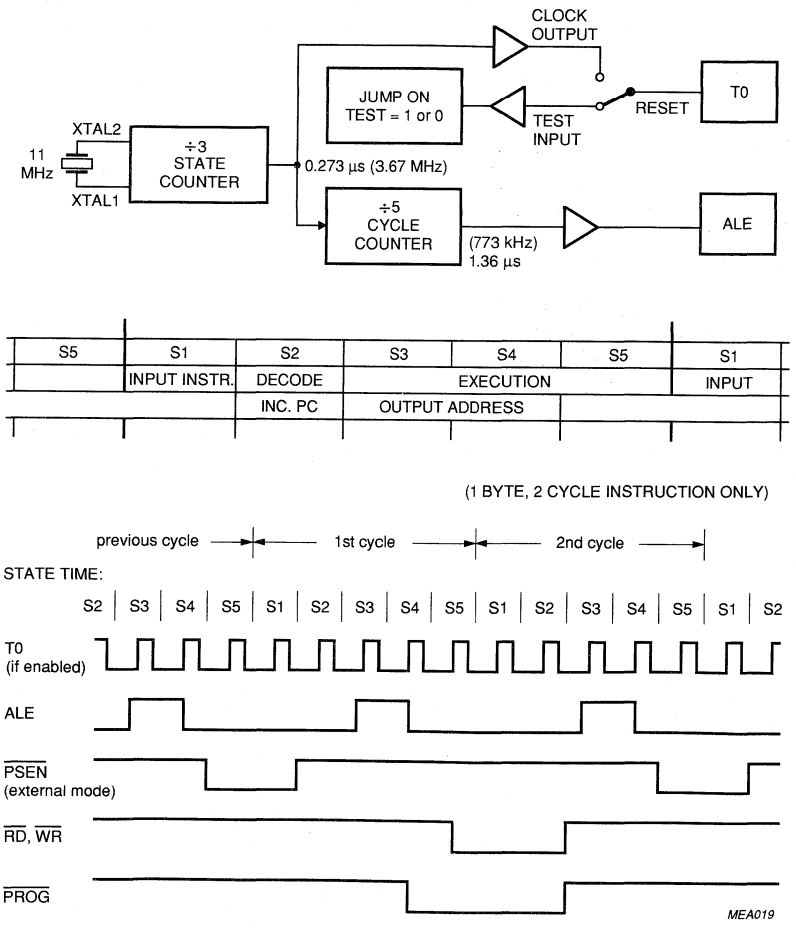


Fig.11 Clock generation and cycle timing.

3.12 The $\overline{\text{RESET}}$ input

The $\overline{\text{RESET}}$ input is used to initialize the microcontroller. This Schmitt-trigger input has an internal pull-up device which, in combination with an external 1.0 μF capacitor, provides an internal reset pulse of sufficient duration to reset the microcontroller (see Fig.12). If the reset pulse is generated externally, the $\overline{\text{RESET}}$ pin must be held LOW for at least 10 milliseconds after V_{CC} has reached the specified operating range. Only 5 machine cycles are required if V_{CC} has already reached the specified operating range and the oscillator has stabilized.

RESET performs the following functions:

- the program counter is reset to zero
- the stack pointer is reset to zero
- register bank 0 is selected
- memory bank 0 is selected
- BUS is placed in the high impedance state (except when EA is HIGH)
- Ports 1 and 2 are placed in the input mode
- the interrupts (timer and external) are disabled
- the timer is stopped
- the timer flag is cleared
- flags F0 and F1 are cleared
- the clock output from T0 is disabled.

3.13 Single step input ($\overline{\text{SS}}$)

This feature provides the user with the debug capability to step through the program one instruction at a time. While stopped, the address of the next instruction to be fetched is available on BUS (8 LSBs of the address) and the lower half of Port 2 (the 4 MSBs of the address are available on P20-P23). The user can therefore follow the program through each of the instruction steps. A timing diagram, showing the relationship between ALE the single step input ($\overline{\text{SS}}$), is shown in Fig.14. The BUS buffer contents are lost during single step; however, a latch may be added to restore the lost I/O capability if needed. Data is valid on the leading edge of ALE.

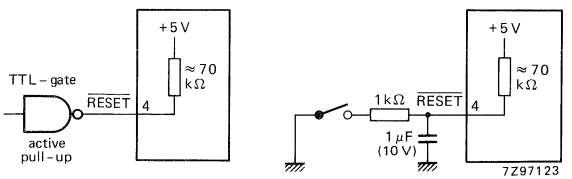


Fig.12 Reset circuitry.

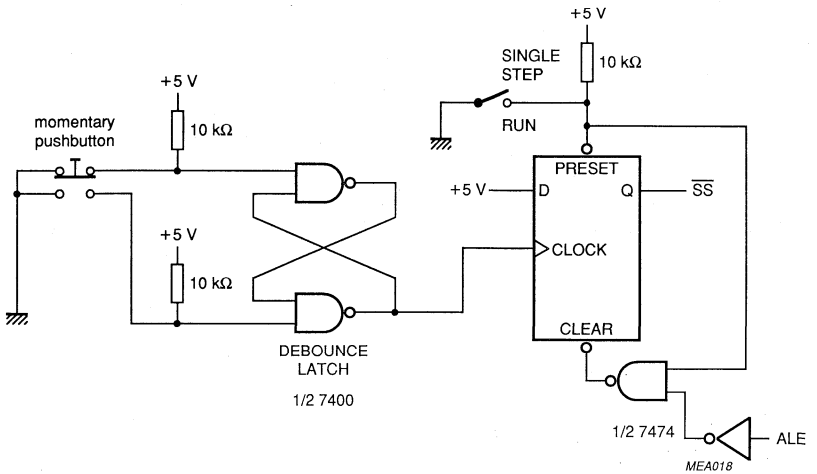


Fig.13 Single step circuit.

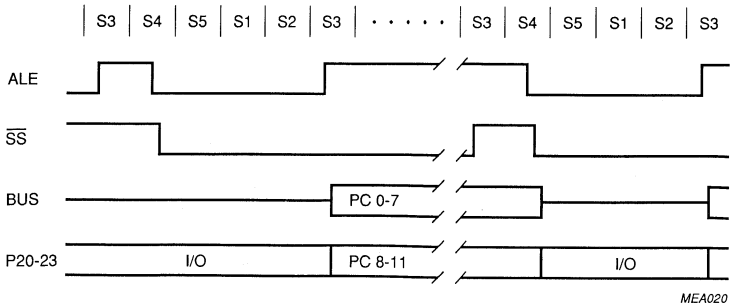


Fig.14 Single step timing.

3.13.1 SINGLE STEP TIMING

Single step operation is as follows:

1. The microcontroller is requested to stop by applying a LOW level on $\overline{SS}$.
2. The microcontroller responds by stopping during the address fetch portion of the next instruction. If a double cycle instruction is in progress when the single step command is received, both cycles will be completed before stopping.
3. Entering the stopped state is acknowledged by a HIGH level on ALE. In this state (which can be maintained indefinitely), the address of the next instruction to be fetched is present on BUS and the lower half of port 2.
4. $\overline{SS}$ goes HIGH to bring the microcontroller out of the stopped mode, allowing it to fetch the next instruction. Leaving the stopped mode is acknowledged a LOW level on ALE.
5. To stop the microcontroller at the next instruction, $\overline{SS}$ must be brought LOW again soon after ALE goes LOW. If $\overline{SS}$ remains HIGH, the microcontroller remains in the run mode.

A diagram for implementing single step operation is shown in Fig.13. A D-type flip-flop with preset and clear is used to generate the input to $\overline{SS}$. In the run mode $\overline{SS}$ is held HIGH by keeping the flip-flop preset (preset has precedence over the clear input). To initiate single step operation, preset is removed allowing ALE to bring $\overline{SS}$ LOW via the clear input. ALE may need to be buffered since the input impedance of the clear input on some TTL flip-flops is low. The microcontroller is now in the stopped state. The next instruction is initiated by clocking a logic 1 through the flip-flop. This logic 1 will not appear on the $\overline{SS}$ input unless ALE is HIGH, removing clear from the flip-flop. In response to $\overline{SS}$ going HIGH the microcontroller begins an instruction fetch which brings ALE LOW resetting $\overline{SS}$ through the clear input and causing the microcontroller to re-enter the stopped state.

3.14 External access mode

In normal operation the first 1 K (8048), 2 K (8049/80C49), or 4 K (8050) bytes of program memory are automatically fetched from internal ROM or EPROM. The EA input pin however enables the user to disable internal program memory by forcing all program memory fetches from external memory. The external access mode is useful for system testing and debugging because it allows the user to disable the internal applications program and substitute an external program of the user's choice - a diagnostic routine for instance. In addition, the data sheet explains how internal program memory can be read externally. A HIGH level on the EA pin initiates the external access mode. For proper operation, RESET should be applied after the EA input has changed.

3.15 Power-down mode

Power can be removed from all but the data RAM array for low power standby operation. In the power-down mode the contents of the data RAM can be maintained using typically 10% to 15% of the normal operating power of the 8048; for the 80C49, supply current in the power-down mode is only a few µA.

V_{CC} serves as the 5 V supply pin for the bulk of circuitry; the V_{DD} pin supplies only the RAM array. In normal operation both pins are at 5 V; in the power-down mode, V_{CC} is at ground and V_{DD} is maintained at its standby level. Resetting the microcontroller by activating the $\overline{RESET}$ pin inhibits any access to the RAM and guarantees that RAM cannot be inadvertently altered as power is removed from V_{CC}.

A typical power-down sequence (see Fig.15) occurs as follows:

- imminent power supply failure is detected by user circuitry. The signal must be generated in time to enable the microcontroller to save all necessary data before V_{CC} falls below normal operating limits.
- the power fail signal is used to interrupt the microcontroller; a power fail routine is located at the interrupt vector.
- the power fail routine saves all important data and machine status in the internal data RAM array. The routine may also initiate transfer of backup supply to the V_{DD} pin and indicate to external circuitry that the power fail routine is complete.
- the $\overline{RESET}$ pin is activated to ensure that RAM data will not be corrupted as the voltage on V_{CC} falls. The $\overline{RESET}$ pin must be held LOW until V_{CC} is at ground.

Recovery from the power-down mode can occur as for any other power-on sequence with an external capacitor on the $\overline{RESET}$ input providing the necessary delay.

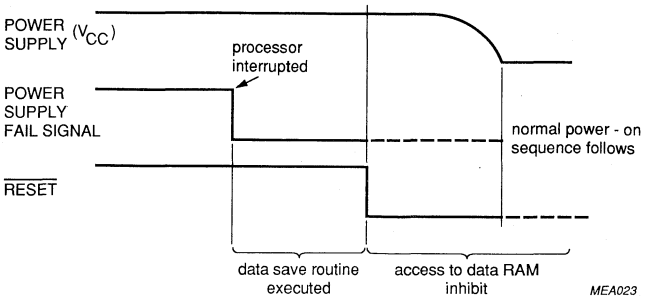


Fig.15 Power-down sequence.

3.16 Idle mode (80C39, 80C49 only)

In the Idle mode the oscillator, internal timer, and the external interrupt and counter pins continue functioning normally. The status of the:
- RAM
- registers
- port 1 and port 2
- BUS

is also maintained.

Executing the IDL instruction places the microcontroller in the Idle mode. Normal operation can be resumed after an enabled interrupt from one of two interrupt sources:
- an external signal applied to the $\overline{INT}$ pin, or
- a timer/counter overflow.

When one of these interrupts occurs, the Idle mode is terminated and program execution resumes at the interrupt vector. Resetting the microcontroller also terminates the Idle mode.

4 INSTRUCTION SET

The instruction set consists of over 90 one and two byte instructions. Program code efficiency is high because all RAM locations and all ROM locations on a 256 byte page require only a single byte address.

Figure 16 shows the instruction map. For more details on the instruction set, see the 'THE 8048 BASED INSTRUCTION SET' chapter.

first hexadecimal character of opcode

second hexadecimal character of opcode

	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
0	NOP	IDLE	OUTL BUS,A	ADD A, #data	JMP page 0	EN I		DEC A	INS A, BUS	IN A,Pp 1	2		4	MOVD A,Pp 5	6	7
1	INC @ Rr 0	1	JB0 addr	ADDC A,#data	CALL page 0	DIS I	JTF addr	INC A	0	1	2	3	INC Rr 4	5	6	7
2	XCH A, @Rr 0	1		MOV A, #data	JMP page 1	EN TCNTI	JNT0 addr	CLR A	0	1	2	3	XCH A,Rr 4	5	6	7
3	XCHD A, @Rr 0	1	JB1 addr		CALL page 1	DIS TCNTI	JT0 addr	CPL A	0	OUTL Pp,A 1	2		4	MOVD Pp,A 5	6	7
4	ORL A, @Rr 0	1	MOV A, T	ORL A, #data	JMP page 2	STRT CNT	JNT1 addr	SWAP A	0	1	2	3	ORL A,Rr 4	5	6	7
5	ANL A, @Rr 0	1	JB2 addr	ANL A, #data	CALL page 2	STRT T	JT1 addr	DA, A	0	1	2	3	ANL A,Rr 4	5	6	7
6	ADD A, @Rr 0	1	MOV T, A		JMP page 3	STOP TCNT		RRC A	0	1	2	3	ADD A,Rr 4	5	6	7
7	ADDC A, @Rr 0	1	JB3 addr		CALL page 3	ENTO CLK	JF1 addr	RR A	0	1	2	3	ADDC A,Rr 4	5	6	7
8	MOVX A, @ Rr 0	1		RET	JMP page 4	CLR F0	JNI addr		ORL BUS, #data	ORL Pp,#data 1	2		4	ORLD Pp,A 5	6	7
9	MOVX @ Rr,A 0	1	JB4 addr	RETR	CALL page 4	CPL F0	JNZ addr	CLR C	ANL BUS #data	ANP Pp,#data 1	2		4	ANLD Pp,A 5	6	7
A	MOV @ Rr,A 0	1		MOVP A,@A	JMP page 5	CLR F1		CPL C	0	1	2	3	MOV Rr,A 4	5	6	7
B	MOV @Rr, #data 0	1	JB5 addr	JMPP @A	CALL page 5	CPL F1	JFO addr		0	1	2	3	MOV Rr,#data 4	5	6	7
C					JMP page 6	SEL RB0	JZ addr	MOV A,PSW	0	1	2	3	DEC Rr 4	5	6	7
D	XRL A, @Rr 0	1	JB6 addr	XRL A,#data	CALL page 6	SEL RB1		MOV PSW,A	0	1	2	3	XRL A,Rr 4	5	6	7
E				MOVP3 A,@A	JMP page 7	SEL MB0	JNC addr	RL A	0	1	2	3	DJNZ Rr,addr 4	5	6	7
F	MOV A, @Rr 0	1	JB7 addr		CALL page 7	SEL MB1	JC addr	RLC A	0	1	2	3	MOV A,Rr 4	5	6	7

MBA365

Fig.16 Instruction map.

5 EXPANDED MICROCONTROLLER SYSTEM

5.1 Introduction

If the capabilities resident on-chip are not sufficient for an application, special on-chip circuitry allows the addition of a wide variety of external memory, I/O, and other peripheral devices. The microcontrollers can be easily expanded as follows:

- program memory: up to 4 K bytes (internal + external)
- data memory: up to 256 bytes additional external data memory
- I/O and peripherals: virtually unlimited.

By using bank switching techniques, maximum use is made of the addressing range. Bank switching is discussed later in this chapter. Expansion is accomplished in two ways:

1. expander I/O: a special I/O expander IC (8243) provides additional four 4-bit I/O ports at the expense of the lower half (4 bits) of port 2 which are used for inter-device communication. Multiple expander ICs may be attached to this 4-bit bus if the required 'chip select' lines are provided.
2. standard bus: one port may be used as an 8-bit bidirectional data bus, allowing interface to numerous standard memories and peripherals.

Systems can be configured using either or both of these expansion features to optimize system resources for an application. Virtually any number and combination of expander ICs, standard memories and peripherals can be added as required.

5.2 Expansion of program memory

Program memory may be expanded beyond the resident 1 K or 2 K bytes using the BUS feature. All program memory fetches from addresses less than 1024 on the 8048 (less than 2048 on the 8049) occur internally; no external signals are generated (except ALE which is always present). At higher addresses, the microcontroller automatically initiates external program memory fetches.

5.2.1 INSTRUCTION FETCH CYCLE (EXTERNAL)

During all instruction fetches from higher addresses (see Fig.17), the following will occur:

- the contents of the 12-bit program counter will be output on BUS and the lower half of port 2.
- Address Latch Enable (ALE) indicates when the address is valid. The falling edge of ALE is used to latch the address externally.
- Program Store Enable ($\overline{\text{PSEN}}$) indicates that an external instruction fetch is in progress and is used to enable the external memory device.
- BUS reverts to the input (high impedance) mode and the microcontroller accepts the 8-bit data on BUS as an instruction byte.

More detailed timing information is available in the microcontroller data sheets.

It is possible to force ALL instruction fetches (including fetches from internal memory) from external memory by pulling the EA pin HIGH. Microcontrollers with no on-chip ROM always operate in the external program memory mode (EA = + 5 V).

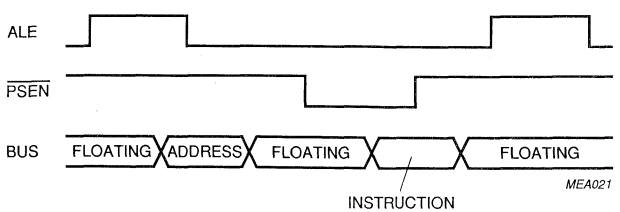


Fig.17 Instruction fetch from external program memory.

5.2.2 Extended program memory addressing (beyond 2 K)

For programs of 2 K bytes or less, the microcontroller addresses program memory in the conventional manner. Addresses beyond 2047 can be accessed by executing a program memory bank switch instruction (SEL MB1) followed by a branch instruction (JMP or CALL). The bank switch feature extends the range of branch instructions beyond their normal 2 K range and also prevents the user from inadvertently crossing the 2 K boundary.

5.2.2.1 Program memory bank switching

The switching of 2 K program memory banks is accomplished by directly setting or resetting the most significant bit of the program counter (bit 11); see Fig.18. Bit 11 is not altered by normal incrementing of the program counter; it is loaded with the contents of a special flip-flop each time a JMP or CALL instruction is executed. This special flip-flop is set by executing the SEL MB1 instruction and reset by SEL MB0. Therefore, the SEL MB instruction may be executed at any time prior to the actual bank switch which occurs during the next branch instruction encountered. Since all twelve bits of the program counter (including bit 11) are stored in the stack, when a CALL is executed, the user may jump to subroutines across the 2 K boundary and the proper bank will be restored upon return. However, the bank switch flip-flop will not be altered on return.

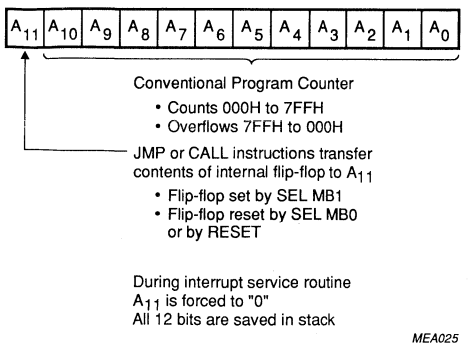


Fig.18 Program counter.

5.2.2.2 Interrupt routines

Interrupts always vector the program counter to location 3 or 7 in the **first** 2 K bank, and bit 11 of the program counter is held at logic 0 during the interrupt service routine. The end of the service routine is signalled by the execution of an RETR instruction. Interrupt service routines should therefore be located entirely in the lower 2 K bytes of program memory. The execution of a SEL MB0 or SEL MB1 instruction within an interrupt routine is not recommended since it will not affect PC11 while in the routine, but will affect the internal flip-flop.

5.2.3 Restoring I/O port information

Although the lower half of Port 2 is used to output the four most significant address bits during an external program memory fetch, the I/O information is still output during certain portions of each machine cycle. I/O information is always present on Port 2's lower 4 bits at the rising edge of ALE and can be sampled or latched at this time.

5.2.4 Expansion example

Figure 19 shows how a 2 K x 8 ROM (2716A) can be connected to provide an extra 2 K bytes of program memory. In this case no chip select decoding is required and $\overline{PSEN}$ enables the memory directly through the chip select input. If the system requires only 2 K of program memory, the same configuration can be used with a microcontroller which has no on-chip ROM.

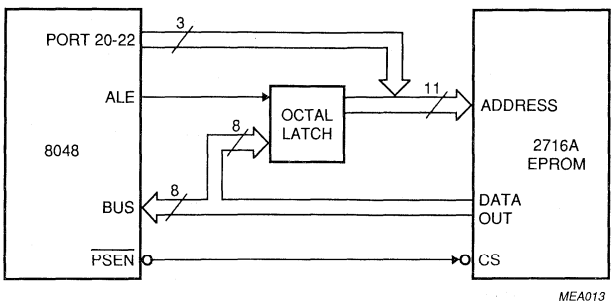


Fig.19 Expanding program memory using standard memory products.

5.3 Expansion of data memory

Data Memory may be expanded beyond the resident 64, 128, or 256 bytes by using the BUS feature.

5.3.1 READ/WRITE CYCLE

All address and data bytes are transferred over the 8 lines of BUS. As shown in Fig.20, a read or write cycle occurs as follows:

1. The contents of register R0 or R1 are output on BUS.
2. Address Latch Enable (ALE) indicates that the address is valid. The falling edge of ALE is used to latch the address externally.
3. A read ($\overline{RD}$) or write ($\overline{WR}$) pulse on the corresponding output pins indicates the type of data memory access in progress. Output data is valid at the falling edge of $\overline{WR}$ and input data must be valid at the falling edge of $\overline{RD}$.
4. Data (8 bits) is transferred in or out via BUS.

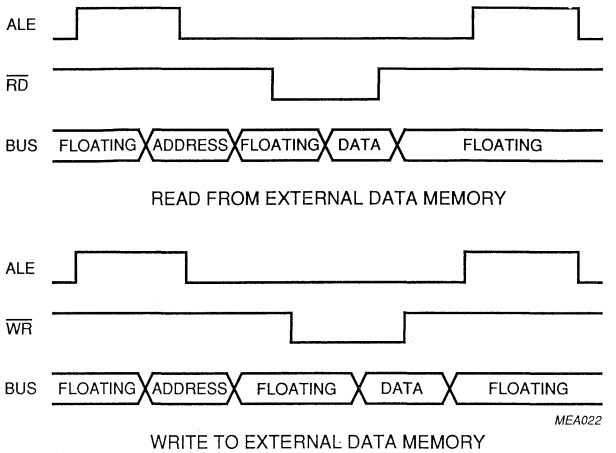


Fig.20 External data memory timing.

5.3.2 Addressing external data memory

External data memory is accessed with two 2-cycle move instructions: MOVX A, @Rr and MOVX @Rr, A; these instructions transfer 8 bits of data between the accumulator and the external data memory location addressed by the contents of one of the RAM pointer registers (R0 and R1). This enables an extra 256 RAM bytes to be addressed in addition to the on-chip RAM. Additional pages may be addressed by 'bank switching' with extra output lines.

5.3.3 Data memory expansion example

Figure 21 shows how the microcontroller can be expanded using a standard 256-byte memory.

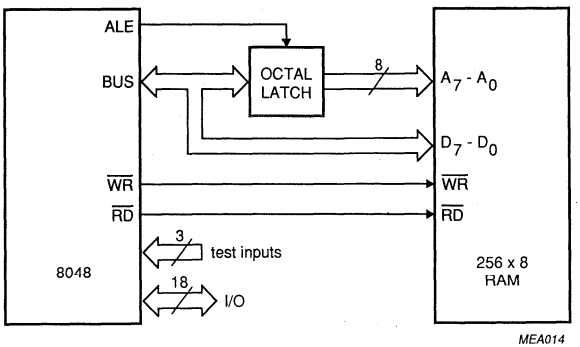


Fig.21 Interfacing the microcontroller to a standard 256 byte memory.

5.4 Expansion of input/output

There are four possible modes of I/O expansion:
- special expander (8243)
- standard I/O devices
- standard TTL compatible devices.

5.4.1 THE 8243 I/O EXPANDER

An efficient means of I/O expansion for small systems is the 8243 I/O expander which requires only 4 ports lines (the lower half of Port 2) for communication with the microcontroller. The 8243 contains four 4-bit I/O ports which serve as an extension of the on-chip I/O and are addressed as ports 4 - 7 (see Fig.22). The following operations may be performed on these ports:
- transfer accumulator contents to port
- transfer port contents to accumulator
- logical AND accumulator with port
- logical OR accumulator with port.

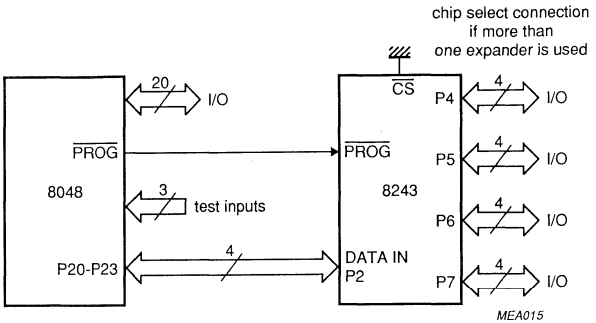


Fig.22 8243 expander I/O interface.

A 4-bit transfer from a port to the lower half of the accumulator sets the most significant four bits to zero. All data transferred between the microcontroller and the 8243 is transferred through the lower four bits of port 2 (P20-P23), with timing provided by an output pulse on the PROG pin of the microcontroller. Each transfer consists of two 4-bit nibbles; the first contains the 'op code' and port address, and the second contains the 4 data bits:

NIBBLE 1				NIBBLE 2			
3	2	1	0	3	2	1	0
Instruction op-code		Port address		Data			
I_1	I_0	A_1	A_0	d	d	d	d

I_1	I_0	FUNCTION
0	0	Read
0	1	Write
1	0	OR
1	1	AND

A_1	A_0	ADDRESS
0	0	Port 4
0	1	Port 5
1	0	Port 6
1	1	Port 7

A HIGH-to-LOW transition on the PROG pin indicates that a valid address and op-code is present; a LOW-to-HIGH transition indicates the presence of data. Additional I/O expanders may be added to the four-bit bus and chip selected using additional microcontroller output lines.

5.5 Memory bank switching

Certain applications may require more than:
- 4 K bytes of program memory directly addressable by the program counter, or
- 256 bytes of data memory and I/O locations directly addressable by the pointer registers R0 and R1.

These requirements can be satisfied by using 'bank switching' techniques. Bank switching involves the selection of various blocks or 'banks' of memory using dedicated microcontroller output port lines. Program memory may be selected in blocks of 4 K bytes; data memory and I/O may be selected in blocks of 256 bytes.

The most important consideration in implementing two or more banks is the software required to cross the bank boundaries. Each crossing of a boundary requires the microcontroller to first write a control bit to an output port before accessing memory or I/O in the new bank. If program memory bank switches are necessary, user programs should be organized to minimize program memory bank boundary crossings. Jumping to subroutines across boundaries should be avoided when possible since the programmer must remember which bank to return to after completion of the subroutine. If these subroutines are to be nested and accessed from various banks, a software 'stack' should be implemented to save the bank switch bit as though it were another program counter bit.

In terms of hardware, bank switching is very straightforward and only involves the connection of one or more I/O lines as bank enable signals. These enable signals should be ANDed with the normal memory and I/O chip select signals to select the required bank.

5.6 Control signal summary

Table 4 summarizes the instructions which activate the various control outputs. During all other instructions these outputs are in the inactive (high impedance) state.

Table 4 Control signals

CONTROL SIGNAL	ACTIVE DURING:
$\overline{RD}$	MOVX A,@R or INS BUS
$\overline{WR}$	MOVX @R,A or OUTL BUS
ALE	Every machine cycle
$\overline{PSEN}$	External program memory fetches (instruction or immediate data)
$\overline{PROG}$	MOVD A,P; ANLD P,A; MOVD P,A; or ORLD P,A

5.7 Port characteristics

5.7.1 BUS PORT OPERATIONS

The BUS port can operate in three different modes: as a latched I/O port, as a bidirectional bus port, or as a program memory address output when external memory is used. The BUS port lines are either active HIGH, active LOW, or high impedance (floating).

The latched mode (INS, OUTL) is intended for use in the single-chip configuration where BUS is not being used as an expander port. OUTL and MOVX instruction can be mixed if necessary. However, a previously latched output will be destroyed by executing a MOVX instruction and BUS will be left in the high impedance state. INS does not place the BUS in a high impedance state. Therefore, the use of MOVX after OUTL to place the BUS in a high impedance state is necessary before an INS instruction intended to read an external word (as opposed to the previously latched value).

OUTL should never be used in a system with external program memory, since latching BUS can cause the next instruction, if external, to be fetched improperly.

5.7.2 PORT 2 OPERATIONS

The lower half of Port 2 can be use in three different ways: as a quasi-bidirectional static port, as an 8243 expander port, and to address external program memory.

In all cases outputs are driven LOW by an active device, and driven HIGH momentarily by a low impedance device and held HIGH by a high impedance device to V_{CC}.

The port may contain latched I/O data prior to its use in another mode without affecting the operation of either. If the lower 4 bits of Port 2 (P20-P23) are used to output address for an external program memory fetch, the I/O information previously latched will be automatically removed temporarily while the address is present, then restored when the fetch is complete. However, if the lower 4 bits of Port 2 are used to communicate with an 8243, P20-3 will be left in the input mode (floating). After an output to the 8243, P20-3 will contain the value written, ANDed, or ORed to the 8243 port.

SINGLE-CHIP 8-BIT MICROCONTROLLER

GENERAL DESCRIPTION

The MAB80XXH family of single-chip 8-bit microcontrollers is fabricated in NMOS.
Three interchangeable (pin compatible) versions are available:

- MAB8048H: 1 K bytes mask-programmed ROM, 64 bytes RAM
- MAB8035HL: ROM-less version of the MAB8048H
- MAB8049H: 2 K bytes mask-programmed ROM, 128 bytes RAM
- MAB8039HL: ROM-less version of the MAB8049H
- MAB8050H: 4 K bytes mask-programmed ROM, 256 bytes RAM
- MAB8040HL: ROM-less version of the MAB8050H

These microcontrollers are designed to be efficient control processors as well as arithmetic processors. Their instruction set allows the user to directly set and reset individual I/O lines as well as test individual bits within the accumulator. A large variety of branch and table look-up instructions enable efficient implementation of standard logic functions. Code efficiency is high; over 70% of the instructions are single byte; all others are two byte.

An on-chip 8-bit counter is provided, which can count either machine cycles (÷32) or external events. The counter can be used to generate an interrupt to the processor.
Program and data memories plus input/output capabilities can be expanded using standard TTL compatible memories and logic. For more detailed information see the 8048 family specification.

Features

- 8-bit CPU, ROM, RAM and I/O
- 8-bit counter/timer
- On-chip oscillator and clock driver circuits
- Single-level interrupts: external and counter/timer
- 17 internal registers: accumulator, 16 addressable registers
- Over 90 instructions: 70% single byte
- All instructions 1 or 2 cycles
- Easily expandable memory and 27 I/O lines
- TTL compatible inputs and outputs
- Single 5 V supply
- Standard and extended temperature ranges (see Table 5): MAB80XX: 0 to +70 °C
  MAF80XX: −40 to +85 °C
  MAF80AXX: −40 to +110 °C

Applications

- Peripheral interfaces and controllers
- Test and measuring instruments
- Sequencers
- Modems and data enciphering
- Environmental control systems
- Audio/video systems

PACKAGE OUTLINES

All versions: with type no. suffix P (see Table 5): 40-lead DIL; plastic (SOT-129).
MAB8035/8048/8039/8049H/HLWP : 44-lead PLCC; plastic leaded chip-carrier (SOT187AA).

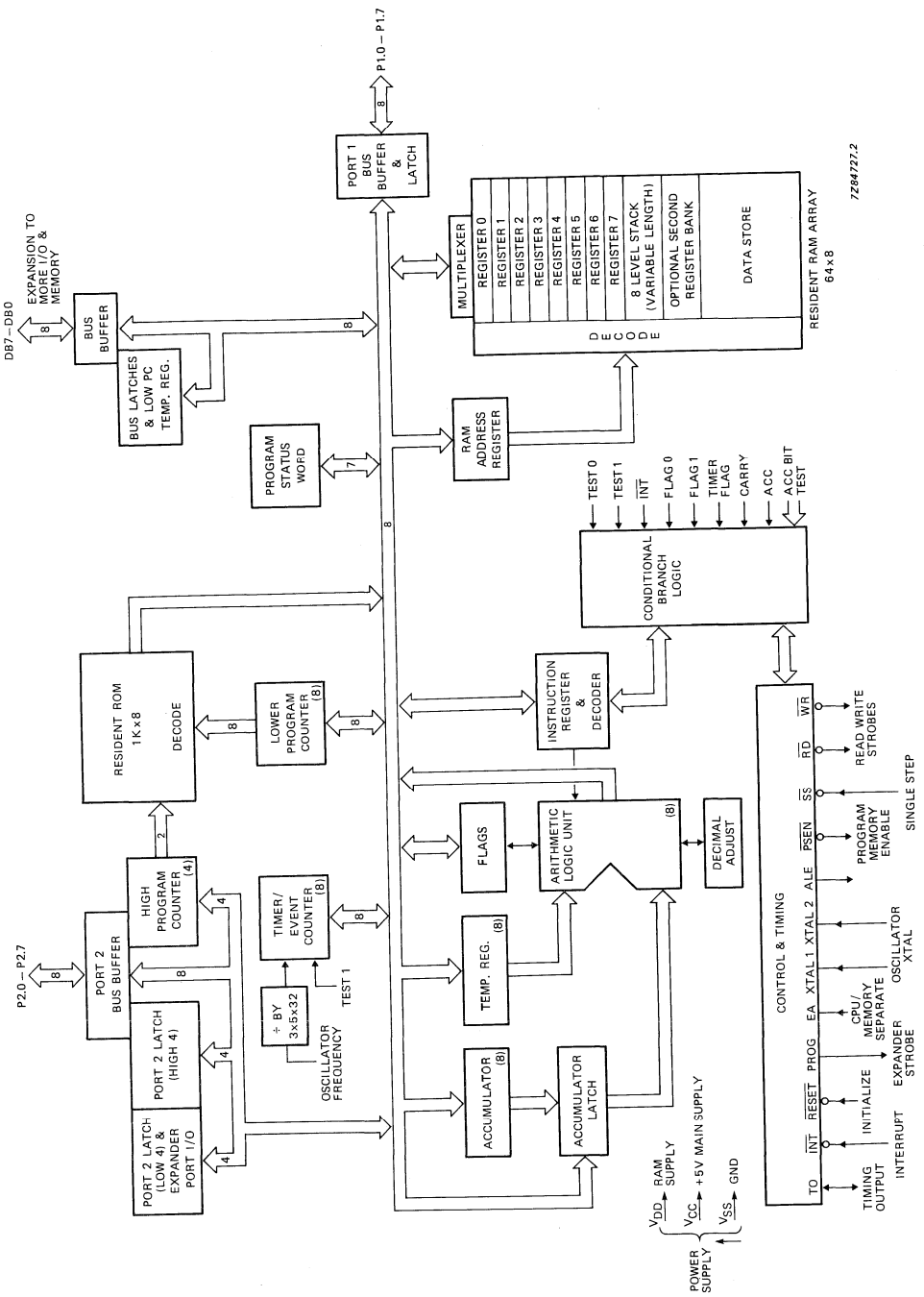


Fig. 1 Block diagram.

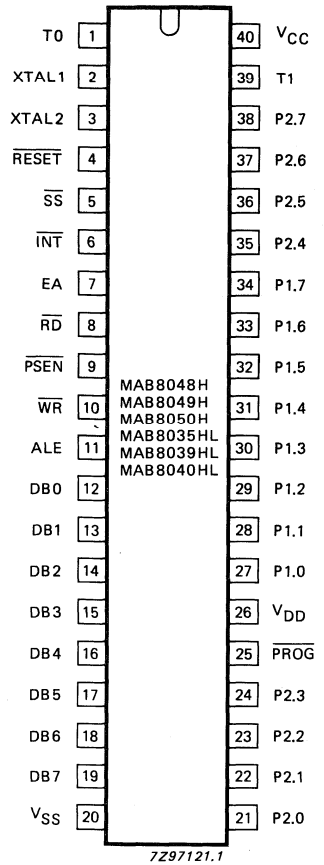


Fig. 2a Pinning diagram; for pin designation see next page.

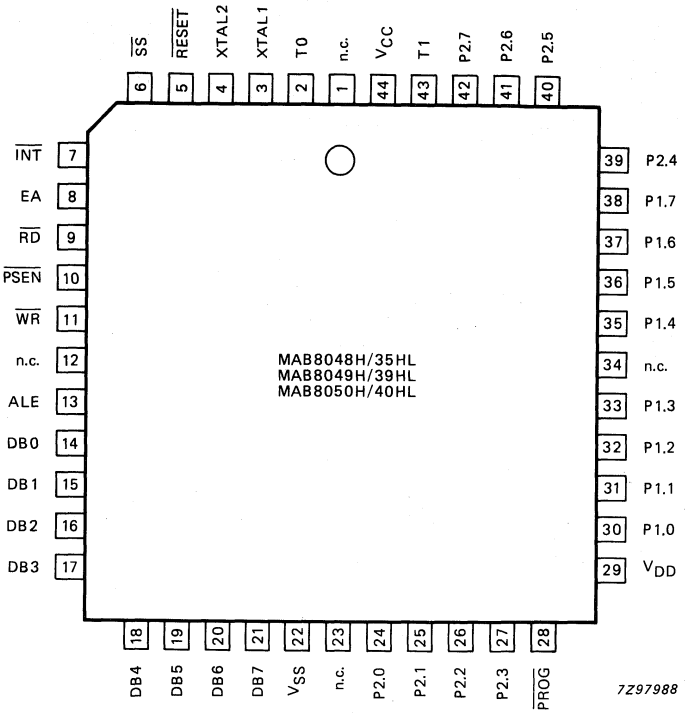


Fig. 2b Pinning diagram for MAB80XXHWP; for pin designation see next page.

Product type numbering European and USA cross-reference scheme

Type numbering reference used mainly in Europe	Type numbering equivalent reference used mainly in USA
MAB8039HLP/HLWP	SCN8039HCB N40/A44
MAF8039HLP/HLWP	SCN8039HAB N40/A44
MAB8049HP/HWP	SCN8049HCB N40/A44
MAF8049HP/HWP	SCN8049HAB N40/A44
MAB8040HLP/HLWP	SCN8040HCB N40/A44
MAF8050HP/HWP	SCN8050HCB N40/A44

PINNING

12–19	DB0–DB7	**Data Bus:** true bidirectional I/O port which can be written or read using the $\overline{RD}$ and $\overline{WR}$ strobes. This port can also be used as an 8-bit latch. It contains the 8 low order address bits during an external memory access and receives the addressed instruction under control of $\overline{PSEN}$. This multiplexed address/data port also contains the address and data during external RAM accesses.
27–34	P1.0–P1.7	**Port 1:** 8-bit quasi-bidirectional I/O port (note 1).
21–24	P2.0–P2.7	**Port 2:** 8-bit quasi-bidirectional I/O port (note 1).
35–38		P2.0–P2.3 contains the 4 higher order address bits during an external program memory access and provides a 4-bit bus for 8243 I/O expanders.
25	$\overline{PROG}$	**Output strobe:** active LOW for 8243 I/O expanders.
1	T0	**Test 0:** input pin which can be tested by the JT0 and JNT0 instructions. **Clock:** T0 can be configured as a clock output using the ENT0 CLK instruction.
39	T1	**Test 1:** input pin which can be tested using the JT1 and JTN1 instructions. T1 can be configured as the timer/counter input using the STRT CNT instruction.
6	$\overline{INT}$	**Interrrupt:** interrupt input pin which can initiate an interrupt if the external interrupt is enabled. Can also be tested using the JNI instruction. Interrupt is disabled during and after $\overline{RESET}$.
4	$\overline{RESET}$	**Reset:** active LOW input used to initialize the microcontroller. During program verification, the address is latched by a 0– to –1 transition on $\overline{RESET}$ and the data at the addressed location is output on BUS (note 2).
11	ALE	**Address latch enable:** occurs each cycle and is useful as a clock output. During an external program or data memory access, ALE is used to latch the address information multiplexed on the DB0 to DB7 outputs.
8	$\overline{RD}$	**Read BUS:** active LOW strobe used to gate data on to BUS lines when reading from an external source.
10	$\overline{WR}$	**Write BUS:** active LOW strobe used to write data from BUS lines to an external designation.
7	EA	**External access input:** when HIGH, all instruction fetches are from external memory.
9	$\overline{PSEN}$	**Program store enable:** active LOW strobe that occurs only during a fetch from external memory.
5	$\overline{SS}$	**Single step:** active LOW input used with ALE to cause the microcontroller to execute a single instruction.
2 3	XTAL 1 XTAL 2	**Crystal inputs:** inputs for a crystal, LC-network or an external timing signal to determine the internal oscillator frequency (note 2).
20	V_{SS}	**Ground:** circuit earth potential.
40	V_{CC}	**Power supply:** + 5 V main power supply pin.
26	V_{DD}	**Power supply:** + 5 V RAM standby power supply; low power

Notes

1. Each port line can be individually configured as an input or an output. A line is designated as an input by first writing a logic 1 to the line. $\overline{\text{RESET}}$ sets all port lines to logic 1.
2. Non-standard TTL V_{IH}.

FUNCTIONAL DESCRIPTION

The following sections provide a detailed functional description of the MAB80XXH microcontroller as shown in Fig. 1. The generic term "MAB80XXH" is used to refer collectively to the MAB8048H/35HL, MAB8049H/39HL and MAB8050H/40HL.

Program memory (see Fig. 3)

The on-chip program memory consists of 1024, 2048 or 4096 bytes of mask programmed ROM (MAB8048H/49H/50H); the MAB8035HL/39HL/40HL versions do not have on-chip program memory. The total addressing capability is 4096 bytes.
The program memory address space is divided into two 2048-byte banks MB0 and MB1. These two 2048 byte banks are each divided into 8 pages of 256 bytes for conditional branches. There are three locations in program memory of special interest. These are:

- Location 0 — contains the first instruction to be executed after a $\overline{\text{RESET}}$
- Location 3 — contains the first instruction of an external interrupt routine
- Location 7 — contains the first instruction of a timer/counter interrupt routine

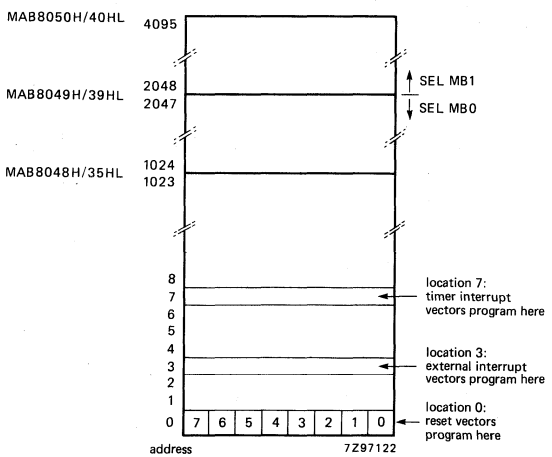


Fig. 3 Program memory map.

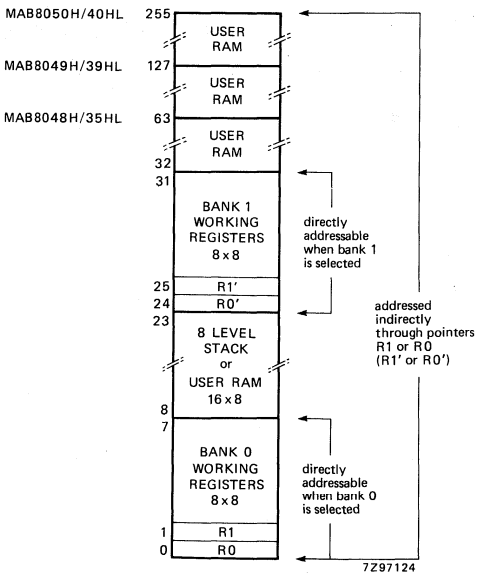


Fig. 4 Data memory map.

FUNCTIONAL DESCRIPTION (continued)

Data memory (see Fig. 4)

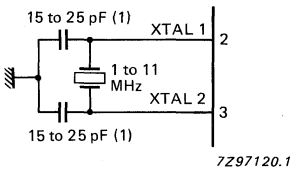
The on-chip data memory consists of a 64, 128 or 256 byte RAM. All locations are indirectly addressable using two RAM pointer registers R0, R1 or R0', R1'. The first 8 RAM locations (0 to 7) are designated as working register bank 0 and are directly addressable. By selecting register bank 1, RAM locations 24 to 31 become the working registers. RAM locations 8 to 23 are designated as the stack. Two bytes are used per CALL allowing up to 8 levels of subroutine nesting.
An extra 256 bytes of RAM may be added and addressed directly using the MOVX instructions.
If more RAM is required, I/O port lines may be used to select additional (256 byte) banks of external memory.

Program counter and stack

The program counter (PC) is a 12-bit counter/register that points to the location from which the next instruction is to be fetched. When EA is logic 0 the PC can address locations 0 to 1023 (8048H), 2047 (8049H) or 4095 (8050H) of internal program memory. At the 1 K (8048H), 2 K (8049H) boundary, an automatic switch-over to external memory occurs. When EA is logic 1 all fetches are from external program memory. The total address space is 4 K bytes. An interrupt or subroutine CALL causes the contents of the program counter to be stored in one of the 8 register pairs of the program counter stack. A 3-bit stack pointer which is part of the program status word (PSW) points to the relevant register pair. Data RAM locations 8 to 23 are available as stack registers and are used to store the program counter and 4 bits of the PSW register. The stack pointer, when initialized to 000, points to RAM locations 8 and 9. The first subroutine jump or interrupt results in the program counter contents being transferred to locations 8 an 9 of the RAM array. The stack pointer is then incremented by one to point to locations 10 and 11. Nesting of subroutines within subroutines can continue up to 8 times without overflowing the stack. If overflow does occur the deepest address stored (locations 8 and 9) will be overwritten and lost since the stack pointer overflows from 111 to 000. It also underflows from 000 to 111. The end of a subroutine, which is signalled by a return instruction (RET or RETR), causes the stack pointer to be decremented and the contents of the appropriate register pair to be transferred to the program counter.

Oscillator and clock (see Figs 5, 6 and 7)

The MAB80XXH has on-chip oscillator and clock driver circuitry. A crystal, LC-network or external timing signal (pulse generator) determines the oscillator frequency. The output of the oscillator is divided-by-three and is available at T0 (pin 1) by executing the ENT0 CLK instruction. This clock signal (CLK) is divided-by-five to define a machine (instruction) cycle. It is available at ALE (pin 11).



(1) Including crystal-socket stray capacities.

Fig. 5 Crystal oscillator mode. Crystal series impedance should be $< 75\ \Omega$ at 6 MHz and $< 180\ \Omega$ at 3,6 MHz. When using a ceramic oscillator both capacitors should be 30 pF.

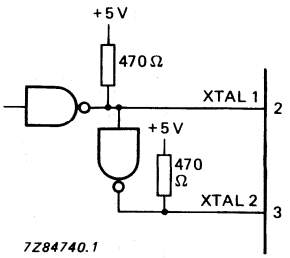
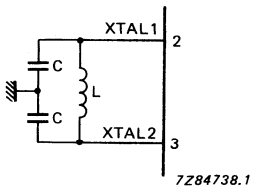


Fig. 6 External clock source. Both XTAL 1 and XTAL 2 should be driven. Resistors to V_{CC} (+ 5 V) are required to ensure V_{IH} = 3,8 V if TTL circuitry is used.
The minimum HIGH and LOW times are 45%.



$$f \approx \frac{1}{2\pi\sqrt{LC'}} \; ; \; C' = \frac{C + 3C_{pp}}{2}$$

L (μH)	C (pF)	nom. f (MHz)
45	20	5,2
120	20	3,2

Fig. 7 LC oscillator. Each capacitor should be ≈ 20 pF including stray capacitance C_{pp} ≈ 5 to 10 pF (pin-to-pin capacitance).

Timer/event counter

An internal counter is available which can count either external events or machine cycles (÷32). The machine cycles are divided-by-32 before they are applied to the input of the 8-bit counter. External events are applied directly to the input of the counter. The maximum clock rate is one third of the machine cycle frequency. The minimum positive duty cycle that can be detected is 0,2 times the cycle period. The counter can be configured to generate an interrupt to the processor when it overflows.

Interrupt

An interrupt may be generated by:

- An external input $\overline{INT}$ (pin 6)
  or
- A timer/counter overflow, when enabled.

In either event, the processor completes execution of the present instruction and then calls the interrupt service routine.
At the end of the interrupt service routine, a RETR instruction restores the machine to the state it was in prior to the interrupt. The external interrupt has priority over the timer/counter interrupt.

Input/output

The MAB80XXH has 27 I/O lines arranged as three 8-bit ports and 3 'test' inputs that can alter program sequences when tested by conditional jump instructions.
Each port line can be individually configured as an input or output.

FUNCTIONAL DESCRIPTION (continued)

Ports 1 and 2 are both 8-bits wide and have identical characteristics. Data written to these ports is latched and remains unchanged until rewritten. In the input mode, these ports are non-latching; inputs must be present until read by an input instruction. Inputs are fully TTL compatible and outputs will drive one standard TTL load.

Ports 1 and 2 are called quasi-bidirectional because they are not high impedance when configured as inputs. Each line is pulled up to + 5 V through a resistor ($\approx 50\ \Omega$). This pull-up provides sufficient source current for a TTL HIGH level, yet can be pulled LOW by a standard TTL gate, thus allowing the pin to be used both as an input and an output. To provide fast switching times during a logic 0 - to - 1 transition, transistor TR 2 is switched on for one fifth of a machine cycle when a logic 1 is written to the line. When a logic 0 is written, transistor TR 1 overcomes the pull-up and provides TTL current sinking capability. Since the pull-down transistor is low impedance, a logic 1 must first be written to any line which is to used as an input. $\overline{\text{RESET}}$ initializes all lines to the high impedance logic 1 state. This structure allows input and output on the same pin. Individual port lines can be read and written using the ANL and ORL instructions.

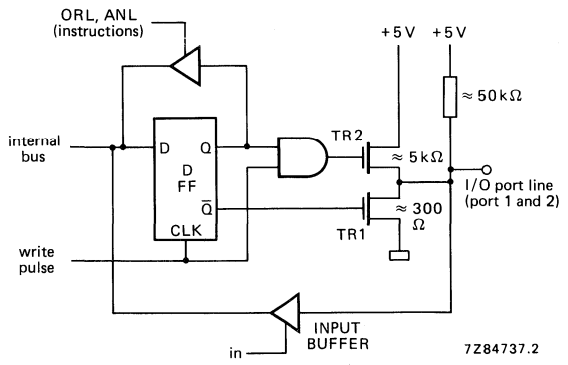
BUS (DB0–DB7)

BUS is a true bidirectional 8-bit port with associated input and output strobes. The BUS port can operate in three different modes: as a latched I/O port, as a bidirectional bus port, or in an expanded system as a program memory address output port. If the bidirectional feature is not needed, BUS can serve as either a statically latched output port or non-latching input port. Input and output lines on this port cannot be mixed. The BUS port lines are either active HIGH, active LOW, or high impedance (floating).

As a static port, data is written and latched using the OUTL instruction and input using the INS in instruction. The INS and OUTL instructions generate pulses on the corresponding $\overline{\text{RD}}$ and $\overline{\text{WR}}$ output strobe lines; however, in the static port mode they are generally not used. As a bidirectional port, the MOVX instructions are used to read and write the port. A write to the port generates a pulse on the $\overline{\text{WR}}$ output line and output data is valid at the trailing edge of $\overline{\text{WR}}$. A read of the port generates a pulse on the $\overline{\text{RD}}$ output line and input data must be valid at the trailing edge of $\overline{\text{RD}}$.

The latched mode (INS, OUTL) is intended for use in the single-chip configuration, where BUS is not being used as an expanded port. OUTL and MOVX instructions can be mixed if required. However, when using a MOVX instruction a previously latched output will be lost and BUS will be left in the high impedance state. INS does not put the BUS in a high impedance state. Therefore, in order to read an external byte (and not the previously latched value) using an INS instruction, it is necessary to precede INS with a MOVX instruction.

OUTL should never be used in a system with external program memory, since latching BUS may cause the next instruction to be incorrectly fetched.



N.B The OUTL, ANL and the ORL instructions relating to BUS are for use with internal program memory only.

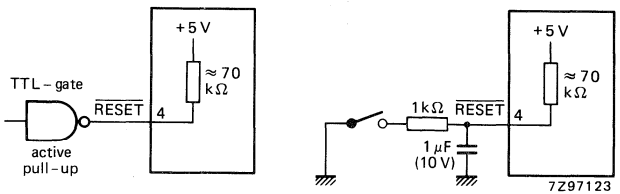
Fig. 8 Quasi-bidirectional port structure.

Test (T0, T1) and $\overline{INT}$

These three pins serve as inputs and may be tested by the conditional jump instruction. They allow inputs to cause program branches without the necessity of loading an input port into the accumulator.

$\overline{RESET}$ (see Fig. 9)

This active LOW input is used to initialize the microcontroller.
This Schmitt-trigger input has an internal pull-up resistor which, in combination with an external 1 μF capacitor, provides an internal reset pulse of sufficient duration to reset all circuitry. If the reset pulse is generated externally, the reset pin must be held at ground (0,45 V) for at least 10 ms after the power supply is within tolerance. Only 5 machine cycles (12,5 μs at 6 MHz) are required if power is already on and the oscillator has stabilized.



(a) (b)

Fig. 9 An external reset is shown in (a) and power-on reset in (b).

Single step ($\overline{SS}$)

This active LOW input when used in combination with ALE will cause the microcontroller to execute a single instruction, then wait until $\overline{SS}$ is reactivated.

Power-down mode (see Fig. 10)

In the MAB80XXH, power can be removed from all but the data RAM array, for low power standby operation. In the power-down mode the contents of the data RAM can be maintained while drawing typically 10% to 15% of the normal operating supply voltage. V_{CC} serves as the + 5 V supply pin for the bulk of the circuitry, while the V_{DD} pin supplies only the RAM array. In normal operation, both pins are at + 5 V. In the standby mode, V_{CC} is at ground and only V_{DD} is maintained at + 5 V. Applying $\overline{RESET}$ to the microcontroller through the reset pin inhibits any access to the RAM and ensures that the RAM cannot be inadvertently altered as power is removed from V_{CC}.

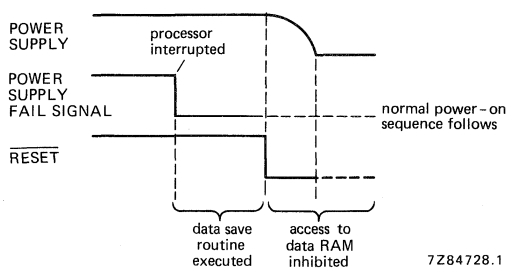


Fig. 10 Power down sequence.

FUNCTIONAL DESCRIPTION (continued)

Instruction set (see Tables 1, 2, 3 and 4)

The MAB80XXH instruction set consists of over 90 one and two-byte instructions. Program code efficiency is high because:

- Working registers and program variables are stored in RAM locations 0 to 127, which require only a single byte to address
- Program memory is divided into pages of 256 bytes, which means that branch destination addresses require only one byte

The instruction set performs logical, arithmetic and test operations on bytes. It also manipulates and tests bits. A set of MOVE instructions operate indirectly on either RAM or ROM, which permits efficient access of pointers and data tables. The indirect jump instruction performs a multi-way branch (up to 256) on the contents of the accumulator to addresses stored in a look-up table. The 'decrement register and jump if not zero' instruction saves a byte each time it is used as opposed to using separate increment and test instructions. The on-chip counter provides the facility for external events or time to be counted by hardware which does not interfere with the main program. The MAB80XXH can either test the counter (under program control) or cause its overflow to generate an interrupt. These features are essential for real-time applications.

Table 1 Symbols and definitions used in Table 2.

symbol	definition description
A	accumulator
addr	program memory address
Bb	bit designation (b = 0–7)
RBS	register bank select
C	carry (bit CY)
CNT	event counter
D	mnemonic for 4-bit digit (nibble)
data	8-bit number or expression
I	interrupt
MB	memory bank
MBFF	memory bank flip-flop
P	mnemonic for 'in-page' operation
PC	program counter
Pp	port designation (p = 0, 1, 2)
PSW	program status word
RB	register bank
Rr	register designation (r = 0–7)
Sn	serial I/O register
SP	stack pointer
T	timer
TF	timer flag
T0	test 0 input
T1	test 1 input
#	immediate data prefix
@	indirect address prefix
(X)	contents of X
((X))	contents of location addressed by X
←	is replaced by
↔	is exchanged with

Table 2 Instruction set

	mnemonic	opcode (hex.)	bytes/ cycles	description	function		notes
ACCUMULATOR	ADD A, Rr	6*	1/1	Add register contents to A	(A)←(A) + (Rr)	r = 0–7	1
	ADD A, @Rr	60 61	1/1	Add RAM data, addressed by Rr, to A	(A)←(A) + ((R0)) (A)←(A) + ((R1))		1
	ADD A, #data	03 data	2/2	Add immediate data to A	(A)←(A) + data		1
	ADDC A, Rr	7*	1/1	Add carry and register contents to A	(A)←(A) + (Rr) + (C)	r = 0–7	1
	ADDC A, @Rr	70 71	1/1	Add carry and RAM data, addresses by Rr, to A	(A)←(A) + ((R0)) + (C) (A)←(A) + ((R1)) + (C)		1
	ADDC A, #data	13 data	2/2	Add carry and immediate data to A	(A)←(A) + data + (C)		1
	ANL A, Rr	5*	1/1	'AND' Rr with A	(A)←(A) AND (Rr)	r = 0–7	
	ANL A, @Rr	50 51	1/1	'AND' RAM data, addressed by Rr, with A	(A)←(A) AND ((R0)) (A)←(A) AND ((R1))		
	ANL A, #data	53 data	2/2	'AND' immediate data with A	(A)←(A) AND data		
	ORL A, Rr	4*	1/1	'OR' Rr with A	(A)←(A) OR (Rr)	r = 0–7	
	ORL A, @Rr	40 41	1/1	'OR' RAM data, addressed by Rr, with A	(A)←(A) OR ((R0)) (A)←(A) OR ((R1))		
	ORL A, #data	43 data	2/2	'OR' immediate data with A	(A)←(A) OR data		
	XRL A, Rr	D*	1/1	'XOR' Rr with A	(A)←(A) XOR (Rr)	r = 0–7	
	XRL A, @Rr	D0 D1	1/1	'XOR' RAM, addressed by Rr, with A	(A)←(A) XOR ((R0)) (A)←(A) XOR ((R1))		
	XRL A, #data	D3 data	2/2	'XOR' immediate data with A	(A)←(A) XOR data		
	INC A	17	1/1	increment A by 1	(A)←(A) + 1		
	DEC A	07	1/1	decrement A by 1	(A)←(A) – 1		
	CLR A	27	1/1	clear A to zero	(A)←0		
	CPL A	37	1/1	one's complement A	(A)←NOT(A)		
	RL A	E7	1/1	rotate A left	$(A_{n+1})\leftarrow(A_n)$ $(A_0)\leftarrow(A_7)$	n = 0–6	

	mnemonic	opcode (hex.)	bytes/ cycles	description	function		notes
ACCUMULATOR (cont.)	RLC A	F7	1/1	rotate A left through carry	$(A_{n+1})\leftarrow(A_n)$ $(A_0)\leftarrow(C)$, $(C)\leftarrow(A_7)$	n = 0–6	2
	RR A	77	1/1	rotate A right	$(A_n)\leftarrow(A_{n+1})$ $(A_7)\leftarrow(A_0)$	n = 0–6	
	RRC A	67	1/1	rotate A right through carry	$(A_n)\leftarrow(A_{n+1})$ $(A_7)\leftarrow(C)$, $(C)\leftarrow(A_0)$	n = 0–6	2
	DA A	57	1/1	decimal adjust A			2
	SWAP A	47	1/1	swap nibbles of A	$(A_{4-7})\leftrightarrow(A_{0-3})$		
DATA MOVES	MOV A, Rr	F*	1/1	move register contents to A	(A)←(Rr)	r = 0–7	
	MOV A, @Rr	F0 F1	1/1	move RAM data, addressed by Rr, to A	(A)←((R0)) (A)←((R1))		
	MOV A, #data	23 data	2/2	move immediate data to A	(A)←data		
	MOV Rr, A	A*	1/1	move accumulator contents to register	(Rr)←(A)	r = 0–7	
	MOV @Rr, A	A0 A1	1/1	move accumulator contents to RAM location addressed by Rr	((R0))←(A) ((R1))←(A)		
	MOV Rr, #data	B* data	2/2	move immediate data to Rr	(Rr)←data	r = 0–7	
	MOV @rR, #data	B0 data B1 data	2/2	move immediate data to RAM location addressed by Rr	((R0))←data ((R1))←data		
	XCH A, Rr	2*	1/1	exchange accumulator contents with Rr	(A)↔(Rr)	r = 0–7	
	XCH A, @Rr	20 21	1/1	exchange accumulator contents with RAM data addressed by Rr	(A)↔((R0)) (A)↔((R1))		
	XCHD A, @Rr	30 31	1/1	exchange lower nibbles of A and RAM data addressed by Rr	$(A_{0-3})\leftrightarrow((R0_{0-3}))$ $(A_{0-3})\leftrightarrow((R1_{0-3}))$		
	MOV A, PSW	C7	1/1	move PSW contents to accumulator	(A)←(PSW)		
	MOV PSW, A	D7	1/1	move accumulator contents to PSW	(PSW)←(A)		3
	MOVP A, @A	A3	1/2	move indirectly addressed data in current page to A	(A)←((A))		
	MOVP3 A,@A	E3	1/2	move data in page 3 to A	(A)←((A))	in page 3	

	MOVX A,@Rr	80 81	1/2	move indirect the contents of external memory to A	(A)←((R))	r = 0–1	
	MOVX @Rr,A	90 91	1/2	move indirect the contents of A to external memory	((R))←(A)	r = 0–1	
FLAGS	CLR C	97	1/1	clear carry bit	(C)←0		2
	CPL C	A7	1/1	complement carry bit	(C)←NOT(C)		2
REGISTER	INC Rr	1*	1/1	increment register by 1	(Rr)←(Rr) + 1	r = 0–7	
	INC @Rr	10 11	1/1	increment RAM data, addressed by Rr, by 1	((R0))←((R0)) + 1 ((R1))←((R1)) + 1		
	DEC Rr	C*	1/1	decrement register by 1	(Rr)←(Rr) −1	r = 0–7	
BRANCH	JMP addr	● 4 address	2/2	unconditional jump within a 2 K bank	(PC_{8-10})←$addr_{8-10}$ (PC_{0-7})←$addr_{0-7}$ (PC_{11-12})←MBFF 0–1		
	JMPP @A	B3	1/2	indirect jump within a page	(PC_{0-7})←((A))		
	DJNZ Rr, addr	E* address	2/2	decrement Rr by 1 and jump if not zero to addr	(Rr)←(Rr) −1 if (Rr) not zero (PC_{0-7})←addr	r = 0–7	
	JF0 addr	B6 address	2/2	jump to addr if F0 = 1	if F0 = 1: (PC_{0-7})←addr		
	JF1 addr	76 address	2/2	jump to addr if F1 = 1	if F1 = 1: (PC_{0-7})←addr		
	JNI addr	86 address	2/2	jump to addr if $\overline{INT}$ = 0	if $\overline{INT}$ = 0: (PC_{0-7})←addr		
	JBb addr	▲ 2 address	2/2	jump to addr if Acc. bit b = 1	if b = 1: (PC_{0-7})←addr	b = 0–7	
	JC addr	F6 address	2/2	jump to addr if C = 1	if C = 1: (PC_{0-7})←addr		
	JNC addr	E6 address	2/2	jump to addr if C = 0	if C = 0: (PC_{0-7})←addr		
	JZ addr	C6 address	2/2	jump to addr if A = 0	if A = 0: (PC_{0-7})←addr		
	JNZ addr	96 address	2/2	jump to addr if A is NOT zero	if A ≠0: (PC_{0-7})←addr		
	JT0 addr	36 address	2/2	jump to addr if T0 = 1	if T0 = 1: (PC_{0-7})←addr		
	JNT0 addr	26 address	2/2	jump to addr if T0 = 0	if T0 = 0: (PC_{0-7})←addr		
	JT1 addr	56 address	2/2	jump to addr if T1 = 1	if T1 = 1: (PC_{0-7})←addr		
	JNT1 addr	46 address	2/2	jump to addr if T1 = 0	if T1 = 0: (PC_{0-7})←addr		
	JTF addr	16 address	2/2	jump to addr if Timer Flag = 1	if TF = 1: (PC_{0-7})←addr		4

	mnemonic	opcode (hex.)	bytes/ cycles	description	function	notes
TIMER/EVENT COUNTER	MOV A, T	42	1/1	move timer/event counter contents to accumulator	(A)←(T)	
	MOV T, A	62	1/1	move accumulator contents to timer/event counter	(T)←(A)	
	STRT CNT	45	1/1	start event counter		
	STRT T	55	1/1	start timer		
	STOP TCNT	65	1/1	stop timer/event counter		
	EN TCNTI	25	1/1	enable timer/event counter interrupt		
	DIS TCNTI	35	1/1	disable timer/event counter interrupt		
CONTROL	EN I	05	1/1	enable external interrupt		
	DIS I	15	1/1	disable external interrupt		
	SEL RB0	C5	1/1	select register bank 0	(RBS)←0	5
	SEL RB1	D5	1/1	select register bank 1	(RBS)←1	5
	SEL MB0	E5	1/1	select program memory bank 0	(MBFF0)←0, (MBFF1)←0	
	SEL MB1	F5	1/1	select program memory bank 1	(MBFF0)←1, (MBFF1)←0	
	ENT0 CLK	75	1/1	enable clock output onto T0		
SUBROUTINE	CALL addr	▲ 4 address	2/2	jump to subroutine	((SP))←(PC), ($PSW_{4, 6, 7}$) (SP)←(SP) + 1 (PC_{8-10})←$addr_{8-10}$ (PC_{0-7})←$addr_{0-7}$ (PC_{11-12})←MBFF 0–1	6
	RET	83	1/2	return from subroutine	(SP)←(SP) −1 (PC)←((SP))	6
	RETR	93	1/2	return from interrupt and restore bits 4, 6, 7 of PSW	(SP)←(SP) −1 ($PSW_{4, 6, 7}$) + (PC)←((SP))	6

INPUT/OUTPUT	OUTL BUS,A	02	1/2	output accumulator to BUS	(BUS)←A		
	IN A,Pp	09 0A	1/2	input port p data to accumulator	(A)←(P1) (A)←(P2)	p = 1–2	7
	INS A,BUS	08	1/2	input strobed BUS data into accumulator	(A)←(BUS)		
	OUTL Pp,A	39 3A	1/2	output accumulator data to port p	(P1)←(A) (P2)←(A)	p = 1–2	
	ANL BUS, # data	98	2/2	logical AND immediate data with BUS	(BUS)←(BUS) AND data		
	ANL Pp, # data	99 9A	2/2	AND port p data with immediate data	(P1)←(P1) AND data (P2)←(P2) AND data	p = 1–2	
	ORL Pp, # data	89 8A	2/2	OR port p data with immediate data	(P1)←(P1) OR data (P2)←(P2) OR data	p = 1–2	
	ORL BUS, # data	88	2/2	logical OR immediate data with BUS	(BUS)←(BUS) OR data		
	MOVD A,Pp	0C 0D 0E 0F	1/2	move contents of designated port (4–7) to A	(A_{0-3})←(Pp) (A_{4-7})←0	p = 4–7	
	MOVD Pp,A	3C 3D 3E 3F	1/1	move contents of A to designated port (4–7)	(Pp)←(A_{0-3})	p = 4–7	
	ANLD Pp,A	9C 9D 9E 9F	1/2	logical AND contents of A with designated port (4–7)	(Pp)←(Pp) AND (A_{0-3})	p = 4–7	
	ORLD Pp,A	8C 8D 8E 8F	1/1	logical OR contents of A with designated port (4–7)	(Pp)←(Pp) OR (A_{0-3})	p = 4–7	
	NOP	00	1/1	no operation			

Notes to Table 2.

1. PSW CY, AC affected
2. PSW CY affected
3. PSW PS affected
4. Execution of JTF and JNTF instruction resets the Timer Flag (TF).
5. PSW RBS affected
6. PSW SP_0, SP_1, SP_2 affected
7. (A) = 111 P23, P22, P21, P20.
8. (S1) has a different meaning for read and write operation, see serial I/O interface.

* : 8, 9, A, B, C, D, E, F
● : 0, 2, 4, 6, 8, A, C, E
▲ : 1, 3, 5, 7, 9, B, D, F

Table 3 Instruction timing (see also Figs 11 and 12)

	cycle 1					cycle 2				
instruction	S1	S2	S3	S4	S5	S1	S2	S3	S4	S5
IN A,P	fetch instruction	increment program counter	–	increment timer	–	–	read port	* –	–	–
OUTL P,A			–		output to port	–	–	* –	–	–
ANL P,#data		*	–		read port	fetch immediate data	–	*increment program counter	output to port	–
ORL P,#data		*	–		read port	fetch immediate data	–	*increment program counter	output to port	–
INS A,BUS			–		–	–	read port	* –	–	–
OUTL BUS,A			–		output to port	–	–	* –	–	–
ANL BUS,#data		*	–		read port	fetch immediate data	–	*increment program counter	output to port	–
ORL BUS,#data		*	–		read port	fetch immediate data	–	*increment program counter	output to port	–
MOVX @R,A			output RAM address		output data to RAM	–	–	* –	–	–
MOVX A,@R			output RAM address		–	–	read data	* –	–	–
MOVD A,P	fetch instruction	increment program counter	output opcode/ address	increment timer	–	–	read P2 lower	* –	–	–

	cycle 1					cycle 2				
instruction	S1	S2	S3	S4	S5	S1	S2	S3	S4	S5
MOVD P,A	fetch instruction	increment program counter	output opcode/ address	increment timer	output data to P2 lower	–	–	* –	–	–
ANLD P,A	¦		output opcode/ address	¦	output data	–	–	* –	–	–
ORLD P,A	¦		output opcode/ address	¦	output data	–	–	* –	–	–
J (conditional)	¦	*	sample condition	increment timer	–	fetch immediate data	–	* opdata program counter	–	–
STRT CNT STRT T	¦	*	–	–	start counter					
STOP TCNT	¦	*	–	–	stop counter					
EN I	¦	*	–	enable interrupt	–					
DIS I	¦	*	–	disable interrupt	–					
ENT0 CLK	fetch instruction	* increment program counter	–	enable clock	–					

* Valid instruction addresses are output at this time if external program memory is being accessed.

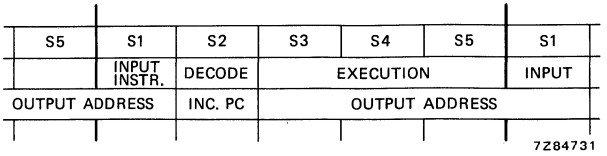


Fig. 11 Instruction cycle.

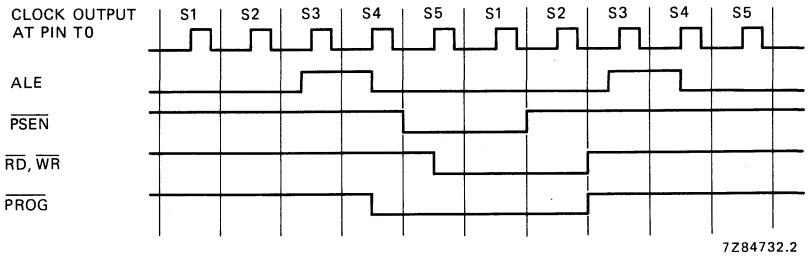


Fig. 12 Instruction cycle timing.

Table 4 Instruction map.

first hexadicimal character of opcode	second hexadicimal character of opcode														
	0 1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
0	NOP	OUTL BUS,A	ADD A, #data	JMP page 0	EN I		DEC A	INS A,BUS	IN A, Pp 1	2		MOVD A,Pp 4	5	6	7
1	INC @Rr 0 1	JB0 addr	ADDC A, #data	CALL page 0	DIS I	JTF addr	INC A	INC Rr 0	1	2	3	4	5	6	7
2	XCH A, @Rr 0 1		MOV A, #data	JMP page 1	EN TCNTI	JNT0 addr	CLR A	XCH A,Rr 0	1	2	3	4	5	6	7
3	XCHD A, @Rr 0 1	JB1 addr		CALL page 1	DIS TCNTI	JT0 addr	CPL A		OUTL Pp,A 1	2		MOVD Pp,A 4	5	6	7
4	ORL A, @Rr 0 1	MOV A, T	ORL A, #data	JMP page 2	STRT CNT	JNT1 addr	SWAP A	ORL A,Rr 0	1	2	3	4	5	6	7
5	ANL A, @Rr 0 1	JB2 addr	ANL A, #data	CALL page 2	STRT T	JT1 addr	DA, A	ANL A,Rr 0	1	2	3	4	5	6	7
6	ADD A, @Rr 0 1	MOV T,A		JMP page 3	STOP TCNT		RRC A	ADD A,Rr 0	1	2	3	4	5	6	7
7	ADDC A, @Rr 0 1	JB3 addr		CALL page 3	ENT0 CLK	JF1 addr	RR A	ADDC A,Rr 0	1	2	3	4	5	6	7
8	MOVX A, @Rr 0 1		RET	JMP page 4	CLR F0	JNI addr		ORL BUS, #data	ORL Pp, #data 1	2		ORLD Pp,A 4	5	6	7
9	MOVX @Rr,A 0 1	JB4 addr	RETR	CALL page 4	CPL F0	JNZ addr	CLR C	ANL BUS, #data	ANP Pp, #data 1	2		ANLD Pp,A 4	5	6	7
A	MOV @Rr, A 0 1		MOVP A, @A	JMP page 5	CLR F1		CPL C	MOV Rr,A 0	1	2	3	4	5	6	7
B	MOV @Rr, #data 0 1	JB5 addr	JMPP @A	CALL page 5	CPL F1	JF0 addr		MOV R, #data 0	1	2	3	4	5	6	7
C				JMP page 6	SEL RB0	JZ addr	MOV A, PSW	DEC Rr 0	1	2	3	4	5	6	7
D	XRL A, @Rr 0 1	JB6 addr	XRL A, #data	CALL page 6	SEL RB1		MOV PSW, A	XRL A,Rr 0	1	2	3	4	5	6	7
E			MOVP3 A @A	JMP page 7	SEL MB0	JNC addr	RL A	DJNZ Rr, addr 0	1	2	3	4	5	6	7
F	MOV A, @Rr 0 1	JB7 addr		CALL page 7	SEL MB1	JC addr	RLC A	MOV A,Rr 0	1	2	3	4	5	6	7

RATINGS

Limiting values in accordance with the Absolute Maximum System (IEC 134)

Input voltage with respect to V_{SS}			
except input EA	V_I	−0,5 to + 7	V
input EA	V_I	−0,5 to + 12	V
DC current into any input or output	$\pm I_I, \pm I_O$	max. 10	mA
Total power dissipation	P_{tot}	max. 1	W
Storage temperature range	T_{stg}	−65 to +150	°C
Operating ambient temperature range	T_{amb}	see Table 5	

Table 5 MAB80XXH versions.

version	internal memory		RAM st/by.	frequency (MHz) min.	frequency (MHz) max.	temperature range (°C)
MAB8048H	1 K x 8 ROM	64 byte RAM	yes	1,0	11,0	0 to +70
MAB8035HL	none	64 byte RAM	yes	1,0	11,0	0 to +70
MAF8048H	1 K x 8 ROM	64 byte RAM	yes	1,0	11,0	−40 to +85
MAF8035HL	none	64 byte RAM	yes	1,0	11,0	−40 to +85
MAF80A48H	1 K x 8 ROM	64 byte RAM	yes	1,0	10,0	−40 to +110
MAF80A35HL	none	64 byte RAM	yes	1,0	10,0	−40 to +110
MAB8049H	2 K x 8 ROM	128 byte RAM	yes	1,0	11,0	0 to +70
MAB8039HL	none	128 byte RAM	yes	1,0	11,0	0 to +70
MAF8049H	2 K x 8 ROM	128 byte RAM	yes	1,0	11,0	−40 to +85
MAF8039HL	none	128 byte RAM	yes	1,0	11,0	−40 to +85
MAF80A49H	2 K x 8 ROM	128 byte RAM	yes	1,0	10,0	−40 to +110
MAF80A39HL	none	128 byte RAM	yes	1,0	10,0	−40 to +110
MAB8050H	4 K x 8 ROM	256 byte RAM	yes	1,0	11,0	0 to +70
MAB8040HL	none	256 byte RAM	yes	1,0	11,0	0 to +70
MAF8050H	4 K x 8 ROM	256 byte RAM	yes	1,0	11,0	−40 to +85
MAF8040HL	none	256 byte RAM	yes	1,0	11,0	−40 to +85
MAF80A50H	4 K x 8 ROM	256 byte RAM	yes	1,0	10,0	−40 to +110
MAF80A40HL	none	256 byte RAM	yes	1,0	10,0	−40 to +110

DC CHARACTERISTICS (MAB8048H/35HL; MAB8049H/39HL; MAB8050H/40HL)

$V_{CC} = V_{DD} = 5$ V (± 10%); $V_{SS} = 0$ V; $T_{amb} = 0$ to + 70 °C; all voltages with respect to V_{SS} unless otherwise specified

parameter	symbol	min.	typ.	max.	unit
Supply current at $V_{DD} = 5$ V ± 10%; $V_{SS} = V_{CC} = 0$ V					
MAB8048H/35HL	I_{DD}	–	–	6	mA
MAB8049H/39HL	I_{DD}	–	–	8	mA
MAB8050H/40HL	I_{DD}	–	–	15	mA
Supply current (total) at $V_{DD} = V_{CC} = 5$ V ± 10%; $V_{SS} = 0$ V					
MAB8048H/35HL	$I_{DD} + I_{CC}$	–	–	80	mA
MAB8049H/39HL	$I_{DD} + I_{CC}$	–	–	90	mA
MAB8050H/40HL	$I_{DD} + I_{CC}$	–	–	100	mA
Inputs					
Input voltage LOW all inputs except XTAL 1, XTAL 2, $\overline{\text{RESET}}$	V_{IL}	−0,5	–	0,8	V
Input voltage LOW XTAL 1, XTAL 2, $\overline{\text{RESET}}$	V_{IL1}	−0,5	–	0,6	V
Input voltage HIGH all inputs except XTAL 1, XTAL 2, $\overline{\text{RESET}}$	V_{IH}	2,0	–	V_{CC}	V
Input voltage HIGH XTAL 1, XTAL 2, $\overline{\text{RESET}}$	V_{IH1}	3,8	–	V_{CC}	V
Outputs					
Output voltage LOW (DB0 to DB7) at $I_{OL} = 2$ mA	V_{OL}	–	–	0,45	V
Output voltage LOW ($\overline{\text{RD}}$, $\overline{\text{WR}}$, $\overline{\text{PSEN}}$, ALE) at $I_{OL1} = 1{,}8$ mA	V_{OL1}	–	–	0,45	V
Output voltage LOW ($\overline{\text{PROG}}$) at $I_{OL2} = 1$ mA	V_{OL2}	–	–	0,45	V
Output voltage LOW (all other outputs) at $I_{OL3} = 1{,}6$ mA	V_{OL3}	–	–	0,45	V
Output voltage HIGH (DB0 to DB7) at $-I_{OH} = 400\ \mu$A	V_{OH}	2,4	–	–	V
Output voltage HIGH ($\overline{\text{RD}}$, $\overline{\text{WR}}$, $\overline{\text{PSEN}}$, ALE) at $-I_{OH1} = 100\ \mu$A	V_{OH1}	2,4	–	–	V
Output voltage HIGH (all other outputs) at $-I_{OH} = 40\ \mu$A	V_{OH2}	2,4	–	–	V

DC CHARACTERISTICS (continued)

parameter	symbol	min.	typ.	max.	unit
Input leakage current (T1, $\overline{INT}$) at $V_{SS} < V_I < V_{CC}$	$\pm I_{IL}$	–	–	10	μA
Output leakage current (DB0 to DB7, T0; high impedance) at $V_{SS} + 0{,}45\ V < V_I < V_{CC}$	$\pm I_{OZ}$	–	–	10	μA
Input load current (P1.0 to P1.7, P2.0 to P2.7, EA, $\overline{SS}$) at $V_{SS} + 0{,}45\ V < V_I < V_{CC}$	$-I_{LI}$	–	–	500	μA
Input load current ($\overline{RESET}$) at $V_{SS} < V_I < V_{CC}$	$-I_{LI1}$	20	–	300	μA

DC CHARACTERISTICS

MAF8048H/35HL; MAF8049H/39HL; MAF8050H/40HL (at $T_{amb} = -40$ to $+85\ ^oC$)
MAF80A48H/A35HL; MAF80A49H/A39HL; MAF80A50H/A40HL (at $T_{amb} = -40$ to $+110\ ^oC$)

$V_{CC} = V_{DD} = 5\ V\ (\pm 10\%)$; $V_{SS} = 0\ V$; T_{amb} as above; all voltages with respect to V_{SS} unless otherwise specified

parameter	symbol	min.	typ.	max.	unit
Supply current at $V_{DD} = 5\ V \pm 10\%$; $V_{SS} = V_{CC} = 0\ V$					
MAF8048H/35HL; MAF80A48H/A35HL	I_{DD}	–	–	8	mA
MAF8049H/39HL; MAF80A49H/A39HL	I_{DD}	–	–	10	mA
MAF8050H/40HL; MAF80A50H/A40HL	I_{DD}	–	–	18	mA
Supply current (total) at $V_{DD} = V_{CC} = 5\ V \pm 10\%$; $V_{SS} = 0\ V$					
MAF8048H/35HL; MAF80A48H/A35HL	$I_{DD} + I_{CC}$	–	–	90	mA
MAF8049H/39HL; MAF80A49H/A39HL	$I_{DD} + I_{CC}$	–	–	100	mA
MAF8050H/40HL; MAF80A50H/A40HL	$I_{DD} + I_{CC}$	–	–	120	mA
Inputs					
Input voltage HIGH all inputs except XTAL 1, XTAL 2, $\overline{RESET}$	V_{IH}	2,2	–	V_{CC}	V
Input load current (P1.0 to P1.7, P2.0 to P2.7, EA, $\overline{SS}$) at $V_{SS} + 0{,}45\ V < V_I < V_{CC}$	$-I_{LI}$	–	–	0,6	mA

AC CHARACTERISTICS (MAB8048H/35HL; MAB8049H/39HL; MAB8050H/40HL)

$V_{CC} = V_{DD} = 5$ V (± 10%); $V_{SS} = 0$ V; $T_{amb} = 0$ to + 70 °C; note 1.
See waveforms Figs 14, 15, 16, 17 and 18

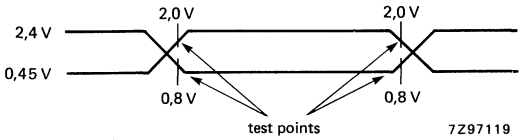
parameter	symbol	$f(t_{CL})$ (note 2)	11 MHz		unit
			min.	max.	
Clock period (note 2)	t_{CL}	$1/(f_{XTAL})$	90,9	1000	ns
ALE pulse duration	t_{LL}	$3{,}5t_{CL}-170$	150	–	ns
Address set-up time to ALE (note 3)	t_{AL}	$2t_{CL}-110$	70	–	ns
Address hold time after ALE	t_{LA}	$t_{CL}-40$	50	–	ns
Control pulse duration $\overline{RD}$, $\overline{WR}$	t_{CC1}	$7{,}5t_{CL}-200$	480	–	ns
Control pulse duration $\overline{PSEN}$	t_{CC2}	$6t_{CL}-200$	350	–	ns
Data set-up time before $\overline{WR}$	t_{DW}	$6{,}5t_{CL}-200$	390	–	ns
Data set-up time after $\overline{WR}$	t_{WD}	$t_{CL}-50$	40	–	ns
Data hold time					
$\overline{RD}$, $\overline{PSEN}$	t_{DR}	$1{,}5t_{CL}-30$	0	110	ns
$\overline{RD}$ to data input	t_{RD1}	$6t_{CL}-170$	–	350	ns
$\overline{PSEN}$ to data input	t_{RD2}	$4{,}5t_{CL}-170$	–	190	ns
Address set-up time to $\overline{WR}$	t_{AW}	$5t_{CL}-150$	300	–	ns
Address set-up time to data input ($\overline{RD}$)	t_{AD1}	$10{,}5t_{CL}-220$	–	730	ns
Address set-up time to data input ($\overline{PSEN}$)	t_{AD2}	$7{,}5t_{CL}-200$	–	460	ns
Address floating to $\overline{RD}$, $\overline{WR}$	t_{AFC1}	$2t_{CL}-40$	140	–	ns
Address floating to $\overline{PSEN}$ (note 3)	t_{AFC2}	$0{,}5t_{CL}-40$	10	–	ns
ALE to control pulse $\overline{RD}$, $\overline{WR}$	t_{LAFC1}	$3t_{CL}-75$	200	–	ns
ALE to control pulse $\overline{PSEN}$	t_{LAFC2}	$1{,}5t_{CL}-75$	60	–	ns
Control pulse to ALE $\overline{RD}$, $\overline{WR}$, $\overline{PROG}$	t_{CA1}	$t_{CL}-40$	50	–	ns
Control pulse to ALE $\overline{PSEN}$	t_{CA2}	$4t_{CL}-40$	320	–	ns
Port control set-up to $\overline{PROG}$	t_{CP}	$1{,}5t_{CL}-80$	50	–	ns
Port control hold to $\overline{PROG}$	t_{PC}	$4t_{CL}-260$	100	–	ns
$\overline{PROG}$ to Port 2 input must be valid	t_{PR}	$8{,}5t_{CL}-120$	–	650	ns
Input data hold time from $\overline{PROG}$	t_{PF}	$1{,}5t_{CL}$	0	150	ns
Output data set-up time	t_{DP}	$6t_{CL}-290$	250	140	ns
Output data hold time	t_{PD}	$1{,}5t_{CL}-90$	40	–	ns
$\overline{PROG}$ pulse duration	t_{PP}	$10{,}5t_{CL}-250$	700	–	ns
Port 2 I/O data set-up time to ALE	t_{PL}	$4t_{CL}-200$	160	–	ns
Port 2 I/O data hold time to ALE	t_{LP}	$1{,}5t_{CL}-120$	15	–	ns

AC CHARACTERISTICS (continued)

parameter	symbol	f(t_{CL}) (note 2)	11 MHz		unit
			min.	max.	
Port output from ALE	t_{PV}	$4{,}5t_{CL}+100$	–	510	ns
T0 repetition rate	t_{0PRR}	$3t_{CL}$	270	–	ns
Cycle time	t_{CY}	$15/f_{XTAL}$	1,36	15	μs
MAF8048H/35HL; MAF8049H/39HL; MAF8050H/40HL					
Clock period (note 2)	t_{CL}	$1/(f_{XTAL})$	90,8	1000	ns

Notes to AC characteristics

1. Control outputs: C_L = 80 pF.
   Bus outputs: C_L = 150 pF.
2. f(t_{CL}) assumes 50% duty cycle on XTAL 1 and XTAL 2; minimum frequency = 1 MHz.
3. Bus high-impedance load: 20 pF.



A.C. testing inputs are driven at 2,4 V for a logic 1 and 0,45 V for a logic 0. Output timing measurements are taken 2,0 V for a logic 1 and 0,8 V for a logic 0.

Fig. 13 A.C. testing input, output waveform.

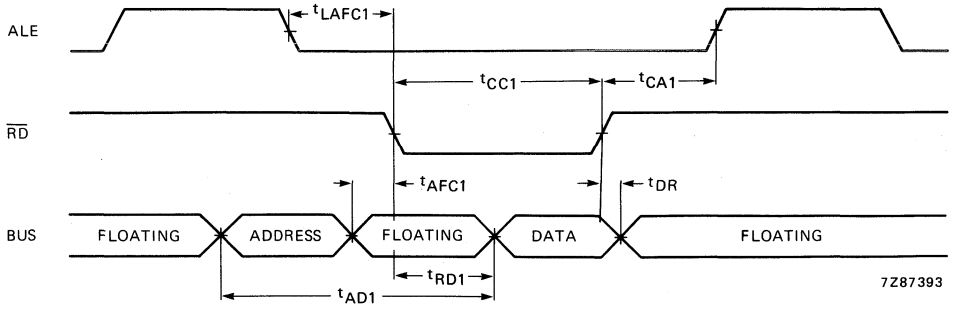


Fig. 14 Read from external data memory.

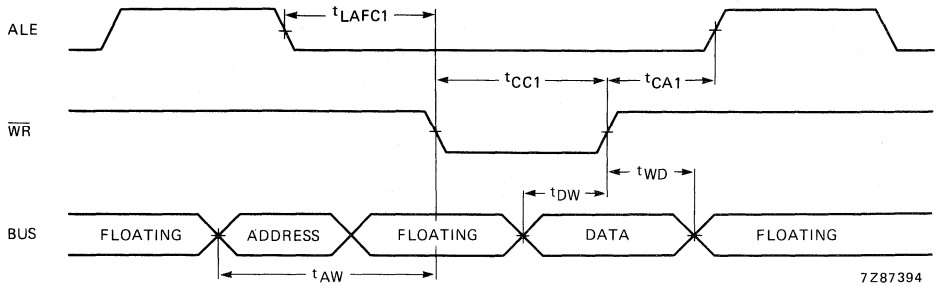


Fig. 15 Write to external memory.

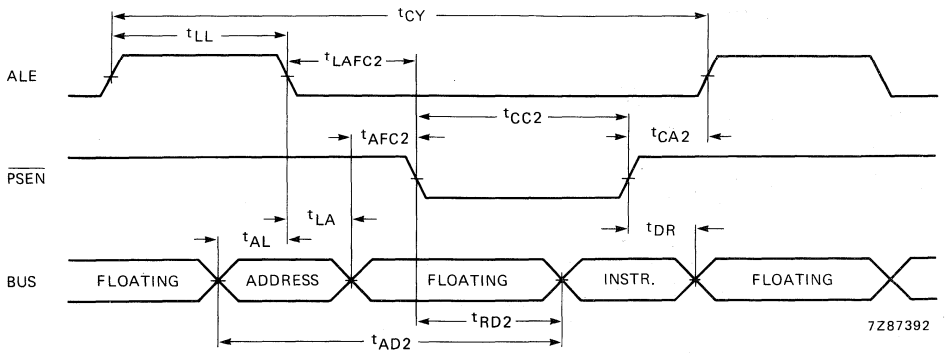


Fig. 16 Instruction fetch from program memory.

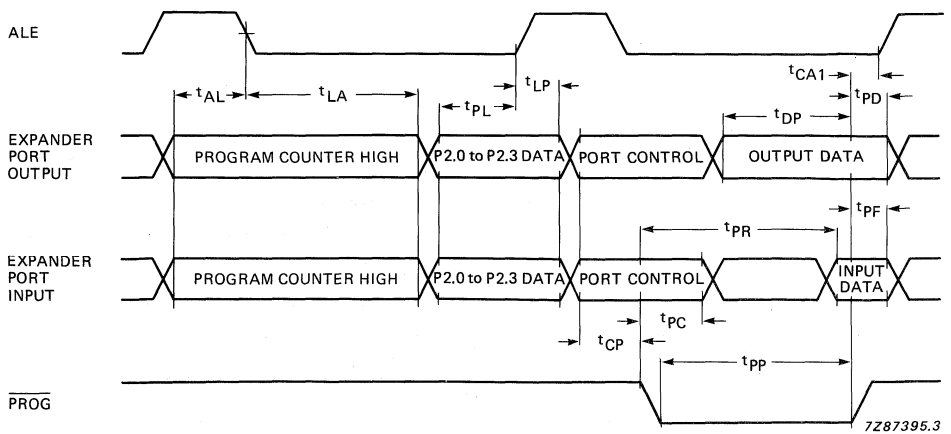


Fig. 17 Port 2 timing.

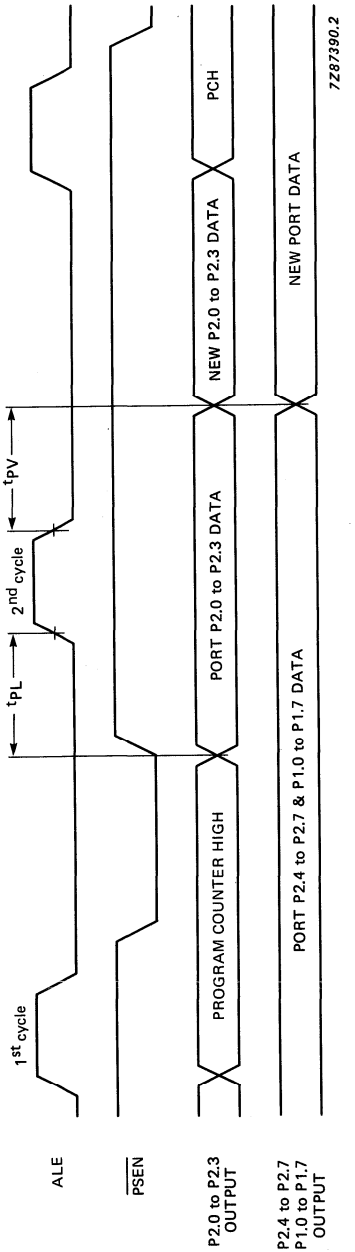


Fig. 18 I/O port timing.

SINGLE-CHIP 8-BIT CMOS MICROCONTROLLER

DESCRIPTION

The PC80CXX family of single-chip 8-bit CMOS microcontrollers consists of:

- The PCB80C49 with resident mask programmed 2 K x 8 ROM, 128 x 8 RAM.
- The PCB80C39 without resident program memory for use with external EPROM/ROM, 128 x 8 RAM.

All versions are pin and function compatible to their NMOS counter parts but with additional features and high performance.

The PC80CXX family are designed to be efficient control processors as well as arithmetic processors. Their instruction set allows the user to directly set and reset individual I/O, and to test individual individual bits within the accumulator. A large variety of branch and table look-up instructions enable efficient implementation of standard logic functions. Code efficiency is high; over 70% of the instructions are single byte; all others are two byte.
An on-chip 8-bit counter is provided, which can count either machine cycles (÷ 32) or external events. The counter can be programmed to cause an interrupt to the processor.
Program and data memories can be expanded using standard devices. Input/output capabilities can be expanded using standard devices.
The family has low power consumption and in addition a power down mode is provided.
For further detailed information see the 8048 family specification.

Features

- 8-bit CPU, ROM, RAM, I/O in a single 40-pin package
- PCB80C49: 2K x 8 ROM, 128 x 8 RAM
- Internal counter/timer
- Internal oscillator, clock driver
- Single-level interrupts: external and counter/timer
- 17 internal registers: accumulator, 16 addressable registers
- Over 90 instructions: 70% single byte
- All instructions: 1 or 2 cycles
- Easily expandable memory and I/O
- TTL compatible inputs and outputs
- Single 5 V supply
- Wide frequency operating range
- Low current consumption
- Available with extended temperature ranges: (PCB version) 0 to + 70 °C
  (PCF version) −40 to + 85 °C
  (PCA version) −40 to + 110 °C
- Frequency range: 1 to 15 MHz for all temperature ranges

APPLICATIONS

- Peripheral interfaces and controllers
- Test and measurement instruments
- Sequencers
- Audio/video systems
- Environmental control systems
- Modems and data enciphering

PACKAGE OUTLINES

PCB/F/A80C39/C49P: 40-lead DIL; plastic (SOT129).
PCB/F/A80C39/C49WP: 44-lead PLCC; plastic leaded chip carrier (SOT187AA).

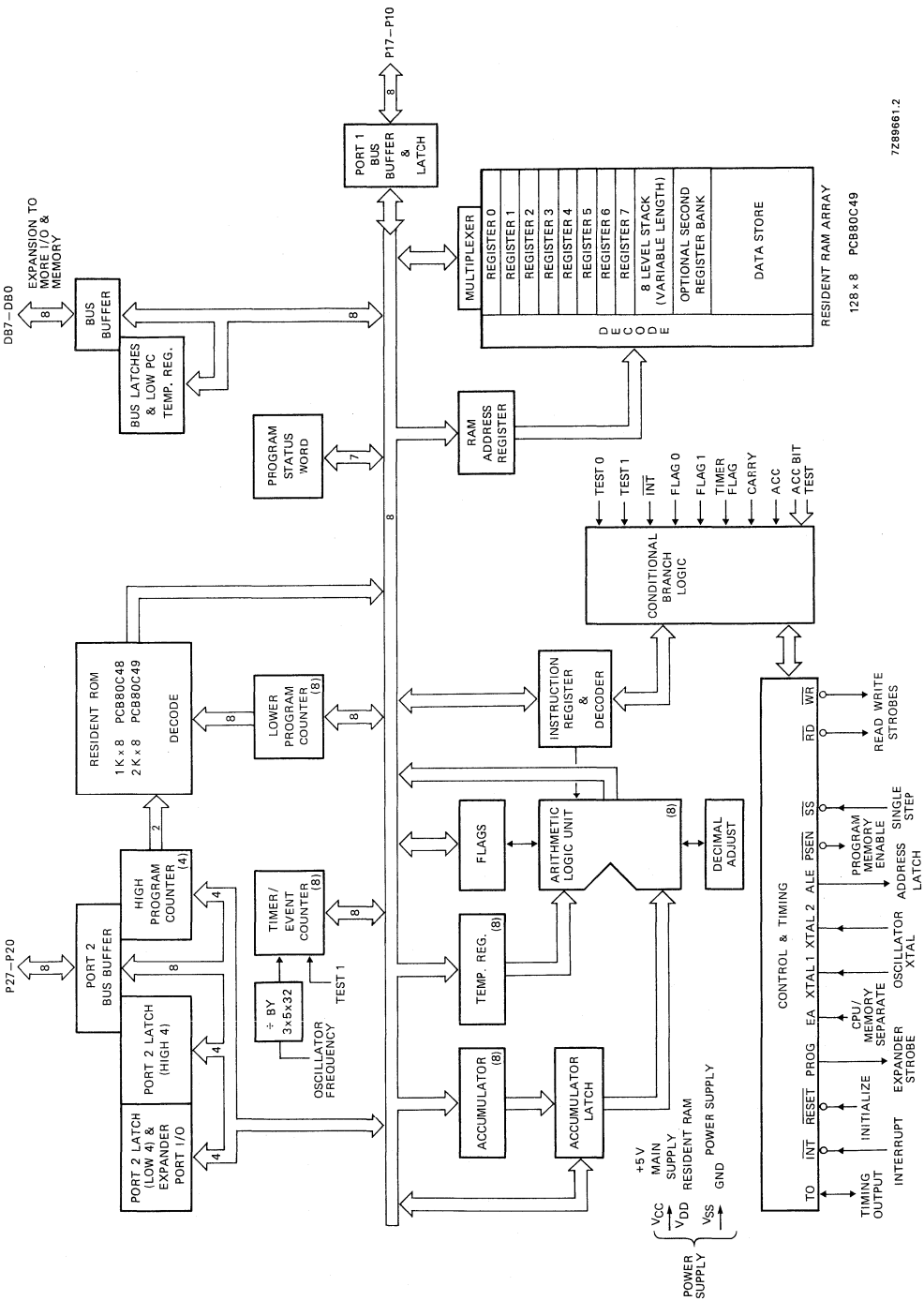


Fig. 1 Block diagram.

DEVELOPMENT DATA

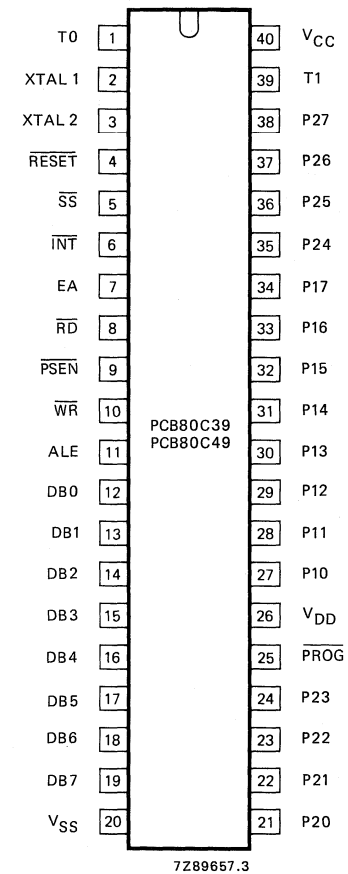


Fig. 2(a) Pinning diagram, DIL; for pin designation see next page.

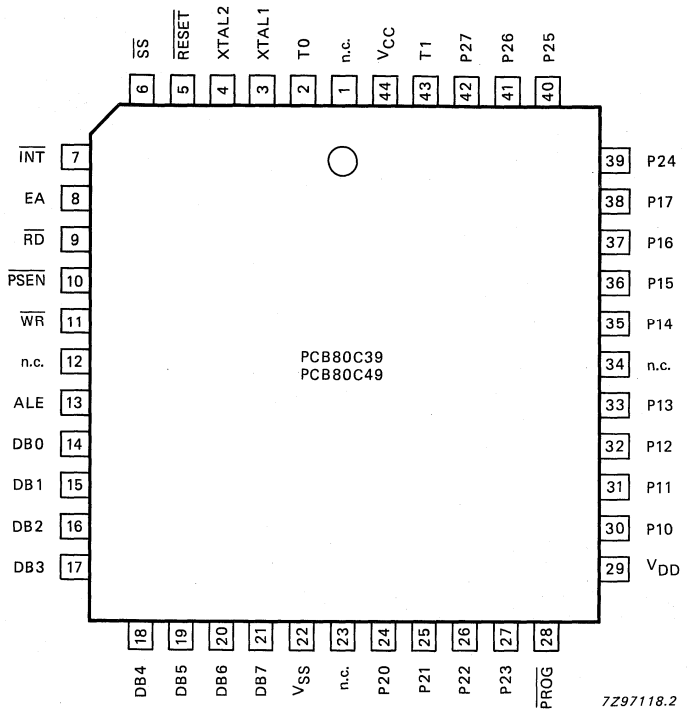


Fig. 2(b) Pinning diagram, PLCC.

Where: n.c. = not connected.

PIN DESIGNATION

designation	pin no.	function
DB0–DB7	12–19	**BUS.** Bidirectional I/O port that can be read or written using the $\overline{RD}$ and $\overline{WR}$ strobes. This port can also be statically latched. Contains the 8 lower order address bits during external memory access and receives the addressed instruction under control of $\overline{PSEN}$. $\overline{PSEN}$, ALE, $\overline{RD}$ and $\overline{WR}$ determine whether the access is an instruction fetch or a read/write access to external RAM.
P10–P17	27–34	**Port 1.** 8-bit quasi-bidirectional I/O port (note 1).
P20–P27	21–24, 35–38	**Port 2.** 8-bit quasi-bidirectional I/O port (note 1). P20–P23 contain the 4 higher order address bits during an access of external program memory.
$\overline{PROG}$	25	**Output strobe** (active LOW) for I/O expander.
T0	1	**Input pin** sensed using the JT0 and JNT0 instructions. **Clock output pin** when designated as such by the ENT0 CLK instruction.
T1	39	**Input pin** sensed using the JT1 and JNT1 instructions. Can be designated as the timer/counter input by the STRT CNT instruction.
$\overline{INT}$	6	**Interrupt input pin.** When LOW causes an interrupt in the current program if external interrupt is enabled. Can also be used as an input, testable using the JNI instruction. Interrupt is disabled during and after a RESET.
$\overline{RESET}$	4	**Reset input pin** used to initialize the microcontroller. Active LOW. During program verification the address is latched by a '0' to '1' transition on $\overline{RESET}$ and the data at the addressed location is output on BUS (note 2).
ALE	11	**Address latch enable.** Occurs once each machine cycle and is useful for timing and sampling. During external program or data memory access, ALE is used to strobe the address information multiplexed on the DB0 to DB7 outputs.
$\overline{RD}$	8	**Read BUS.** Active LOW strobe used to gate data onto BUS lines when reading from an external source.
$\overline{WR}$	10	**Write BUS.** Active LOW strobe used to write data from BUS lines to an external designation.
EA	7	**External access input.** When HIGH, forces instruction fetch from external memory.
$\overline{PSEN}$	9	**Program store enable.** Active LOW strobe that occurs only during a fetch from external program memory.
$\overline{SS}$	5	**Single step input.** Active LOW which is used with ALE to cause the microcontroller to execute a single instruction.
V_{DD}	26	**RAM power supply,** + 5 V during normal operation and power-down mode.

designation	pin no.	function
XTAL1	2	**One side of crystal** (or inductor) input for internal oscillator. Can also be used as an input for an external timing source (note 2).
XTAL2	3	**Other side of crystal.** XTAL2 must be driven with the inverted signal of XTAL1 when an external timing source of $11 < f_{osc} \leqslant 15$ MHz is used.
V_{SS}	20	**Ground.**
V_{CC}	40	**Mains power supply,** +5 V during normal operation.

Notes

1. Each port line can be designated as an input or an output. A line is designated as an input by first writing a logic 1 to the line. $\overline{\text{RESET}}$ sets all lines to logic 1.
2. Non-standard TTL V_{IH}.

FUNCTIONAL DESCRIPTION

Program memory (see Fig. 3)

The resident program memory is:
2048 byte ROM
The PCB80C39 has no resident program memory.
The total addressing capability is 4096 bytes.
The program memory address space is divided into two 2048-bytes banks MB0 and MB1.
The program memory is also divided into pages of 256 bytes for conditional branches.

There are three locations in program memory of special importance. These locations contain the first instruction to be executed after one of three events.
The three locations and their contents are:
location 0 – activation, then deactivation of the $\overline{\text{RESET}}$ line,
location 3 – activation of the $\overline{\text{INT}}$ line when the external interrupt is enabled,
location 7 – an overflow of the timer/counter if the T/C interrupt is enabled.

Data memory (see Fig. 4)

The resident data memory is: 128 byte RAM.

All locations are indirectly addressable by either of two RAM pointer registers at locations 0 and 1.
The first eight locations of RAM (0 to 7) are designated as working registers and are directly addressable by several instructions. By selecting register bank 1, RAM locations 24 to 31 become the working registers, replacing those in register bank 0 (0 to 7).
RAM locations 8 to 23 are designated as the stack. Two locations (bytes) are used per CALL, allowing up to eight levels of subroutine nesting.
If additional RAM is required, up to 256 bytes may be added and addressed directly using the MOVX instructions. If more RAM is required, an I/O port can be used to select one (256 byte) bank of external memory at a time.

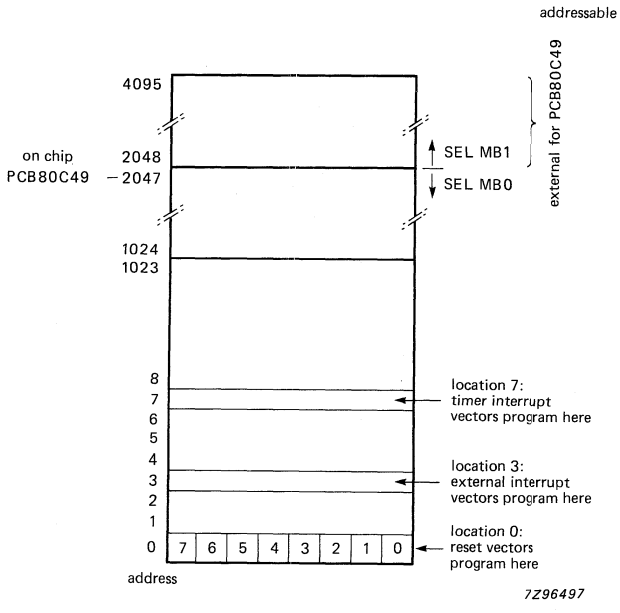


Fig. 3 Program memory map.

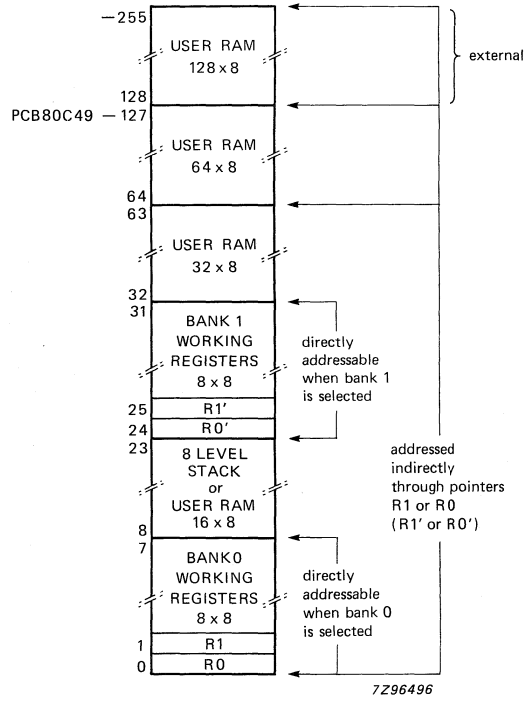


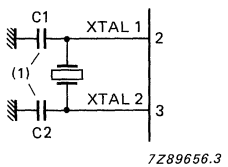
Fig. 4 Data memory map. In addition R0 or R1 (R0' or R1') may be used to address 256 words of an external RAM.

Program counter and stack

The program counter (PC) is a 12-bit counter/register that points to the location from which the next instruction is to be fetched. When EA is '0', the PC addresses an internal program memory. At the boundary of the internal program memory an automatic switch over to external memory is made. When EA is '1', all the program is fetched from external ROM/EPROM. The total address space is 4K bytes. An interrupt or CALL to a subroutine causes the contents of the program counter to be stored in one of the 8 register pairs of the program counter stack. The pair to be used is determined by a 3-bit stack pointer which is part of the program status word (PSW). Data RAM locations 8 to 23 are available as stack registers and are used to store the program counter and 4 bits of PSW. The stack pointer, when initialized to 000B, points to RAM locations 8 and 9. The first subroutine jump or interrupt results in the program counter contents being transferred to locations 8 and 9 of the RAM array. The stack pointer is then incremented by one to point to locations 10 and 11 in anticipation of another CALL. Nesting of subroutines within subroutines can continue up to 8 times without overflowing the stack. If overflow does occur the deepest address stored (location 8 and 9) will be overwritten and lost since the stack pointer overflows from 111 to 000. It also underflows from 000 to 111.
The end of a subroutine, which is signalled by a return instruction (RET or RETR), causes the stack pointer to be decremented and the contents of the resulting register pair to be transferred to the program counter.

Oscillator and clock

The PCB80C49 contains its own internal oscillator and clock driver.
A crystal, inductor or external pulse generator determines the oscillator frequency (see Figs 5, 6 and 7). The output of the oscillator is divided-by-three and is available at T0 (pin 1) by executing the ENT0 CLK instruction. This CLK signal is divided-by-five to define a machine (instruction) cycle. It is available at ALE (pin 11).



For quartz crystal
1 to 15 MHz: C1 = C2 = 15 to 25 pF
For ceramic resonators
1 to 15 MHz: C1 = C2 = 30^{+5}_{-10} pF

(1) Including crystal-socket stray capacitance.

Fig. 5 Crystal oscillator mode. Typical values are given. Crystal serial impedance should be $< 75\ \Omega$ at 6 MHz and $< 180\ \Omega$ at 3,6 MHz.

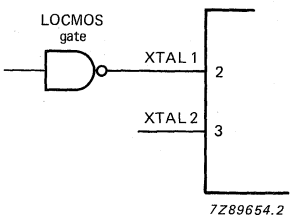
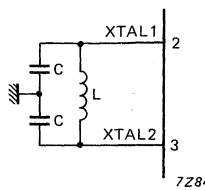


Fig. 6 Driving from an external source.
Test conditions at XTAL1;
Minimum HIGH ($> 0{,}7$ of V_{CC}) and LOW ($< 0{,}13$ of V_{CC}) times, should be at least 45% of a clock period.
XTAL2 must be driven with the inverted signal of XTAL1 when an external timing source of $11 < f_{osc} \leqslant 15$ MHz is used.



L (μH)	C (pF)	f_{nom} (MHz)
45	20	5,2
120	20	3,2

Fig. 7 LC oscillator mode.

FUNCTIONAL DESCRIPTION (continued)

Timer/event counter

An internal counter is available which can count either external events or machine cycles (÷ 32). The machine cycles are divided-by-32 before they are applied to the input of the 8-bit counter. External events are applied directly to the input of the counter. The maximum frequency that can be counted is one third of the machine cycle frequency. The minimum positive duty cycle that can be detected is 0,2 times the cycle period. The counter is under program control and can be made to generate an interrupt to the processor when it overflows.

Interrupt

An interrupt may be generated by either an external input ($\overline{INT}$, pin 6) or the overflow of the internal timer/event counter, when enabled. In either case, the processor completes execution of the present instruction and then does a CALL to the interrupt service routine, After service, a RETR instruction restores the machine to the state it was prior to the interrupt. The external interrupt has priority over the internal interrupt.

Input/output

The PCB80CXX family has 27 I/O lines. These lines are arranged as three 8-line ports, which serve individually as either inputs, outputs or together as bidirectional ports, plus 3 'test' inputs which can alter program sequences when tested by conditional jump instructions.

Ports 1 and 2

Ports 1 and 2 are each 8-bits wide and have identical characteristics. Data written to these ports is statically latched and remains unchanged until rewritten. As input ports these lines are non-latching, e.g., inputs must be present until read by an input instruction. Inputs are fully TTL compatible and outputs will drive one standard TTL load.

The lines of ports 1 and 2 are called quasi-bidirectional because of a special output circuit structure which allows each line to serve as an input, an output, or both even though outputs are statically latched. The circuit configuration is shown in Fig. 8. Each line has a unique high-impedance pull-up transistor TR3, this is turned on when the line is pulled above 2 V by an external source or by writing a logic 1 to the port. This pull-up is sufficient to provide the source current for a TTL HIGH level, yet can be pulled LOW by a standard TTL gate, thus allowing the same pin to used for both input and output. When a logic 1 is written to a line, a second high impedance transistor TR2, pulls the line up to 5 V. To provide fast switching during a '0' to '1' transition, a relatively low-impedance transistor TR1 (approx. 750 Ω) is switched on for 1/5 of a machine cycle whenever a '1' is written to the line. Whenever a '0' is written to the line, a low-impedance (approx. 250 Ω) transistor TR4, overcomes the weak light pull-up and provides TTL current sinking capability.

Since the pull-down transistor TR4 is a low-impedance device, a '1' must first be written to any line which is to be used as an input. RESET initializes all lines to the high impedance '1' state. This structure allows input and output on the same pin and also allows a mixture of input lines and output lines on the same port. The quasi-bidirectional port in combination with the ANL and ORL logical instructions provide an efficient means for handling single line inputs and outputs within an 8-bit processor.

BUS

BUS is a true bidirectional 8-bit port with associated input and output strobes. The BUS port can operate in three different modes: as a latched I/O port, as a bidirectional bus port, or in an expanded system as a program memory address output port. If the bidirectional feature is not needed, BUS can serve as either a statically latched output port or non-latching input port. Input and output lines on this port cannot be mixed. The BUS port lines are either active HIGH, active LOW, or high impedance (floating).

As a static port, data is written and latched using the OUTL instruction and input using the INS instruction. The INS and OUTL instructions generate pulses on the corresponding $\overline{RD}$ and $\overline{WR}$ output strobe lines; however, in the static port mode they are generally not used. As a bidirectional port, the MOVX instructions are used to read and write the port. A write to the port generates a pulse on the $\overline{WR}$ output line and output data is valid at the trailing edge of $\overline{WR}$. A read of the port generates a pulse on the $\overline{RD}$ output line and input data must be valid at the trailing edge of $\overline{RD}$.

The latched mode (INS, OUTL) is intended for use in the single-chip configuration, where BUS is not being used as an expanded port. OUTL and MOVX instructions can be mixed if required. However, when using a MOVX instruction a previously latched output will be lost and BUS will be left in the high impedance state. INS does not put the BUS in a high impedance state. Therefore, in order to read an external byte (and not the previously latched value) using the INS instruction, it is necessary to follow an OUTL with a MOVX instruction to place BUS in a high impedance state.

OUTL should never be used in a system with external program memory, since latching BUS may cause the next instruction to be incorrectly fetched.

Test (T0, T1) and $\overline{INT}$ inputs

Three pins serve as inputs and are testable with the conditional jump instruction. These pins are T0, T1, and $\overline{INT}$ and they allow inputs to cause program branches without the necessity to load an input port into the accumulator. The T0, T1 and $\overline{INT}$ pins have other possible functions as well.

DEVELOPMENT DATA

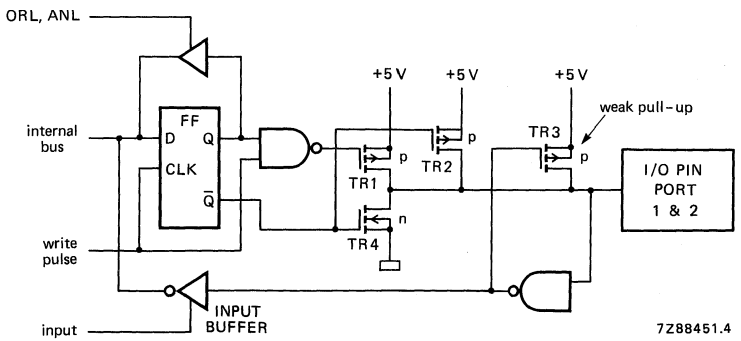


Fig. 8 Quasi-bidirectional port structure.

FUNCTIONAL DESCRIPTION (continued)

$\overline{\text{RESET}}$ input (see Fig. 10)

The $\overline{\text{RESET}}$ input provides a means to initialize the processor. This Schmitt-trigger input has an internal pull-up resistor. The combination of an external 47 kΩ resistor and a 1 μF capacitor provides a reset pulse of sufficient duration to guarantee that all circuitry is reset. If the reset pulse is generated otherwise, the $\overline{\text{RESET}}$ pin must be held at ground for at least 10 ms after the (0,13 V_{CC}) power supply is within tolerance. Only five machine cycles (2,5 μs at 6 MHz) are required if power is already on and the oscillator has stabilized.

Single step input ($\overline{\text{SS}}$)

Under control of the $\overline{\text{SS}}$ line, the processor can be forced to execute one instruction and then to wait until the single step switch is activated again.

IDLE mode

The PCB80CXX family is provided with a IDLE mode in which the internal oscillator, the internal timer and the external interrupt and counter are still functioning, while the status of following parts is maintained: RAM and register/Port 1 and 2/Bus. The IDLE mode is entered after execution of the IDLE instruction (opcode 1H). The IDLE mode is terminated by one of the two possible interrupts, if enabled, or a RESET signal. If an external interrupt terminates the IDLE mode, the next instruction that is executed is at location 3 of the program store. If a timer/counter interrupt terminates the IDLE mode, the next instruction is at location 7. The reset signal will terminate the IDLE mode, and also initialize the processor.

Power-down mode

In the PCB80CXX family, power can be removed from all but the 64 x 8 bit and 128 x 8 bit data RAM array, for low power standby operation. In the power-down mode the contents of the data RAM are maintained. V_{CC} serves as the + 5 V supply pin for most of the circuitry, while the V_{DD} pin supplies only the RAM array. In normal operation, both pins are at + 5 V. In standby, V_{CC} is at ground and V_{DD} is maintained at + 5 V.

Applying $\overline{\text{RESET}}$ to the processor through the $\overline{\text{RESET}}$ pin inhibits any access to the RAM by the processor and guarantees that the RAM cannot be inadvertently altered when power is removed from V_{CC}. Fig. 9 shows a typical power-down sequence.

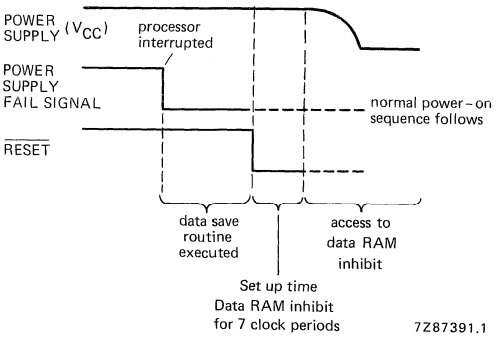


Fig. 9 Power-down sequence.

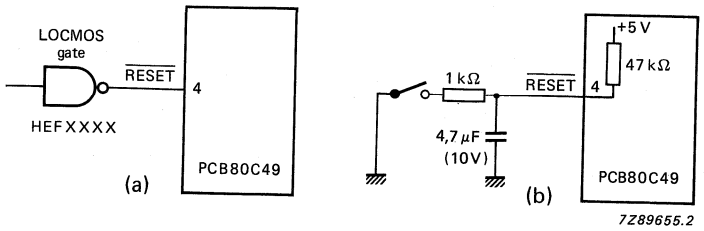


Fig. 10(a) External reset circuit; (b) power-on reset.

DEVELOPMENT DATA

Instruction set

The PCB80CXX instruction set consists of over 90 one and two-byte instructions (see Table 2). Program code efficiency is high because:

- working registers and program variables are stored in the RAM, which require only a single byte to address,
- program memory is divided into pages of 256 bytes, which means that branch destination addresses require one byte.

In addition to performing logical and arithmetic operations, the instruction set manipulates and tests both bits and bytes. A set of MOVE instructions operates indirectly upon either RAM or ROM, which permits efficient access of pointers and data tables. The indirect jump instruction performs a multi-way branch (up to 256) upon the content of the accumulator to addresses stored in a look-up table. The 'decrement register and jump if not zero' DJNZ instruction saves a byte every time it is used as opposed to using separate increment and test instructions.

The on-chip counter enables either external events or time to be counted off-line from the main program. The PCB80CXX can either test the counter (under program control) or cause its overflow to generate an interrupt. These features are required for real-time applications. Instruction timing is shown in Table 3.

Note: The OUTL, ANL and ORL instructions relating to BUS, are used with internal program memory only.

Table 1

Symbol definitions used in Table 2.

symbol	description
A	the accumulator
AC	the auxiliary carry flag
addr	program memory address (11-bits)
Bb	bit designation (b = 0–7)
BS	the bank switch
C	carry flag
CLK	clock signal
CNT	event counter
D	nibble designation (4-bits)
DBF	program memory bank flip-flop
data	number or expression (8-bits)
F0, F1	flags 0 and 1
I	interrupt
$\overline{INT}$	external interrupt
P	'in-page' operation designation
Pp	port designation (p = 1, 2 or 4–7)
PSW	program status word
Rr	register designation (r = 0, 1 or 0–7)
SP	stack pointer
T	timer
TF	timer flag
T0, T1	test 0 and 1 inputs
#	immediate data prefix
@	indirect address prefix
$	current value of program counter
←	is replaced by
↔	is exchanged with

Notes to Table 2.

1. Instruction code designations r and p form the binary representation of the registers and ports involved.
2. The dot under the appropriate flag bit indicates that its content is subject to change by the instruction it appears in.
3. Numerical subscripts appearing in the FUNCTION column reference the specific bits affected.

Table 2 Instruction set

	mnemonic	function	description	instruction code D_7	D_6	D_5	D_4	D_3	D_2	D_1	D_0	cycles	bytes	flags C	AC	F0	F1	BS
ACCUMULATOR	ADD A,Rr	(A)←(A) + (Rr) for r = 0–7	Add contents of designated register to A	0	1	1	0	1	r	r	r	1	1	•	•			
	ADD A,@Rr	(A)←(A) + ((Rr)) for r = 0–1	Add indirect the contents of the data memory location to A	0	1	1	0	0	0	0	r	1	1	•	•			
	ADD A,#data	(A)←(A) + data	Add immediate data to A	0 d_7	0 d_6	0 d_5	0 d_4	0 d_3	0 d_2	1 d_1	1 d_0	2	2	•	•			
	ADDC A,Rr	(A)←(A) + (C) + (Rr) for r = 0–7	Add with carry the contents of designated register to A	0	1	1	1	1	r	r	r	1	1	•	•			
	ADDC A,@Rr	(A)←(A) + (C) + ((Rr)) for r = 0–1	Add indirect with carry the contents of the data memory location to A	0	1	1	1	0	0	0	r	1	1	•	•			
	ADDC A,#data	(A)←(A) + (C) + data	Add immediate data with carry to A	0 d_7	0 d_6	0 d_5	1 d_4	0 d_3	0 d_2	1 d_1	1 d_0	2	2	•	•			
	ANL A,Rr	(A)←(A) AND (Rr) for r = 0–7	Logical AND contents of designated register with A	0	1	0	1	1	r	r	r	1	1					
	ANL A,@Rr	(A)←(A) AND ((Rr)) for r = 0–1	Logical AND indirect the contents of data memory with A	0	1	0	1	0	0	0	r	1	1					
	ANL A,#data	(A)←(A) AND data	Logical AND immediate data with A	0 d_7	1 d_6	0 d_5	1 d_4	0 d_3	0 d_2	1 d_1	1 d_0	2	2					
	CLR A	(A)←0	Clear the contents of A	0	0	1	0	0	1	1	1	1	1					
	CPL A	(A)←NOT(A)	Complement the contents of A	0	0	1	1	0	1	1	1	1	1					
	DA A		Decimal adjust the contents of A	0	1	0	1	0	1	1	1	1	1	•	•			
	DEC A	(A)←(A) −1	Decrement the contents of A by 1	0	0	0	0	0	1	1	1	1	1					
	INC A	(A)←(A) + 1	Increment the contents of A by 1	0	0	0	1	0	1	1	1	1	1					
	ORL A,Rr	(A)←(A) OR (Rr) for r = 0–7	Logical OR contents of designated register with A	0	1	0	0	1	r	r	r	1	1					
	ORL A,@Rr	(A)←(A) OR ((Rr)) for r = 0–1	Logical OR indirect the contents of data memory location with A	0	1	0	0	0	0	0	r	1	1					

DEVELOPMENT DATA

ACCUMULATOR (continued)	ORL A,#data	(A)←(A) OR data	Logical OR immediate data with A	0 d_7	1 d_6	0 d_5	0 d_4	0 d_3	0 d_2	1 d_1	1 d_0	2	2						
	RL A	(A_{n+1})←(A_n) (A_0)←(A_7) for n = 0–6	Rotate A left by 1-bit without carry	1	1	1	0	0	1	1	1	1	1						
	RLC A	(A_{n+1})←(A_n) (A_0)←(C) (C)←(A_7) for n = 0–6	Rotate A left by 1-bit through carry	1	1	1	1	0	1	1	1	1	1	•					
	RR A	(A_n)←(A_{n+1}) (A_7)←(A_0) for n = 0–6	Rotate A right by 1-bit without carry	0	1	1	1	0	1	1	1	1	1						
	RRC A	(A_n)←(A_{n+1}) (A_7)←(C) (C)←(A_0) n = 0–6	Rotate A right by 1-bit through carry	0	1	1	0	0	1	1	1	1	1	•					
	SWAP A	(A_{4-7})↔(A_{0-3})	Swap the two 4-bit nibbles in A	0	1	0	0	0	1	1	1	1	1						
	XRL A,Rr	(A)←(A) XOR (Rr)	Logical XOR contents of designated register with A	1	1	0	1	1	r	r	r	1	1						
	XRL A,@Rr	(A)←(A) XOR ((Rr))	Logical XOR indirect the contents of data memory location with A	1	1	0	1	0	0	0	r	1	1						
	XRL A,#data	(A)←(A) XOR data	Logical XOR immediate data with A	1 d_7	1 d_6	0 d_5	1 d_4	0 d_3	0 d_2	1 d_1	1 d_0	2	2						

Table 2 (continued)

	mnemonic	function	description	instruction code D_7 D_6 D_5 D_4 D_3 D_2 D_1 D_0	cycles	bytes	flags C	AC	F0	F1	BS
BRANCH	DJNZ Rr, addr	(Rr)←(Rr) −1 if (Rr) not zero: (PC_{0-7})←addr	Decrement the specified register and test contents	1 1 1 0 1 r r r a_7 a_6 a_5 a_4 a_3 a_2 a_1 a_0	2	2					
	JBb addr	(PC_{0-7})←addr if Bb = 1; (PC)←(PC) + 2 if Bb = 0	Jump to specified address if accumulator bit is set	b_2 b_1 b_0 1 0 0 1 0 a_7 a_6 a_5 a_4 a_3 a_2 a_1 a_0	2	2					
	JC addr	(PC_{0-7})←addr if C = 1; (PC)←(PC) + 2 if C = 0	Jump to specified address if carry flag is set	1 1 1 1 0 1 1 0 a_7 a_6 a_5 a_4 a_3 a_2 a_1 a_0	2	2					
	JF0 addr	(PC_{0-7})←addr if (F0 = 1; (PC)←(PC) + 2 if F0 = 0	Jump to specified address if flag F0 is set	1 0 1 1 0 1 1 0 a_7 a_6 a_5 a_4 a_3 a_2 a_1 a_0	2	2					
	JF1 addr	(PC_{0-7})←addr if F1 = 1; (PC)←(PC) + 2 if F1 = 0	Jump to specified address if flag F1 is set	0 1 1 1 0 1 1 0 a_7 a_6 a_5 a_4 a_3 a_2 a_1 a_0	2	2					
	JMP addr	(PC_{8-10})←$addr_{8-10}$ (PC_{0-7})←$addr_{0-7}$ (PC_{11})←(DBF)	Direct jump to specified address within the 2K address block	a_{10} a_9 a_8 0 0 1 0 0 a_7 a_6 a_5 a_4 a_3 a_2 a_1 a_0	2	2					
	JMPP @A	(PC_{0-7})←((A))	Jump indirect to specified address within address page	1 0 1 1 0 0 1 1	2	1					
	JNC addr	(PC_{0-7})←addr if C = 0; (PC)←(PC) + 2 if C = 1	Jump to specified address if carry flag is LOW	1 1 1 0 0 1 1 0 a_7 a_6 a_5 a_4 a_3 a_2 a_1 a_0	2	2					
	JNI	(PC_{0-7})←addr if $\overline{INT}$ = 0; (PC)←(PC) + 2 if $\overline{INT}$ = 1	Jump to specified address if $\overline{INT}$ input is LOW	1 0 0 0 0 1 1 0 a_7 a_6 a_5 a_4 a_3 a_2 a_1 a_0	2	2					
	JNT0 addr	(PC_{0-7})←addr if T0 = 0; (PC)←(PC) + 2 if T0 = 1	Jump to specified address if T0 is LOW	0 0 1 0 0 1 1 0 a_7 a_6 a_5 a_4 a_3 a_2 a_1 a_0	2	2					
	JNT1 addr	(PC_{0-7})←addr if T1 = 0; (PC)←(PC) + 2 if T1 = 1	Jump to specified address if T1 is LOW	0 1 0 0 0 1 1 0 a_7 a_6 a_5 a_4 a_3 a_2 a_1 a_0	2	2					

BRANCH (continued)	JNZ addr	$(PC_{0-7}) \leftarrow$addr if $A \neq 0$; $(PC) \leftarrow (PC) + 2$ if $A = 0$	Jump to specified address if A is non-zero	1 a_7	0 a_6	0 a_5	1 a_4	0 a_3	1 a_2	1 a_1	0 a_0	2	2					
	JTF addr	$(PC_{0-7}) \leftarrow$addr if TF = 1; $(PC) \leftarrow (PC) + 2$ if TF = 0	Jump to specified address if timer flag is set to 1	0 a_7	0 a_6	0 a_5	1 a_4	0 a_3	1 a_2	1 a_1	0 a_0	2	2					
	JT0 addr	$(PC_{0-7}) \leftarrow$addr if T0 = 1; $(PC) \leftarrow (PC) + 2$ if T0 = 0	Jump to specified address if T0 = 1	0 a_7	0 a_6	1 a_5	1 a_4	0 a_3	1 a_2	1 a_1	0 a_0	2	2					
	JT1 addr	$(PC_{0-7}) \leftarrow$addr if T1 = 1; $(PC) \leftarrow (PC) + 2$ if T1 = 0	Jump to specified address if T1 = 1	0 a_7	1 a_6	0 a_5	1 a_4	0 a_3	1 a_2	1 a_1	0 a_0	2	2					
	JZ addr	$(PC_{0-7}) \leftarrow$addr if $A = 0$; $(PC) \leftarrow (PC) + 2$ if $A \neq 0$	Jump to specified address if A is zero	1 a_7	1 a_6	0 a_5	0 a_4	0 a_3	1 a_2	1 a_1	0 a_0	2	2					
CONTROL	EN I		Enable external ($\overline{INT}$) interrupt	0	0	0	0	0	1	0	1	1	1					
	DIS I		Disable external ($\overline{INT}$) interrupt	0	0	0	1	0	1	0	1	1	1					
	SEL RB0	(BS)←0	Select bank 0 (locations 0–7) of data memory	1	1	0	0	0	1	0	1	1	1					•
	SEL RB1	(BS)←1	Select bank 1 (locations 24–31) of data memory	1	1	0	1	0	1	0	1	1	1					•
	SEL MB0	(DBF)←0	Select program memory bank 0; addresses 0–2047	1	1	1	0	0	1	0	1	1	1					
	SEL MB1	(DBF)←1	Select program memory bank 1; addresses 2048–4095	1	1	1	1	0	1	0	1	1	1					
	ENT0 CLK		Enable clock output onto T0	0	1	1	1	0	1	0	1	1	1					

Table 2 (continued)

	mnemonic	function	description	instruction code D_7 D_6 D_5 D_4 D_3 D_2 D_1 D_0	cycles	bytes	flags C	AC	F0	F1	BS
DATA MOVES	MOV A,#data	(A)←data	Move immediate data into A	0 0 1 0 0 0 1 1 d_7 d_6 d_5 d_4 d_3 d_2 d_1 d_0	2	2					
	MOV A,Rr	(A)←(Rr) for r = 0–7	Move the contents of the designated register into A	1 1 1 1 1 r r r	1	1					
	MOV A,@Rr	(A)←((R)) for r = 0–1	Move indirect the contents of data memory into A	1 1 1 1 0 0 0 r	1	1					
	MOV A,PSW	(A)←(PSW)	Move contents of the program status word into A	1 1 0 0 0 1 1 1	1	1					
	MOV Rr,#data	(Rr)←data for r = 0–7	Move immediate data into the designated register	1 0 1 1 1 r r r d_7 d_6 d_5 d_4 d_3 d_2 d_1 d_0	2	2					
	MOV Rr,A	(Rr)←(A) for r = 0–7	Move A contents into the designated register	1 0 1 0 1 r r r	1	1					
	MOV @Rr,A	((Rr))←(A) for r = 0–1	Move indirect A contents into data memory location	1 0 1 0 0 0 0 r	1	1					
	MOV @Rr,#data	((Rr))←data for r = 0–1	Move indirect the specified data into data memory	1 0 1 1 0 0 0 r d_7 d_6 d_5 d_4 d_3 d_2 d_1 d_0	2	2					
	MOV PSW,A	(PSW)←A	Move contents of A into the program status word	1 1 0 1 0 1 1 1	1	1	•	•	•		•
	MOVP A,@A	(A)←((A))	Move data in the current page into A	1 0 1 0 0 0 1 1	2	1					
	MOVP3 A,@A	(A)←((A)) in page 3	Move data in page 3 of memory bank Ø into A	1 1 1 0 0 0 1 1	2	1					
	MOVX A,@Rr	(A)←((R)) for r = 0–1	Move indirect the contents of external memory location into A	1 0 0 0 0 0 0 r	2	1					
	MOVX @Rr,A	((Rr))←(A) for r = 0–1	Move indirect the contents of A into external memory	1 0 0 1 0 0 0 r	2	1					
	XCH A,Rr	(A)↔(Rr) for r = 0–7	Exchange A with designated register contents	0 0 1 0 1 r r r	1	1					
	XCH A,@Rr	(A)↔((R)) for r = 0–1	Exchange indirect A contents with location in data memory	0 0 1 0 0 0 0 r	1	1					

DEVELOPMENT DATA

D.M.	XCHD A,@Rr	$(A_{0-3}) \leftrightarrow (Rr_{0-3})$ for r = 0–1	Exchange indirect 4-bit contents of A with data memory	0	0	1	1	0	0	0	r	1	1					
FLAGS	CPL C	(C)←NOT (C)	Complement content of carry bit	1	0	1	0	0	1	1	1	1	1	•				
	CPL F0	(F0)←NOT (F0)	Complement content of flag F0	1	0	0	1	0	1	0	1	1	1			•		
	CPL F1	(F1)←NOT (F1)	Complement content of flag F1	1	0	1	1	0	1	0	1	1	1				•	
	CLR C	(C)←0	Clear content of carry bit to 0	1	0	0	1	0	1	1	1	1	1	•				
	CLR F0	(F0)←0	Clear content of flag F0 to 0	1	0	0	0	0	1	0	1	1	1			•		
	CLR F1	(F1)←0	Clear content of flag F1 to 0	1	0	1	0	0	1	0	1	1	1				•	
INPUT/OUTPUT	ANL BUS,#data	(BUS)←(BUS) AND data	Logical AND immediate data with BUS	1 d_7	0 d_6	0 d_5	1 d_4	1 d_3	0 d_2	0 d_1	0 d_0	2	2					
	ANL Pp,#data	(Pp)←(Pp) AND data; p = 1–2	Logical AND immediate data with designated port (1 or 2)	1 d_7	0 d_6	0 d_5	1 d_4	1 d_3	0 d_2	p d_1	p d_0	2	2					
	ANLD Pp,A	(Pp)←(Pp) AND (A_{0-3}); p = 4–7	Logical AND contents of A with designated port (4–7)	1	0	0	1	1	1	p	p	2	1					
	IN A,Pp	(A)←(Pp) p = 1–2	Input data from designated port (1–2) into A	0	0	0	0	1	0	p	p	2	1					
	INS A,BUS	(A)←(BUS)	Input strobed BUS data into A	0	0	0	0	1	0	0	0	2	1					
	MOVD A,Pp	(A_{0-3})←(Pp); p = 4–7 (A_{4-7})←0	Move contents of designated port (4–7) into A	0	0	0	0	1	1	p	p	2	1					
	MOVD Pp,A	(Pp)←(A_{0-3}) p = 4–7	Move contents of A to designated port (4–7)	0	0	1	1	1	1	p	p	1	1					
	ORLD Pp,A	(Pp)←(Pp) OR (A_{0-3}); p = 4–7	Logical OR contents of A with designated port (4–7)	1	0	0	0	1	1	p	p	1	1					
	ORL BUS,#data	(BUS)←(BUS) OR data	Logical OR immediate data with BUS	1 d_7	0 d_6	0 d_5	0 d_4	1 d_3	0 d_2	0 d_1	0 d_0	2	2					
	ORL Pp,#data	(Pp)←(Pp) OR data p = 1–2	Logical OR immediate data with designated port (1–2)	1 d_7	0 d_6	0 d_5	0 d_4	1 d_3	0 d_2	p d_1	p d_0	2	2					
	OUTL BUS,A	(BUS)←(A)	Output contents A onto BUS	0	0	0	0	0	0	1	0	2	1					
	OUTL Pp,A	(Pp)←(A) p = 1–2	Output contents A to designated port (1–2)	0	0	1	1	1	0	p	p	2	1					

Table 2 (continued)

	mnemonic	function	description	instruction code D_7 D_6 D_5 D_4 D_3 D_2 D_1 D_0	cycles	bytes	flags C	AC	F0	F1	BS
REGISTER	DEC Rr	(Rr)←(Rr) −1 for r = 0–7	Decrement contents of designated register by 1	1 1 0 0 1 r r r	1	1					
	INC Rr	(Rr)←(Rr) + 1 for r = 0–7	Increment contents of designated register by 1	0 0 0 1 1 r r r	1	1					
	INC @Rr	((R))←((Rr)) + 1 for r = 0–1	Increment indirect the contents of data memory location by 1	0 0 0 1 0 0 0 r	1	1					
SUBROUTINE	CALL addr	((SP))←(PC), (PSW_{4-7}) (SP)←(SP) + 1 (PC_{8-10})←$addr_{8-10}$ (PC_{0-7})←$addr_{0-7}$	Call designated subroutine	a_{10} a_9 a_8 1 0 1 0 0 a_7 a_6 a_5 a_4 a_3 a_2 a_1 a_0	2	2					
	RET	(SP)←(SP) −1 (PC)←((SP))	Return from subroutine without restoring program status word	1 0 0 0 0 0 1 1	2	1					
	RETR	(SP)←(SP) −1 (PC)←((SP)) (PSW_{4-7})←((SP))	Return from subroutine restoring program status word	1 0 0 1 0 0 1 1	2	1					
TIMER/COUNTER	EN TCNTI		Enable timer/counter interrupt	0 0 1 0 0 1 0 1	1	1					
	DIS TCNTI		Disable timer/counter interrupt	0 0 1 1 0 1 0 1	1	1					
	MOV A,T	(A)←(T)	Move contents of timer/counter into A	0 1 0 0 0 0 1 0	1	1					
	MOV T,A	(T)←(A)	Move contents of A into timer/counter	0 1 1 0 0 0 1 0	1	1					
	STOP TCNT		Stop count for event counter or timer	0 1 1 0 0 1 0 1	1	1					
	STRT CNT		Start count for event counter	0 1 0 0 0 1 0 1	1	1					
	STRT T		Start count for timer	0 1 0 1 0 1 0 1	1	1					
	IDLE		Entering IDLE mode	0 0 0 0 0 0 0 1	1	1					
	NOP		No operation	0 0 0 0 0 0 0 0	1	1					

Table 3 Instruction timing (see also Figs 11 and 12)

	cycle 1					cycle 2				
instruction	S1	S2	S3	S4	S5	S1	S2	S3	S4	S5
IN A,P	fetch instruction	increment program counter	–	increment timer	–	–	read port	* –	–	–
OUTL P,A	|	|	–	|	output to port	–	–	* –	–	–
ANL P,#data	|	* |	–	|	read port	fetch immediate data	–	*increment program counter	output to port	–
ORL P,#data	|	* |	–	|	read port	fetch immediate data	–	*increment program counter	output to port	–
INS A,BUS	|	|	–	|	–	–	read port	* –	–	–
OUTL BUS,A	|	|	–	|	output to port	–	–	* –	–	–
ANL BUS,#data	|	* |	–	|	read port	fetch immediate data	–	*increment program counter	output to port	–
ORL BUS,#data	|	* |	–	|	read port	fetch immediate data	–	*increment program counter	output to port	–
MOVX @R,A	|	|	output RAM address	|	output data to RAM	–	–	* –	–	–
MOVX A,@R	|	|	output RAM address	|	–	–	read data	* –	–	–
MOVD A,P	| fetch instruction	increment program counter	output opcode/ address	| increment timer	–	–	read P2 lower	* –	–	–

Table 3 Instruction timing (*continued*)

	cycle 1					cycle 2				
instruction	S1	S2	S3	S4	S5	S1	S2	S3	S4	S5
MOVD P,A	fetch instruction	increment program counter	output opcode/ address	increment timer	output data to P2 lower	–	–	* –	–	–
ANLD P,A	| | |		output opcode/ address	| | |	output data	–	–	* –	–	–
ORLD P,A	| | |		output opcode/ address	| | |	output data	–	–	* –	–	–
J (conditional)	| | |	*	sample condition	| increment timer	–	fetch immediate data	–	*update program counter	–	–
STRT CNT STRT T	| |	*	–	–	start counter					
STOP TCNT	| |	*	–	–	stop counter					
EN I	| |	*	–	enable interrupt	–					
DIS I	| |	*	–	disable interrupt	–					
ENT0 CLK	fetch instruction	*increment program counter	–	enable clock	–					

* Valid instruction addresses are output at this time if external program memory is being accessed.

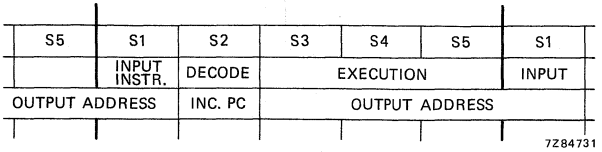


Fig. 11 Instruction cycle.

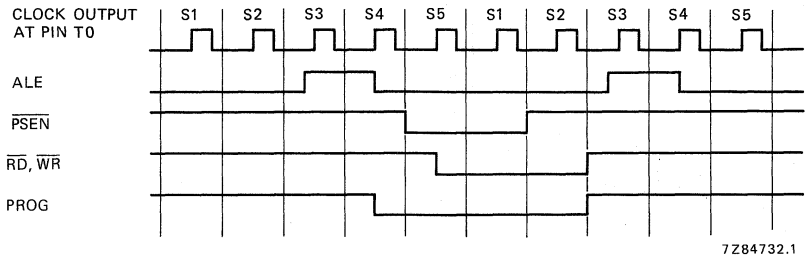


Fig. 12 Instruction cycle timing.

DEVELOPMENT DATA

Table 4 Instruction map.

first hexadicimal character of opcode	second hexadicimal character of opcode														
	0 1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
0	NOP IDLE mode	OUTL BUS,A	ADD A, #data	JMP page 0	EN I		DEC A	INS A,BUS	IN A, Pp 1	2		MOVD A,Pp 4	5	6	7
1	INC @Rr 0 1	JB0 addr	ADDC A, #data	CALL page 0	DIS I	JTF addr	INC A	INC Rr 0	1	2	3	4	5	6	7
2	XCH A, @Rr 0 1		MOV A, #data	JMP page 1	EN TCNTI	JNT0 addr	CLR A	XCH A,Rr 0	1	2	3	4	5	6	7
3	XCHD A, @Rr 0 1	JB1 addr		CALL page 1	DIS TCNTI	JT0 addr	CPL A		OUTL Pp,A 1	2		MOVD Pp,A 4	5	6	7
4	ORL A, @Rr 0 1	MOV A, T	ORL A, #data	JMP page 2	STRT CNT	JNT1 addr	SWAP A	ORL A,Rr 0	1	2	3	4	5	6	7
5	ANL A, @Rr 0 1	JB2 addr	ANL A, #data	CALL page 2	STRT T	JT1 addr	DA, A	ANL A,Rr 0	1	2	3	4	5	6	7
6	ADD A, @Rr 0 1	MOV T,A		JMP page 3	STOP TCNT		RRC A	ADD A,Rr 0	1	2	3	4	5	6	7
7	ADDC A, @Rr 0 1	JB3 addr		CALL page 3	ENT0 CLK	JF1 addr	RR A	ADDC A,Rr 0	1	2	3	4	5	6	7
8	MOVX A, @Rr 0 1		RET	JMP page 4	CLR F0	JN1 addr		ORL BUS, #data	ORL Pp, #data 1	2		ORLD Pp,A 4	5	6	7
9	MOVX @Rr,A 0 1	JB4 addr	RETR	CALL page 4	CPL F0	JNZ addr	CLR C	ANL BUS, #data	ANP Pp, #data 1	2		ANLD Pp,A 4	5	6	7
A	MOV @Rr, A 0 1		MOVP A, @A	JMP page 5	CLR F1		CPL C	MOV Rr,A 0	1	2	3	4	5	6	7
B	MOV @Rr, #data 0 1	JB5 addr	JMPP @A	CALL page 5	CPL F1	JF0 addr		MOV R, #data 0	1	2	3	4	5	6	7
C				JMP page 6	SEL RB0	JZ addr	MOV A, PSW	DEC Rr 0	1	2	3	4	5	6	7
D	XRL A, @Rr 0 1	JB6 addr	XRL A, #data	CALL page 6	SEL RB1		MOV PSW, A	XRL A,Rr 0	1	2	3	4	5	6	7
E			MOVP3 A @A	JMP page 7	SEL MB0	JNC addr	RL A	DJNZ Rr, addr 0	1	2	3	4	5	6	7
F	MOV A, @Rr 0 1	JB7 addr		CALL page 7	SEL MB1	JC addr	RLC A	MOV A,Rr 0	1	2	3	4	5	6	7

RATINGS

Limiting values in accordance with the Absolute Maximum System (IEC 134)

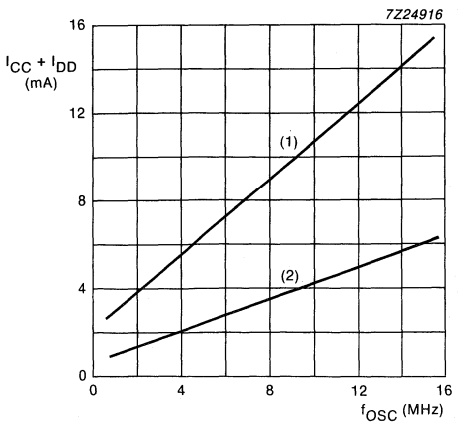
parameter	symbol	min.	max.	unit
Input, output current on any single pin	I_I, I_O	–	± 10	mA
Total power dissipation	P_{tot}	–	0.5	W
Storage temperature range	T_{stg}	–65	+ 150	°C
Operating ambient temperature range				
PCB80C39/49 version	T_{amb}	0	+ 70	°C
PCF80C39/49 version	T_{amb}	–40	+ 85	°C
PCA80C39/49 version	T_{amb}	–40	+ 110	°C

DC CHARACTERISTICS

$V_{CC} = V_{DD} = 5$ V (± 10%); $V_{SS} = 0$ V; $T_{amb} = 0$ to + 70 °C (PCB80C39/49); –40 to + 85 °C (PCF80C39/49); –40 to + 110 °C (PCA80C39/49). All voltages with respect to V_{SS} unless otherwise specified.

DEVELOPMENT DATA

parameter	conditions	symbol	min.	max.	unit
Supply voltage	$V_{CC} = V_{DD}$ (note 2)	V_{CC}	4.5	5.5	V
Supply current					
operating	f_{CLK} = 15 MHz	$I_{CC} + I_{DD}$	–	15	mA
IDLE mode	f_{CLK} = 15 MHz	I_{IDLE}	–	6	mA
power down mode	V_{DD} = 2 V; $\overline{RESET}$ = LOW	I_{PD}	–	2	μA
Inputs					
Input voltage LOW all inputs except $\overline{RESET}$; XTAL1; XTAL2		V_{IL}	–0.5	0.18 V_{CC}	V



(1) Operational mode.
(2) Idle mode.

Fig.13 Typical values of maximum supply current ($I_{CC} + I_{DD}$) as a function of the oscillator frequency (f_{OSC}) at $V_{CC(max.)}$ = 5.5 V; T_{amb} = 0 to + 70 °C.

DC CHARACTERISTICS (continued)

parameter	conditions	symbol	min.	max.	unit
Input voltage LOW $\overline{RESET}$; XTAL1; XTAL2		V_{IL1}	−0.5	0.13 V_{CC}	V
Input voltage HIGH all inputs except $\overline{RESET}$; XTAL1; XTAL2	(PCB80C39/49 version)	V_{IH}	0.4 V_{DD}	V_{CC}	V
	(PCF/PCA80C39/49 version note 1)	V_{IH}	0.43 V_{DD}	V_{CC}	V
Input voltage HIGH $\overline{RESET}$; XTAL1; XTAL2		V_{IH1}	0.7 V_{DD}	V_{CC}	V
Outputs					
Output voltage LOW BUS	I_{OL} = 2 mA	V_{OL}	–	0.45	V
Output voltage LOW $\overline{RD}$; $\overline{WR}$; $\overline{PSEN}$; ALE	I_{OL} = 1.8 mA	V_{OL1}	–	0.45	V
Output voltage LOW $\overline{PROG}$	I_{OL} = 1 mA	V_{OL2}	–	0.45	V
Output voltage LOW all other outputs	I_{OL} = 1.6 mA	V_{OL3}	–	0.45	V
Output voltage HIGH BUS	$-I_{OH}$ = 400 μA	V_{OH}	0.75 V_{CC}	–	V
Output voltage HIGH $\overline{RD}$; $\overline{WR}$; $\overline{PSEN}$; ALE	$-I_{OH}$ = 100 μA	V_{OH1}	0.75 V_{CC}	–	V
Output voltage HIGH all other outputs	$-I_{OH}$ = 40 μA	V_{OH2}	0.75 V_{CC}	–	V
Input leakage current $\overline{INT}$; T1; EA	without internal pull-up $V_{SS} < V_I < V_{CC}$	$\pm I_{IL}$	–	10	μA
Input leakage current P10-P17; P20-P27; $\overline{SS}$	with internal pull-up (PCB80C39/49 version) $V_{SS} + 0.45 < V_I < V_{CC}$	$-I_{IL1}$	–	500	μA
	PCF/PCA80C39/49 versions $V_{SS} + 0.45 < V_I < V_{CC}$	$-I_{IL1}$	–	600	μA

parameter	conditions	symbol	min.	max.	unit
Input leakage current $\overline{RESET}$	(PCB80C39/49 version) $V_{SS} < V_I < V_{IH1}$	$-I_{ILR}$	20	300	μA
	(PCF/PCA80C39/49 version) $V_{SS} < V_I < V_{IH1}$	$-I_{ILR}$	20	350	μA
Output leakage current BUS; TO at high impedance state	$V_{SS}+0.45 < V_I < V_{CC}$	$\pm I_{OL}$	–	10	μA

Notes to the DC characteristics

1. Levels are not fully compatible according to the TTL specification.
2. Note here that V_{DD} and V_{CC} refer to pins 26 and 40. These pins are the RAM and processor power supplies respectively.

DEVELOPMENT DATA

AC CHARACTERISTICS

$V_{CC} = V_{DD} = 5\ V \pm 10\%$; $V_{SS} = 0\ V$; T_{amb} (PCB version) = 0 to + 70 °C; note 1;
T_{amb} (PCF version) = −40 to + 85 °C; note 1; T_{amb} (PCA version) = −40 to + 110 °C; note 1).

See waveforms Figs 14, 15, 16, 17 and 18.

parameter	f (t_{CL}) (note 2)	symbol	11 MHz		unit
			min.	max.	
ALE pulse width	$7/30t_{CL}-170$	t_{LL}	150	–	ns
Address set-up time to ALE	$2/15t_{CL}-110$	t_{AL}	70	–	ns
Address hold time from ALE	$1/15t_{CL}-40$	t_{LA}	50	–	ns
Control pulse width $\overline{RD}$, $\overline{WR}$	$1/2t_{CL}-200$	t_{CC1}	480	–	ns
Control pulse width $\overline{PSEN}$	$2/5t_{CL}-200$	t_{CC2}	350	–	ns
Data set-up time before $\overline{WR}$	$13/30t_{CL}-200$	t_{DW}	390	–	ns
Data hold time after $\overline{WR}$ (note 3)	$1/15t_{CL}-50$	t_{WD}	40	–	ns
Data hold time $\overline{RD}$, $\overline{PSEN}$	$1/10t_{CL}-30$	t_{DR}	0	110	ns
$\overline{RD}$ to data input	$2/5t_{CL}-170$	t_{RD1}	–	375	ns
$\overline{PSEN}$ to data input	$3/10t_{CL}-170$	t_{RD2}	–	240	ns
Address set-up time to $\overline{WR}$	$1/3t_{CL}-150$	t_{AW}	300	–	ns
Address set-up time to data input ($\overline{RD}$)	$7/10t_{CL}-250$	t_{AD1}	–	730	ns
Address set-up time to data input ($\overline{PSEN}$)	$1/2t_{CL}-220$	t_{AD2}	–	460	ns
Address floating to $\overline{RD}$, $\overline{WR}$	$2/15t_{CL}-40$	t_{AFC1}	140	–	ns
Address floating to $\overline{PSEN}$	$1/30t_{CL}-40$	t_{AFC2}	10	–	ns
ALE to control pulse $\overline{RD}$, $\overline{WR}$	$1/5t_{CL}-75$	t_{LAFC1}	200	–	ns
ALE to control pulse $\overline{PSEN}$	$1/10t_{CL}-75$	t_{LAFC2}	60	–	ns
Control pulse to ALE $\overline{RD}$, $\overline{WR}$, $\overline{PROG}$	$1/15t_{CL}-40$	t_{CA1}	50	–	ns
Control pulse to ALE $\overline{PSEN}$	$4/15t_{CL}-40$	t_{CA2}	320	–	ns
Port control set-up to $\overline{PROG}$	$1/10t_{CL}-80$	t_{CP}	50	–	ns

AC CHARACTERISTICS (continued)

parameter	f (t_{CL})	symbol	11 MHz		unit
			min.	max.	
Port control hold to $\overline{PROG}$	$4/15t_{CL}-260$	t_{PC}	100	–	ns
$\overline{PROG}$ to time port 2 input must be valid	$17/30t_{CL}-120$	t_{PR}	–	650	ns
Input data hold time from $\overline{PROG}$	$1/10t_{CL}$	t_{PF}	0	140	ns
Output data set-up time	$2/5t_{CL}-290$	t_{DP}	250	–	ns
Output data hold time	$1/10t_{CL}-90$	t_{PD}	40	–	ns
$\overline{PROG}$ pulse width	$7/10t_{CL}-250$	t_{PP}	700	–	ns
Port 2 I/O data set-up time to ALE	$4/15t_{CL}-200$	t_{PL}	160	–	ns
Port 2 I/O data hold time to ALE	$1/10t_{CL}-120$	t_{LP}	15	–	ns
Port output from ALE	$3/10t_{CL}+100$	t_{PV}	–	510	ns
Cycle time	$(1/f_{XTAL}) \times 15$	t_{CL}	1,36	15	μs
T0 repetition rate	$3/15t_{CL}$	t_{0PRR}	270	–	ns
Clock period (note 2)	$1/(f_{XTAL})$	t_{CY}	90,9	1000	ns

Notes to AC characteristics

1. Control outputs: C_L = 80 pF
   Bus outputs: C_L = 150 pF.
2. f(t_{CL}) assumes 50% duty cycle on XTAL1 and XTAL2; minimum frequency = 1 MHz; maximum frequency = 15 MHz for all versions.
3. Bus high-impedance load: 20 pF.

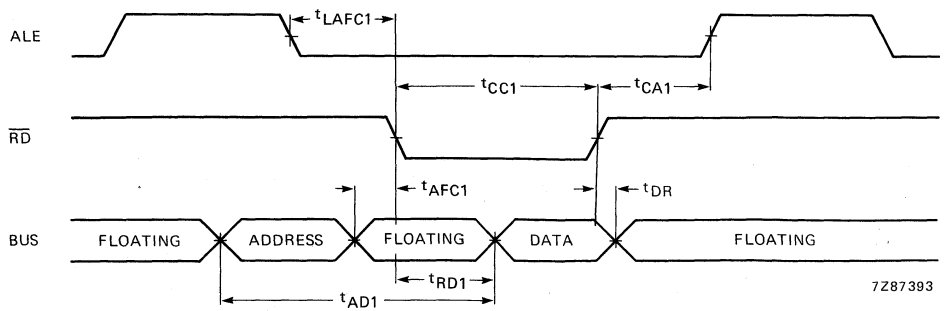


Fig.14 Read from external data memory.

DEVELOPMENT DATA

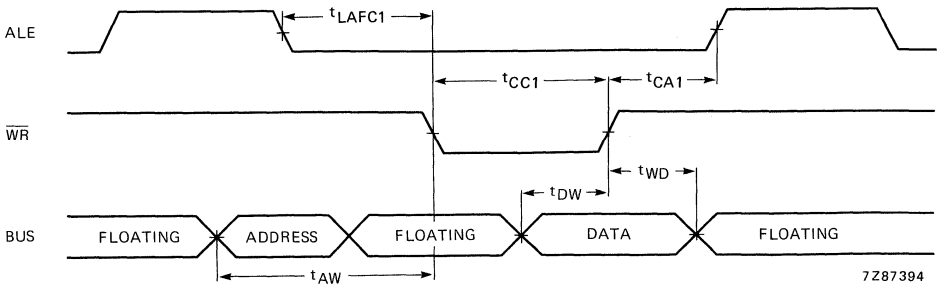


Fig.15 Write to external data memory.

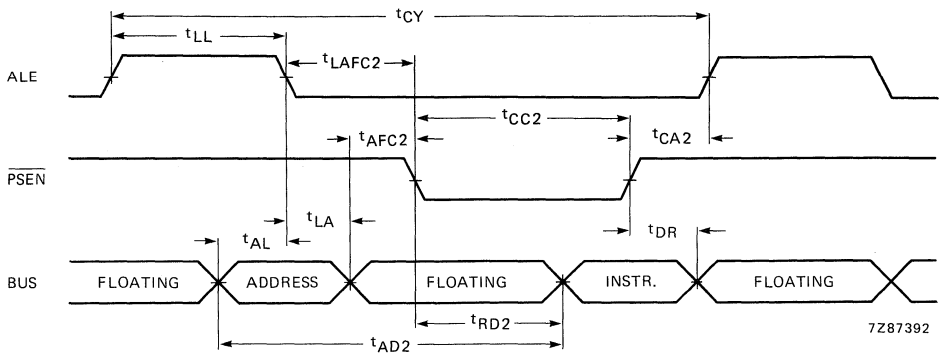


Fig.16 Instruction fetch from external program memory.

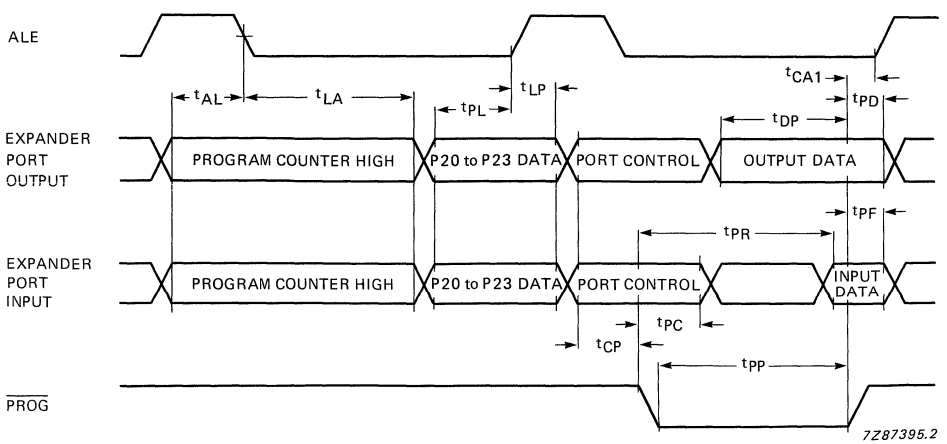


Fig.17 Port 2 timing.

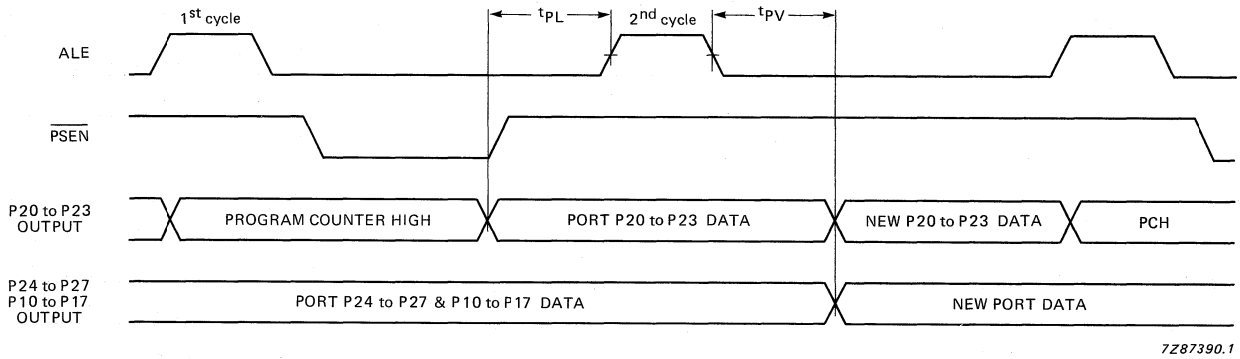


Fig.18 I/O port timing.

Section 3 - 80XX Family and Derivatives

MAB84XX

Single-chip 8-bit microcontroller family specification

CONTENTS

Section

Single-chip 8-bit microcontroller family specification

MAB84XX

1 INTRODUCTION

This family data sheet describes the microcontroller core which is common for all members of the MAB84XX family. For complete information of a particular microcontroller, consult both the specific microcontroller data sheet and this family data sheet.

2 FEATURES

- 8-bit CPU
- Up to 8 K bytes ROM
- Up to 128 bytes RAM
- Over 80 instructions, all instructions 1 or 2 cycles
- Up to 20 quasi bi-directional I/O port lines
- 8-bit programmable timer/event counter
- 3 single-level vectored interrupts (external, timer/ counter and serial I/O)
- 2 Test inputs: T0 (may also be used as an interrupt) and T1 (may also be used as an input to an 8-bit counter)
- Serial I/O interface
- On-chip oscillator

The family is well supported with:

- Cross assemblers
- In-circuit emulation tools
- Window debugger
- Piggy-back versions for prototyping.

3 GENERAL DESCRIPTION

The MAB84XX single-chip 8-bit microcontroller family is fabricated in NMOS. Members contain on-chip mask programmable program ROM (up to 6 K bytes, except the ROM-less version which can address up to 8 K bytes of external ROM), up to 128 bytes RAM, an interrupt input, a test input (directly testable), a serial I/O and general purpose I/O lines. The instruction set is based on that of the MAB8048. A number of the MAB84XX family members have similar CMOS counterparts in the PCF84CXXX microcontroller family.

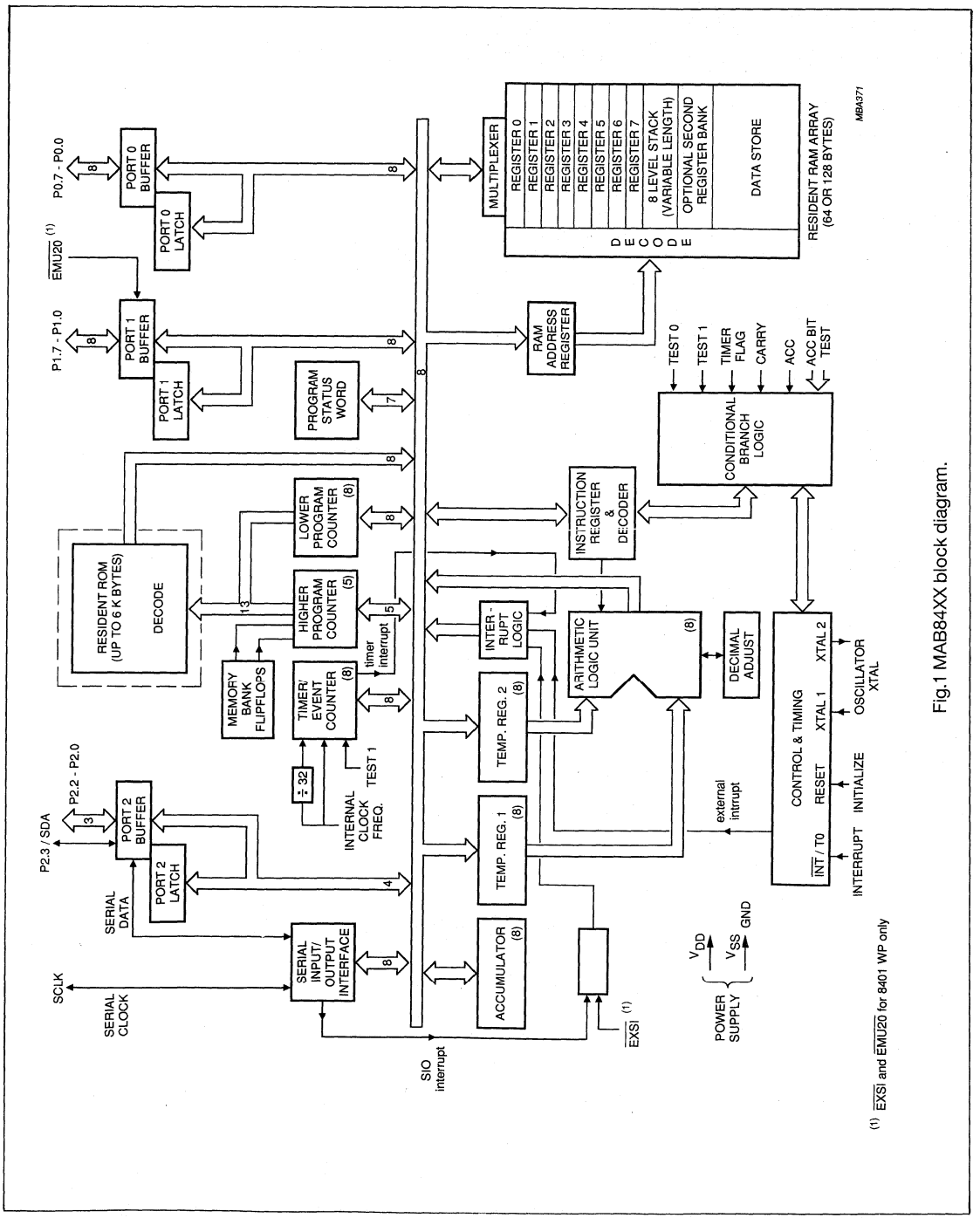


Fig.1 MAB84XX block diagram.

4 CENTRAL PROCESSING UNIT (CPU)

The CPU performs arithmetic, logical and program control functions. The CPU consists of:

- **ALU** (arithmetic and logic unit) which performs the following operations:
  arithmetic operations: ADD, INCREMENT, DECREMENT, ROTATE. The DA A, SWAP A, and XCHD instructions and a half carry flag simplify BCD arithmetic and the handling of nibbles.
  logical operations: OR, AND, EXCLUSIVE-OR, and COMPLEMENT.
- **ACCUMULATOR**; in most operations this is the source or destination register.
- **Program status word**; indicates the status of the microcontroller.
- **Program counter**; supplies a 13-bit address to the program memory.
- **Instruction decoder**; decodes the instructions and supplies control signals to several parts of the microcontroller.
- **Control and timing**; contains the control, oscillator and timing circuitry. A machine cycle consists of 10 states, each state contains 3 XTAL clock periods. Thus, one machine cycle consists of 30 XTAL clock periods. All instructions take 1 or 2 machine cycles.

5 MEMORY AND REGISTERS

5.1 Program memory

Members of the MAB84XX family contain 1 K, 2 K, 4 K, or 6 K bytes of on-chip read-only memory (ROM); there are also ROM-less versions which can address up to 8 K bytes of external ROM. Each location is directly addressable by the program counter. The program memory is mask-programmed at the factory. Figure 2a shows the program memory map.

Four program memory locations are of special importance:

- Location 0: first instruction to be executed after the controller is reset.
- Location 3: first instruction of an external interrupt ($\overline{INT}$/T0) service routine.
- Location 5: first instruction of a serial I/O interrupt service routine.
- Location 7: first instruction of a timer/event counter interrupt service routine.

Program memory is arranged in banks of 2 K bytes. Memory banks are preselected by the SEL MB instructions. Memory bank boundaries can only be crossed by using the unconditional jump (JMP) or subroutine call (CALL) instructions, after the appropriate memory bank has been selected. The program memory is further divided into 'pages', each of 256 bytes. Page boundaries can not be crossed by conditional jumps and indirect jump (JMPP) instructions.

The MOVP A,@A instruction permits efficient table look-up from the current ROM page.

5.2 Data memory

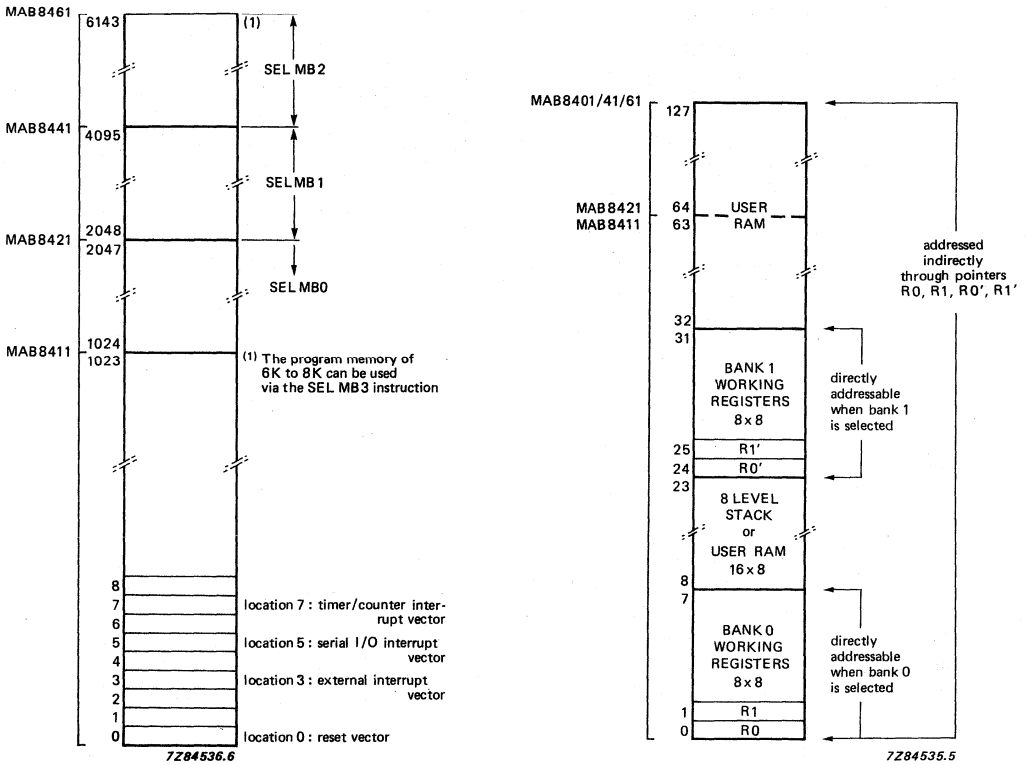
Members of the MAB84XX family contain up to 128 bytes of random access data memory (RAM). All locations are indirectly addressable using RAM pointer registers; up to 16 designated locations are directly addressable (register banks 0 and 1). Another 16 bytes are designated to an 8-level program counter stack addressed by a 3-bit stack pointer. Figure 2b shows the data memory map.

5.2.1 Registers R0 to R7

Registers R0 to R7 are directly addressable by the direct register instructions. Ease of addressing, and a minimum requirement of instruction bytes to manipulate their contents, makes these locations suitable for storing frequently addressed intermediate results.

Executing the SEL RB0 (SELect Register Bank 0) instruction designates R0-R7 to data memory locations 0-7. Executing the SEL RB1 instruction designates R0-R7 to data memory locations 24-31. This second register bank may be used as an extension of the first, or it may be reserved for use during interrupt service routines leaving the first bank available for the main program.

The first 2 locations of each bank contain the RAM pointer registers R0, R1 and R0', R1' which indirectly address all data memory locations. Every data memory location can operate as a loop counter when used with the decrement and test instruction DJNZ @Rr,addr.



(a) Program memory map (b) Data memory map

Fig.2 MAB84XX memory and register map.

5.2.2 Stack

Locations 8 to 23 of the data memory are designated to an 8-level program counter stack (2 locations per level). See Fig.3. A 3-bit stack pointer (SP) points to the next free location on the stack. After a RESET, the SP contents are 000 and the SP points to data memory locations 8 and 9.

During a subroutine CALL or interrupt, the contents of the 13 bit program counter (PC) and bits 4, 6 and 7 of the program status word (PSW) are first transferred to the stack. The SP is then incremented. During a RET or RETR instruction, the 13 bit program counter is restored and the stack pointer is decremented. Only the RETR instruction transfers the saved PSW bits to the PSW.

Nesting of subroutines and interrupt service routines can continue to a depth of 8 levels without overflowing the stack. When an overflow occurs the deepest address stored (locations 8 and 9) will be overwritten and lost since the stack pointer overflows from 111 to 000. The stack pointer also underflows from 000 to 111.

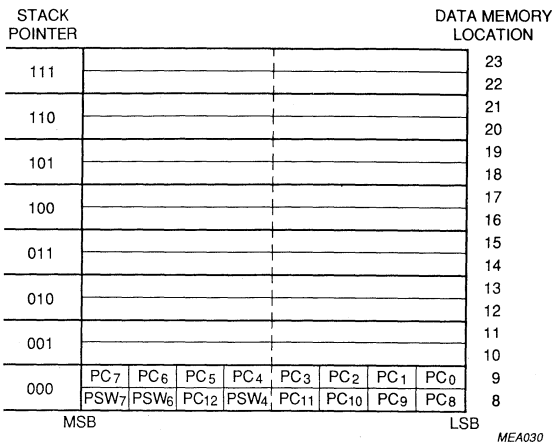


Fig.3 Program counter stack.

5.2.3 User RAM

Data memory locations 32 to 127 are designated as user RAM and are indirectly addressable with the RAM pointers R0, R1 or R0', R1'. Unused register and stack locations are also available as user RAM and are addressed in the same way.

6 PROGRAM COUNTER

The 13-bit program counter can address up to 8 K bytes of ROM (see Fig.4). The least significant 11 bits (PC0 to PC10) are auto-incrementing. The two most significant bits (PC11 and PC12) are loaded with the contents of two internal flip flops (MBFF0 and MBFF1 respectively) when a JMP or CALL instruction is executed. The contents of these two internal flip-flops are altered using the SEL MB instructions. During an interrupt service routine PC11 and PC12 are forced to logic 0.

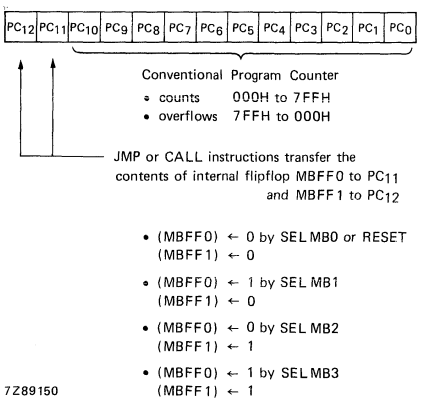


Fig.4 Program counter.

7 PROGRAM STATUS WORD

The program status word (PSW) is an 8-bit CPU register which stores information about the current status of the microcontroller (see Fig.5). All bits can be read using the MOV A,PSW instruction. Only the PS bit can be written using the MOV PSW,A instruction. Table 1 shows the function of the PSW bits and when they are affected.

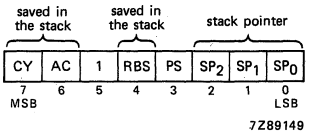


Fig.5 Program status word.

Table 1 Program Status Word.

BIT	NAME	FUNCTION	AFFECTED BY
7	CY	Carry, signals accumulator overflow	ADD, ADDC, DA, RLC, RRC, CLR C and CPL C instructions
6	AC	Auxiliary Carry, half carry	ADD and ADDC instructions
5	-	Not used, always 1 when read	
4	RBS	Register Bank Select 0: select register bank 0 1: select register bank 1	SEL RB instructions
3	PS	Timer Prescaler Select 0: prescaler selected (÷32) 1: prescaler not selected (÷1)	MOV PSW,A instruction
2 1 0	SP2 SP1 SP0	Stack Pointer bits 2-0	CALL, RET, RETR instructions and interrupts

8 CONDITIONAL JUMP LOGIC

The conditional jump logic enables several conditions to be tested by the user's program. Table 2 lists the conditional jump instructions which may be used to change the program sequence.

The DJNZ instruction decrements a designated register or data memory location and jumps if the contents are not zero. This instruction is useful for looping control.

Table 2 Conditional jumps.

TEST	JUMP CONDITION	JUMP INSTRUCTION
accumulator	all bits zero any bit non-zero	JZ JNZ
accumulator bit test	1	JB0 to JB7
carry flag	1 0	JC JNC
timer overflow flag	1 0	JTF JNTF
test input T0	1 0	JT0 JNT0
test input T1	1 0	JT1 JNT1
register	non-zero	DJNZ Rr
data memory location	non-zero	DJNZ @Rr

9 INTERRUPT LOGIC

The MAB84XX family handles external, serial I/O, and timer/event counter interrupts. The interrupt mechanism is single level; an executing interrupt service routine can not be interrupted by other interrupts. If several interrupt requests are detected simultaneously, they are serviced in order of priority (see Table 3). An interrupt request will only be serviced if the interrupt is enabled; each interrupt type has individual enable and disable instructions (see Table 3).

Before an interrupt is serviced, execution of the current instruction is first completed. The contents of the program counter, and bits 4, 6 and 7 of the program status word are then saved on the program counter stack, and a CALL to the corresponding interrupt vector is executed (see Table 3). The maximum interrupt latency time is 4 machine cycles. The interrupt service routine must be terminated by the RETR (return and restore) instruction. At least one instruction in the main program will then be executed before another interrupt service routine is serviced.

Table 3 Interrupts.

INTERRUPT			INSTRUCTION	
PRIORITY	**TYPE**	**VECTOR LOCATION**	**ENABLE**	**DISABLE**
1 (highest)	external	003 (ROM)	EN I	DIS I
2	serial I/O	005 (ROM)	EN SI	DIS SI
3 (lowest)	timer/event counter	007 (ROM)	EN TCNTI	DIS TCNTI

Notes on interrupt service routines

- While an interrupt service routine is in progress, the two most significant bits of the program counter are frozen at zero.
  Therefore: interrupt service routines and any subroutines called within interrupt service routines must reside (entirely) in memory bank 0.
  Within an interrupt service routine and within any subroutines called within an interrupt service routine, other memory banks cannot be selected and SEL MB instructions must not be used.
- Subroutines and nested subroutines called within an interrupt service routine must all end with RET. RETR would clear the interrupt in progress flag and further pending interrupts would then interfere with the interrupt service routine in progress.

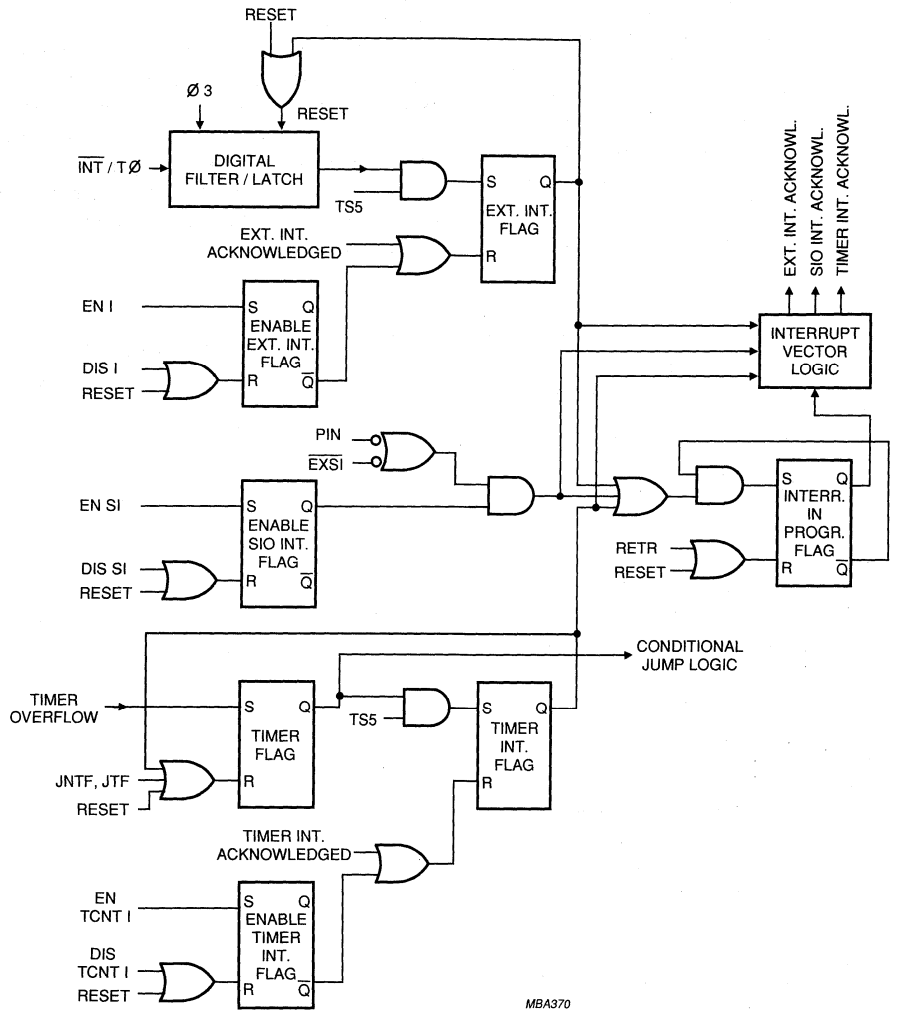


Fig.6 Interrupt logic.

Notes:

1. When the Interrupt In Progress (IIP) flag is set, other interrupts are latched but ignored until the RETR instruction is executed.
2. When the timer/counter interrupt flag (TF) has been enabled, the JTF and JNTF instructions will always find the flag at logic 0; this is because the timer flag is set at time slot 3 and reset at time slot 5 of the same cycle.

9.1 External interrupt

A HIGH-to-LOW transition on the $\overline{INT}$/T0 pin generates an external interrupt request which is always latched in a digital filter/latch (see Fig.6). To ensure that the transition is latched, the LOW state must exceed 7 XTAL clock periods after a HIGH state of more than 4 XTAL clock periods. When the External Interrupt Flag (EIF) is set, it resets the digital filter/latch (see Fig.6). If the external interrupt is enabled and no interrupt service routine is in progress, the external interrupt will be serviced. Simultaneously, the hardware removes the latched interrupt request.

During the forced CALL to the external interrupt vector location, EIF is reset and the digital filter/latch is re-enabled; the Interrupt In Progress (IIP) flag bit is also set so that other interrupts are latched but ignored until the RETR instruction is executed. After the RETR instruction has been executed, latched interrupts will be serviced in order of priority. Execution of a DIS I (disable external interrupt) instruction cancels a pending external interrupt request which has been latched in the EIF. An interrupt request which is latched in the digital filter/latch after the external interrupt has been disabled can't be cancelled.

9.2 Serial I/O interrupt

The serial I/O interrupt may be requested by the serial I/O interface. An interrupt request from the SIO interface will force the PIN interrupt flag to logic 0. If the serial I/O interrupt is enabled and no interrupt service routine is in progress, the serial I/O interrupt will be serviced. The interrupt flag is NOT automatically reset to the inactive state (by hardware) when a serial I/O interrupt is serviced; this must be done by user software.

9.3 Timer/event counter interrupt

A timer/event counter overflow generates a timer/event counter interrupt request. This interrupt request is always latched. If the timer/event counter interrupt has been enabled and if no interrupt service routine is in progress, the timer/event counter interrupt will be serviced. Simultaneously, the hardware removes the latched interrupt request. Execution of a DIS TCNTI (disable timer/event counter interrupt) instruction cancels a pending timer/event counter interrupt request which has been latched in the Timer Interrupt Flag.

9.4 Interrupt operation examples

Figures 7 to 10 show examples of how the interrupt mechanism operates.

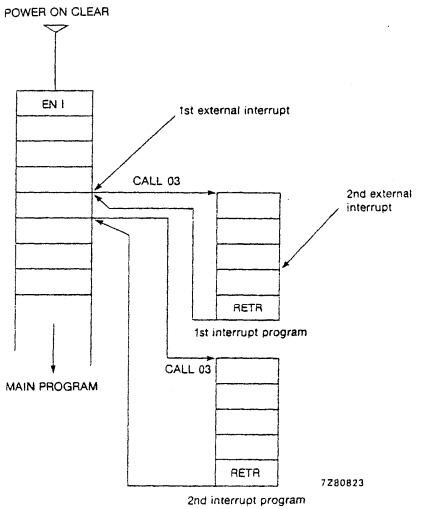


Fig.7 An external interrupt which occurs while another interrupt service routine in being executed remains latched until the executing routine is complete.

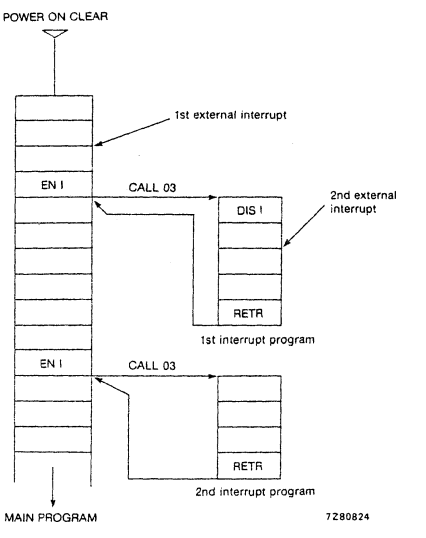


Fig.8 External interrupts are postponed until enabled.

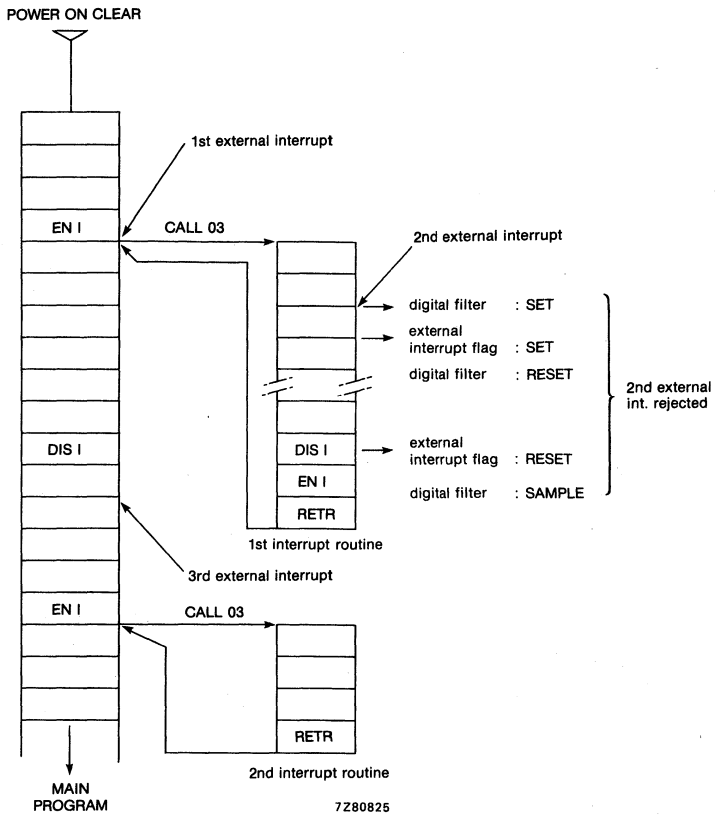


Fig.9 A previous external interrupt is cleared.

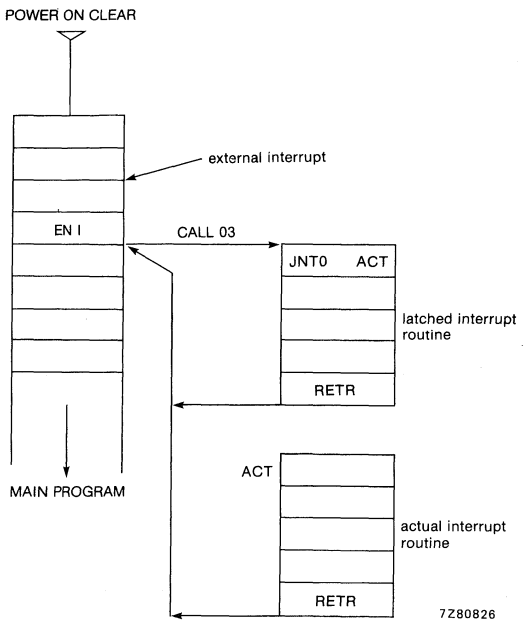


Fig.10 Detection of previous and actual external interrupt.

10 TIMER/EVENT COUNTER

The Timer/event counter (see Fig.11) can operate either as a timer or as an event counter.

- **Timer mode**; depending on the state of the PS-bit in the program status word, the internal 8-bit up-counter is incremented every machine cycle (PS = 1) or every 32 machine cycles (PS = 0).
- **Counter mode**; the internal 8-bit up-counter is incremented on every LOW-to-HIGH transition on pin T1. The HIGH state of T1 must exceed 4 XTAL clock periods after a LOW state of more than 4 XTAL clock periods. The maximum count rate is one increment per machine cycle.

When the 8-bit up-counter overflows, the timer overflow flag is set. If the timer/event counter interrupt is enabled, the timer interrupt flag (see Fig.6) is also set and the timer overflow flag is reset again in the same machine cycle. Table 4 details the instructions that control the timer/event counter. Testing the timer overflow flag, using the timer JTF (jump if timer flag = 1) or JNTF instructions, also clears the timer overflow flag. Starting the timer, using the STRT T instruction, also clears the pre-scaler. Reading the 8-bit up-counter does not disturb the counting process.

Table 4 Timer/event counter control.

FUNCTION	TIMER MODE	COUNTER MODE
clear	MOV T,A (A)=0 or RESET	MOV T,A (A)=0 or RESET
preset	MOV T,A	MOV T,A
start	STRT T	STRT CNT
stop	STOP TCNT or RESET	STOP TCNT or RESET
test	JTF/JNTF	JTF/JNTF
read	MOV A,T	MOV A,T

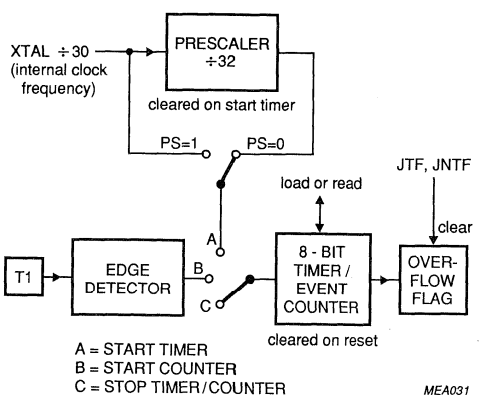


Fig.11 Timer/event counter.

11 I/O

Members of the MAB84XX family have up to 23 I/O lines arranged as follows:

- two 8-bit parallel ports
- one 4-bit parallel port
- a serial I/O which consists of the following two lines:
  - P2.3/SDA; data line shared with port line P2.3
  - SCLK, dedicated clock line
- $\overline{INT}$/T0, an external interrupt/ test input
- T1, a test input which may also be used as an input to the counter/timer

11.1 Parallel I/O ports

Up to three parallel ports are controlled by dedicated port instructions.

These ports can have up to 20 I/O port lines arranged as:

- Port 0: 8-bit parallel port (P0.0 to P0.7)
- Port 1: 8-bit parallel port (P1.0 to P1.7)
- Port 2: 4-bit parallel port (P2.0 to P2.3).

Output data written to a port is latched and remains unchanged until rewritten. Input data is not latched and must therefore remain present until read by an input instruction. Fig.12 shows the timing diagram for all ports using IN, OUTL, ANL and ORL instructions. For the OUTL instruction, data changes on time slot 7 of cycle 1. For the ANL and ORL instructions, port data changes on time slot 7 of cycle 2.

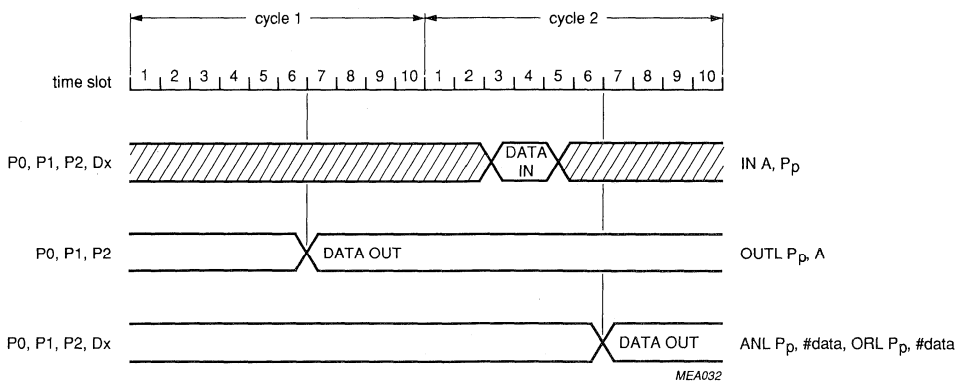


Fig.12 Port timing.

Input port lines are fully TTL compatible; output port lines can drive one TTL load. The high drive capability of Port 1 was intended to enable the 8 Port 1 output lines to drive 8 LEDs.

Three I/O mask options make it possible for parallel I/O port lines to be individually configured as follows:

- **Option 1** STANDARD I/O; quasi-bidirectional I/O (see Fig.13) consisting of a push-pull output with a weak pull-up transistor (TR3). If the output latch contains a logic 1, then the port line is pulled up to V_{CC} via TR3. TR3 provides sufficient current for a HIGH level to one TTL input, yet can be pulled LOW by an external TTL or CMOS device, thus allowing the same port line to be used for both input and output. When used as an input, the port latch must be in a logic one state. When a logic 0 is written to the port latch, a low impedance transistor (TR1) in turned on to provide TTL current sinking capability. To provide a fast logic 0 to logic 1 transition, TR2 is active during 1 XTAL clock cycle when a logic 1 is written to the port latch.
- **Option 2** OPEN DRAIN I/O WITH PULL-UP; open drain output with pull-up transistor, TR3 (see Fig.14).
- **Option 3** OPEN DRAIN I/O WITHOUT PULL-UP; open drain output without pull-up transistor (see Fig.15). Application as an output requires the connection of an external pull-up resistor.

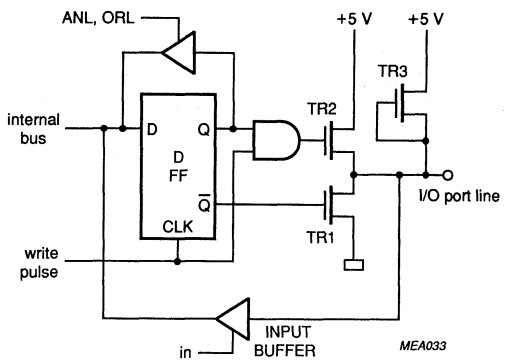


Fig.13 Standard push-pull I/O with pull-up transistor (option 1).

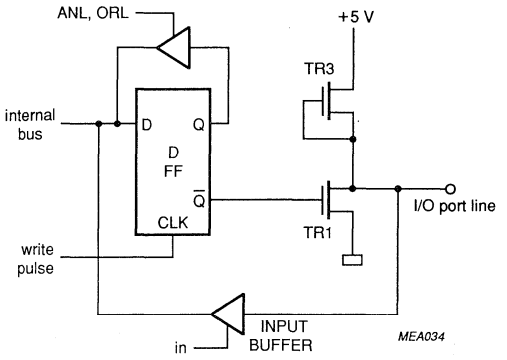


Fig.14 Open drain I/O with pull-up transistor (option 2).

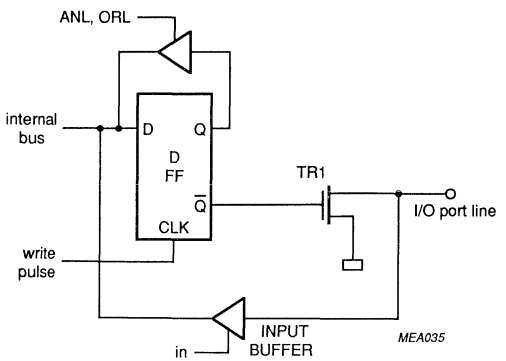


Fig.15 Open drain I/O without pull-up transistor (option 3).

11.2 Test inputs T1 and $\overline{INT}$/T0

The T1 input line can be used as:

- a test input for jump instructions JT1 and JNT1
- an input for zero voltage cross-over detection
- an external input to the event counter.

A pull-up resistor can be provided as a ROM mask option; this is useful when the input is from a switch or a standard TTL output.

When used as a test input, the JT1 and JNT1 instructions test the T1 pin for logic 1 and logic 0 respectively.

The T1 input has a self biasing circuit which can detect when an AC signal crosses zero (see Fig.16).

A LOW-to-HIGH transition on the T1 input increments the timer/event counter when the timer/event counter is in the counter mode.

When used in conjunction with the timer/event counter interrupt, zero cross-over detection is useful in thyristor control of power equipment.

The $\overline{INT}$/T0 input line can be used as:

- a test input for jump instructions JT0 and JNT0
- an external interrupt input.

11.3 Serial I/O

The MAB84XX on-chip serial I/O (SIO) hardware enables MAB84XX family members to be interconnected via the two-line serial I^2C bus. The SIO allows two or more devices to communicate without interrupting other devices on the bus. This is achieved by allocating a specific 7-bit address to each device on the bus. Each device only reacts to messages prefixed with the specific address of the device or the general call address. Address recognition is handled by the SIO hardware, and the microcontroller is interrupted only when a valid address has been detected. The SIO hardware thus saves considerable CPU resources and memory requirements (compared to a software serial interface).

When address recognition is not required (e.g. a configuration with only two microcontrollers connected to the serial bus), direct data transfer without addressing can be performed. In multi-master systems, an automatic arbitration procedure stops two or more devices from transmitting simultaneously.

For more information on the SIO, see the chapter 'SERIAL I/O'.

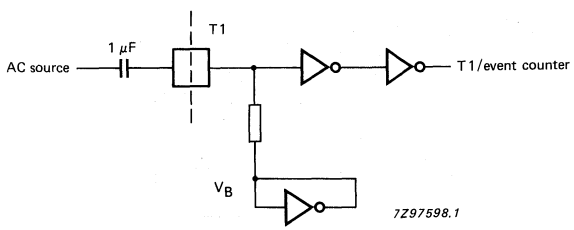


Fig.16 Self biasing circuit.

12 OSCILLATOR

The oscillator circuit must be completed by connecting one of the following timing elements between the XTAL1 and XTAL2 pins:

- a quartz crystal
- a ceramic resonator
- an inductor.

Two external load capacitors are also required.

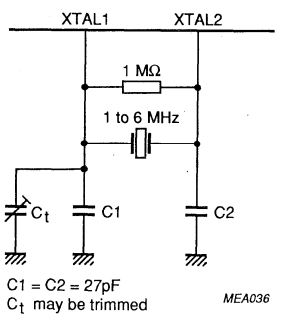
If a quartz crystal is used as the timing element, the crystal must operate in the fundamental mode of vibration. The clock may also be supplied by an external source. In this case the rise and fall times of the external clock (t_r, t_f) must be less that 10 ns. A pull-up resistor is required if the clock source is a TTL device. Table 5 gives the recommended L and C values for various oscillator frequencies when the timing element is an inductor.

Table 5 LC oscillator timing.

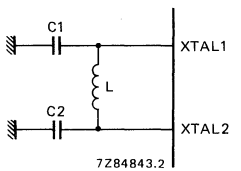
FREQUENCY (MHz)	C1 = C2 (pF)	L (μH)
3.0	33	100
4.0	33	56
4.4	33	47
5.0	33	33
6.0	33	22

Oscillator start-up time depends on the external timing element. The start-up time of a quartz crystal is several milliseconds. The start-up time of a ceramic resonator is tenths of a millisecond.

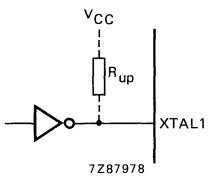
The XTAL clock may also be supplied by an external clock signal at pin XTAL1. In this case, XTAL2 is not connected.



(a) Crystal or ceramic resonator



(b) Inductor



(c) External

Fig.17 Oscillator circuit with various timing elements.

13 RESET

A reset signal initializes the microcontroller to a defined state.

13.1 Reset sequence

To ensure a correct reset, the reset signal at the RESET pin must be HIGH for at least 2 machine cycles after the power supply and clock have stabilized. The HIGH level at the RESET pin:

- sets the program counter to zero
- selects memory bank 0 and register bank 0
- sets the stack pointer to zero (000), pointing to RAM addresses 8 and 9
- resets interrupt flags (external, timer/event counter and serial I/O) to the inactive state
- disables interrupts (external, timer/event counter and serial I/O)
- stops the timer/event counter, then sets it to zero
- sets the timer prescaler to divide by 32
- resets the timer overflow flag
- sets all ports latches to logic one (input mode), except P2.3/SDA which is high impedance
- sets the serial I/O to the slave receiver mode and disables the serial I/O.

A reset does not affect the data memory contents.

The external power-on-reset circuit should consist of a 1 μF capacitor connected between V_{CC} and the RESET pin (see Fig.18); a diode may also be connected between the RESET pin and ground to ensure a correct reset if the supply voltage falls momentarily. Alternatively the device can be reset by an external TTL compatible signal (see Fig.19). Figure 20 shows typical input characteristics.

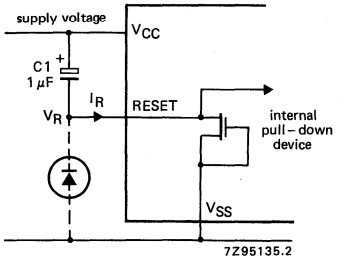


Fig.18 Power-on-reset.

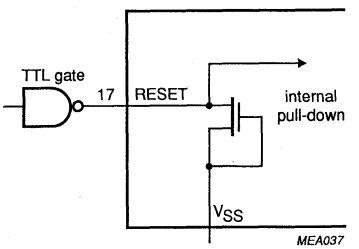


Fig.19 External reset.

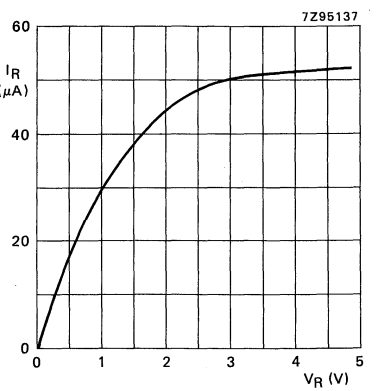


Fig.20 Typical input characteristics.

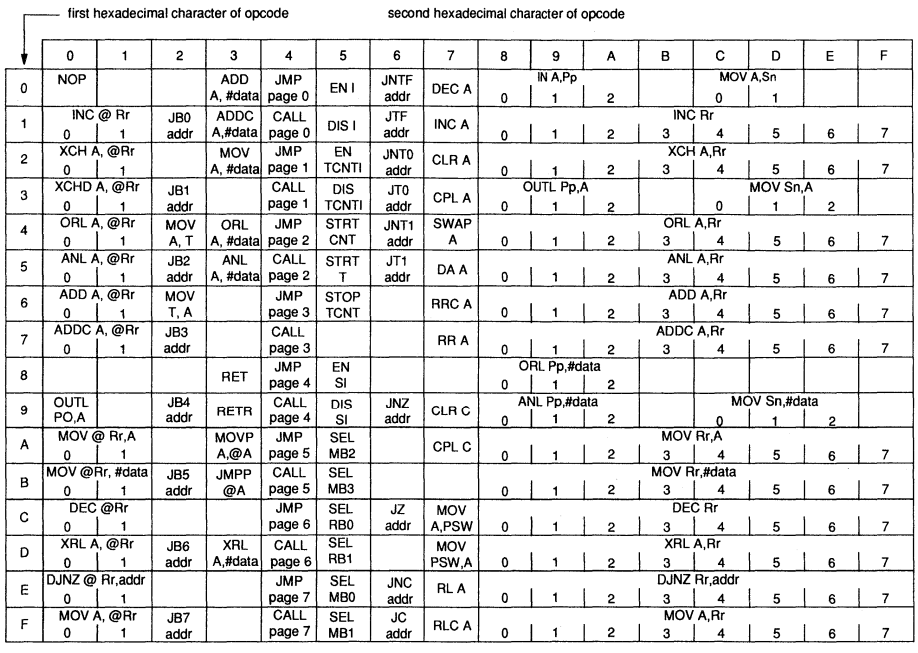
14 INSTRUCTION SET

The MAB84XX family instruction set consists of over 80 one and two byte instructions. Program code efficiency is high because all RAM locations and all ROM locations on a 256 byte page require only a single byte address.

Figure 21 shows the instruction map and Table 6 describes the symbols that are used.

Table 6 Symbols and definitions.

SYMBOL	DESCRIPTION
A	accumulator
addr	program memory address
Bb	bit designation (b = 0...7)
C	carry bit (bit CY)
CNT	event counter
data	8-bit immediate data
I	interrupt
MBn	memory bank (n = 0...3)
MBFFn	memory bank flip-flop (n = 0, 1)
P	mnemonic for 'in-page' operation
PC	program counter
Pp	port designation (p = 0, 1, 2)
PSW	program status word
RB	register bank
RBS	register bank select flag
@Rr	8-bit address register (r = 0, 1)
Rr	8-bit register (r = 0...7)
Sn	serial I/O register (n = 0, 1, 2)
SP	stack pointer
T	timer
TCNT	timer/event counter
TF	timer flag
T0, T1	test 0 and test 1 inputs
#	immediate data prefix
@	indirect address prefix
(X)	contents of X
((X))	contents of location addressed by X
←	is replaced by
↔	is exchanged with



first hexadecimal character of opcode ↓	second hexadecimal character of opcode →															
	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
0	NOP			ADD A, #data	JMP page 0	EN I	JNTF addr	DEC A	IN A,Pp 0	IN A,Pp 1	IN A,Pp 2		MOV A,Sn 0	MOV A,Sn 1		
1	INC @ Rr 0	INC @ Rr 1	JB0 addr	ADDC A,#data	CALL page 0	DIS I	JTF addr	INC A	INC Rr 0	INC Rr 1	INC Rr 2	INC Rr 3	INC Rr 4	INC Rr 5	INC Rr 6	INC Rr 7
2	XCH A, @Rr 0	XCH A, @Rr 1		MOV A, #data	JMP page 1	EN TCNTI	JNT0 addr	CLR A	XCH A,Rr 0	XCH A,Rr 1	XCH A,Rr 2	XCH A,Rr 3	XCH A,Rr 4	XCH A,Rr 5	XCH A,Rr 6	XCH A,Rr 7
3	XCHD A, @Rr 0	XCHD A, @Rr 1	JB1 addr		CALL page 1	DIS TCNTI	JT0 addr	CPL A	OUTL Pp,A 0	OUTL Pp,A 1	OUTL Pp,A 2		MOV Sn,A 0	MOV Sn,A 1	MOV Sn,A 2	
4	ORL A, @Rr 0	ORL A, @Rr 1	MOV A, T	ORL A, #data	JMP page 2	STRT CNT	JNT1 addr	SWAP A	ORL A,Rr 0	ORL A,Rr 1	ORL A,Rr 2	ORL A,Rr 3	ORL A,Rr 4	ORL A,Rr 5	ORL A,Rr 6	ORL A,Rr 7
5	ANL A, @Rr 0	ANL A, @Rr 1	JB2 addr	ANL A, #data	CALL page 2	STRT T	JT1 addr	DA A	ANL A,Rr 0	ANL A,Rr 1	ANL A,Rr 2	ANL A,Rr 3	ANL A,Rr 4	ANL A,Rr 5	ANL A,Rr 6	ANL A,Rr 7
6	ADD A, @Rr 0	ADD A, @Rr 1	MOV T, A		JMP page 3	STOP TCNT		RRC A	ADD A,Rr 0	ADD A,Rr 1	ADD A,Rr 2	ADD A,Rr 3	ADD A,Rr 4	ADD A,Rr 5	ADD A,Rr 6	ADD A,Rr 7
7	ADDC A, @Rr 0	ADDC A, @Rr 1	JB3 addr		CALL page 3			RR A	ADDC A,Rr 0	ADDC A,Rr 1	ADDC A,Rr 2	ADDC A,Rr 3	ADDC A,Rr 4	ADDC A,Rr 5	ADDC A,Rr 6	ADDC A,Rr 7
8				RET	JMP page 4	EN SI			ORL Pp,#data 0	ORL Pp,#data 1	ORL Pp,#data 2					
9	OUTL PO,A		JB4 addr	RETR	CALL page 4	DIS SI	JNZ addr	CLR C	ANL Pp,#data 0	ANL Pp,#data 1	ANL Pp,#data 2		MOV Sn,#data 0	MOV Sn,#data 1	MOV Sn,#data 2	
A	MOV @ Rr,A 0	MOV @ Rr,A 1		MOVP A,@A	JMP page 5	SEL MB2		CPL C	MOV Rr,A 0	MOV Rr,A 1	MOV Rr,A 2	MOV Rr,A 3	MOV Rr,A 4	MOV Rr,A 5	MOV Rr,A 6	MOV Rr,A 7
B	MOV @Rr, #data 0	MOV @Rr, #data 1	JB5 addr	JMPP @A	CALL page 5	SEL MB3			MOV Rr,#data 0	MOV Rr,#data 1	MOV Rr,#data 2	MOV Rr,#data 3	MOV Rr,#data 4	MOV Rr,#data 5	MOV Rr,#data 6	MOV Rr,#data 7
C	DEC @Rr 0	DEC @Rr 1			JMP page 6	SEL RB0	JZ addr	MOV A,PSW	DEC Rr 0	DEC Rr 1	DEC Rr 2	DEC Rr 3	DEC Rr 4	DEC Rr 5	DEC Rr 6	DEC Rr 7
D	XRL A, @Rr 0	XRL A, @Rr 1	JB6 addr	XRL A,#data	CALL page 6	SEL RB1		MOV PSW,A	XRL A,Rr 0	XRL A,Rr 1	XRL A,Rr 2	XRL A,Rr 3	XRL A,Rr 4	XRL A,Rr 5	XRL A,Rr 6	XRL A,Rr 7
E	DJNZ @ Rr,addr 0	DJNZ @ Rr,addr 1			JMP page 7	SEL MB0	JNC addr	RL A	DJNZ Rr,addr 0	DJNZ Rr,addr 1	DJNZ Rr,addr 2	DJNZ Rr,addr 3	DJNZ Rr,addr 4	DJNZ Rr,addr 5	DJNZ Rr,addr 6	DJNZ Rr,addr 7
F	MOV A, @Rr 0	MOV A, @Rr 1	JB7 addr		CALL page 7	SEL MB1	JC addr	RLC A	MOV A,Rr 0	MOV A,Rr 1	MOV A,Rr 2	MOV A,Rr 3	MOV A,Rr 4	MOV A,Rr 5	MOV A,Rr 6	MOV A,Rr 7

MBA369

Fig.21 Instruction map.

SINGLE-CHIP 8-BIT MICROCONTROLLER

DESCRIPTION

The MAB84X1 family of microcontrollers is fabricated in NMOS. The family consists of 5 devices:

- MAB8401 — 128 bytes RAM, external program memory, with 8-bit LED-driver (10mA), emulation of MAB/F8422/42* possible
- MAB/MAF8411 — 1K byte ROM/64 bytes RAM plus 8-bit LED-driver
- MAB/MAF8421 — 2K bytes ROM/64 bytes RAM plus 8-bit LED-driver
- MAB/MAF8441 — 4K bytes ROM/128 bytes RAM plus 8-bit LED-driver
- MAB/MAF8461 — 6K bytes ROM/128 bytes RAM plus 8-bit LED-driver

Each version has 20 quasi-bidirectional I/O port lines, one serial I/O line, one single-level vectored interrupt, an 8-bit timer/event counter and on-board clock oscillator and clock circuits. Two 20-pin versions, MAB/F8422 and MAB/F8442* are also available.

This microcontroller family is designed to be an efficient controller as well as an arithmetic processor. The instruction set is based on that of the MAB8048. The microcontrollers have extensive bit handling abilities and facilities for both binary and BCD arithmetic.

For detailed information see the 84XX family specification.

* See data sheet on MAB/F8422/42.

Features

- 8-bit: CPU, ROM, RAM and I/O in a single 28-lead DIL package
- 1K, 2K, 4K or 6K ROM bytes plus a ROM-less version
- 64 or 128 RAM bytes
- 20 quasi-bidirectional I/O port lines
- Two testable inputs: one of which can be used to detect zero cross-over, the other is also the external interrupt input
- Single level vectored interrupts: external, timer/event counter, serial I/O
- Serial I/O that can be used in single or multi-master systems (serial I/O data via an existing port line and clock via a dedicated line)
- 8-bit programmable timer/event counter
- Internal oscillator, generated with inductor, crystal, ceramic resonator or external source
- Over 80 instructions (based on MAB8048) all of 1 or 2 cycles
- Single 5 V power supply (± 10%)
- Operating temperature ranges:

0 to + 70 °C	MAB84X1 family
−40 to + 85 °C	MAF84X1 family only
−40 to + 110 °C	MAF84AX1 family only

PACKAGE OUTLINES

MAB8401B: 28-lead 'Piggy-back' package (with up to 28-pin EPROM on top).
MAB8401WP: 68-lead plastic leaded chip-carrier (PLCC) (SOT188).
MAB/MAF8411/21/41/61P: 28-lead DIL; plastic with internal heat spreader (SOT117).
MAF84A11/21/41/61P: 28-lead DIL; plastic with internal heat spreader (SOT117).
MAB8411/21/41/61T: 28-lead mini-pack; plastic (SO28; SOT136A).

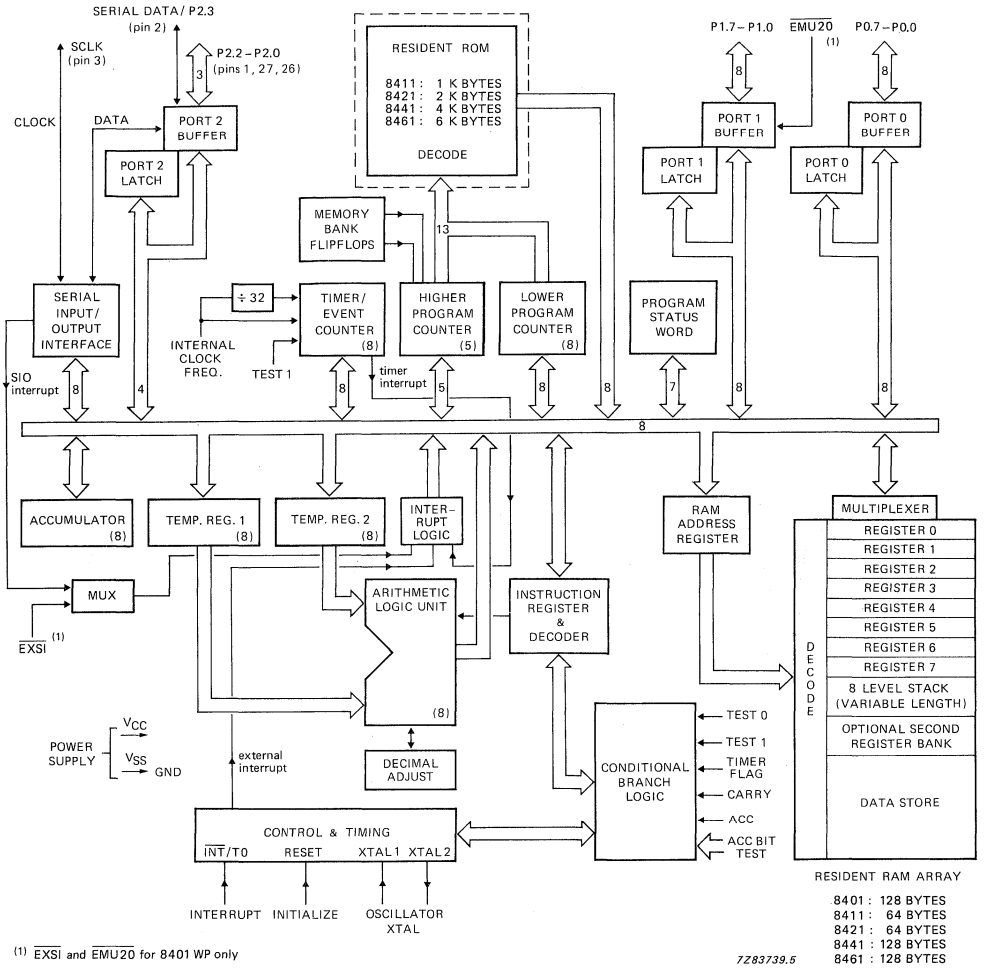


Fig. 1a Block diagram of the MAB84X1 family.

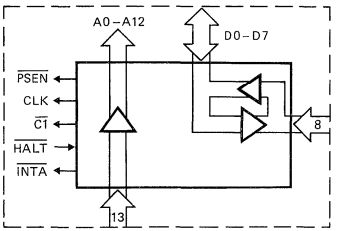


Fig. 1b Replacement for dotted part in Fig. 1a for the MAB8401WP bond-out version.

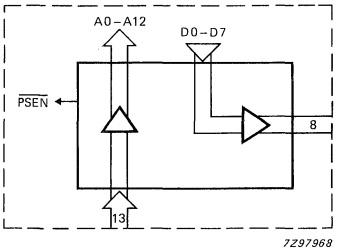


Fig. 1c Replacement of dotted part in Fig. 1a for the MAB8401B 'Piggy-back' version.

PINNING

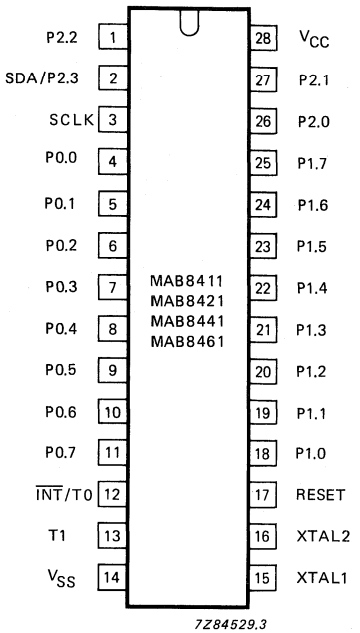
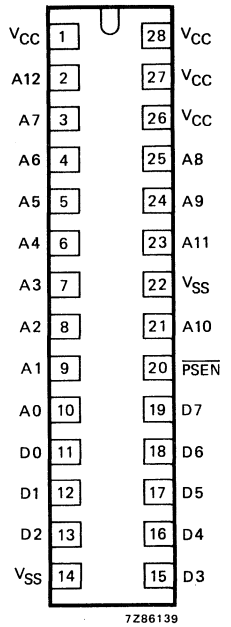


Fig. 2 Pinning diagram for mask-programmable devices MAB8411, MAB8421, MAB8441, MAB8461 and for MAB8401 'Piggy-back' version bottom pinning (for top pinning see Fig. 3).

PINNING DESIGNATION

V_{SS}	14	**Ground**
V_{CC}	28	**Power supply,** + 5 V
P0.0 – P0.7	4 – 11	**Port 0,** 8-bit quasi-bidirectional I/O port
P1.0 – P1.7	18 – 25	**Port 1,** 8-bit quasi-bidirectional I/O port with 8-bit LED driver
P2.0 – P2.3	26, 27, 1, 2	**Port 2,** 4-bit quasi-bidirectional I/O port; SDA/P2.3 is the serial data I/O in serial I/O mode
SCLK	3	Bidirectional clock for serial I/O
$\overline{INT}$/T0	12	External interrupt input (sensitive to a negative-going edge min LOW > 7 clock pulses, min HIGH > 4 clock pulses), testable using the JT0 or JNT0 instructions.
T1	13	Input pin, testable using the JT1 or JNT1 instructions. It can be designated as event counter input using the STRT CNT instruction. It can also be used to detect zero cross-over of slowly moving AC inputs.
RESET	17	Input to initialize the processor (active HIGH).
XTAL1	15	Connection to timing component (crystal) that determines the frequency of the internal oscillator. It is also the input for an external clock source.
XTAL2	16	Connection to other side of the timing component.

MAB8401B (top pinning)



PIN DESIGNATION

designation	pin	function
V_{SS}	14, 22	**Ground**
V_{CC}	1, 26–28	**Power supply,** + 5 V
A0–A12	10–3, 25, 24, 21, 23, 2	Address outputs
D0–D7	11–13, 15–19	Data inputs
$\overline{PSEN}$	20	Program store enable

Fig. 3 Pinning diagram for MAB8401B 'Piggy-back' version top pinning (for bottom pinning see Fig. 2); to access a 2732 or 2764 EPROM.

Note

Access times for ROMS/EPROMS to be below 1 μs.

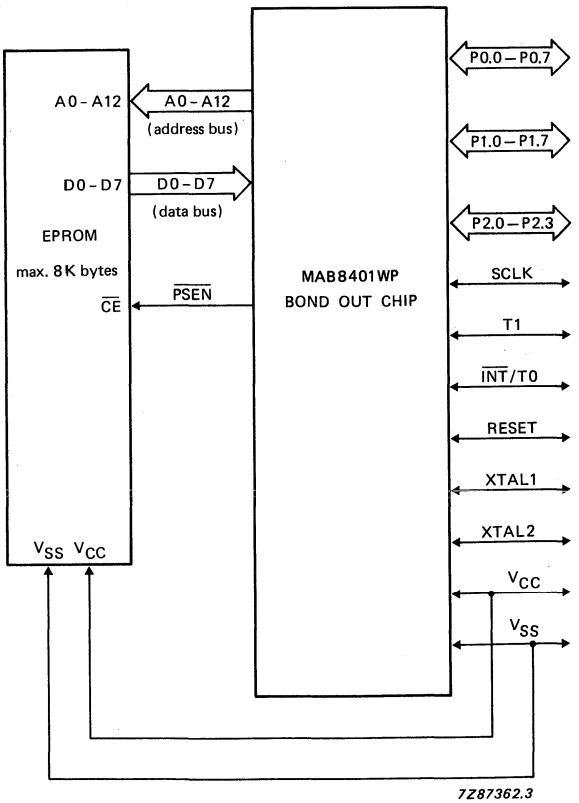


Fig. 3a Connection of EPROM to 'Piggy-back' package MAB8401B.

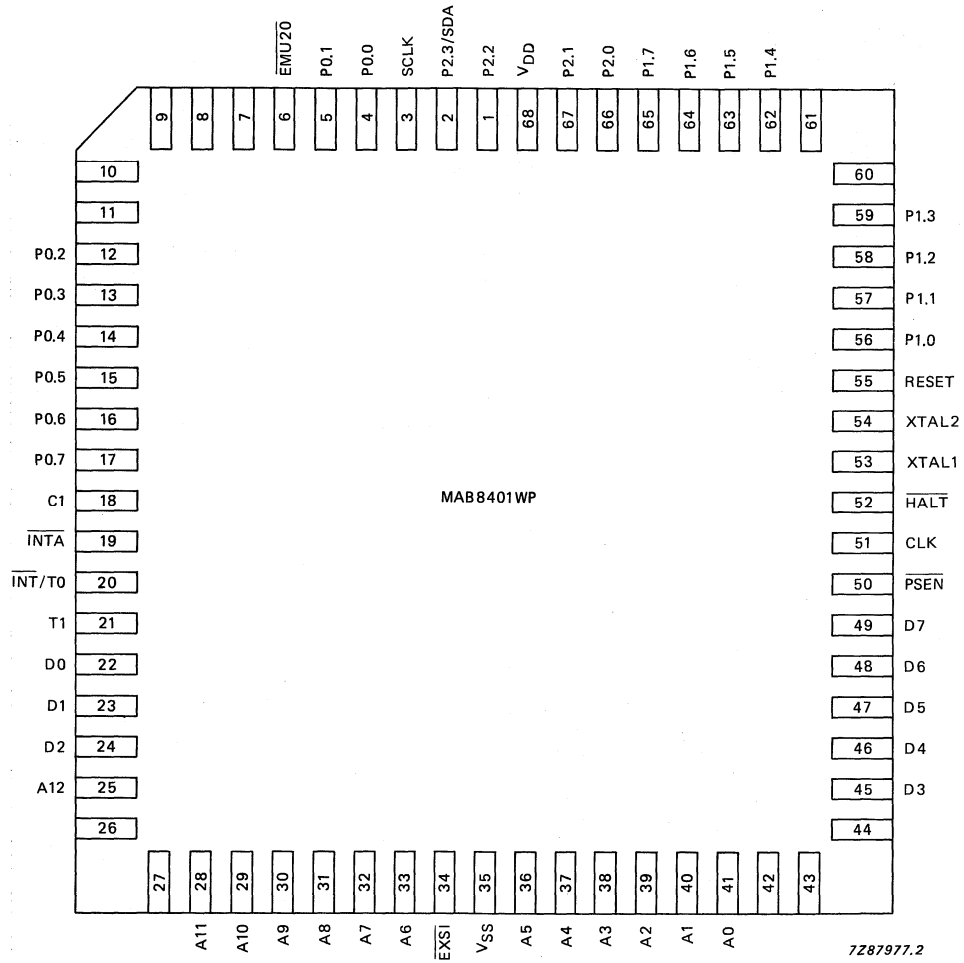


Fig. 4 Pinning diagram; PLCC.

CHIP CARRIER DESIGNATION

designation	pad no.	function
V_{SS}	35	**Ground**
V_{CC}	68	**Power supply,** + 5 V
P0.0 – P0.7	4–5, 12–17	**Port 0,** 8-bit quasi-bidirectional I/O port
P1.0 – P1.7	56–59, 62–65	**Port 1,** 8-bit quasi-bidirectional I/O port with 8-bit LED driver
P2.0 – P2.3	66, 67, 1, 2	**Port 2,** 4-bit quasi-bidirectional I/O port; SDA/P2.3 is the serial data I/O in serial I/O mode
SCLK	3	Bidirectional clock for serial I/O
INT/T0	20	External interrupt input (sensitive to a negative-going edge), testable using the JT0 or JNT0 instructions

T1	21	Input pin, testable using the JT1 or JNT1 instructions. It can be designated as event counter input using the STRT CNT inctruction. It can also be used to detect zero cross-over of slowly moving a.c. inputs.
RESET	55	Input to initialize the processor (active HIGH)
XTAL1	53	Connection to timing component (e.g. crystal) that determines the frequency of the internal oscillator. It is also the input for an external clock source.
XTAL2	54	Connection to other side of the timing component
$\overline{\text{EXSI}}$	34	External serial I/O interrupt (active-LOW) for emulation of MAB/F8422/42.
A0–A12	41–36, 33–28 25	Program memory address outputs (active HIGH); A0 = LSB, A12 = MSB. Address output change after begin ϕ3 of TS8.
D0–D7	22–24, 45–49	Data input lines (active HIGH) used for reading external program memory. D0 = LSB, D7 = MSB.
CLK	51	Clock output buffered from XTAL2. On the positive-going edge the (internal) ϕ clock goes HIGH.
$\overline{\text{PSEN}}$	50	Program store enable. This signal is used for enabling the external EPROM (e.g. on the 'Piggy-back' version). For emulation, it enables the emulation memory and it indicates machine cycles. Active LOW during TS9,*TS10 of each machine cycle and TS1 of the following machine cycle.
$\overline{\text{C1}}$	18	Cycle 1 indication output (active LOW). During emulation, this signal indicates the opcode fetch cycle (useful for external instruction decoding, real-time trace). Active from start of TS10 of the cycle preceding cycle 1, until the start of TS10 of cycle 1.
$\overline{\text{HALT}}$	52	Halt input (active LOW). If activated, the current instruction is finished and the microcontroller stops execution (HALT mode). The next program counter address is available on the address bus. Program counter and timer/event counter are no longer updated. The serial I/O finishes the current transmit/receive action and goes into the idle state. Interrupts are *not* sampled in the HALT mode, they are only sampled when the microcontroller is running. Interrupt routines can be single-stepped as a normal program.
$\overline{\text{INTA}}$	19	Interrupt acknowledge output (active LOW). It indicates any interrupt acception. Active from start of TS8 of the interrupted cycle, until start of TS7 of the second cycle of the (internally forced 'CALL vector address' instruction. During $\overline{\text{INTA}}$ active, the address bus shows the address that has been saved in the stack (return address); the C1 output indicates opcode fetch cycles as if a user CALL was executed.
$\overline{\text{EMU20}}$	6	Emulate 20-pin version MAB/F8422/42 (active-LOW).

* TS = Time slot, where 10 TS = 1 cycle.

FUNCTIONAL DESCRIPTION (for more detail see 84XXX family specification)

Bond-out version MAB8401WP

The bond-out version is a microcontroller that contains no on-board ROM, but has all address and data lines brought out to access an external ROM or EPROM. Thus, this version has more pins than the standard microcontrollers with on-board ROM. It has all the features of the other members of the MAB84X1 family, including emulation facilities for the MAB/F8422/42 (20-pin version). It can address 8K bytes of external ROM. The RAM has 128 bytes.

Piggy-back version MAB8401B

The Piggy-back version is a special package that has standard pinning to the bottom which facilitates insertion as a mask-programmed device. An EPROM is mounted on top in an additional socket. Thus, the total package height is greater than the standard DIL package. Emulation of the 8422/42 is not possible.

Program and data memory

The program memory (ROM) is mask-programmed at our factory. Because the MAB84X1 family offers a range of ROM capacities to suit the application, ROM expansion is not required. Figure 5 shows the program memory map. Program memory is arranged in banks of 2K bytes, that are selected by SEL MB instructions.

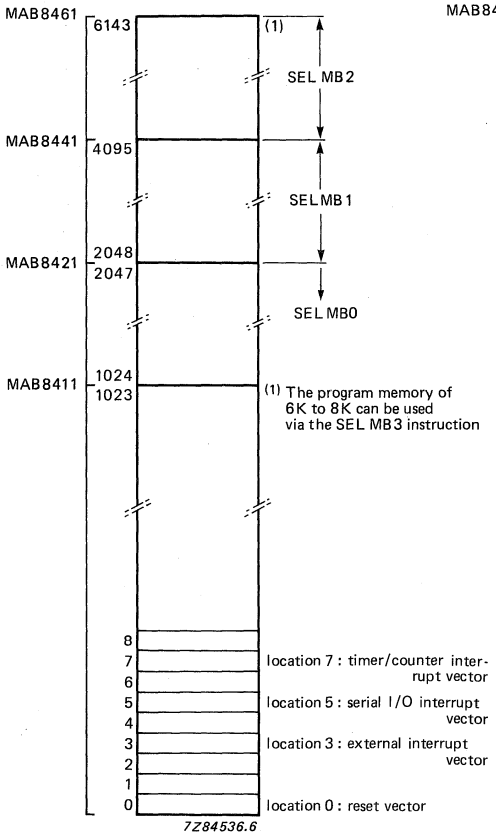


Fig. 5 The program memory map.

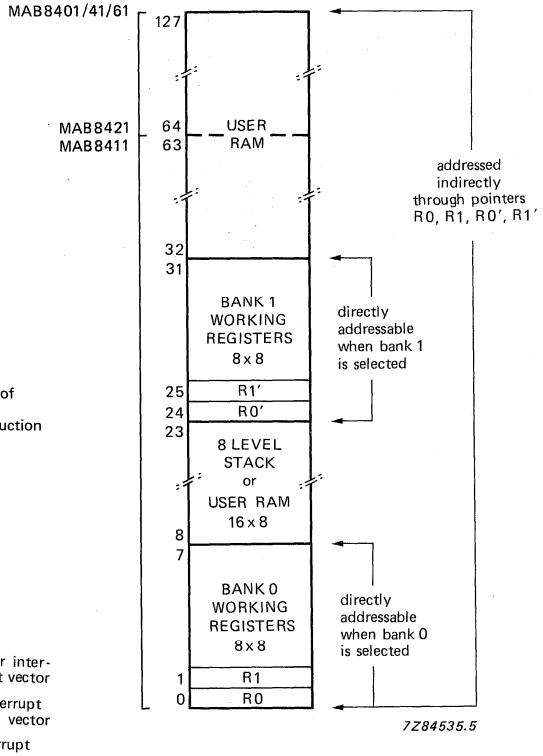


Fig. 6 The data memory map.

FUNCTIONAL DESCRIPTION (continued)

The data memory (RAM) consists of 64 or 128 bytes (8-bit words). All locations are indirectly addressable using RAM pointer registers and up to 16 designated location can be addressed directly. The memory also includes an 8-level program counter stack addressed by a 3-bit stack pointer. Figure 6 shows the data memory map.

On-chip peripheral functions

In addition to the CPU and memories, an interrupt system, I/O facilities, and an 8-bit timer/event counter are integrated on-chip to assist the CPU in repetitions, complicated or time-critical tasks. The I/O facilities include the I/O pins, parallel ports and a serial I/O port, consisting of a data line SDA shared with a parallel port line (P2.3), and a dedicated clock line SCLK.

I/O facilities

The MAB84XX family has 23 I/O lines arranged as:

- Two parallel ports of 8 lines (P0.0–P0.7, P1.0–P1.7). Each line of Port 1 can sink 10 mA.
- A parallel port of 4 lines (P2.0–P2.3).
- A serial I/O consisting of a data line shared with a parallel port line (P2.3) and a separate clock line SCLK;
- An external interrupt and test input $\overline{\text{INT}}$/T0, which when used as a test input can be tested by the conditional jump instructions JT0 or JNT0;
- A test input T1, which can alter program sequences when tested by conditional jump instructions JT1 or JNT1. T1 can also be used as an input to the timer/event counter or to detect zero cross-over of slowly moving AC signals.

All parallel port lines are available in three optional output configurations (except P2.3 – option 1 only):

- Option 1; open drain output without pull-up transistor (Fig. 7(a))
- Option 2; open drain output with pull-up transistor (Fig. 7(b))
- Option 3; push-pull output with pull-up transistor (Fig. 7(c))

If the inputs and outputs on a port are mixed (mixed-mode), the inputs should be options 1 or 2 but not option 3. This prevents cross-currents via TR2 and an external connection to ground, while switching the output on the same port and in parallel, masking the inputs with logic 1s.

The MAB84X1 family serial I/O interface has been designed to eliminate the heavy processing load imposed upon a normal microcontroller performing serial data transfer. Whereas a normal microcontroller must regularly monitor the serial data bus for the presence of data, the serial I/O interface detects, receives and converts the serial data stream into a parallel format without interrupting the execution of the current program. An interrupt is sent to the microcontroller only when a complete byte is received. Then, the microcontroller reads the data byte in one instruction. Likewise, for transmission, the serial I/O interface performs parallel to serial conversion and subsequent serial output of the data and the microcontroller is only interrupted in the execution of its programmed tasks when a complete byte has been transmitted. The design of the serial I/O interface allows any number of MAB84X1 family devices and peripheral circuits with I^2C bus compatibility to be interconnected by the two-line serial bus. This is achieved by allocating a specific 7-bit address to each device and ensuring that a device reacts only to a message preceded by its own address or the 'general call' address.

Address recognition is performed by the interface hardware so that the microcontroller need only be interrupted when a valid address is received. This saves significant processing time and memory space compared to a conventional microcontroller with a software serial interface. When the address facility is not required, for instance in a system with only two microcontrollers, direct data transfer is possible. In multi-master systems, an automatically invoked arbitration procedure prevents two or more devices transmitting simultaneously.

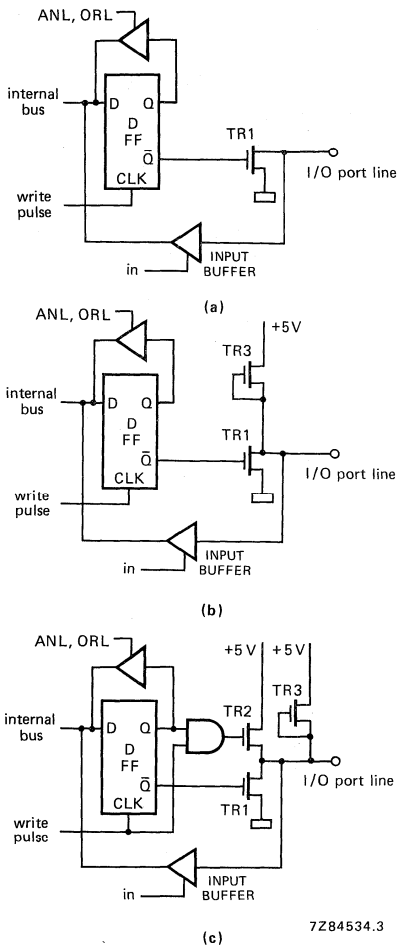


Fig. 7 Quasi-bidirectional I/O interface with (a) open drain output without pull-up transistor, (b) open drain output with pull-up transistor, (c) push-pull output with pull-up transistor.

Serial I/O interface

Figure 8 shows the serial I/O interface. The clock line of the serial bus has exclusive use of pin 3 (SCLK) while the data line shares pin 2 (serial data) with the I/O line P2.3 of port 2. When the serial I/O is enabled, P2.3 is disabled as a parallel port line (P2.3 and SCLK only open drain).

The microcontroller and interface communicate via the internal microcontroller bus and the Serial Interrupt Request line. Data and information controlling the operation of the interface are stored in four registers:

- data shift register S0,
- serial I/O interface status word S1,
- serial clock control word S2,
- address register S0'

FUNCTIONAL DESCRIPTION (continued)

Serial I/O interface (continued)

Data shift register S0

S0 is the shift register that converts serial data to parallel format and vice versa. A pending interrupt is generated only after a complete byte has been transmitted, or after a complete data byte, specific or general call address has been received. The most significant bit is transmitted first.

Serial I/O interface status word S1

S1 provides information about the state of the interface and stores interface control information from the microcontroller. The four most significant bits are common to both read and write instructions, with a separate 4 read-only control bits and 4 write-only interface status bits.

MST and TRX

These bits determine the operating mode of the serial I/O interface (Table 2).

Table 1 Operating modes of the serial I/O interface.

MST	TRX	mode
0	0	slave receiver
1	0	master receiver
0	1	slave transmitter
1	1	master transmitter

BB: Bus Busy

This bit indicates the status of the bus.

PIN: Pending Interrupt Not

PIN = '0' indicates that there is an interrupt pending. This causes a Serial Interrupt Request when the serial interrupt mechanism is enabled.

ESO: Enable Serial Output

The ESO flag enables/disables the serial I/O interface: ESO = logic 1 enables
ESO = logic 0 disables

BC0, BC1 and BC2

These bits indicate the number of bits received or transmitted in a serial data stream.

Bits ES0, BC0, BC1 and BC2 can only be written via software.

AL: Arbitration Lost

The AL flag is set via the hardware when the serial I/O interface, as a master transmitter, loses the bus arbitration procedure.

AAS: Addressed As Slave

This flag is set via the hardware when the interface detects either its own address or the 'general call' address as the first byte of a transfer and if the interface has been programmed to operate in the address recognition mode.

AD0: Address Zero

This flag is set via the hardware after the general call address is detected when the interface is operating in the address recognition mode.

LRB: Last Received Bit

This contains either the last data bit received or, for a transmitting device in the acknowledge mode, the acknowledge from the receiving device.

Bits AL, AAS, AD0 and LRB can only be read via software.

Serial clock control register S2

Bits 0 to 4 of S2 are used to set the frequency of the serial clock signal. When a 4,43 MHz crystal is used, the frequency of the serial clock can be varied between 100 kHz and 720 Hz. An asymmetrical clock with a HIGH to LOW ratio of 3 to 1 is produced by setting bit 5. The asymmetrical clock allows a microcontroller more time per clock period for sampling the data line, making the timing of this action less critical. Bit 6 is used to activate the acknowledge mode of the serial I/O. S2 is a write-only register.

Address register S0'

The address register contains the 7-bit address back-up latches and the bit (ALS) used to enable/disable the address recognition mode. Only when ES0 = 0 can the address register be written using the MOV S0,A and MOV S0,#data instructions.

Serial I/O interrupt logic

The interrupt logic is enabled by the EN SI instruction and disabled by DIS SI. When the interrupt logic is enabled, a pending interrupt results in a serial I/O interrupt to the controller, causing a jump to location 5 in the ROM. When the logic is disabled, the presence of an interrupt is still indicated by the PIN bit in register S1. Therefore, an interrupt can still be serviced but a vectored interrupt will not occur.

Interrupt system

External events and real-time on-chip peripherals require servicing by the CPU asynchronous to the execution of any particular section of code. To tie the asynchronous activities of these functions to normal program execution, three single-level nested interrupts are provided.

Each interrupt vectors to a separate location in the program memory for its service program. Each source can be individually enabled or disabled. When more than one interrupt occurs simultaneously, their priority will be: (1) external, (2) serial I/O and, (3) timer/event counter. An additional external interrupt can be created using the timer/event counter interrupt.

FUNCTIONAL DESCRIPTION (continued)

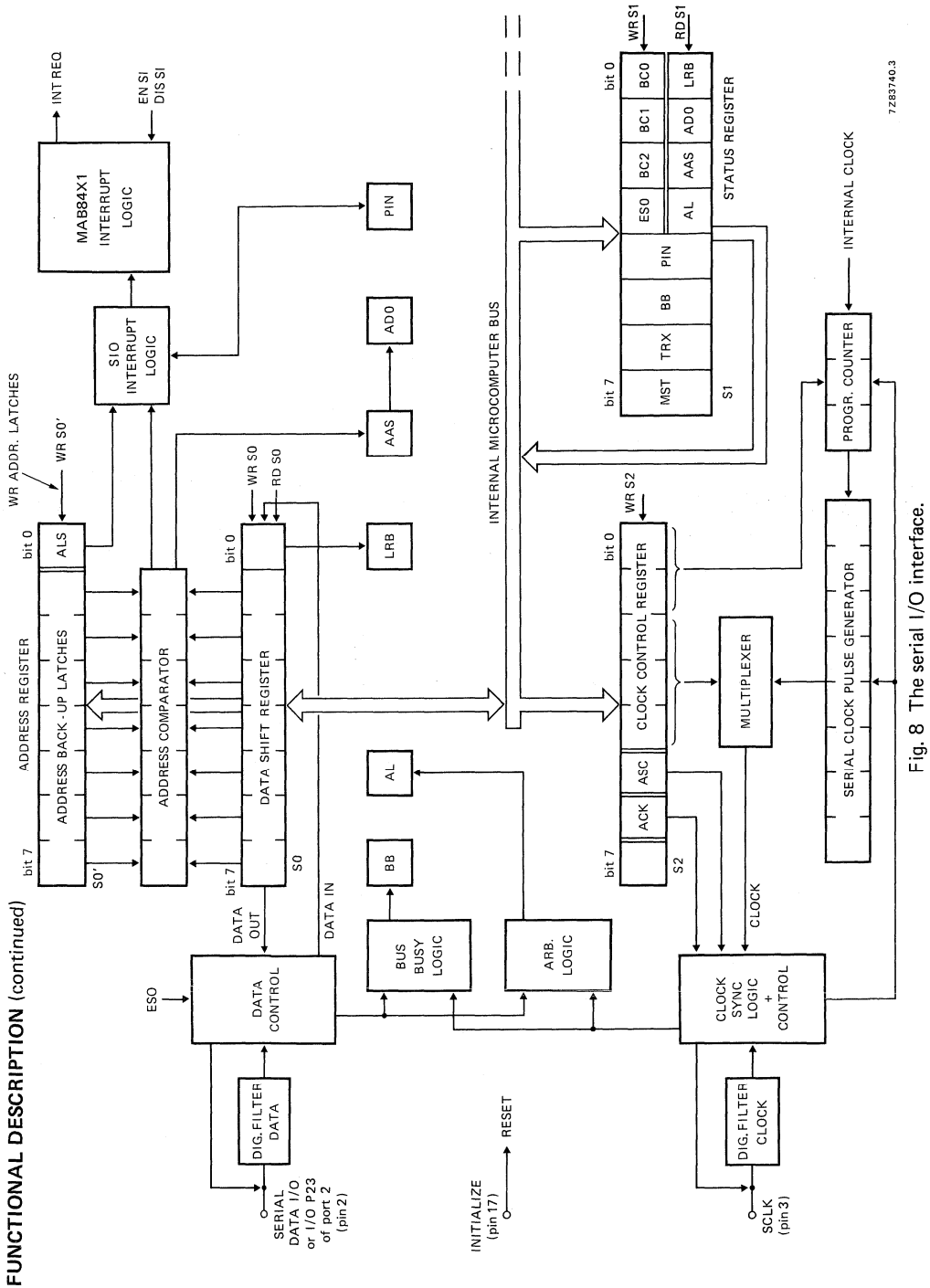


Fig. 8 The serial I/O interface.

Test input T1

The T1 input line can be used as:

- a test input for branch instructions,
- an input for zero voltage cross-over detection,
- an external input to the event counter.

An internal pull-up transistor is provided as a ROM mask option. This is useful when the input is from a switch or standard TTL output.

When T1 is used as a test input, the JT1 or JNT1 instructions test for a HIGH or a LOW respectively.

When used for zero-cross detection purposes, the T1 input must be coupled through a capacitor of typical value 1 μF and operation carried out using the T1 input without the pull-up transistor. The maximum input voltage amplitude is 3 V (peak-to-peak), with a maximum operational frequency of 1 kHz. The T1 input has an on-chip DC offset circuit which self-biases the input to its exact switching level of 1 V. As a consequence a small change will cause a digital transition to occur. The switching level of the T1 input circuit is within the bias voltage of ± 135 mV. Upon each positive cycle on the pin, the event counter is incremented and an overflow will set the timer flag TF. Zero cross-over detection used in conjunction with the timer/event counter interrupt, is useful in thyristor control of power equipment. Figure 9 illustrates, (a) the input waveform, (b) the input diagram and (c) the on-chip self-stabilized bias.

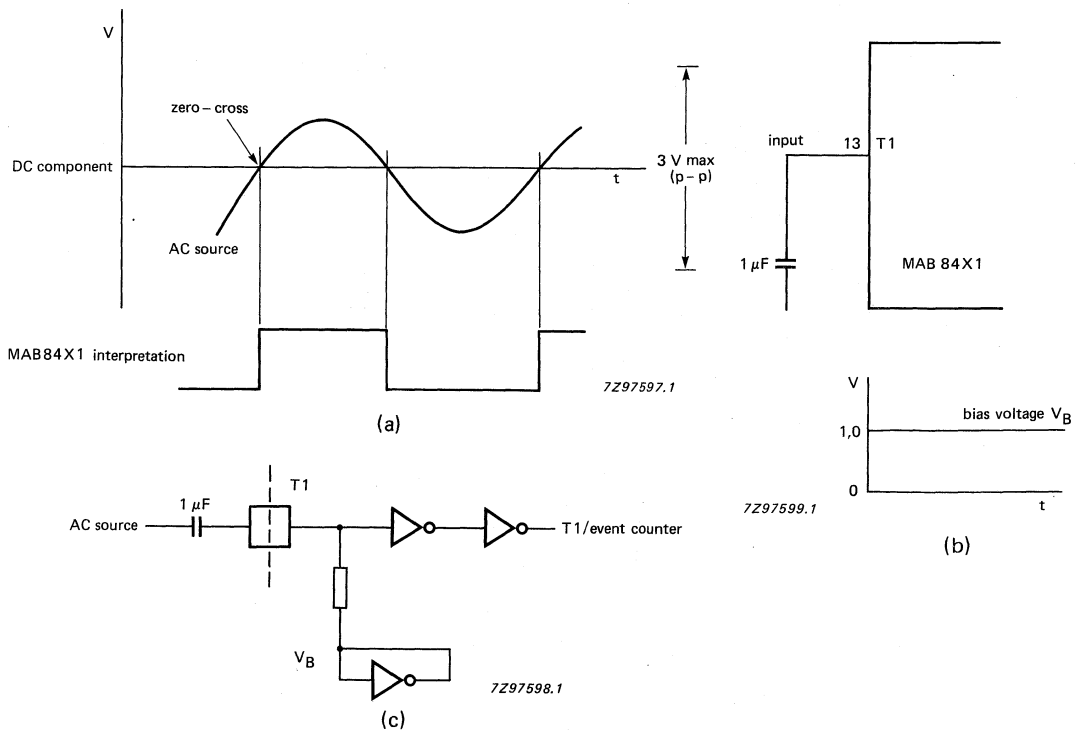


Figure 9 Zero-cross detection circuitry; (a) input waveform, (b) input diagram, (c) on-chip self-stabilized bias.

The operation of T1 as an input to the timer/event counter is described under the heading Timer/ event counter.

High current outputs

Ten pins are provided that can sink high currents:

– P2.3 (serial data), pin 2	5 mA at 0,45 V (open drain),
– SCLK, pin 3	5 mA at 0,45 V (open drain),
– P1.0 – P1.7 *	10 mA at 1 V

Timer/event counter

An 8-bit binary up-counter is provided. This can count external events, machine cycles divided by 32, or machine cycles directly. When used as a timer, the input to the counter is either the overflow or input of a 5-bit prescaler. When used as an event counter, LOW to HIGH transitions on T1 (pin 13) are counted. The maximum rate at which the counter may be incremented is once every machine cycle (200 kHz for a 5 μs machine cycle). Figure 10 illustrates the timer/event counter.

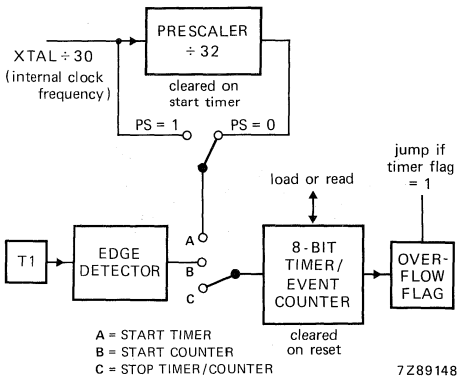


Fig. 10 The timer/event counter.

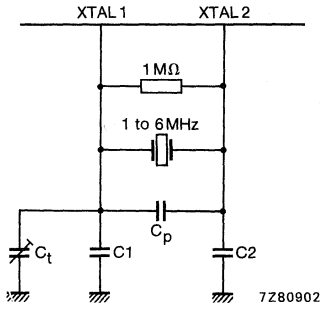
Differences between the MAB8021 and MAB8048 microcontrollers, and the MAB84X1 family.

	8021	8048	8401, 8411 8421, 8441, 8461
ROM capacity (bytes)	1K	1K	ROMless, 1K, 2K, 4K, 6K
RAM capacity (bytes)	64	64	128, 64, 64, 128, 128
paralel I/O lines	8 + 8 + 4	8 + 8 + 8	8 + 8 + 4
single inputs	1	3	2
serial I/O	no	no	yes, 2-line multi-transmitter
timer	8 bit	8 bit	8 bit
prescaler	mod. 32	mod. 32	mod. 1 & mod. 32
machine cycle time (μs)	10	2,5	5
for clock (MHz)	3	6	6
instruction set	8021	8048	8048 with omissions; 5 new serial I/O instructions; 2 new register instructions; 2 new control instructions; 1 new cond. branch instruction
interrupts	none	2 external timer/ event counter	3 external serial I/O timer/event counter
no. of pins (DIL)	28	40	68 (PLCC), 28

* P1.0 to P1.7 may be connected in parallel if their logic outputs are always the same.

OSCILLATOR CIRCUITRY

Clock frequency is determined by using the internal oscillator or by connecting an external clock to XTAL1. Where the internal oscillator is used, the frequency is set by a crystal between XTAL1 and XTAL2, or by a ceramic resonator or an inductor, each with two associated capacitors, between XTAL1 and XTAL2 (see Fig. 11a). A machine cycle consists of 10 states, each state being 3 oscillator periods. The common 6 MHz crystal gives a 5 μs machine cycle. The MAB84X1 family has dynamic logic, and therefore, for adequate refreshing the oscillator frequency must be at least 1 MHz.



1. Crystal – AT-cut
2. Ceramic resonator
   C1 = C2 = 27 pF
   C1 may be trimmed
   $C_p \leqslant 6{,}75$ pF (parasitic capacitance)

Fig. 11a Quartz crystal or ceramic resonator mode.

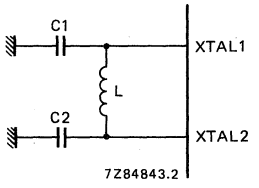
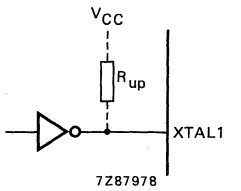


Fig. 11b LC pi-network.

LC oscillator timing

frequency	C1 = C2	L
3,0 MHz	33 pF	100 μH
4,0 MHz	33 pF	56 μH
4,4 MHz	33 pF	47 μH
5,0 MHz	33 pF	33 μH
6,0 MHz	33 pF	22 μH



Drive XTAL1
Leave XTAL2 open

Driver may be high-speed CMOS or any TTL
t_r, $t_f < 10$ ns

Fig. 11c External drive.

PROGRAM STATUS WORD

The program status word (PSW) is an 8-bit word in the CPU which stores information about the current status of the microcontroller (Fig. 12). The PSW bits are:

bits 0, 1 and 2 – stack pointer bits (SP_0, SP_1, SP_2);

bit 3 – prescaler select (PS); 0 = divide-by-32; 1 = no prescaling;

bit 4 – working register bank select (RBS):
0 = register bank 0
1 = register bank 1;

bit 5 – not used (1);

bit 6 – auxiliary carry (AC):
half-carry bit is generated by an ADD instruction and used by the decimal adjust instruction DA A;

bit 7 – carry (CY):
the carry flag indicates that the previous operation has resulted in an overflow of the accumulator.

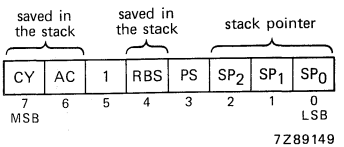


Fig. 12 Program status word.

All bits can be read using MOV A, PSW and bit 3 can be written with MOV PSW, A.

Bits 6 and 7 can be set and cleared by CPU operation. Bit 4 is changed by the SEL RB instruction, bit 3 by the MOV PSW,A instruction, and bits 0, 1 and 2 by the CALL, RET or RETR instructions and when an interrupt occurs. Bits 4, 6 and 7 are stored in the program counter stack during subroutine and interrupt calls. These bits are restored to the PSW with RETR (return and restore) instruction.

Note: The RET instruction has no restore feature and should not be used at the end of an interrupt because this would leave any further interrupts disabled.

The MAB84X1 family has arithmetic, logical and branching capabilities. The DA A, SWAP A, and XCHD instructions simplify BCD arithmetic and the handling of nibbles. The MOVP A,@A instruction permits efficient table look up from the current ROM page.

The conditional branch logic within the processor enables several conditions, internal and external to the processor, to be tested by the user's program. Table 2 lists the conditional branch instructions used to change the program execution sequence. The DJNZ instruction decrements a designated register and branches if the contents are not zero. This instruction makes the register an efficient program loop counter. The JMPP @A instruction allows multiway branches to destinations indirectly addressed by the contents of the accumulator.

Table 2 Conditional branches

TEST	JUMP CONDITION	JUMP INSTRUCTION
accumulator	0 or non-zero	JZ, JNZ
accumulator bit test	1	JB0 to JB7
carry flag	0 or 1	JNC, JC
timer overflow flag	1	JTF
test input $\overline{INT}$	0 or 1	JNT0, JT0
test input T1	0 or 1	JNT1, JT1
test flag 0	1	JF0
test flag 1	1	JF1
register	non-zero	DJNZ

RESET

A positive-going signal on the RESET input:

- sets the program counter to zero,
- selects location 0 of memory bank 0, and register bank 0,
- sets the stack pointer to zero ('000'B); pointing to RAM address 8,
- disable the interrupts (external, timer and serial I/O),
- stops the timer/event counter, then sets it to zero,
- sets the timer prescaler to divide-by-32,
- resets the timer flag,
- sets all ports to logic '1' (input mode),
- sets the serial I/O to slave receiver mode and disables serial I/O.

Automatic reset at power-up may be obtained by connecting the RESET pin to V_{CC} through a 1 μF capacitor C, together with a diode to V_{SS} (cathode to RESET pin). This arrangement is satisfactory, if both the voltage (V_{CC}) rise time and the oscillator start-up time do not exceed either 1 or 10 ms respectively.

The power-on reset circuit is shown in figure 13. At power-on the current drawn by RESET commences to charge the capacitor C. The difference between this increasing capacitor voltage and V_{CC} is known as V_{RESET}. The charging circuit is designed to hold V_{RESET} above the lower threshold of a Schmitt trigger arrangement long enough to effect a complete reset. The minimum time required; is the oscillator start-up time plus two machine cycles.

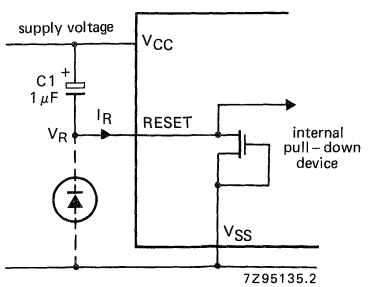


Fig. 13 Typical power-on reset circuitry.

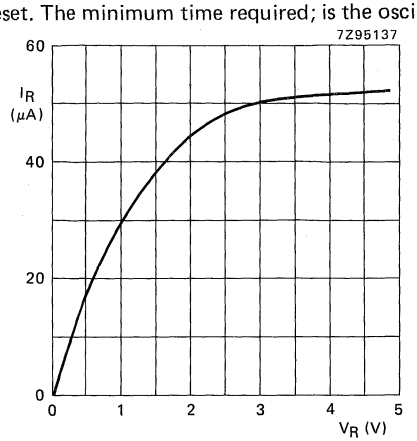


Fig. 14 Power-on reset input characteristics (typical).

INSTRUCTION SET

The instruction set consists of over 80 one and two byte instructions and is based on the MAB8048 instruction set. New instructions include those for serial I/O operation and memory bank selection. Program code efficiency is high because all ROM locations on a 256 byte page require only a single byte address.

Table 3 gives the instruction set of the MAB84X1 family and Table 4 shows the instruction map. The following symbols and abbreviations are used.

Note: During development of software on a PMDS or similar system, it is important to ensure that no jump instruction (direct or indirect), outreaches the final address range of the device.

symbol	description
A	the accumulator
AC	the auxiliary carry flag
addr	program memory address (11-bits)
Bb	bit designation (b = 0–7)
BS	the bank switch
C	carry flag
CLK	clock signal
CNT	event counter
D	nibble designation (4-bits)
DBF	program memory bank flip-flop
data	number or expression (8-bits)
F0, F1	flags 0 and 1
I	interrupt
$\overline{\text{INT}}$	external interrupt
P	'in-page' operation designation
Pp	port designation (p = 1, 2 or 4–7)
PSW	program status word
Rr	register designation (r = 0, 1 or 0–7)
SP	stack pointer
T	timer
TF	timer flag
T0, T1	test 0 and 1 inputs
#	immediate data prefix
@	indirect address prefix
$	current value of program counter
←	is replaced by
↔	is exchanged with

Table 3 MAB84XX family instruction set

	mnemonic	opcode (hex.)	bytes/ cycles	description	function		notes
ACCUMULATOR	ADD A, Rr	6*	1/1	Add register contents to A	(A)←(A) + (Rr)	r = 0–7	1
	ADD A, @Rr	60 61	1/1	Add RAM data, addressed by Rr, to A	(A)←(A) + ((R0)) (A)←(A) + ((R1))		1
	ADD A, #data	03 data	2/2	Add immediate data to A	(A)←(A) + data		1
	ADDC A, Rr	7*	1/1	Add carry and register contents to A	(A)←(A) + (Rr) + (C)	r = 0–7	1
	ADDC A, @Rr	70 71	1/1	Add carry and RAM data, addressed by Rr, to A	(A)←(A) + ((R0)) + (C) (A)←(A) + ((R1)) + (C)		1
	ADDC A, #data	13 data	2/2	Add carry and immediate data to A	(A)←(A) + data + (C)		1
	ANL A, Rr	5*	1/1	'AND' Rr with A	(A)←(A) AND (Rr)	r = 0–7	
	ANL A, @Rr	50 51	1/1	'AND' RAM data, addressed by Rr, with A	(A)←(A) AND ((R0)) (A)←(A) AND ((R1))		
	ANL A, #data	53 data	2/2	'AND' immediate data with A	(A)←(A) AND data		
	ORL A, Rr	4*	1/1	'OR' Rr with A	(A)←(A) OR (Rr)	r = 0–7	
	ORL A, @Rr	40 41	1/1	'OR' RAM data, addressed by Rr, with A	(A)←(A) OR ((R0)) (A)←(A) OR ((R1))		
	ORL A, #data	43 data	2/2	'OR' immediate data with A	(A)←(A) OR data		
	XRL A, Rr	D*	1/1	'XOR' Rr with A	(A)←(A) XOR (Rr)	r = 0–7	
	XRL A, @Rr	D0 D1	1/1	'XOR' RAM, addressed by Rr, with A	(A)←(A) XOR ((R0)) (A)←(A) XOR ((R1))		
	XRL A, #data	D3 data	2/2	'XOR' immediate data with A	(A)←(A) XOR data		
	INC A	17	1/1	increment A by 1	(A)←(A) + 1		
	DEC A	07	1/1	decrement A by 1	(A)←(A) – 1		
	CLR A	27	1/1	clear A to zero	(A)←0		
	CPL A	37	1/1	one's complement A	(A)←NOT(A)		
	RL A	E7	1/1	rotate A left	$(A_{n+1})\leftarrow(A_n)$ $(A_0)\leftarrow(A_7)$	n = 0–6	

	mnemonic	opcode (hex.)	bytes/ cycles	description	function	notes
ACCUMULATOR (cont.)	RLC A	F7	1/1	rotate A left through carry	$(A_{n+1})\leftarrow A_n$ n = 0–6 $(A_0)\leftarrow(C)$, $(C)\leftarrow(A_7)$	2
	RR A	77	1/1	rotate A right	$(A_n)\leftarrow(A_{n+1})$ n = 0–6 $(A_7)\leftarrow(A_0)$	
	RRC A	67	1/1	rotate A right through carry	$(A_n)\leftarrow(A_{n+1})$ n = 0–6 $(A_7)\leftarrow(C)$, $(C)\leftarrow(A_0)$	2
	DA A	57	1/1	decimal adjust A		2
	SWAP A	47	1/1	swap nibbles of A	$(A_{4-7})\leftrightarrow(A_{0-3})$	
DATA MOVES	MOV A, Rr	F*	1/1	move register contents to A	(A)←(Rr) r = 0–7	
	MOV A, @Rr	F0 F1	1/1	move RAM data, addressed by Rr, to A	(A)←((R0)) (A)←((R1))	
	MOV A, #data	23 data	2/2	move immediate data to A	(A)←data	
	MOV Rr, A	A*	1/1	move accumulator contents to register	(Rr)←(A) r = 0–7	
	MOV @Rr, A	A0 A1	1/1	move accumulator contents to RAM location addressed by Rr	((R0))←(A) ((R1))←(A)	
	MOV Rr, #data	B* data	2/2	move immediate data to Rr	(Rr)←data	
	MOV @Rr, #data	B0 data B1 data	2/2	move immediate data to RAM location addressed by Rr	((R0))←data ((R1))←data	
	XCH A, Rr	2*	1/1	exchange accumulator contents with Rr	(A)↔(Rr) r = 0–7	
	XCH A, @Rr	20 21	1/1	exchange accumulator contents with RAM data addressed by Rr	(A)↔((R0)) (A)↔((R1))	
	XCHD A, @Rr	30 31	1/1	exchange lower nibbles of A and RAM data addressed by Rr	$(A_{0-3})\leftrightarrow((R0_{0-3}))$ $(A_{0-3})\leftrightarrow((R1_{0-3}))$	
	MOV A, PSW	C7	1/1	move PSW contents to accumulator	(A)←(PSW)	
	MOV PSW, A	D7	1/1	move accumulator bit 3 to PSW_3	$(PSW_3)\leftarrow(A_3)$	3
	MOVP A, @A	A3	1/2	move indirectly addressed data in current page to A	$(PC_{0-7})\leftarrow(A)$, $(A)\leftarrow((PC))$	

FLAGS	CLR C	97	1/1	clear carry bit	(C)←0		2
	CPL C	A7	1/1	complement carry bit	(C)←NOT(C)		2
REGISTER	INC Rr	1*	1/1	increment register by 1	(Rr)←(Rr) + 1	r = 0–7	
	INC @Rr	10 11	1/1	increment RAM data, addressed by Rr, by 1	((R0))←((R0)) + 1 ((R1))←((R1)) + 1		
	DEC Rr	C*	1/1	decrement register by 1	(Rr)←(Rr) – 1	r = 0–7	
	DEC @Rr	C0 C1	1/1	decrement RAM data, addressed by Rr, by 1	((R0))←((R0)) – 1 ((R1))←((R1)) – 1		
BRANCH	JMP addr	• 4 address	2/2	unconditional jump within a 2K bank	(PC_{8-10})←$addr_{8-10}$ (PC_{0-7})←$addr_{0-7}$ (PC_{11-12})←MBFF 0–1		
	JMPP @A	B3	1/2	indirect jump within a page	(PC_{0-7})←((A))		
	DJNZ Rr, addr	E* address	2/2	decrement Rr by 1 and jump if not zero to addr	(Rr)←(Rr) – 1 if (Rr) not zero (PC_{0-7})←addr	r = 0–7	
	DJNZ @Rr, addr	E0 address	2/2	decrement RAM data, addressed by Rr, by 1 and jump if not zero to addr	((R0))←((R0)) – 1 if ((R0)) not zero (PC_{0-7})←addr		
		E1 address			((R1))←((R1)) – 1 if ((R1)) not zero (PC_{0-7})←addr		
	JBb addr	▲ 2 address	2/2	jump to addr if Acc. bit b = 1	if b = 1: (PC_{0-7})←addr	b = 0–7	
	JC addr	F6 address	2/2	jump to addr if C = 1	if C = 1: (PC_{0-7})←addr		
	JNC addr	E6 address	2/2	jump to addr if C = 0	if C = 0: (PC_{0-7})←addr		
	JZ addr	C6 address	2/2	jump to addr if A = 0	if A = 0: (PC_{0-7})←addr		
	JNZ addr	96 address	2/2	jump to addr if A is NOT zero	if A ≠ 0: (PC_{0-7})←addr		
	JT0 addr	36 address	2/2	jump to addr if T0 = 1	if T0 = 1: (PC_{0-7})←addr		
	JNT0 addr	26 address	2/2	jump to addr if T0 = 0	if T0 = 0: (PC_{0-7})←addr		
	JT1 addr	56 address	2/2	jump to addr if T1 = 1	if T1 = 1: (PC_{0-7})←addr		
	JNT1 addr	46 address	2/2	jump to addr if T1 = 0	if T1 = 0: (PC_{0-7})←addr		
	JTF addr	16 address	2/2	jump to addr if Timer Flag = 1	if TF = 1: (PC_{0-7})←addr		4
	JNTF addr	06 address	2/2	jump to addr if Timer Flag = 0	if TF = 0: (PC_{0-7})←addr		

	mnemonic	opcode (hex.)	bytes/ cycles	description	function	notes
TIMER/EVENT COUNTER	MOV A, T	42	1/1	move timer/event counter contents to accumulator	(A)←(T)	
	MOV T, A	62	1/1	move accumulator contents to timer/event counter	(T)←(A)	
	STRT CNT	45	1/1	start event counter		
	STRT T	55	1/1	start timer		
	STOP TCNT	65	1/1	stop timer/event counter		
	EN TCNTI	25	1/1	enable timer/event counter interrupt		
	DIS TCNTI	35	1/1	disable timer/event counter interrupt		
CONTROL	EN I	05	1/1	enable external interrupt		
	DIS I	15	1/1	disable external interrupt		
	SEL RB0	C5	1/1	select register bank 0	(RBS)←0	5
	SEL RB1	D5	1/1	select register bank 1	(RBS)←1	5
	SEL MB0	E5	1/1	select program memory bank 0	(MBFF0)←0, (MBFF1)←0	
	SEL MB1	F5	1/1	select program memory bank 1	(MBFF0)←1, (MBFF1)←0	
	SEL MB2	A5	1/1	select program memory bank 2	(MBFF0)←0, (MBFF1)←1	
	SEL MB3	B5	1/1	select program memory bank 3	(MBFF0)←1, (MBFF1)←1	
SUBROUTINE	CALL addr	▲ 4 address	2/2	jump to subroutine	((SP))←(PC), ($PSW_{4, 6, 7}$) (SP)←(SP) + 1 (PC_{9-10})←$addr_{8-10}$ (PC_{0-7})←$addr_{0-7}$ (PC_{11-12})←MBFF 0–1	6
	RET	83	1/2	return from subroutine	(SP)←(SP) – 1 (PC)←((SP))	6
	RETR	93	1/2	return from interrupt and restore bits 4, 6, 7 of PSW	(SP)←(SP) – 1 ($PSW_{4, 6, 7}$) + (PC)←((SP))	6

PARALLEL INPUT/OUTPUT	IN A, Pp	08 09 0A	1/2	input port p data to accumulator	(A)←(P0) (A)←(P1) (A)←(P2)	7
	OUTL Pp, A	38 39 3A	1/2	output accumulator data to port p	(P0)←(A) (P1)←(A) (P2)←(A)	
	ANL Pp, #data	98 data 99 data 9A data	2/2	AND port p data with immediate data	(P0)←(P0) AND data (P1)←(P1) AND data (P2)←(P2) AND data	
	ORL Pp, #data	88 data 89 data 8A data	2/2	OR port p data with immediate data	(P0)←(P0) OR data (P1)←(P1) OR data (P2)←(P2) OR data	
	OUTL PO,A	90	1/2	Output accumulator data to port ϕ	(P0)←(A)	9
SERIAL INPUT/OUTPUT	MOV A, S_n	0C 0D	1/2	move serial I/O register contents to accumulator	(A)←(S0) (A)←(S1)	8
	MOV S_n, A	3C 3D 3E	1/2	move accumulator contents to serial I/O register	(S0)←(A) (S1)←(A) (S2)←(A)	
	MOV S_n, #data	9C data 9D data 9E data	2/2	move immeidate date to serial I/O register	(S0)←data (S1)←data (S2)←data	
	EN SI	85	1/1	enable serial I/O interrupt		
	DIS SI	95	1/1	disable serial I/O interrupt		
	NOP	00	1/1	no operation		

Notes to Table 3.

1. PSW CY, AC affected
2. PSW CY affected
3. PSW PS affected
4. Execution of JTF and JNTF instructions resets the Timer Flag (TF).
5. PSW RBS affected
6. PSW SP_0, SP_1, SP_2 affected
7. (A) = 1111 P2.3, P2.2, P2,1, P2,0.
8. (S1) has a different meaning for read and write operation, see serial I/O interface.
9. Only for software-transfer from the MAB8021.

* : 8, 9, A, B, C, D, E, F
● : 0, 2, 4, 6, 8, A, C, E
▲ : 1, 3, 5, 7, 9, B, D, F

Table 4 MAB84X1 family instruction set

first hexadecimal chracter of opcode ↓ / second hexadecimal chracter of opcode →

	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
0	NOP			ADD A, # data	JMP page 0	EN I	JNTF addr	DEC A	IN A,Pp 0	1	2		0	MOV A,Sn 1		
1	INC @Rr 0	1	JB0 addr	ADDC A, # data	CALL page 0	DIS I	JTF addr	INC A	0	1	2	INC Rr 3	4	5	6	7
2	XCH A, @Rr 0	1		MOV A, # data	JMP page 1	EN TCNTI	JNT0 addr	CLR A	0	1	2	XCH A,Rr 3	4	5	6	7
3	XCHD A, @Rr 0	1	JB1 addr		CALL page 1	DIS TCNTI	JT0 addr	CPL A	OUTL Pp,A 0	1	2		MOV Sn,A 0	1	2	
4	ORL A, @Rr 0	1	MOV A, T	ORL A, # data	JMP page 2	STRT CNT	JNT1 addr	SWAP A	0	1	2	ORL A,Rr 3	4	5	6	7
5	ANL A, @Rr 0	1	JB2 addr	ANL A, # data	CALL page 2	STRT T	JT1 addr	DA A	0	1	2	ANL A,Rr 3	4	5	6	7
6	ADD A, @Rr 0	1	MOV T, A		JMP page 3	STOP TCNT		RRC A	0	1	2	ADD A,Rr 3	4	5	6	7
7	ADDC A, @Rr 0	1	JB3 addr		CALL page 3			RR A	0	1	2	ADDC A,Rr 3	4	5	6	7
8				RET	JMP page 4	EN SI			ORL Pp, # data 0	1	2					
9	OUTL PO, A		JB4 addr	RETR	CALL page 4	DIS SI	JNZ addr	CLR C	ANL Pp, # data 0	1	2		MOV Sn, # data 0	1	2	
A	MOV @Rr, A 0	1		MOVP A, @A	JMP page 5	SEL MB2		CPL C	0	1	2	MOV Rr, A 3	4	5	6	7
B	MOV @Rr, # data 0	1	JB5 addr	JMPP @A	CALL page 5	SEL MB3			0	1	2	MOV Rr, # data 3	4	5	6	7
C	DEC @Rr 0	1			JMP page 6	SEL RB0	JZ addr	MOV A, PSW	0	1	2	DEC Rr 3	4	5	6	7
D	XRL A, @Rr 0	1	JB6 addr	XRL A, # data	CALL page 6	SEL RB1		MOV PSW, A	0	1	2	XRL A, Rr 3	4	5	6	7
E	DJNZ @Rr, addr 0	1			JMP page 7	SEL MB0	JNC addr	RL A	0	1	2	DJNZ Rr, addr 3	4	5	6	7
F	MOV A, @Rr 0	1	JB7 addr		CALL page 7	SEL MB1	JC addr	RLC A	0	1	2	MOV A, Rr 3	4	5	6	7

Table 5 shows the additional MAB84X1 family instructions (including the five for serial I/O operation) that are not part of the MAB8048 instruction set.

Table 5 MAB84X1 family instructions not in the MAB8048 instruction set

serial I/O	register	control	conditional branch

MOV A, S_n
MOV S_n, A
MOV S_n, #data
EN SI
DIS SI
	DEC @Rr
	DJNZ @Rr, addr
		SEL MB2
		SEL MB3
			JNTF addr

Table 6 shows the MAB8048 instructions omitted from the MAB84X1 family instruction set.

Table 6 MAB8048 instructions not in the MAB84X1 family instruction set

data moves	flags	branch	control

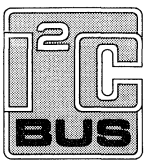
MOVX A, @R
MOVX @R, A
MOVP3 A, @A
MOVD A, P
MOVD P, A
ANLD P, A
ORLD P, A
	CLR F0
	CPL F0
	CLR F1
	CPL F1
		* JNI addr
		JF0 addr
		JF1 addr
		
		* replaced by
		JT0 JNT0.
			ENT0 CLK

ABSOLUTE MAXIMUM RATINGS

Limiting values in accordance with the Absolute Maximum System (IEC 134)

Stress above those listed under 'Absolute maximum ratings' may cause permanent damage to the device. This is a stress rating only and functional operation of the device, at these, or any other conditions above those indicated in the operational sections of this specification is not implied. Exposure to absolute maximum rating conditions for extended periods may affect device reliability.

parameter	symbol	min.	max.	unit
Input voltage on any pin with respect to ground (V_{SS})	V_I	−0,5	+7	V
Total power dissipation				
SOT-117, 28-lead DIL	P_{tot}	–	1	W
SOT-136, 28-lead DIL	P_{tot}	–	0,6	W
Input/output current for all pins except port 1	I_I, I_O	–	10	mA
Input/output current for port 1	I_I, I_O	–	20	mA
Storage temperature	T_{stg}	−65	+150	°C
Operating temperature				
standard	T_{amb}	0	+70	°C
extended	T_{amb}	−40	+85	°C
automotive	T_{amb}	−40	+110	°C



Purchase of Philips' I^2C components conveys a license under the Philips' I^2C patent to use the components in the I^2C-system provided the system conforms to the I^2C specifications defined by Philips.

DC CHARACTERISTICS

V_{CC} = 5 V (10%); V_{SS} = 0 V; all voltages with respect to V_{SS} unless otherwise specified

parameter	conditions	symbol	min.	max.	unit
Supply current					
MAB	0 to + 70 °C	I_{CC}	–	85	mA
MAF	–40 to + 85 °C	I_{CC}	–	100	mA
MAF84A	–40 to + 110 °C	I_{CC}	–	100	mA
Inputs					
Input voltage LOW (except P2.3 and SCLK)		V_{IL}	–0,5	0,8	V
Input voltage LOW (P2.3 and SCLK)		V_{IL1}	–0,5	1,5	V
Input voltage HIGH (all inputs except XTAL1, P2.3 and SCLK)		V_{IH}	2	V_{CC} + 0,5	V
Input voltage HIGH (XTAL1, P2,3 and SCLK)		V_{IH1}	3,0	V_{CC} + 0,5	V
Outputs					
Output voltage LOW (P0.0–P0.7)	I_{OL} = 1,6 mA	V_{OL}	–	0,45	V
Output voltage LOW (P1.0–P1.7 for 8401/11/21/41/61)	I_{OL12} = 10 mA	V_{OL12}	–	1,0	V
Output voltage LOW (P2.0–P2.2)	I_{OL2} = 1,6 mA	V_{OL2}	–	0,45	V
Output voltage LOW (P2.3, SCLK)	I_{OL3} = 5 mA	V_{OL3}	–	0,45	V
Output voltage LOW (non-standard pins of bond-out versions)	I_{OL4} = 0,4 mA	V_{OL4}	–	0,45	V
Output voltage HIGH (all outputs unless open drain)	I_{OH} = –50 μA	V_{OH}	2,4	–	V
Output leakage current	$V_{SS} < V_I < V_{CC}$	$\pm I_{OL}$	–	10	μA

AC CHARACTERISTICS (all versions except bond-out)

V_{CC} = 5 V ± 10%; V_{SS} = 0 V.

parameter	symbol		min.	max.	unit
Frequency	f_{XTAL}	MAB/MAF84X1	1	6	MHz
		MAF84AX1	1	5	MHz
Cycle time	t_{CY}	MAB/MAF84X1	5	30	μs
		MAF84AX1	6	30	μs

AC CHARACTERISTICS (bond-out versions)

V_{CC} = 5 V ± 10%; V_{SS} = 0 V.

parameter	symbol	min.	max.	unit
f_{CL} = 6 MHz				
Control pulse duration $\overline{PSEN}$ (9CP)	t_{CC}	1,5	9	μs
Address to $\overline{PSEN}$ L set-up (1CP)	t_{AS}	167	–	ns
Data to $\overline{PSEN}$ H set-up (1CP + 120 ns)	t_{DS}	600	–	ns
Data hold time	t_{DR}	0	–	ns
Address to data-in (10CP–t_{DS})	t_{AD}	–	1,07	μs
Time from $\overline{PSEN}$ L to C1 (3CP)	t_{PC}	500	–	ns
Time from $\overline{INTA}$ L to $\overline{PSEN}$ (3CP)	t_{IP0}	500	–	ns
Time from $\overline{INTA}$ H to $\overline{PSEN}$ (6CP)	t_{IP1}	1	–	μs
$\overline{HALT}$ set-up to $\overline{PSEN}$ (15CP)	t_{HS}	2,5	–	μs
$\overline{HALT}$ hold time from $\overline{PSEN}$ (3CP)	t_{HH}	500	–	ns

Note: CP = clock pulse.

T1 ZERO-CROSS CHARACTERISTICS

T_{amb} = 0 to + 70 °C; V_{CC} = 5 V ± 10%, V_{SS} = 0 V; C_L = 80 pF

parameter	conditions	symbol	min.	max.	unit
Zero-cross detection input (T1) peak-topeak	AC coupled, C = 1,0 μF	$V_{ZX(p-p)}$	1	3	V
Zero-cross accuracy	50 Hz sine wave	A_{ZX}	–	± 135	mV
Zero-cross detection input frequency (T1)		F_{ZX}	0,05	1	kHz

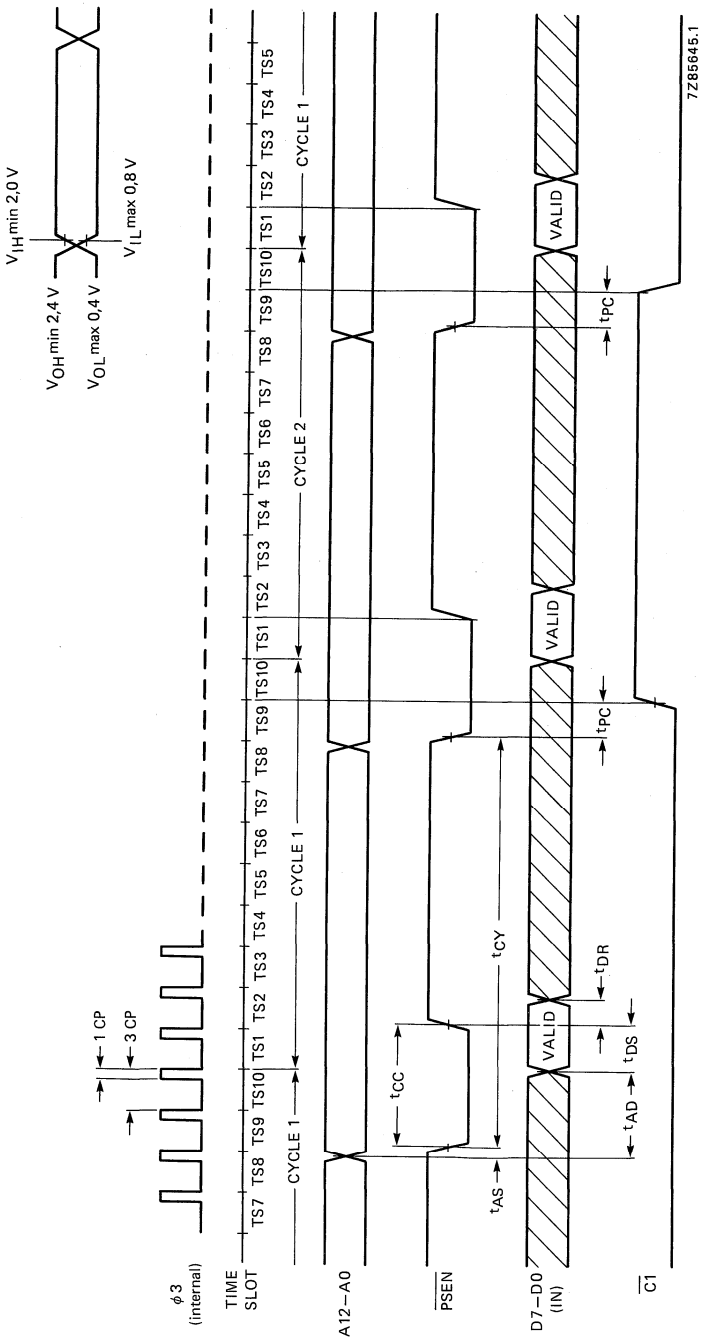


Fig. 15 Memory access timing MAB8401B/WP and I/O voltage parameters.

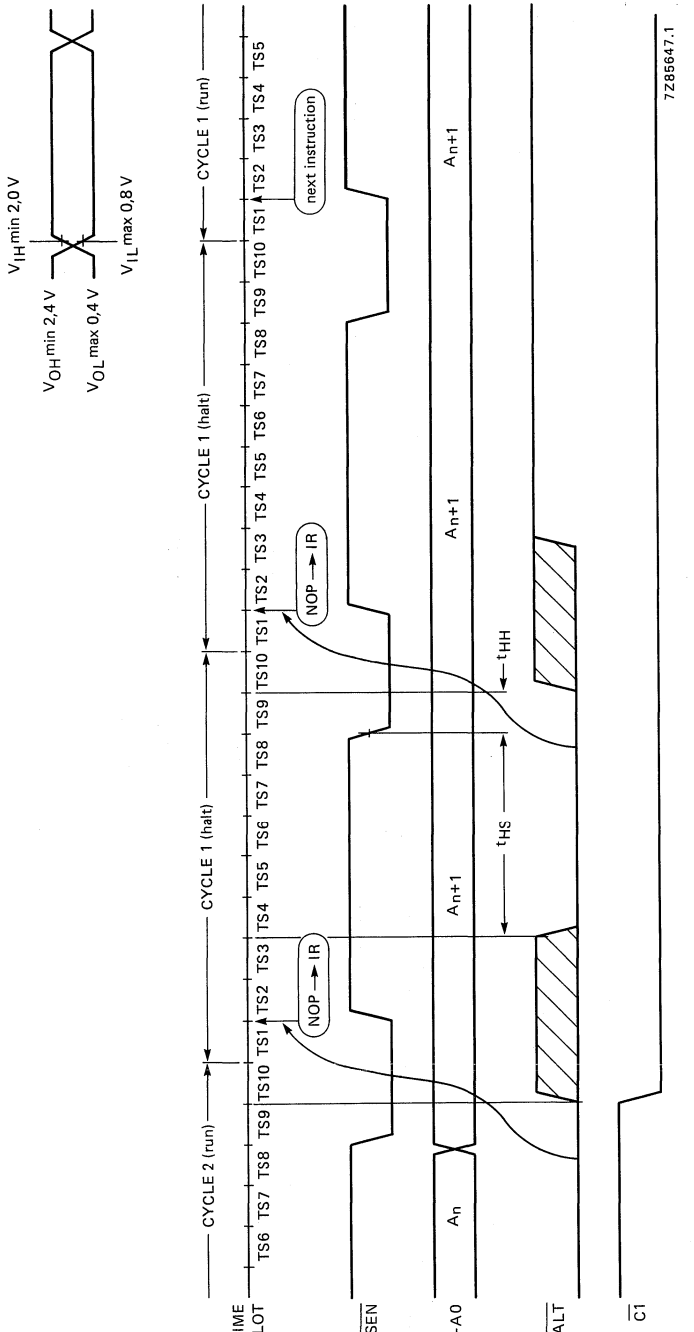


Fig. 16 $\overline{HALT}$ timing MAB8401WP and I/O voltage parameters.

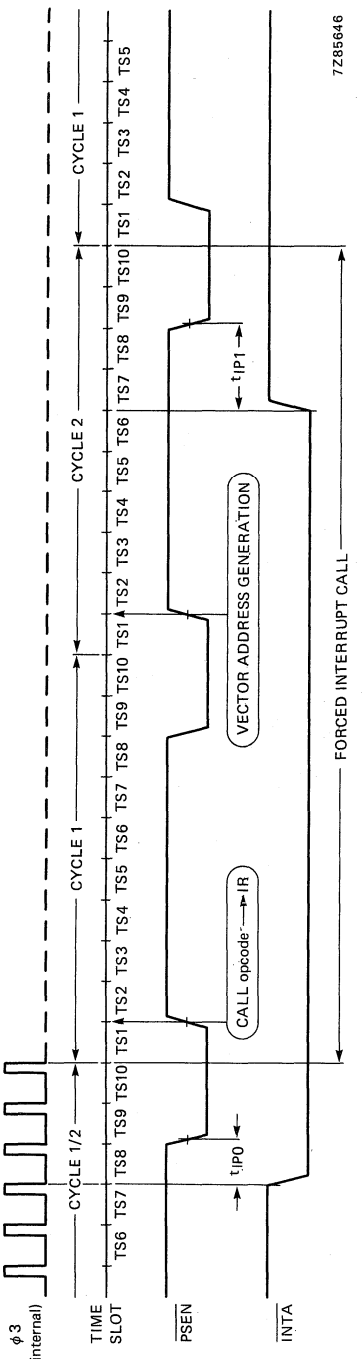


Fig. 17 $\overline{\text{INTA}}$ timing MAB8401WP.

SINGLE-CHIP 8-BIT MICROCONTROLLER

DESCRIPTION

The MAB8422/8442 is a high-performance microcontroller incorporating dedicated hardware, memory capacity and I/O lines. This dedication means a microcontroller can be economically installed in high-volume products where its main function is control.

The MAB8422/8442 is a 20 pin, single-chip 8-bit microcontroller that has been developed from the 28 pin MAB8421/8441 microcontrollers. The versions are:

- MAB8422 - 2K x 8 ROM/64 bytes RAM
- MAB8442 - 4K x 8 ROM/128 bytes RAM

Each version has 15 I/O port lines comprising one 8-bit parallel port (P0), one 2-bit parallel port (P1.0 and P1.1 that are shared with the serial I/O lines SDA and SCL), one 3-bit parallel port (P2.0 - P2.2) and two input lines ($\overline{INT}$/T0 and T1).

The serial I/O interface is I^2C compatible and therefore the MAB8422/8442 can operate as a slave or a master in single and multi-master systems. Conversion from parallel to serial data when transmitting, and vice versa when receiving, is done mainly in software. There is a minimum of hardware for the serial I/O implemented. This hardware is controlled by the status of the SDA and SCL lines and can be read or written under software control. Standard software for I^2C-bus control is available upon request. For detailed information see the 84XX family specification.

Features

- 8-bit: CPU, ROM, RAM and I/O
- 20 pin package
- MAB8422: 2K x 8 ROM/64 bytes RAM
- MAB8442: 4K x 8 ROM/128 bytes RAM
- 13 quasi-bidirectional I/O port lines
- Two testable inputs T1 and $\overline{INT}$/T0
- High current output on P0 (I_{OL} = 10 mA at V_{OL} = 1 V)
- One interrupt line combined with the testable input line $\overline{INT}$/T0
- Single-level interrupts: external, timer/event counter, serial I/O
- I^2C-compatible serial I/O that can be used in single or multi-master systems (serial I/O data and clock via P1.0 and P1.1 port lines, respectively)
- 8-bit programmable timer/event counter
- Internal oscillator, generated with inductor, crystal, ceramic resonator or external source
- Over 80 instructions (based on MAB8048)
- All instructions 1 or 2 cycles, cycle time dependent on oscillator frequency
- Single power supply
- Operating temperature ranges: 0 to +70 °C (MAB84X2)
  −40 to +85 °C (MAF84X2)
  −40 to +110 °C (MAF84AX2)

PACKAGE OUTLINES

MAB/MAF84X2, MAF84AX2: 20-lead DIL; plastic (SOT146).

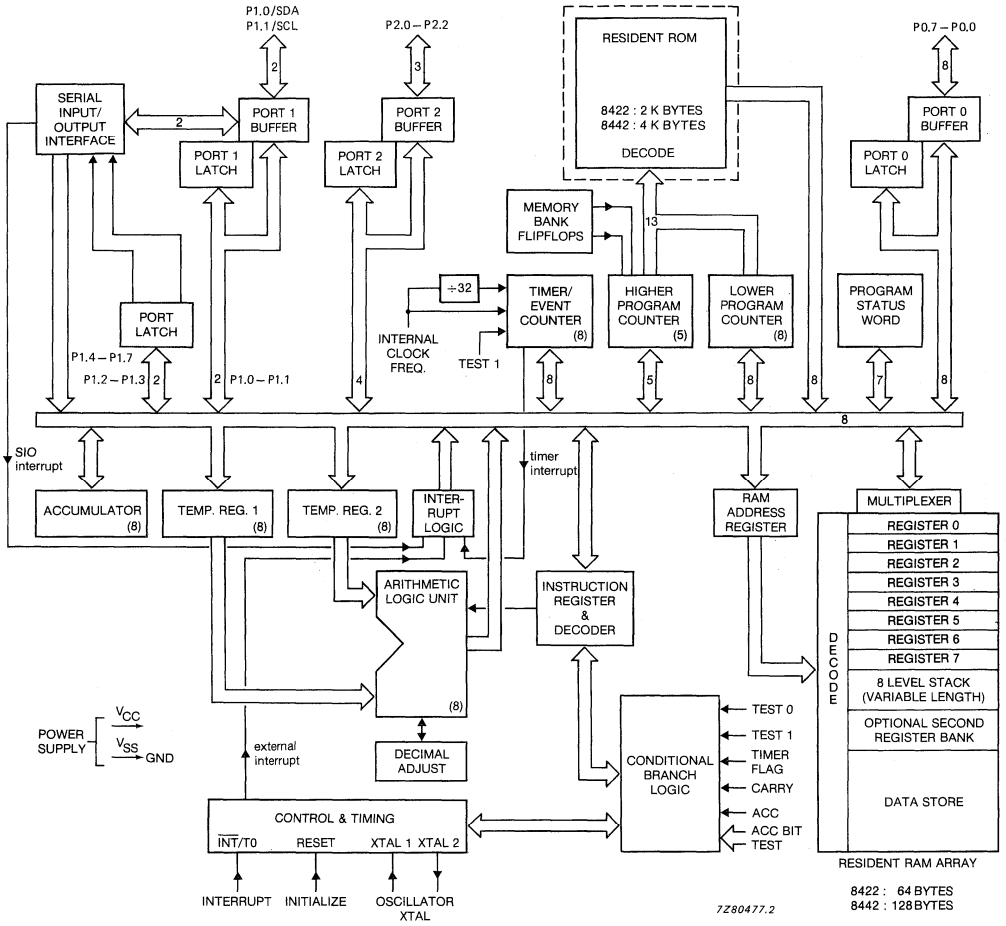


Fig. 1 Block diagram of the MAB8422/8442.

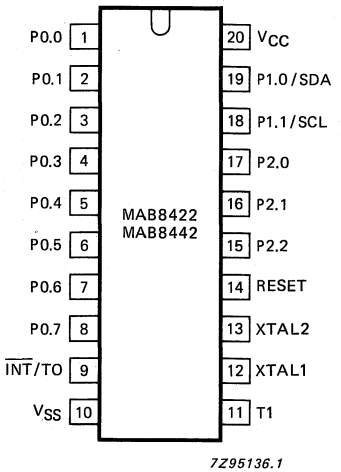


Fig. 2 Pinning diagram.

PINNING

Designation	Pin number	Function
P0.0 - P0.7	1-8	8-bit quasi-bidirectional I/O port (Port 0 high current output).
$\overline{INT}$/T0	9	External interrupt input (sensitive to a negative going edge) and/or input, testable using the JT0 or JNT0 instructions.
V_{SS}	10	Ground.
T1	11	Input pin, testable using the JT1 or JNT1 instructions. It can be designated as event counter input using the STRT CNT instruction. It can also be used to detect zero cross-over of slowly moving a.c. inputs.
XTAL1	12	Connection to timing component that determines the frequency of the internal oscillator. It is also the input for an external clock source.
XTAL2	13	Connection to the other side of the timing component.
RESET	14	Input to initialize the processor (active HIGH).
P2.0-P2.2	17-15	Quasi-bidirectional port.
P1.1/SCL	18	Quasi-bidirectional port in parallel port mode. Serial clock in serial I/O mode.
P1.0/SDA	19	Quasi-bidirectional port in parallel port mode. Serial data I/O in serial I/O mode.
V_{CC}	20	Power supply.

FUNCTIONAL DESCRIPTION

Program and data memory

The non-volatile program memory (ROM), as shown in Fig. 3, is arranged in two banks of 2K bytes, that are selected by SEL MB instructions, and each bank is further divided into 256-byte pages. Only the unconditional jump instructions (JMP and CALL) can be used to cross page boundaries. Memory bank boundaries can also be crossed using these instructions provided that the appropriate memory bank has been selected.

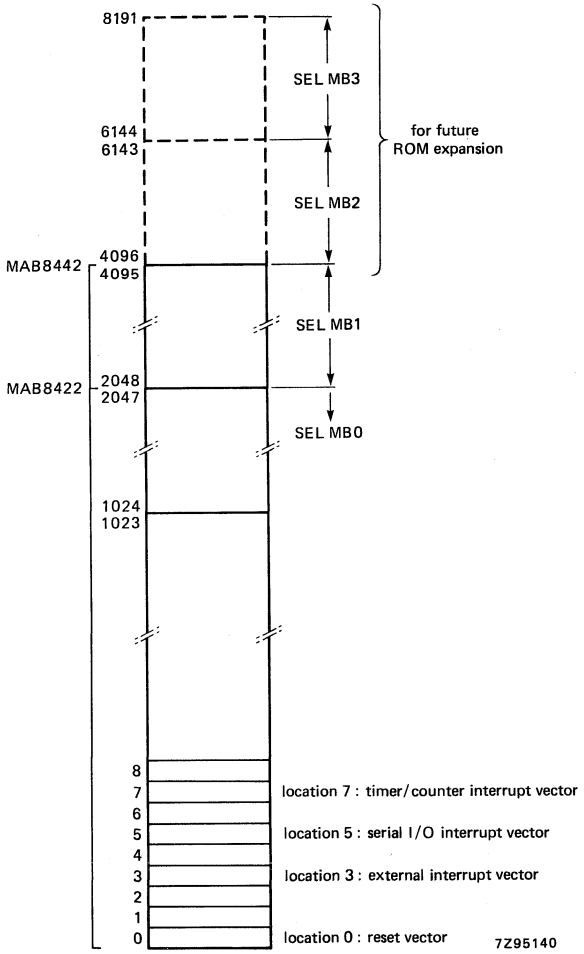


Fig. 3 Program memory map.

In the volatile data memory (RAM), all locations are indirectly addressable using RAM pointer registers and up to 16 designated locations can be addressed directly. The memory also includes an 8-level program counter stack addressed by a 3-bit stack pointer and two register banks, each with 8 registers. The data memory is shown in Fig. 4.

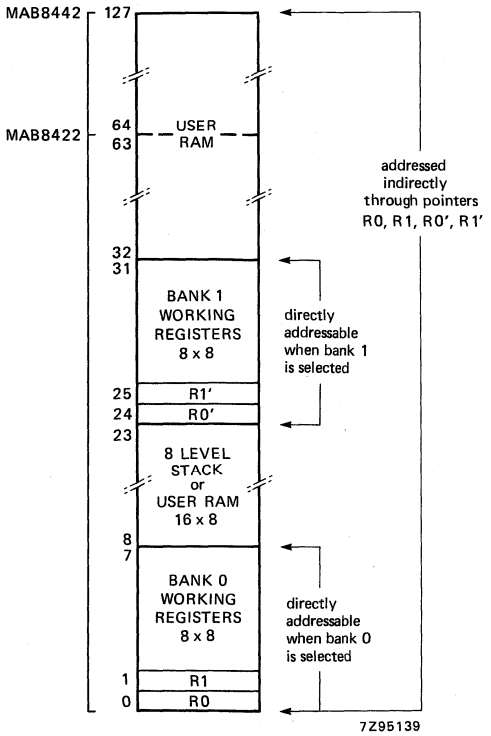


Fig. 4 Data memory map.

Instruction set

The instruction set consists of over 80 one and two byte instructions. It is identical to the MAB84X1 instruction set except that the instructions MOV Sn, A, MOV A, Sn and MOV Sn, #data are not used. Program code efficiency is high because all RAM locations on a 256 byte page require only a single byte address.

On-chip peripheral functions

In addition to the CPU and memories, an interrupt system, I/O facilities, and an 8-bit timer/event counter are integrated on-chip to assist the CPU in repetitious, complicated or time-critical tasks. The I/O facilities include the I/O pins, parallel ports and a serial I/O port sharing the two pins of the parallel port P1.

FUNCTIONAL DESCRIPTION (continued)

I/O facilities (see Fig. 5)

The MAB8422/8442 has 13 I/O lines and 2 testable inputs arranged as:

- An 8 line parallel port P0.0-P0.7, high current outputs with three optional output configurations:
  - A push-pull output with pull-up (Fig. 5 (a))
  - Open drain with pull-up (Fig. 5 (b))
  - Open drain without pull-up (Fig. 5 (c))
- A 2 line parallel/serial port P1.0/SDA and P1.1/SCL, open-drain without pull-up, as output configuration. Schmitt-trigger input. After RESET, P1.0/SDA and P1.1/SCL will be in the parallel port mode. To stay in this mode the internal port latches, P1.4 and P1.3 must be kept in the logic '1' state. Inputs P1.2-P1.7 are not valid in the parallel port mode. After a RESET, the microcontroller remains in the parallel port mode until the serial I/O mode is enabled.
- A 3 line parallel port P2.0-P2.2 with the same output configurations as P0.0-P0.7 but without high-current output;
- An external interrupt and test input INT/T0, which when used as a test input can be tested by the conditional jump instructions JT0 and JNT0;
- A test input T1, tested by the conditional jump instructions JT1 and JNT1. T1 can also be used as an input to the timer/event counter or to detect zero cross-over of slowly moving AC signals. This test or input line can be ordered with port option 5(b) or 5(c).

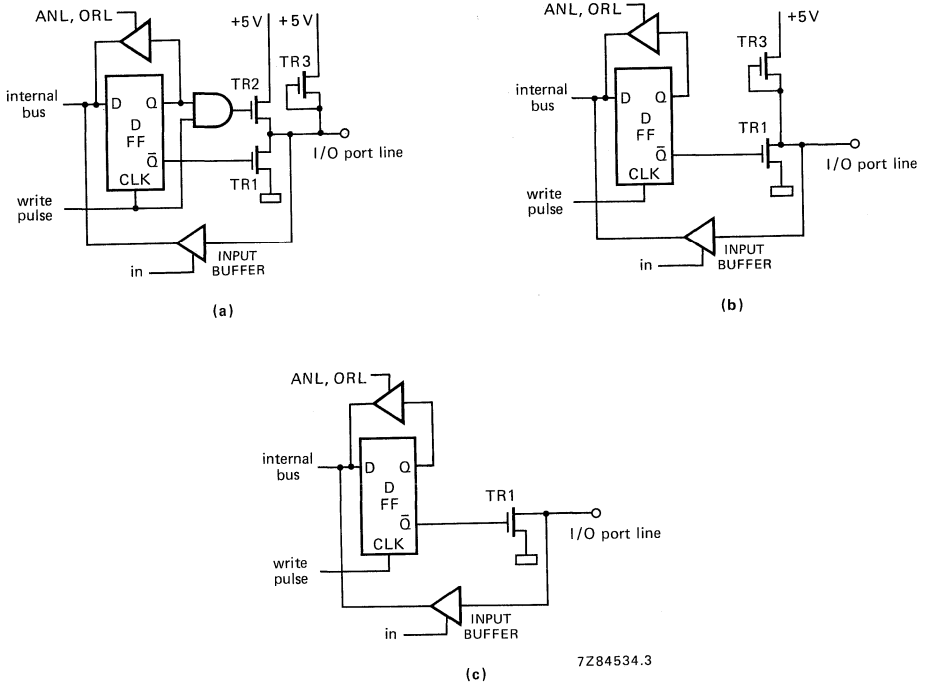


Fig. 5 Port option configurations.

Test input T1

The T1 input line can be used as:

- a test input for branch instructions,
- an input for zero voltage cross-over detection,
- an external input to the event counter.

An internal pull-up transistor is provided as a ROM mask option. This is useful when the input is from a switch or standard TTL output.

When T1 is used as a test input, the JT1 or JNT1 instructions test for a HIGH or a LOW respectively.

Zero cross-over detection

When used for zero-cross detection purposes, the T1 input must be coupled through a capacitor of typical value 1 μF and operation carried out using the T1 input without the pull-up transistor. The maximum input voltage amplitude is 3 V (peak-to-peak), with a maximum operational frequency of 1 kHz. The T1 input has an on-chip DC offset circuit which self-biases the input near to its exact switching level of 1 V. As a consequence a small change will cause a digital transition to occur. The switching level of the T1 input circuit is within the bias voltage of ±135 mV. Upon each positive cycle on the pin, the event counter is incremented and an overflow will set the timer flag TF. Zero cross-over detection used in conjunction with the timer/event counter interrupt, is useful in thyristor control of power equipment. Figure 6 illustrates, (a) the input waveform, (b) the input diagram and (c) the on-chip self-stabilized bias.

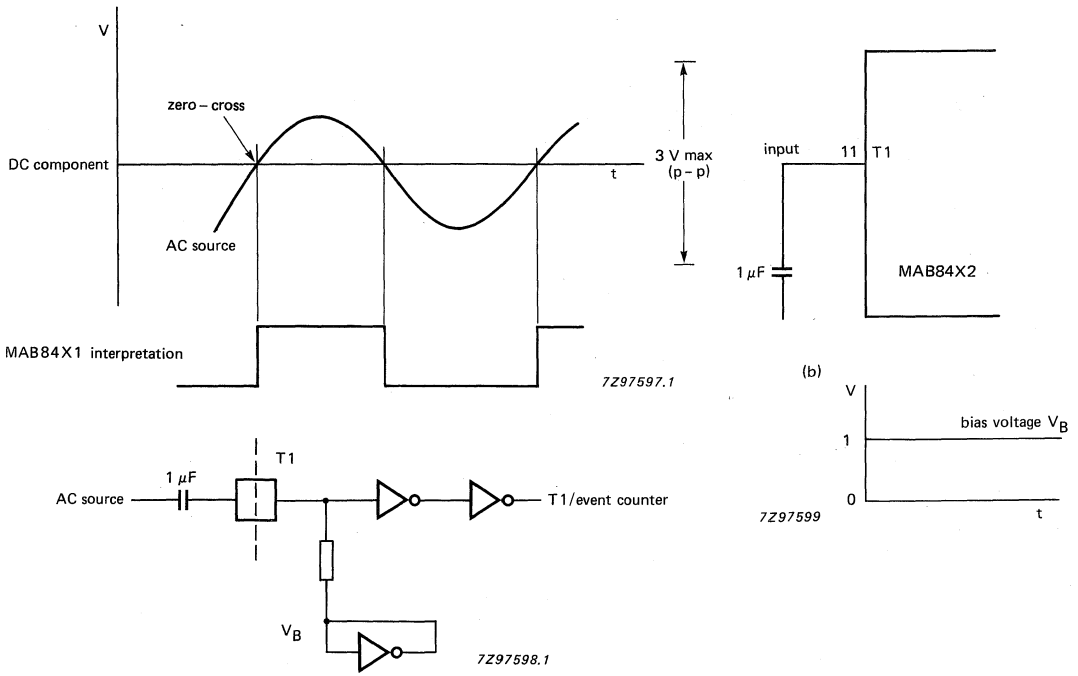


Fig. 6 Zero-cross voltage detection circuitry.

FUNCTIONAL DESCRIPTION (continued)

Timer/event counter

An 8-bit binary up-counter is provided. This can count external events, modulo-32 machine cycles, or machine cycles directly. When used as a timer, the input to the counter is either the overflow or input of a 5-bit prescaler. When used as an event counter, LOW to HIGH transitions on T1 (pin 13) are counted. The maximum rate at which the counter may be incremented is once every machine cycle (200 kHz for a 5 μs machine cycle). Fig. 7 illustrates the timer/event counter.

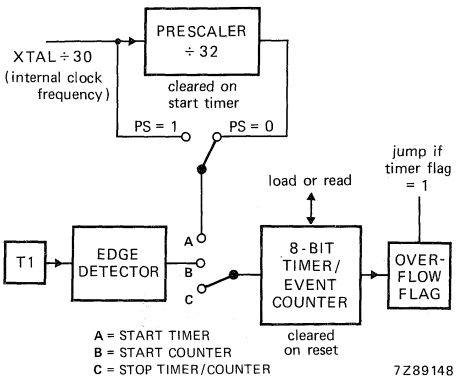


Fig. 7 Timer/event counter.

Interrupt system

External events and real-time on-chip peripherals require servicing by the CPU asynchronous to the execution of any particular section of code. To tie the asynchronous activities of these functions to normal program execution a multiple-source, single-level nested interrupt system is provided.

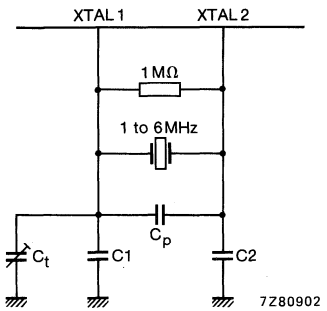
The MAB8422/8442 handles interrupts from three sources as follows:

- $\overline{\text{INT}}$/T0; externally via pin 9
- $\overline{\text{SIOINT}}$; from the internal serial I/O port
- TCNT interrupt; from the internal timer/event counter.

Each interrupt vectors to a separate location in the program memory for its service program. Each source can be individually enabled or disabled. When more than one interrupt occurs simultaneously, their priority will be: (1) external, (2) serial I/O and, (3) timer/event counter. An additional external interrupt can be created using the timer/event counter interrupt.

OSCILLATOR CIRCUITRY (see Fig. 8)

The clock frequency is determined by using the internal oscillator or by connecting an external clock to XTAL1. Where the internal oscillator is used the frequency is set by a crystal, a ceramic resonator or an inductor (each with associated capacitors) between XTAL1 and XTAL2. A machine cycle consists of 10 states, each state being 3 oscillator periods. The MAB8422/8442 has a dynamic logic, and therefore, for adequate refreshing the oscillator frequency must be at least 1 MHz.



1. Crystal - AT-cut
2. Ceramic resonator
   $C_1 = C_2 = 27$ pF
   C_t is optional
   $C_p < 6{,}75$ pF (parasitic capacitance)

Fig. 8(a) Quartz crystal or ceramic resonator mode.

If the frequency has to be trimmed, then a trimmer capacitor C_t should be connected in parallel with the fixed capacitor C_1.

Table 1 shows the LC values for timing generation with the LC oscillator.

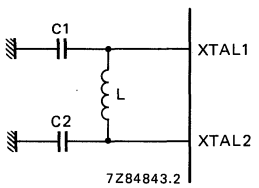
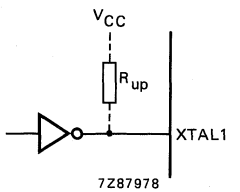


Fig. 8(b) LC pi-network.

Table 1 LC oscillator timing

frequency	C1 = C2	L
3,0 MHz	33 pF	100 μH
4,0 MHz	33 pF	56 μH
4,4 MHz	33 pF	47 μH
5,0 MHz	33 pF	33 μH
6,0 MHz	33 pF	22 μH



Drive XTAL1
Leave XTAL2 open

Driver may be high-speed CMOS or any TTL
$t_r, t_f < 10$ ns

Fig. 9 External drive.

RESET

A positive-going signal on the RESET input:

- sets the program counter to zero,
- selects location 0 of memory bank 0, and register bank 0,
- sets the stack pointer to zero (000); pointing to RAM address 8,
- disables the interrupts (external), timer and serial I/O),
- stops the timer/event counter, then sets it to zero,
- sets the timer prescaler to modulo-32,
- resets the timer flag,
- sets all ports to logic '1' (input mode),
- sets ports P1.0/SDA and P1.1/SCL to the parallel part mode and disables the serial I/O.

Automatic reset at power-up may be obtained by connecting the RESET pin to V_{CC} through a 1 μF capacitor C1, together with a diode to V_{SS} (cathode to RESET pin). This arrangement is satisfactory, if both the voltage (V_{CC}) rise time and the oscillator start-up time do not exceed either 1 or 10 ms respectively.

The power-on reset circuit is shown in Fig. 11; the input characteristics are shown in Fig. 12. At power-on the current drawn by RESET commences to charge the capacitor C1. The difference between this increasing capacitor voltage and V_{CC} is known as V_{RESET}. The charging circuit is designed to hold V_{RESET} above the lower threshold of a schmitt trigger arrangement, long enough to effect a complete reset. The minimum time required is the oscillator start-up time, plus two machine cycles.

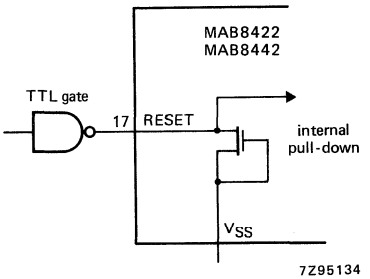


Fig. 10 External reset.

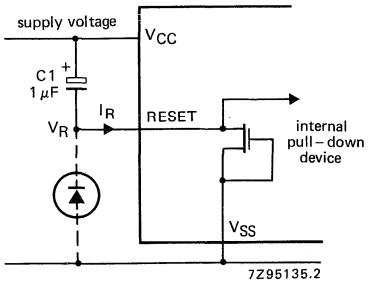


Fig. 11 Power-on reset circuitry.

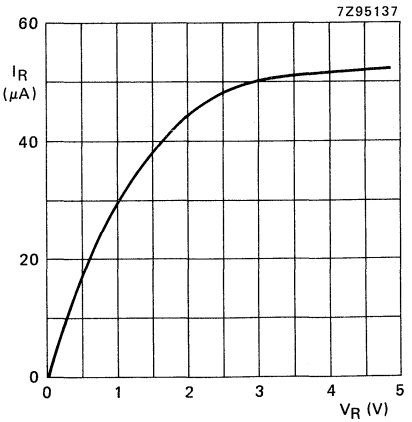


Fig. 12 Power-on reset input characteristics.

INSTRUCTION SET

The instruction set consists of over 80 one and two byte instructions and is based on the MAB8048 instruction set. New instructions include those for serial I/O operation and memory bank selection. Program code efficiency is high because all ROM locations on a 256 byte page require only a single byte address.

Table 2 gives the instruction set of the MAB84X2 family and Table 3 shows the instruction map. The following symbols and abbreviations are used.

symbol	description
A	the accumulator
AC	the auxiliary carry flag
addr	program memory address (11-bits)
Bb	bit designation (b = 0–7)
BS	the bank switch
C	carry flag
CLK	clock signal
CNT	event counter
D	nibble designation (4-bits)
DBF	program memory bank flip-flop
data	number of expression (8-bits)
F0, F1	flags 0 and 1
I	interrupt
$\overline{INT}$	external interrupt
P	'in-page' operation designation
Pp	port designation (p = 1, 2 or 4–7)
PSW	program status word
Rr	register designation (r = 0, 1 or 0–7)
SP	stack pointer
T	timer
TF	timer flag
T0, T1	test 0 an 1 inputs
#	immediate data prefix
@	indirect address prefix
S	current value of program counter
←	is replaced by
↔	is exchanged with

Table 2 Instruction set

	mnemonic	opcode (hex.)	bytes/ cycles	description	function		notes
ACCUMULATOR	ADD A, Rr	6*	1/1	Add register contents to A	(A)←(A) + (Rr)	r = 0–7	1
	ADD A, @Rr	60 61	1/1	Add RAM data, addressed by Rr, to A	(A)←(A) + ((R0)) (A)←(A) + ((R1))		1
	ADD A, #data	03 data	2/2	Add immediate data to A	(A)←(A) + data		1
	ADDC A, Rr	7*	1/1	Add carry and register contents to A	(A)←(A) + (Rr) + (C)	r = 0–7	1
	ADDC A, @Rr	70 71	1/1	Add carry and RAM data, addressed by Rr, to A	(A)←(A) + ((R0)) + (C) (A)←(A) + ((R1)) + (C)		1
	ADDC A, #data	13 data	2/2	Add carry and immediate data to A	(A)←(A) + data + (C)		1
	ANL A, Rr	5*	1/1	'AND' Rr with A	(A)←(A) AND (Rr)	r = 0–7	
	ANL A, @Rr	50 51	1/1	'AND' RAM data, addressed by Rr, with A	(A)←(A) AND ((R0)) (A)←(A) AND ((R1))		
	ANL A, #data	53 data	2/2	'AND' immediate data with A	(A)←(A) AND data		
	ORL A, Rr	4*	1/1	'OR' Rr with A	(A)←(A) OR (Rr)	r = 0–7	
	ORL A, @Rr	40 41	1/1	'OR' RAM data, addressed by Rr, with A	(A)←(A) OR ((R0)) (A)←(A) OR ((R1))		
	ORL A, #data	43 data	2/2	'OR' immediate data with A	(A)←(A) OR data		
	XRL A, Rr	D*	1/1	'XOR' Rr with A	(A)←(A) XOR (Rr)	r = 0–7	
	XRL A, @Rr	D0 D1	1/1	'XOR' RAM, addressed by Rr, with A	(A)←(A) XOR ((R0)) (A)←(A) XOR ((R1))		
	XRL A, #data	D3 data	2/2	'XOR' immediate data with A	(A)←(A) XOR data		
	INC A	17	1/1	increment A by 1	(A)←(A) + 1		
	DEC A	07	1/1	decrement A by 1	(A)←(A) −1		
	CLR A	27	1/1	clear A to zero	(A)←0		
	CPL A	37	1/1	one's complement A	(A)←NOT(A)		
	RL A	E7	1/1	rotate A left	$(A_{n+1})\leftarrow(A_n)$ $(A_0)\leftarrow(A_7)$	n = 0–6	

	mnemonic	opcode (hex.)	bytes/ cycles	description	function		notes
ACCUMULATOR (cont.)	RLC A	F7	1/1	rotate A left through carry	$(A_{n+1}) \leftarrow A_n$ $(A_0) \leftarrow (C)$, $(C) \leftarrow (A_7)$	n = 0–6	2
	RR A	77	1/1	rotate A right	$(A_n) \leftarrow (A_{n+1})$ $(A_7) \leftarrow (A_0)$	n = 0–6	
	RRC A	67	1/1	rotate A right through carry	$(A_n) \leftarrow (A_{n+1})$ $(A_7) \leftarrow (C)$, $(C) \leftarrow (A_0)$	n = 0–6	2
	DA A	57	1/1	decimal adjust A			2
	SWAP A	47	1/1	swap nibbles of A	$(A_{4-7}) \leftrightarrow (A_{0-3})$		
DATA MOVES	MOV A, Rr	F*	1/1	move register contents to A	(A)←(Rr)	r = 0–7	
	MOV A, @Rr	F0 F1	1/1	move RAM data, addressed by Rr, to A	(A)←((R0)) (A)←((R1))		
	MOV A, #data	23 data	2/2	move immediate data to A	(A)←data		
	MOV Rr, A	A*	1/1	move accumulator contents to register	(Rr)←(A)	r = 0–7	
	MOV @Rr, A	A0 A1	1/1	move accumulator contents to RAM location addressed by Rr	((R0))←(A) ((R1))←(A)		
	MOV Rr, #data	B* data	2/2	move immediate data to Rr	(Rr)←data		
	MOV @Rr, #data	B0 data B1 data	2/2	move immediate data to RAM location addressed by Rr	((R0))←data ((R1))←data		
	XCH A, Rr	2*	1/1	exchange accumulator contents with Rr	(A)↔(Rr)	r = 0–7	
	XCH A, @Rr	20 21	1/1	exchange accumulator contents with RAM data addressed by Rr	(A)↔((R0)) (A)↔((R1))		
	XCHD A, @Rr	30 31	1/1	exchange lower nibbles of A and RAM data addressed by Rr	$(A_{0-3}) \leftrightarrow ((R0_{0-3}))$ $(A_{0-3}) \leftrightarrow ((R1_{0-3}))$		
	MOV A, PSW	C7	1/1	move PSW contents to accumulator	(A)←(PSW)		
	MOV PSW, A	D7	1/1	move accumulator bit 3 to PSW_3	$(PSW_3) \leftarrow (A_3)$		3
	MOVP A, @A	A3	1/2	move indirectly addressed data in current page to A	$(PC_{0-7}) \leftarrow (A)$, $(A) \leftarrow ((PC))$		

Table 2 Instruction set (continued)

FLAGS	CLR C	97	1/1	clear carry bit	(C)←0		2
	CPL C	A7	1/1	complement carry bit	(C)←NOT(C)		2
REGISTER	INC Rr	1*	1/1	increment register by 1	(Rr)←(Rr) + 1	r = 0–7	
	INC @Rr	10 11	1/1	increment RAM data, addressed by Rr, by 1	((R0))←((R0)) + 1 ((R1))←((R1)) + 1		
	DEC Rr	C*	1/1	decrement register by 1	(Rr)←(Rr) – 1	r = 0–7	
	DEC @Rr	C0 C1	1/1	decrement RAM data, addressed by Rr, by 1	((R0))←((R0)) – 1 ((R1))←((R1)) – 1		
BRANCH	JMP addr	• 4 address	2/2	unconditional jump within a 2 K bank	(PC_{8-10})←$addr_{8-10}$ (PC_{0-7})←$addr_{0-7}$ (PC_{11-12})←MBFF 0–1		
	JMPP @A	B3	1/2	indirect jump within a page	(PC_{0-7})←((A))		
	DJNZ Rr, addr	E* address	2/2	decrement Rr by 1 and jump if not zero to addr	(Rr)←(Rr) – 1 if (Rr) not zero (PC_{0-7})←addr	r = 0–7	
	DJNZ @Rr, addr	E0 address	2/2	decrement RAM data, addressed by Rr by 1 and jump if not zero to addr	((R0))←((R0)) – 1 if ((R0)) not zero (PC_{0-7})←addr		
		E1 address			((R1))←((R1)) – 1 if ((R1)) not zero (PC_{0-7})←addr		
	JBb addr	▲ 2 address	2/2	jump to addr if Acc. bit b = 1	if b = 1 : (PC_{0-7})←addr	b = 0–7	
	JC addr	F6 address	2/2	jump to addr if C = 1	if C = 1 : (PC_{0-7})←addr		
	JNC addr	E6 address	2/2	jump to addr if C = 0	if C = 0 : (PC_{0-7})←addr		
	JZ addr	C6 address	2/2	jump to addr if A = 0	if A = 0 : (PC_{0-7})←addr		
	JNZ addr	96 address	2/2	jump to addr if A is NOT zero	if A≠0 : (PC_{0-7})←addr		
	JT0 addr	36 address	2/2	jump to addr if T0 = 1	if T0 = 1: (PC_{0-7})←addr		
	JNT0 addr	26 address	2/2	jump to addr if T0 = 0	if T0 = 0: (PC_{0-7})←addr		
	JT1 addr	56 address	2/2	jump to addr if T1 = 1	if T1 = 1: (PC_{0-7})←addr		
	JNT1 addr	46 address	2/2	jump to addr if T1 = 0	if T1 = 0: (PC_{0-7})←addr		
	JTF addr	16 address	2/2	jump to addr if Timer Flag = 1	if TF = 1: (PC_{0-7})←addr		4
	JNTF addr	06 address	2/2	jump to addr if Timer Flag = 0	if TF = 0: (PC_{0-7})←addr		

	mnemonic	opcode (hex.)	bytes/ cycles	description	function	notes
TIMER/EVENT COUNTER	MOV A, T	42	1/1	move timer/event counter contents to accumulator	(A)←(T)	
	MOV T, A	62	1/1	move accumulator contents to timer/event counter	(T)←(A)	
	STRT CNT	45	1/1	start event counter		
	STRT T	55	1/1	start timer		
	STOP TCNT	65	1/1	stop timer/event counter		
	EN TCNTI	25	1/1	enable timer/event counter interrupt		
	DIS TCNTI	35	1/1	disable timer/event counter interrupt		
CONTROL	EN I	05	1/1	enable external interrupt		
	DIS I	15	1/1	disable external interrupt		
	SEL RB0	C5	1/1	select register bank 0	(RBS)←0	5
	SEL RB1	D5	1/1	select register bank 1	(RBS)←1	5
	SEL MB0	E5	1/1	select program memory bank 0	(MBFF0)←0, (MBFF1)←0	
	SEL MB1	F5	1/1	select program memory bank 1	(MBFF0)←1, (MBFF1)←0	
	SEL MB2	A5	1/1	select program memory bank 2	(MBFF0)←0, (MBFF1)←1	
	SEL MB3	B5	1/1	select program memory bank 3	(MBFF0)←1, (MBFF1)←1	
SUBROUTINE	CALL addr	▲ 4 address	2/2	jump to subroutine	((SP))←(PC), ($PSW_{4, 6, 7}$) (SP)←(SP) + 1 (PC_{8-10})←$addr_{8-10}$ (PC_{0-7})←$addr_{0-7}$ (PC_{11-12})←MBFF 0–1	6
	RET	83	1/2	return from subroutine	(SP)←(SP) – 1 (PC)←((SP))	6
	RETR	93	1/2	return from interrupt and restore bits 4, 6, 7 of PSW	(SP)←(SP) – 1 ($PSW_{4, 6, 7}$) + (PC)←((SP))	6

Table 2 Instruction set (continued)

PARALLEL INPUT/OUTPUT	IN A, Pp	08 09 0A	1/2	input port p data to accumulator	(A)←(P0) (A)←(P1) (A)←(P2)	7
	OUTL Pp, A	38 39 3A	1/2	output accumulator data to port p	(P0)←(A) (P1)←(A) (P2)←(A)	
	ANL Pp, #data	98 data 99 data 9A data	2/2	AND port p data with immediate data	(P0)←(P0) AND data (P1)←(P1) AND data (P2)←(P2) AND data	
	ORL Pp, #data	88 data 89 data 8A data	2/2	OR port p data with immediate data	(P0)←(P0) OR data (P1)←(P1) OR data (P2)←(P2) OR data	
	OUTL PO,A	90	1/2	Output accumulator data to port ϕ	(P0)←(A)	8
SERIAL INPUT/OUTPUT	EN SI	85	1/1	enable serial I/O interrupt		
	DIS SI	95	1/1	disable serial I/O interrupt		
	NOP	00	1/1	no operation		

Notes to Table 2.

1. PSW CY, AC affected
2. PSW CY affected
3. PSW PS affected
4. Execution of JTF and JNTF instructions resets the Timer Flag (TF).
5. PSW RBS affected
6. PSW SP_0, SP_1, SP_2 affected
7. (A) = 1111 P2.3, P2.2, P2.1, P2.0.
8. Only for software-transfer from the MAB8021.

* : 8, 9, A, B, C, D, E, F
● : 0, 2, 4, 6, 8, A, C, E
▲ : 1, 3, 5, 7, 9, B, D, F

TABLE 3 MAB8422/8442 INSTRUCTION MAP

first hexadecimal character of opcode

second hexadecimal character of opcode

	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
0	NOP			ADD A, # data	JMP page 0	EN I	JNTF addr	DEC A	IN A,Pp 0	1	2					
1	INC ∂Rr 0	1	JB0 addr	ADDC A, # data	CALL page 0	DIS I	JTF addr	INC A	0	1	2	INC Rr 3	4	5	6	7
2	XCH A, ∂Rr 0	1		MOV A, # data	JMP page 1	EN TCNTI	JNT0 addr	CLR A	0	1	2	XCH A,Rr 3	4	5	6	7
3	XCHD A,∂Rr 0	1	JB1 addr		CALL page 1	DIS TCNTI	JT0 addr	CPL A	OUTL Pp,A 0	1	2					
4	ORL A, ∂Rr 0	1	MOV A, T	ORL A, # data	JMP page 2	STRT CNT	JNT1 addr	SWAP A	0	1	2	ORL A,Rr 3	4	5	6	7
5	ANL A, ∂Rr 0	1	JB2 addr	ANL A, # data	CALL page 2	STRT T	JT1 addr	DA A	0	1	2	ANL A,Rr 3	4	5	6	7
6	ADD A, ∂Rr 0	1	MOV T, A		JMP page 3	STOP TCNT		RRC A	0	1	2	ADD A,Rr 3	4	5	6	7
7	ADDC A, ∂Rr 0	1	JB3 addr		CALL page 3			RR A	0	1	2	ADDC A,Rr 3	4	5	6	7
8				RET	JMP page 4	EN SI			ORL Pp,# data 0	1	2					
9	OUTL PO, A		JB4 addr	RETR	CALL page 4	DIS SI	JNZ addr	CLR C	ANL Pp,# data 0	1	2					
A	MOV ∂Rr, A 0	1		MOVP A, ∂A	JMP page 5	SEL MB2		CPL C	0	1	2	MOV Rr, A 3	4	5	6	7
B	MOV ∂Rr, # data 0	1	JB5 addr	JMPP ∂A	CALL page 5	SEL MB3			0	1	2	MOV Rr, # data 3	4	5	6	7
C	DEC ∂Rr 0	1			JMP page 6	SEL RB0	JZ addr	MOV A, PSW	0	1	2	DEC Rr 33	4	5	6	7
D	XRL A, ∂Rr 0	1	JB6 addr	XRL A, # data	CALL page 6	SEL RB1		MOV PSW, A	0	1	2	XRL A,Rr 3	4	5	6	7
E	DJNZ ∂Rr, addr 0	1			JMP page 7	SEL MB0	JNC addr	RL A	0	1	2	DJNZ Rr, addr 3	4	5	6	7
F	MOV A, ∂Rr 0	1	JB7 addr		CALL page 7	SEL MB1	JC addr	RLC A	0	1	2	MOV A, Rr 3	4	5	6	7

ABSOLUTE MAXIMUM RATINGS

Limiting values in accordance with the Absolute Maximum System (IEC 134)

parameter	symbol	min.	max.	unit
Input voltage on any pin with respect to ground (V_{SS})	V_I	−0,5	+7	V
Total power dissipation	P_{tot}	–	1	W
Input/output current for all pins except port 0	I_I, I_O	–	10	mA
Input/output current for port 0	I_I, I_O	–	20	mA
Storage temperature	T_{stg}	−65	+150	°C
Operating temperature				
standard	T_{amb}	0	+70	°C
extended	T_{amb}	−40	+85	°C
automotive	T_{amb}	−40	+110	°C

DC CHARACTERISTICS

V_{CC} = 5 V (± 10%); V_{SS} = 0 V

parameter	conditions	symbol	min.	max.	unit
Supply current					
MAB8422/42	at 0 °C	I_{CC}	–	70	mA
MAF8422/42	at −40 °C	I_{CC}	–	90	mA
MAF84A22/42	at −40 °C	I_{CC}	–	90	mA
Inputs					
Input voltage LOW (except P1.0/SDA and P1.1/SCL)		V_{IL}	−0,5	0,8	V
Input voltage LOW (P1.0/SDA and P1.1/SCL)		V_{IL1}	−0,5	1,5	V
Input voltage HIGH all inputs except XTAL1, P1.0/SDA and P1.1/SCL		V_{IH}	2,0	V_{CC} + 0,5	V
Input voltage HIGH to XTAL1, P1.0/SDA and P1.1/SCL		V_{IH1}	3,0	V_{CC} + 0,5	V
Outputs					
Output voltage LOW (P0 only)	I_{OL} = 10 mA	V_{OL}		1,0	V
Output voltage LOW (P1.0/SDA and P1.1/SCL)	I_{OL} = 5 mA	V_{OL1}		0,45	V
Output voltage LOW (P0 and P2)	I_{OL} = 1,6 mA	V_{OL2}		0,45	V
Output voltage HIGH (all outputs unless open-drain)	I_{OH} = −50 μA	V_{OH}	2,4		V
Output leakage current	$V_{CC} > V_I > V_{SS}$	± I_{OL}		10	μA

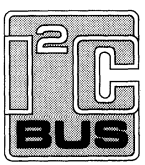
AC CHARACTERISTICS

parameter	symbol	min.	max.	unit
Frequency				
MAB/MAF8422/42	f_{XTAL}	1	6	MHz
MAF84A22/42	f_{XTAL}	1	5	MHz
Cycle time				
MAB/MAF8422/42	t_{CY}	5	30	μs
MAF84A22/42	t_{CY}	6	30	μs

T1 ZERO-CROSS CHARACTERISTICS

T_{amb} = 0 to + 70 °C; V_{CC} = 5 V ± 10%; V_{SS} = 0 V; C_L = 80 pF

parameter	conditions	symbol	min.	max.	unit
Zero-cross detection input (T1) peak-to-peak	AC coupled, C = 1 μF	$V_{ZX(p-p)}$	1	3	V
Zero-cross accuracy	50 Hz sine wave	A_{ZX}	–	± 135	mV
Zero-cross detection input frequency (T1)		F_{ZX}	0,05	1	kHz



Purchase of Philips' I²C components conveys a license under the Philips' I²C patent to use the components in the I²C-system provided the system conforms to the I²C specifications defined by Philips.

I²C-BUS SOFTWARE EXAMPLES

The following subroutines can be used separately, where the processor acts as either a slave or a master, or used together, where the processor is required to be both master and slave. When these two subroutines are used together then line 123 in the slave subroutine and line 126 in the master subroutine should be changed to MASTER EQU USE and SLAVE EQU USED, respectively.

Slave transmitter/receiver subroutine

Description

This subroutine transmits up to C bytes or receives up to D bytes in one transfer. The limits C and D depend upon the number of bytes (C + D + 1) that can be allocated in the data memory. With a crystal clock frequency of 6 MHz, a maximum transfer rate of 10 kbits/s is possible. However, clock synchronization slows high speed transfers (100 kbits/s) on the I²C-bus to this value. When the MAB8422/42 is not involved in a transfer, it has no effect on I²C-bus speed.

Since this subroutine forms part of the serial I/O interrupt routine, it should reside in the lower 2 K bytes of the program memory. Entering the subroutine via a serial I/O interrupt call, it starts with the slave address and the subsequent R/$\overline{W}$ bit determines the mode of operation (slave transmitter/receiver). The MAB8422/42 acknowledges the master if its own slave address is received, and if it can perform the desired operation. If not, the microcontroller returns a not-acknowledge signal and exits the subroutine by a "return to main program" instruction. The subroutine then enters either the receiver or the transmitter program section, depending on the value of the R/$\overline{W}$ bit.

In the slave receiver mode, data is stored in the allocated registers, and, after each byte is received an acknowledge is sent. On receiving a STOP condition or when all available registers are full, the processor sets the data-valid flag and exits the subroutine by a "return to main program" instruction. Subsequent data bytes are ignored, resulting in a not-acknowledge to the master-transmitter. When a bus error (e.g. a bus obstruction where SDA and/or SCL are held at a fixed level) or spurious START and/or STOP conditions occur, the processor exits the subroutine without setting the data-valid flag. As long as the data-valid flag is set, the slave receiver does not acknowledge its address, indicating that it cannot receive data (it can still operate as a slave transmitter). After the data registers have been read, the main program must clear the data-valid flag.

In the slave transmitter section, data available in the allocated registers is sent but between each byte transmitted, an acknowledge must be received. On receiving a not-acknowledge signal or when all available bytes have been transmitted, the transmission stops, the data-ready flag is cleared and the processor exits the subroutine by a "return to main program" instruction. If a bus error or spurious START and/or STOP condition occurs the I^2C-bus is released and the processor exits the subroutine without clearing the data-ready flag. As long as the data-ready flag is not set, the slave transmitter does not acknowledge its address, indicating that it is not ready to transmit data (it can still operate as a slave receiver).

Note* If a master routine is present, line 123 must be changed into:

```
MASTER   EQU   USED
```

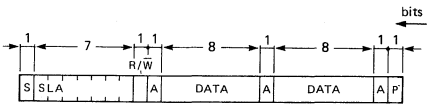
this slave subroutine is also called when in the master routine the arbitration is lost and the devices own slave address is called.

DATA FORMATS

SLAVE TRANSMITTER

R/$\overline{W}$ BIT = '0'

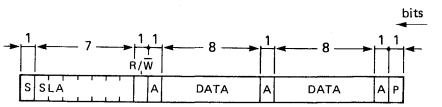
FINAL ACKNOWLEDGE BIT = '1'



7Z91771.1

SLAVE RECEIVER

R/$\overline{W}$ BIT = '1'



7Z91771.1

S = START CONDITION
SLA = 7-BIT SLAVE ADDRESS
R/$\overline{W}$ = READ/NOT WRITE BIT
A = ACKNOWLEDGE BIT
DATA = 8-BIT DATA BYTE
P = STOP CONDITION

Fig.13 Slave transmitter/receiver modes.

Table 4 Serial port equates

signal	definition	pin name	pin number	function
SDA	EQU 01 H	SDA/P10	19	I^2C-bus data I/O
SCL	EQU 02 H	SCL/P11	18	I^2C-bus clock I/O

The internal flags of Port 1 are used to control the serial I/O hardware. These flags cannot be accessed directly by the user via the port pinning.

signal	definition	port bit	R/W	function
NXTBN	EQU 04 H	P12	W	Next bit transfer NOT
STCHN	EQU 08 H	P13	W	Stretch SCL LOW period NOT
BBFN	EQU 10 H	P14	R	Bus busy flag NOT
STAF	EQU 20 H	P15	R	Start condition flag
CLLHN	EQU 40 H	P16	R	SCL LOW to HIGH NOT
DATB	EQU 80 H	P17	R	Data bit received

NOT = active-LOW

```
108         EJECT
109 *       SYMBOL DEFINITION.
110 *       ##################
111 *
112 ERRF2   EQU     H'40'           SCL PERIOD EXEEDS TIME LIMIT.
113 SCLC    EQU     250             250 X 12 MACHINE CYCLES TO DEFINE SCL PERIOD
114 *                               TIME LIMIT.(15 MSEC. @ 6 MHZ XTAL)
115 NTBY    EQU     4               MAX. NUMBER OF BYTES TO BE TRANSMITTED.
116 *                               (MUST BE ADAPTED TO THE APPLICATION).
117 NRBY    EQU     4               MAX. NUMBER OF BYTES TO BE RECEIVED.
118 *                               (MUST BE ADAPTED TO THE APPLICATION).
119 OWNAD   EQU     H'50'           OWN SLAVE ADDRESS IN THE 7 MSB. BIT 0 ="0".
120 *                               (MUST BE ADAPTED TO THE APPLICATION).
121 *
122 USED    EQU     1               FOR THE PRESENTS OF A MASTER SUBROUTINE.
123 MASTER  EQU     .NOT.USED       MASTER SUBROUTINE IS NOT PRESENT.
124 *                               IF MASTER SUBROUTINE IS PRESENT, CHANGE INTO:
125 *                               MASTER   EQU    USED
126 *
127 *       DATA MEMORY ALLOCATION.
128 *       ######################
129 *
130 SLVST   EQU     H'30'           SLAVE STATUS REGISTER. CAN BE RELOCATED.
131 *
132 *               SLAVE STATUS REGISTER CONTAINS:
133 *-------------------------------------------------------------------------------
134 SRDAVF  EQU     H'80'           SLV/REC DATA-VALID FLAG.
135 STDRF   EQU     H'40'           SLV/TRX DATA-READY FLAG.
136 *-------------------------------------------------------------------------------
137 *
138 STDTR1  EQU     SLVST+1         SLV/TRX DATA BYTE REGISTER 1
139 STDTR2  EQU     STDTR1+1                ,,                 2
140 STDTR3  EQU     STDTR2+1                ,,                 3
141 STDTR4  EQU     STDTR3+1                ,,                 4
142 *ETC.
143 SRDTR1  EQU     STDTR4+1        SLV/REC DATA BYTE REGISTER 1 (LAST STDTR+1)
144 SRDTR2  EQU     SRDTR1+1                ,,                 2
145 SRDTR3  EQU     SRDTR2+1                ,,                 3
146 SRDTR4  EQU     SRDTR3+1                ,,                 4
147 *ETC.
148 *THE NUMBER OF DATA BYTE REGISTERS MUST BE ADAPTED TO THE APPLICATION
149 *-------------------------------------------------------------------------
```

```
150         EJECT
151 *
152 *       INITIALIZATION.
153 *       ###############
154 *
155 *       THE FOLLOWING INITIALIZATION MUST BE CARRIED OUT BY THE MAIN
156 *       PROGRAM TO ENABLE THE SLAVE OPERATION (IN THE GIVEN ORDER) :
157 *
158 *    1) CLEAR SRDAVF AND STDRF BY WRITING H'00' TO SLVST REGISTER.
159 *       THIS INDICATES NO RECEIVED DATA AVAILABLE AND NO DATA READY TO
160 *       TRANSMIT.
161 *
162 *    2) ENABLE SERIAL INTERRUPT.
163 *
164 *    3) WRITE .NOT.NXTBN (H'FB') TO PORT 1 TO ENABLE THE SERIAL HARDWARE.
165 *
166 *       AFTER THIS INITIALIZATION THE MICROCONTROLLER CAN BE ADDRESSED AS
167 *       SLAVE RECEIVER AND RECEIVE SUBSEQUENT DATA BYTES.
168 *-------------------------------------------------------------------------
169 *
170 *       THIS SLAVE SUBROUTINE WILL AFFECT THE MICROCONTROLLER AS FOLLOWS:
171 *
172 *     - R1, R2, R3, R4, R5 AND R6 IN REGISTER BANK 1 ARE MODIFIED.
173 *
174 *     - SUBROUTINE NESTING = 1 (EXCLUDED THE SERIAL INTERRUPT CALL).
175 *
176 *     - NUMBER OF DATA MEMORY BYTES USED = 1+C+D STARTING AT LOCATION
177 *       H'30' (C = NUMBER OF DATA BYTES TRANSMITTED, D = THE NUMBER OF DATA
178 *       BYTES RECEIVED).
179 *-------------------------------------------------------------------------
```

```
                        180         EJECT
                        181 *
                        182 *       ##########################################################
                        183 *       #       START SLAVE TRANSMITTER/RECEIVER SUBROUTINE       #
                        184 *       ##########################################################
                        185 *
                        186 *ENTRY:  STDTR1 UP TO STDTR(C) CONTAIN UP TO C BYTES TO BE TRANSMITTED.
                        187 *         STDRF   SLV/TRX DATA-READY FLAG; HAS TO BE SET TO "1" AFTER STDTR
                        188 *                 DATA MEMORY LOCATIONS HAVE BEEN LOADED.
                        189 *         SRDAVF  SLV/REC DATA-VALID FLAG; MUST BE CLEARED TO "0" IF SRDTR
                        190 *                 DATA MEMORY LOCATIONS ARE FREE TO RECEIVE DATA.
                        191 *
                        192 *
                        193 *EXIT:   SRDTR1 UP TO STDTR(D) CONTAIN UP TO D RECEIVED BYTES.
                        194 *         SRDAVF  IS SET TO "1" IF DATA IS CORRECTLY RECEIVED.
                        195 *         STDRF   IS CLEARED TO "0" IF DATA CORRECTLY TRANSMITTED.
                        196 *
                        197 *
                        198 SLAVE1 ASECT   ROM
                        199         PAGE    256
000000                  200         ORG     H'05'           SIO INTERRUPT VECTOR
                        201         IF      MASTER=USED
                        202         ENTRY   SIV             ENTRY FROM MULTI-MASTER SUBROUTINE.
                        203         ENTRY   SAR             ENTRY FROM MULTI-MASTER SUBROUTINE.
                        204         ENDIF
                        205 *
000005 0410        1    206 SIV     JMP     SLV             JUMP TO BEGIN OF THE SLAVE SUBROUTINE.
                        207 *
000007                  208         ORG H'010'
                        209 *-------------------------------------------------------------------------
                        210 *               RECEPTION OF SLAVE ADDRESS + READ/WRITE BIT.
                        211 *-------------------------------------------------------------------------
000010 D5          2    212 SLV     SEL     RB1             SELECT REGISTER BANK 1
000011 FE          3    213         MOV     A,R6            SAVE ACCU.
                        214 *
000012 1474        4    215 SLV1    CALL    SRDB            CALL SLV/REC DATA BYTE SUBROUTINE.
000014 B212        5    216         JB5     SLV1            JUMP IF START CONDITION DETECTED.
000016 9643        6    217         JNZ     SLVEX           JUMP IF STOP OR SCL ERROR.
000018 FC          7    218 SAR     MOV     A,R4            FETCH RECEIVED SLAVE ADDRESS.
000019 D5          8    219         SEL     RB1             POSSIBLE ENTRY AT SAR FROM MASTER SUBROUTINE.
00001A D350        9    220         XRL     A,#OWNAD        COMPARE WITH OWN SLAVE ADDRESS.
00001C C649       10    221         JZ      SRDT            JUMP IF ADDRESSED AS SLAVE RECEIVER.
00001E 07         11    222         DEC     A               COMPLEMENT READ/WRITE BIT.
00001F 9643       12    223         JNZ     SLVEX           JUMP IF NOT ADDRESSED AS SLAVE TRANSMITTER.
```

```
                        224         EJECT
                        225 *       SLAVE TRANSMITTER ROUTINE.
                        226 *       ==========================
                        227 *-------------------------------------------------------------------------
                        228 *           - GENERATION OF AN ACKNOWLEDGE ("0") IF THE DATA-READY FLAG
                        229 *             "STDRF" HAS BEEN SET, OTHERWISE THE ROUTINE IS EXITTED.
                        230 *           - TRANSMISSION OF DATA BYTES STORED IN THE DATA MEMORY
                        231 *             LOCATIONS STDTR1, STDTR2 ETC.
                        232 *           - RECEPTION OF THE ACKNOWLEDGE BIT AFTER EACH TRANSMITTED
                        233 *             BYTE. WHEN A NOT-ACKNOWLEDGE IS RECEIVED, THE
                        234 *             TRANSMISSION STOPS AND "STDRF" IS CLEARED.
                        235 *           - IF THE MAX. NUMBER OF BYTES "NTBY" HAS BEEN
                        236 *             TRANSMITTED, THE TRANSMISSION STOPS AND "STDRF" IS CLEARED.
                        237 *           - IF A BUS ERROR IS DETECTED, THE SLAVE SUBROUTINE TERMINATES
                        238 *             WITHOUT CLEARING "STDRF".
                        239 *-------------------------------------------------------------------------
                        240 *
000021 B930       13    241 STDT    MOV     R1,#SLVST       LOAD SLAVE STATUS LOCATION IN R1
000023 F1         14    242         MOV     A,@R1           FETCH SLAVE STATUS.
000024 37         15    243         CPL     A
000025 D243       16    244         JB6     SLVEX           JUMP IF STDRF IS NOT SET.
000027 14B8       17    245         CALL    SRAC            OUTPUT ACKNOWLEDGE.
000029 9643       18    246         JNZ     SLVEX           JUMP IF SCL ERROR.
00002B 19         19    247         INC     R1              LOCATION OF FIRST SLV/TRX DATA REG. TO R1.
00002C BD04       20    248         MOV     R5,#NTBY        MAXIMUM NUMBER OF BYTES IN R5.
                        249 *
00002E F1         21    250 STDT1   MOV     A,@R1           FETCH DATA BYTE TO BE TRANSMITTED.
00002F 19         22    251         INC     R1              NEXT SLV/TRX DATA REG. LOCATION.
000030 1496       23    252         CALL    STDB            OUTPUT DATA BYTE.
000032 9643       24    253         JNZ     SLVEX           JUMP IF START,STOP OR SCL ERROR.
000034 14BD       25    254         CALL    STAC            INPUT ACKNOWLEDGE BIT.
000036 9643       26    255         JNZ     SLVEX           JUMP IF START, STOP OR SCL ERROR.
000038 FC         27    256         MOV     A,R4            FETCH ACKNOWLEDGE BIT
000039 963D       28    257         JNZ     STEX            JUMP IF NOT-ACKNOWLEDGE.
00003B ED2E       29    258         DJNZ    R5,STDT1        JUMP IF NOT MAX. NUMBER OF BYTES TRANSMITTED.
                        259 *
00003D B930       30    260 STEX    MOV     R1,#SLVST       LOAD SLAVE STATUS LOCATION IN R1.
00003F F1         31    261         MOV     A,@R1           FETCH SLAVE STATUS.
000040 53BF       32    262         ANL     A,#.NOT.STDRF   CLEAR SLV/TRX DATA READY FLAG.
000042 A1         33    263         MOV     @R1,A           SAVE SLAVE STATUS.
                        264 *
                        265 *
                        266 *       SLAVE MODE EXIT ROUTINE.
                        267 *       ========================
000043 89FF       34    268 SLVEX   ORL     P1,#SDA+SCL+NXTBN+STCHN+H'F0'    SET SDA TO HIGH AND RELEASE
                        269 *                                   SCL(STILL STRETCHED); DISABLE STRETCHING.
000045 99FB       35    270         ANL     P1,#.NOT.NXTBN SET SCL OUTPUT TO HIGH.
                        271 *
000047 FE         36    272 SIOEX   MOV     A,R6            RESTORE ACCU.
000048 93         37    273         RETR
```

```
                        274          EJECT
                        275 *
                        276 *        SLAVE RECEIVER ROUTINE.
                        277 *        =======================
                        278 *-------------------------------------------------------------------------
                        279 *              - GENERATION OF AN ACKNOWLEDGE ("0") IF THE DATA-VALID FLAG
                        280 *                "SRDAVF" HAS BEEN CLEARED, OTHERWISE THE ROUTINE IS EXIT.
                        281 *              - RECEPTION OF DATA BYTES THAT ARE THEN STORED IN THE DATA
                        282 *                MEMORY LOCATIONS SRDTR1, SRDTR2, ETC.
                        283 *              - TRANSMISSION OF AN ACKNOWLEDGE ("0") AFTER EACH RECEIVED
                        284 *                BYTE.
                        285 *              - ON RECEIPT OF A STOP CONDITION, OR IF ALL DATA MEMORY
                        286 *                LOCATIONS ARE OCCUPIED, THE "SRDAVF" IS SET AND THE
                        287 *                ROUTINE IS LEFT.
                        288 *              - IF A BUS ERROR IS DETECTED THE SLAVE SUBROUTINE TERMINATES
                        289 *                WITHOUT SETTING "SRDAVF".
                        290 *-------------------------------------------------------------------------
                        291 *
000049 B930      38     292 SRDT     MOV     R1,#SLVST       LOAD SLAVE STATUS LOCATION IN R1.
00004B F1        39     293          MOV     A,@R1           FETCH SLAVE STATUS.
00004C F243      40     294          JB7     SLVEX           JUMP IF SRDAVF IS SET.
00004E B935      41     295          MOV     R1,#SRDTR1      LOCATION OF FIRST SLV/REC.
000050 BD04      42     296          MOV     R5,#NRBY        MAX.NUMBER OF BYTES IN R5.
                        297 *
000052 14B8      43     298 SRDT1    CALL    SRAC            OUTPUT ACKNOWLEDGE.
000054 9643      44     299          JNZ     SLVEX           JUMP IF SCL ERROR.
000056 1474      45     300          CALL    SRDB            INPUT DATA BYTE.
000058 966B      46     301          JNZ     SRDT3           JUMP IF START,STOP COND. OR SCL ERROR.
00005A FC        47     302          MOV     A,R4            FETCH RECEIVED DATA BYTE.
00005B A1        48     303          MOV     @R1,A           SAVE RECEIVED DATA BYTE.
00005C 19        49     304          INC     R1              NEXT SLV/REC DATA REG. LOCATION.
00005D ED52      50     305          DJNZ    R5,SRDT1        JUMP IF NOT MAX NUMBER OF BYTES.
00005F 14B8      51     306          CALL    SRAC            OUTPUT ACKNOWLEDGE.
000061 9643      52     307          JNZ     SLVEX           JUMP IF STOP OR SCL ERROR.
000063 B930      53     308 SRDT2    MOV     R1,#SLVST       LOAD SLAVE STATUS LOCATION IN R1.
000065 F1        54     309          MOV     A,@R1           FETCH SLAVE STATUS.
000066 4380      55     310          ORL     A,#SRDAVF       SET "SRDAVF" FLAG.
000068 A1        56     311          MOV     @R1,A           SAVE SLAVE STATUS.
000069 0443      57     312          JMP     SLVEX
                        313 *
00006B 5243      58     314 SRDT3    JB2     SLVEX           JUMP IF SCL ERROR.
                        315 *                                OTHERWISE A STOP CONDITION HAS BEEN RECEIVED,
                        316 *                                POSSIBLY FOLLOWED BY A NEW START CONDITION.
00006D FB        59     317          MOV     A,R3            FETCH BIT COUNTER.
00006E D308      60     318          XRL     A,#8            TEST IF BIT COUNTER=8
000070 C663      61     319          JZ      SRDT2           JUMP IF STOP CONDITION WAS AT THE END OF THE
                        320 *                                BYTE.
000072 0443      62     321          JMP     SLVEX           ERROR, STOP CONDITION WAS WITHIN THE BYTE.
                        322 *
```

```
                                EJECT
                        *       ##########################################################
                        *       #       SINGLE BYTES AND ACKNOWLEDGE SUBROUTINES.        #
                        *       ##########################################################
                        *
                        *       SLAVE RECEIVER SINGLE DATA BYTE SUBROUTINE.
                        *       ==========================================
                        *---------------------------------------------------------------------
                        * SUBROUTINE INPUTS 8 BITS OF DATA IN SLAVE RECEIVER MODE
                        * EXIT: A = 00; DATA IS RECEIVED CORRECTLY
                        *                R4 CONTAINS DATA BYTE AND
                        *                BIT COUNTER R3 = 0
                        *        A = STAF; START CONDITION OR STOP CONDITION FOLLOWED BY
                        *                  A START CONDITION IS DETECTED.
                        *        A = BBNF; STOP CONDITION DETECTED
                        *        A = ERRF2; SCL EXCEEDS TIME LIMIT
                        *---------------------------------------------------------------------
000074 BB08       63    SRDB    MOV     R3,#8           SET BIT COUNTER.
000076 8907       64    SRDB1   ORL     P1,#SDA+SCL+NXTBN  SET P10/SDA HIGH, RELEASE P11/SCL.
000078 99F3       65            ANL     P1,#.NOT.(NXTBN+STCHN)  SET P11/SCL HIGH; ENABLE STRETCHING.
00007A BAFA       66            MOV     R2,#SCLC        LOAD SCL TIME-OUT COUNTER.
                        *
00007C 09         67    SRDB2   IN      A,P1            INPUT STATUS.
00007D F7         68            RLC     A               SHIFT RECEIVED BIT IN CARRY.
00007E 53E4       69            ANL     A,#B'11100100'  TEST CLLHN*STAF*BBNF*SCL=0
000080 C68B       70            JZ      SRDB4           JUMP IF DATA BIT HAS BEEN RECEIVED.
                        *
000082 B292       71    SRDB3   JB5     SRDB5           JUMP IF STOP CONDITION HAS BEEN RECEIVED.
000084 D292       72            JB6     SRDB5           JUMP IF START CONDITION HAS BEEN RECEIVED.
000086 EA7C       73            DJNZ    R2,SRDB2        JUMP IF SCL COUNTER NOT ZERO.
000088 2340       74            MOV     A,#ERRF2        SET ERROR FLAG; SCL PERIOD TIME-OUT.
00008A 83         75            RET
                        *
00008B 2C         76    SRDB4   XCH     A,R4            DATA BYTE TO ACCU.
00008C F7         77            RLC     A               SHIFT RECEIVED BIT IN DATA BYTE.
00008D 2C         78            XCH     A,R4            SAVE DATA BYTE IN R4.
00008E EB76       79            DJNZ    R3,SRDB1        DECR. AND JUMP IF BIT COUNTER NOT ZERO.
000090 27         80            CLR     A               SET FLAGS TO ZERO.
000091 83         81            RET
                        *
000092 77         82    SRDB5   RR A
000093 5330       83            ANL     A,#STAF+BBFN    MASK FLAGS.
000095 83         84            RET
                        *
```

```
                        367         EJECT
                        368 *
                        369 *       SLAVE TRANSMITTER SINGLE DATA BYTE SUBROUTINE.
                        370 *       ==============================================
                        371 *-------------------------------------------------------------------------------
                        372 * SUBROUTINE OUTPUTS 8 BITS OF DATA IN SLAVE TRANSMITTER MODE.
                        373 * ENTRY: A CONTAINS DATA BYTE TO BE TRANSMITTED.
                        374 * EXIT: A = 00; DATA BYTE IS OUTPUT CORRECTLY
                        375 *                 R4 CONTAINS THE TRANSMITTED DATA BYTE
                        376 *         A = STAF; START CONDITION OR STOP CONDITION FOLLOWED BY
                        377 *                   A START CONDITION IS DETECTED.
                        378 *         A = BBNF; STOP CONDITION DETECTED
                        379 *         A = ERRF2; SCL EXCEEDS TIME LIMIT.
                        380 *-------------------------------------------------------------------------------
000096 BB08       85    381 STDB    MOV     R3,#8           SET BIT COUNTER.
000098 AC         86    382         MOV     R4,A            DATA BYTE TO R4.
000099 FC         87    383 STDB0   MOV     A,R4            FETCH DATA BYTE.
00009A E7         88    384         RL      A               DATA BIT TO BIT 0 OF ACCU.
00009B AC         89    385         MOV     R4,A            SHIFTED DATA BYTE TO R4.
00009C 43FE       90    386 STDB1   ORL     A,#SCL+NXTBN+STCHN+H'F0'  PREPARE P1 SIGNALS.
00009E 39         91    387         OUTL    P1,A            WRITE DATA BIT TO P10/SDA, RELEASE P11/SCL.
00009F 99F3       92    388         ANL     P1,#.NOT.(NXTBN+STCHN)  SET P11/SCL HIGH.
0000A1 BAFA       93    389         MOV     R2,#SCLC        LOAD SCL TIME-OUT COUNTER.
                        390 *
0000A3 09         94    391 STDB2   IN      A,P1            FETCH STATUS.
0000A4 5372       95    392         ANL     A,#CLLHN+STAF+BBFN+SCL  MASK FLAGS.
0000A6 C6B1       96    393         JZ      STDB4           JUMP IF DATA BIT CORRECTLY TRANSMITTED.
                        394 *
0000A8 92B5       97    395 STDB3   JB4     STDB5           JUMP IF STOP CONDITION HAS BEEN RECEIVED.
0000AA B2B5       98    396         JB5     STDB5           JUMP IF START CONDITION HAS BEEN RECEIVED.
0000AC EAA3       99    397         DJNZ    R2,STDB2        JUMP IF SCL TIME OUT COUNTER NOT ZERO
0000AE 2340      100    398         MOV     A,#ERRF2        SET ERROR FLAG; SCL PERIOD TIME OUT.
0000B0 83        101    399         RET
                        400 *
0000B1 EB99      102    401 STDB4   DJNZ    R3,STDB0        JUMP IF BIT COUNTER NOT ZERO.
0000B3 27        103    402         CLR     A               CLEAR FLAGS
0000B4 83        104    403         RET
                        404 *
0000B5 5330      105    405 STDB5   ANL     A,#STAF+BBFN    MASK FLAGS.
0000B7 83        106    406         RET
```

```
                        407         EJECT
                        408 *       SLAVE RECEIVER ACKNOWLEDGE SUBROUTINE.
                        409 *       ======================================
                        410 *-------------------------------------------------------------------------
                        411 *SUBROUTINE OUTPUTS AN ACKNOWLEDGE ("0") IN SLAVE RECEIVER MODE
                        412 * EXIT: A = 00; ACKNOWLEDGE IS OUTPUT CORRECTLY
                        413 *       A = ERRF2; SCL EXCEEDS TIME LIMIT
                        414 *-------------------------------------------------------------------------
0000B8 BB01       107   415 SRAC    MOV     R3,#1           SET BIT COUNTER.
0000BA 27         108   416         CLR     A               TO SEND A "0" AS ACKNOWLEDGE.
0000BB 049C       109   417         JMP     STDB1           JUMP TO SLV/TRX MODE.
                        418 *
                        419 *
                        420 *
                        421 *       SLAVE RECEIVER NOT-ACKNOWLEDGE SUBROUTINE OR
                        422 *       SLAVE TRANSMITTER INPUT ACKNOWLEDGE SUBOUTINE.
                        423 *       ==============================================
                        424 *-------------------------------------------------------------------------
                        425 * SUBROUTINE OUTPUTS A NOT-ACKNOWLEDGE ("1") IN SLAVE RECEIVER MODE.
                        426 * SUBROUTINE INPUTS ACKNOWLEDGE IN SLAVE TRANSMITTER MODE.
                        427 * EXIT: A = 00; ACKNOWLEDGE BIT IS OUTPUT OR RECEIVED CORRECTLY.
                        428 *               R4 = 0; ACKNOWLEDGE ("0") RECEIVED.
                        429 *               R4 = 1; NOT-ACKNOWLEDGE ("1") RECEIVED.
                        430 *       A = STAF; START CONDITION OR STOP CONDITION FOLLOWED BY
                        431 *                 A START CONDITION IS DETECTED.
                        432 *       A = BBNF; STOP CONDITION DETECTED
                        433 *       A = ERRF2; SCL EXCEEDS TIME LIMIT
                        434 *-------------------------------------------------------------------------
0000BD BB01       110   435 STAC    MOV     R3,#1           SET BIT COUNTER.
0000BF BC00       111   436         MOV     R4,#0           CLEAR DATA REGISTER.
0000C1 0476       112   437         JMP     SRDB1           JUMP TO SLV/REC MODE.
                        438 *
0000C3                  439         END

  NO ERRORS DETECTED
```

Multi-master subroutine

Description

This subroutine transmits up to E bytes or receives up to F bytes in one transfer, with or without pointer byte and repeated START condition. The limits E and F depend upon the number of bytes (4 + E + F) that can be allocated in the data memory. The I^2C interface of the MAB8422/42 filters out spikes of less than 2 crystal clock periods, improving data transfer reliability. With a crystal clock frequency of 6 MHz, transfer rates between 8 and 9,5 kbits/s are possible. However, clock synchronization slows high speed transfers (100 kbits/s) on the I^2C-bus to this value. When the MAB8422/42 is not involved in a transfer, it has no effect on I^2C-bus speed. Arbitration on the serial data line and synchronization on the clock line follow the I^2C specification permitting multi-master operation.

If another master transmits a START condition, this subroutine reacts via a serial I/O interrupt routine by stretching the SCL line during the START condition. It releases the bus after the START condition, and the rest of the data transfer is unaffected. When arbitration is lost, another subroutine is used to release the bus. These two subroutines are found at the end of the program. Also, if the slave routine is present, these interrupt routines are deleted by changing line 126 to:

```
SLAVE   EQU     USED
```

Error conditions

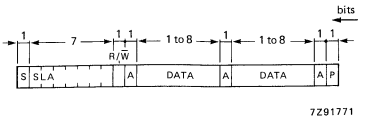
Any additional clock pulses are dealt with in the synchonization of the serial clocks, and require no software action. However, transfer is terminated and appropriate error flags set when spurious START and/or STOP conditions occur. Furthermore, noise that causes an error in the level of a bit will set a flag indicating arbitration lost. Due to the wired-AND structure of the bus, noise that distorts a 0 to form a 1 is unlikely and is not handled by this subroutine. A software timer monitors the SCL LOW period and if a device stretches the period beyond the 7,5 ms time-out value, transfer is terminated and an error flag is set. When the bus is occupied for more than 1 second (6 MHz crystal), up to 10 forced STOP conditions are generated to free the bus and transfer is attempted again. If this attempt is also unsuccessful, an error flag is set.

DATA FORMATS

(a)

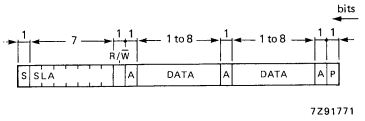
MASTER TRANSMITTER NO POINTER

R/$\overline{W}$ BIT = '0'

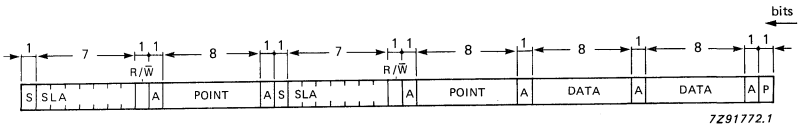


MASTER RECEIVER NO POINTER

R/$\overline{W}$ BIT = '1', FINAL ACKNOWLEDGE BIT = '1'



(b)



MASTER TRANSMITTER WITH POINTER AND
REPEATED START CONDITION

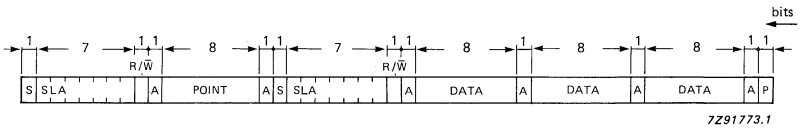


Fig.14 Multi-master transmitter/receiver modes.

MASTER RECEIVER WITH POINTER AND
REPEATED START CONDITION
FINAL ACKNOWLEDGE BIT = '0'

S = START CONDITION
SLA = 7-BIT SLAVE ADDRESS
R/$\overline{W}$ = READ/NOT WRITE BIT
A = ACKNOWLEDGE/ NOT ACKNOWLEDGE BIT
POINT = 8-BIT POINTER BYTE
DATA = 8-BIT DATA BYTE
P = STOP CONDITION

The number of data bytes transmitted is specified in register 'DBCNTR'

Table 5 serial port equates

signal definition			pin name	pin number	function
SDA	EQU	01 H	SDA/P10	19	I^2C-bus data I/O
SCL	EQU	02 H	SCL/P11	18	I^2C-bus clock I/O

The internal flags of Port 1 are used to control the serial I/O hardware. These flags cannot be accessed directly by the user via the port pinning.

signal definition			port bit	R/W	Function
NXTBN	EQU	04 H	P12	W	Next bit transfer NOT
STCHN	EQU	08 H	P13	W	Stretch SCL LOW period NOT
BBFN	EQU	10 H	P14	R	Bus busy flag NOT
STAF	EQU	20 H	P15	R	Start condition flag
CLLHN	EQU	45 H	P16	R	SCL LOW to HIGH NOT
DATB	EQU	80 H	P17	R	Data bit received

NOT = active-LOW

```
107         EJECT
108 *       SYMBOL DEFINITION.
109 *       ##################
110 *
111 ERRF1   EQU     H'80'           ARBITRATION LOST.
112 ERRF2   EQU     H'40'           BUS OBSTRUCTED; SCL AND/OR SDA STAY LOW.
113 ERRF3   EQU     H'20'           SPURIOUS START CONDITION.
114 ERRF4   EQU     H'10'           SPURIOUS STOP CONDITION.
115 NAACK   EQU     H'02'           NO ADDRESS ACKNOWLEDGE FROM SLAVE.
116 NDACK   EQU     H'01'           NO DATA ACKNOWLEDGE FROM SLAVE RECEIVER.
117 *
118 RWB     EQU     H'01'           READ/WRITE BIT AFTER SLAVE ADDRESS.
119 SCLC    EQU     250             250 X 6 MACHINE CYCLES TO DEFINE SCL PERIOD
120 *                               TIME LIMIT. (7.5 MSEC @ 6 MHZ XTAL)
121 *                               CAN BE ADAPTED TO THE APPLICATION.
122 BTOC    EQU     100             THIS NUMBER TIMES 10 MSEC DEFINES THE BUS.
123 *                               TIME-OUT LIMIT IN CASE OF A BUS OBSTRUCTION.
124 *                               CAN BE ADAPTED TO THE APPLICATION.
125 USED    EQU     1               FOR THE PRESENCE OF A SLAVE SUBROUTINE.
126 SLAVE   EQU     .NOT.USED       SLAVE SUBROUTINE IS NOT PRESENT.
127 *                               IF SLAVE SUBROUTINE IS PRESENT, CHANGE INTO:
128 *                               SLAVE  EQU    USED
129 *
130 *       DATA MEMORY ALLOCATION.
131 *       ########################
132 *
133 SIOMDR  EQU     H'20'           SIO MODE REGISTER. CAN BE RELOCATED.
134 *               SIO MODE REGISTER CONTAINS:
135 *-------------------------------------------------------------------------
136 SLVMF   EQU     H'00'           SLAVE MODE; I.E. NOT ACTIVE AS MASTER.
137 MTMF    EQU     H'08'           MASTER/TRANSMITTER MODE WITHOUT POINTER-BYTE
138 MRMF    EQU     H'04'           MASTER/RECEIVER MODE WITHOUT POINTER-BYTE.
139 MTPMF   EQU     H'02'           MASTER/TRANSMITTER MODE WITH POINTER-BYTE AND
140 *                               REPEATED START CONDITION.
141 MRPMF   EQU     H'01'           MASTER/RECEIVER MODE WITH POINTER-BYTE AND
142 *                               REPEATED START CONDITION.
143 *-------------------------------------------------------------------------
144 SLVADR  EQU     SIOMDR+1        SLAVE ADDRESS REGISTER
145 DBCNTR  EQU     SLVADR+1        DATA BYTE COUNTER REGISTER
146 POINTR  EQU     DBCNTR+1        POINTER VALUE REGISTER
147 *
148 MTDTR1  EQU     POINTR+1        MST/TRX DATA BYTE REGISTER 1
149 MTDTR2  EQU     MTDTR1+1                ,,                 2
150 MTDTR3  EQU     MTDTR2+1                ,,                 3
151 MTDTR4  EQU     MTDTR3+1                ,,                 4
152 * ETC.
153 MRDTR1  EQU     MTDTR4+1        MST/REC DATA BYTE REGISTER 1 (LAST MTDTR+1)
154 MRDTR2  EQU     MRDTR1+1                ,,                 2
155 MRDTR3  EQU     MRDTR2+1                ,,                 3
156 MRDTR4  EQU     MRDTR3+1                ,,                 4
157 * ETC.
158 *THE NUMBER OF DATA BYTE REGISTERS MUST BE ADAPTED TO THE APPLICATION.
159 *-------------------------------------------------------------------------
```

```
160         EJECT
161 *
162 *       INITIALIZATION.
163 *       ###############
164 *
165 *       THE FOLLOWING INITIALIZATION MUST BE CARRIED OUT BY THE MAIN
166 *       PROGRAM TO ENABLE THE MULTI-MASTER OPERATION (IN THE ORDER GIVEN) :
167 *
168 *    1) SELECT REGISTER BANK 0 (THE SLAVE SUBROUTINE USES REGISTER BANK 1).
169 *
170 *    2) ENABLE SERIAL INTERRUPT.
171 *
172 *    3) WRITE .NOT.NXTBN (H'FB') TO PORT 1 TO ENABLE THE SERIAL I/O
173 *       HARDWARE.
174 *-------------------------------------------------------------------------
175 *       THIS  MULTI-MASTER SUBROUTINE WILL AFFECT THE MICROCONTROLLER
176 *       AS FOLLOWS:
177 *
178 *     - R1, R2, R3, R4 AND R5 IN REGISTER BANK 0 ARE MODIFIED.
179 *
180 *     - THE ACCUMULATOR AND CARRY BIT ARE MODIFIED.
181 *
182 *     - SUBROUTINE NESTING = 1 (INCLUDING THE SERIAL INTERRUPT CALL).
183 *
184 *     - NUMBER OF DATA MEMORY BYTES USED = 4+E+F STARTING AT LOCATION
185 *       H'20' (E = NUMBER OF DATA BYTES TRANSMITTED, F = THE NUMBER OF DATA
186 *       BYTES RECEIVED).
187 *-------------------------------------------------------------------------
```

```
       EJECT
*      ###########################################################
*      #          START MULTI-MASTER I2C SUBROUTINE              #
*      ###########################################################
*
*ENTRY:  LOAD THE NEXT LOCATIONS IN THE DATA MEMORY:
*
*        SIOMDR   WITH THE MODE (E.G. #MTMF = MST/TRX WITHOUT POINTER).
*        SLVADR   IN THE 7 MSB WITH THE ADDRESS OF THE SLAVE TO BE
*                 SELECTED. IN BIT 0 THE READ/WRITE BIT MUST BE LOADED.
*        DBCNTR   WITH THE NUMBER OF DATA BYTES TO BE TRANSMITTED OR
*                 RECEIVED.
*        POINTR   WITH THE POINTER BYTE.
*        MTDTR(1)
*        UP TO
*        MTDTR(E) WITH UP TO E BYTES TO BE TRANSMITTED.
*
*        THE MULTI-MASTER SUBROUTINE IS INVOKED BY: CALL MMST
*                                                   ---------
*
*EXIT:   THE LOCATIONS IN DATA MEMORY MRDTR(1) UP TO MRDTR(F) CONTAIN:
*                 UP TO F RECEIVED DATA BYTES.
*
*        THE ACCUMULATOR (A) CONTAINS THE FOLLOWING FLAGS:
*        A=H'00'  DATA TRANSMITTED OR RECEIVED CORRECTLY.
*        A=H'80'  ARBITRATION LOST AND BYTE COMPLETED WITH CLOCK PULSES.
*                 THE BUS IS LEFT IN A RELEASED STATE.
*        A=H'40'  BUS OBSTRUCTED; SCL AND/OR SDA STAY LOW; NO TRANSFER
*                 POSSIBLE.
*        A=H'20'  SPURIOUS START CONDITION; TRANSFER ABORTED AND CONCLUDED
*                 WITH A STOP CONDITION.
*        A=H'10'  SPURIOUS STOP CONDITION; TRANSFER ABORTED; BUS RELEASED.
*        A=H'02'  NO ADDRESS ACKNOWLEDGE FROM SLAVE; TRANSFER CONCLUDED
*                 WITH A STOP CONDITION.
*        A=H'01'  NO DATA ACKNOWLEDGE FROM SLAVE RECEIVER; TRANSFER
*                 CONCLUDED WITH A STOP CONDITION.
*
*        A COMBINATION OF FLAGS IS POSSIBLE.
*
*
*
MUMST1 ASECT  ROM
       PAGE   256
000000         ORG    H'100'
       ENTRY  MMST
       IF SLAVE=USED
       EXTRN  SIV
       EXTRN  SAR
OWNAD  EQU    H'50'          OWN SLAVE ADDRESS IN THE 7 MSB. BIT 0 ="0".
*                            (MUST BE ADAPTED TO THE APPLICATION).
       ENDIF
```

```
                        239          EJECT
                        240 *        MASTER TRANSMITTER/RECEIVER WITH OR WITHOUT POINTER-BYTE.
                        241 *        ========================================================
                        242 *-------------------------------------------------------------------------
                        243 *              TEST BUS STATUS. FORCE STOP CONDITIONS IF OBSTRUCTED.
                        244 *-------------------------------------------------------------------------
000100 95          1    245 MMST     DIS     SI              DISABLE SERIAL INTERRUPT.
000101 BB0A        2    246          MOV     R3,#10          LOAD NUMBER FORCED STOP COND TO FREE THE BUS.
000103 BA64        3    247 MS0      MOV     R2,#BTOC        LOAD BUS TIME-OUT COUNTER.
000105 B985        4    248 MS1      MOV     R1,#133         INITIALIZE 10 MS TEST LOOP.
                        249 *
000107 23FB        5    250 MS2      MOV     A,#.NOT.NXTBN
000109 39          6    251          OUTL    P1,A            TO ENSURE THE CORRECT STATE OF P1.
00010A 09          7    252 MS3      IN      A,P1            FETCH STATUS
00010B B220        8    253          JB5     MS5             JUMP IF A START COND. FROM ANOTHER MASTER
                        254 *                                 IS RECEIVED.
00010D 43EC        9    255          ORL     A,#.NOT.(SDA+SCL+BBFN)  TEST IF BUS IS FREE.
00010F 37         10    256          CPL     A
000110 C626       11    257          JZ      MST1            JUMP IF THE BUS IS FREE,SCL AND SDA ARE HIGH.
000112 E907       12    258 MS4      DJNZ    R1,MS2          DECR. AND CHECK TEST LOOP COUNTER.
000114 EA05       13    259          DJNZ    R2,MS1          DECR. AND CHECK TIME OUT COUNTER.
000116 99FD       14    260          ANL     P1,#.NOT.SCL    SET SCL OUTPUT TO LOW.
000118 5474       15    261          CALL    MSSTP           FORCE A STOP CONDITION TO FREE THE BUS.
00011A EB03       16    262          DJNZ    R3,MS0          RETRY TO GET THE BUS.
00011C 2340       17    263          MOV     A,#ERRF2        LOAD ERROR FLAG: BUS OBSTRUCTED.
00011E 24BB       18    264          JMP     MSTER           JUMP TO MASTER ERROR ROUTINE.
                        265 *
000120 3212       19    266 MS5      JB1     MS4             WAIT UNTIL SCL GOES LOW.
000122 1405       20    267          CALL    SIV             CALL SLAVE INTERRUPT VECTOR.
000124 240A       21    268          JMP     MS3             RETRY TO GET THE BUS.
                        269 *-------------------------------------------------------------------------
                        270 *        TRANSMISSION OF START CONDITION+SLAVE ADDRESS+READ/WRITE BIT
                        271 *        RECEPTION OF ACKNOWLEDGE BIT.
                        272 *-------------------------------------------------------------------------
                        273 *
000126 99FE       22    274 MST1     ANL     P1,#.NOT.SDA    OUTPUT START CONDITION.
000128 99F5       23    275          ANL     P1,#.NOT.(SCL+STCHN)  SCL TO LOW AND STRETCHING ENABLED.
00012A B920       24    276          MOV     R1,#SIOMDR      LOCATION OF SIO MODE REG. TO R1.
00012C F1         25    277          MOV     A,@R1           FETCH SIO MODE
00012D AD         26    278          MOV     R5,A            SAVE SIO MODE IN R5.
00012E 5301       27    279          ANL     A,#MRPMF        PREPARE R/WN BIT;ONLY FOR MRPM IT IS INVERTED
000130 19         28    280          INC     R1              LOCATION OF SLAVE ADDRESS REGISTER TO R1.
000131 D1         29    281          XRL     A,@R1           PUT SLAVE ADDRESS AND R/WN BIT IN ACCU.
000132 5400       30    282          CALL    MTDB            TRANSMIT SLAVE ADDRESS + R/WN BIT.
000134 E7         31    283          RL      A               SET FLAGS IN RIGHT POSITION.
000135 67         32    284          RRC     A                "    "   "   "      "
000136 F2BB       33    285          JB7     MAL             JUMP IF ARBITRATION LOST; OWN ADDR. RECEIVED?
000138 96BB       34    286          JNZ     MSTER           JUMP IF BUS ERROR.
00013A 546E       35    287          CALL    MTAC            RECEIVE ACNOWLEDGE BIT.
00013C 2C         36    288          XCH     A,R4            SET FLAGS IN RIGHT POSITION.
00013D E7         37    289          RL      A                "    "   "   "      "
00013E 4C         38    290          ORL     A,R4            COLLECT THE FLAGS IN ACCU.
00013F 96BB       39    291          JNZ     MSTER           JUMP IF BUS ERROR OR IF NOT-ACKNOWLEDGE.
000141 FD         40    292          MOV     A,R5            FETCH SIO MODE.
000142 729F       41    293          JB3     MST5            JUMP IF MST/TRX WITHOUT POINTER-BYTE.
000144 5284       42    294          JB2     MST4            JUMP IF MST/REC WITHOUT POINTER-BYTE.
```

```
                        295         EJECT
                        296 *       MASTER TRANSMITTER/RECEIVER WITH A POINTER-BYTE.
                        297 *       ===============================================
                        298 *-------------------------------------------------------------------------
                        299 *               TRANSMISSION OF THE POINTER-BYTE.
                        300 *               RECEPTION OF THE ACKNOWLEDGE BIT.
                        301 *-------------------------------------------------------------------------
                        302 *
000146 B923       43    303 MST2    MOV     R1,#POINTR      LOCATION OF THE POINTER-BYTE TO R1.
000148 F1         44    304         MOV     A,@R1           FETCH POINTER BYTE.
000149 5400       45    305         CALL    MTDB            TRANSMIT POINTER BYTE.
00014B E7         46    306         RL      A               SET FLAGS IN ACCU IN RIGHT POSITION.
00014C 67         47    307         RRC     A                "   "   "  "   "   "     "
00014D 96BB       48    308         JNZ     MSTER           JUMP IF BUS ERROR OR IF ARBITRATION LOST.
00014F 546E       49    309         CALL    MTAC            INPUT ACKNOWLEDGE BIT.
000151 4C         50    310         ORL     A,R4            COLLECT FLAGS.
000152 96BB       51    311         JNZ     MSTER           JUMP IF BUS ERROR OR NOT-ACKNOWLEDGE.
                        312 *
                        313 *       MASTER TRANSMITTER/RECEIVER WITH POINTER BYTE.
                        314 *       =============================================
                        315 *-------------------------------------------------------------------------
                        316 *               TRANSMISSION OF A REPEATED START CONDITION
                        317 *                                 + SLAVE ADDRESS + READ/WRITE BIT.
                        318 *               RECEPTION OF THE ACKNOWLEDGE BIT.
                        319 *-------------------------------------------------------------------------
                        320 *
000154 8906       52    321 MST3    ORL     P1,#SCL+NXTBN   RELEASE P11/SCL (STILL STRETCHED).
000156 99FB       53    322         ANL     P1,#.NOT.NXTBN  SET SCL TO HIGH.
000158 BAFA       54    323         MOV     R2,#SCLC        LOAD SCL TIME OUT COUNTER.
                        324 *
00015A 09         55    325 MP1     IN      A,P1            FETCH STATUS.
00015B 37         56    326         CPL     A
00015C D264       57    327         JB6     MP2             JUMP IF SCL WENT TO HIGH.
00015E EA5A       58    328         DJNZ    R2,MP1          DECREMENT AND TEST TIME-OUT.
000160 2340       59    329         MOV     A,#ERRF2        LOAD ERROR FLAG.
000162 24BB       60    330         JMP     MSTER           JUMP BECAUSE OF SCL TIME-OUT.
                        331 *
000164 99FE       61    332 MP2     ANL     P1,#.NOT.SDA    OUTPUT REPEATED START CONDITION.
000166 99FD       62    333         ANL     P1,#.NOT.SCL    SET SCL OUTPUT TO LOW.
000168 B921       63    334         MOV     R1,#SLVADR      LOCATION OF SLAVE ADDRESS IN R1.
00016A F1         64    335         MOV     A,@R1           PUT SLAVE ADDRESS + R/WN BIT IN ACCU.
00016B 5400       65    336         CALL    MTDB            TRANSMIT SLAVE ADDRESS + R/WN BIT.
00016D E7         66    337         RL      A               SET FLAGS AT THE RIGHT POSITION.
00016E 67         67    338         RRC     A                "   "   "  "   "   "     "
00016F 96BB       68    339         JNZ     MSTER           JUMP IF BUS ERROR OR ARBITRATION LOST.
000171 546E       69    340         CALL    MTAC            RECEPTION OF THE ACKNOWLEDGE BIT.
000173 2C         70    341         XCH     A,R4
000174 E7         71    342         RL      A               SET ACK FLAG IN THE CORRECT POSITION.
000175 4C         72    343         ORL     A,R4            COLLECT THE FLAGS IN ACCU.
000176 96BB       73    344         JNZ     MSTER           JUMP IF BUS ERROR OR NOT-ACKNOWLEDGE.
000178 FD         74    345         MOV     A,R5            FETCH SIO MODE.
000179 1284       75    346         JB0     MST4            JUMP IF MASTER RECEIVER.
```

```
                             EJECT
                       *------------------------------------------------------------------
                       *              TRANSMISSION OF THE POINTER BYTE.
                       *------------------------------------------------------------------
                       *
00017B B922       76   MTP   MOV    R1,#DBCNTR    LOCATION OF DATA BYTE COUNTER TO R1.
00017D F1         77         MOV    A,@R1         NUMBER OF DATA BYTES IN ACCU.
00017E 17         78         INC    A             INCREMENT NUMBER FOR POINTER BYTE.
00017F AD         79         MOV    R5,A          TOTAL NUMBER IN R5.
000180 B923       80         MOV    R1,#POINTR    LOCATION OF POINTER BYTE TO R1.
000182 24A5       81         JMP    MT1           JUMP TO TRANSMISSION OF DATA BYTES.
                       *------------------------------------------------------------------
                       *
                       *     MASTER RECEIVER WITH OR WITHOUT POINTER-BYTE.
                       *     ============================================
                       *------------------------------------------------------------------
                       *              RECEPTION OF A NUMBER OF DATA BYTES. (NUMBER IN DBCNTR).
                       *              TRANSMISSION OF ACKNOWLEDGE BITS.
                       *              TRANSMISSION OF NOT-ACKNOWLEDGE AFTER LAST DATA BYTE.
                       *------------------------------------------------------------------
                       *
000184 B922       82   MST4  MOV    R1,#DBCNTR    LOCATION OF DBCNTR TO R1.
000186 F1         83         MOV    A,@R1         FETCH DATA-BYTE COUNTER.
000187 AD         84         MOV    R5,A          SAVE DATA-BYTE COUNTER.
000188 B928       85         MOV    R1,#MRDTR1    LOCATION OF FIRST DATA REGISTER IN R1.
00018A 543D       86   MR1   CALL   MRDB          INPUT DATA BYTE.
00018C 96BB       87         JNZ    MSTER         JUMP IF BUS ERROR.
00018E FC         88         MOV    A,R4          FETCH RECEIVED DATA BYTE.
00018F A1         89         MOV    @R1,A         SAVE RECEIVED DATA BYTE.
000190 19         90         INC    R1            NEXT DATA REGISTER LOCATION.
000191 ED99       91         DJNZ   R5,MR2        DECR. AND TEST DATA-BYTE CNTR.
000193 5468       92         CALL   MRNAC         OUTPUT NOT-ACK.
000195 96BB       93         JNZ    MSTER         JUMP IF BUS ERROR.
000197 24B4       94         JMP    MSTSTP        OUTPUT STOP CONDITION.
                       *
000199 5462       95   MR2   CALL   MRAC          OUTPUT ACK. BIT.
00019B 96BB       96         JNZ    MSTER         JUMP IF BUS ERROR.
00019D 248A       97         JMP    MR1           JUMP TO RECEIVE NEXT DATA BYTE.
```

```
                        385         EJECT
                        386 *       MASTER TRANSMITTER WITH OR WITOUT POINTER-BYTE.
                        387 *       ==============================================
                        388 *-------------------------------------------------------------------------
                        389 *             TRANSMISSION OF A NUMBER OF DATA BYTES.(NUMBER IN DBCNTR).
                        390 *             RECEPTION OF ACKNOWLEDGE BITS.
                        391 *-------------------------------------------------------------------------
                        392 *
00019F B922      98     393 MST5    MOV     R1,#DBCNTR      LOCATION OF DATA-BYTE COUNTER TO R1.
0001A1 F1        99     394         MOV     A,@R1           FETCH DATA-BYTE COUNTER.
0001A2 AD       100     395         MOV     R5,A            DATA-BYTE COUNTER TO R5.
0001A3 B924     101     396         MOV     R1,#MTDTR1      LOCATION OF FIRST MST/TRX DATA REG. TO R1.
                        397 *
0001A5 F1       102     398 MT1     MOV     A,@R1           FETCH DATA BYTE TO BE TRANSMITTED.
0001A6 5400     103     399         CALL    MTDB            TRANSMIT DATA BYTE.
0001A8 E7       104     400         RL      A               SET FLAGS IN RIGHT POSITION.
0001A9 67       105     401         RRC     A                "    "    "   "      "
0001AA 96BB     106     402         JNZ     MSTER           JUMP IF BUS ERROR OR IF DATA DISTURBED.
0001AC 546E     107     403         CALL    MTAC            INPUT ACKNOWLEDGE BIT.
0001AE 4C       108     404         ORL     A,R4            COLLECT FLAGS.
0001AF 96BB     109     405         JNZ     MSTER           JUMP IF BUS ERROR OR NOT-ACKNOWLEDGE.
0001B1 19       110     406         INC     R1              NEXT MST/TRX DATA REG. LOCATION.
0001B2 EDA5     111     407         DJNZ    R5,MT1          DECR. AND TEST DATA BYTE COUNTER.
                        408 *
                        409 *       STOP CONDITION.
                        410 *       ===============
                        411 *-------------------------------------------------------------------------
                        412 *             TRANSMISSION OF THE STOP CONDITION.
                        413 *-------------------------------------------------------------------------
0001B4 5474     112     414 MSTSTP  CALL    MSSTP           OUTPUT STOP CONDITION.
0001B6 D2BB     113     415         JB6     MSTER           JUMP IF SCL AND/OR SDA STAY LOW; BUS
                        416 *                               OBSTRUCTED.
0001B8 27       114     417 MT2     CLR     A               CLEAR FLAGS.
0001B9 85       115     418         EN      SI              ENABLE SERIAL INTERRUPT.
0001BA 83       116     419         RET                     RETURN TO MAIN PROGRAM.
                        420 *
                        421 *       ARBITRATION LOST DURING TRANSMISSION OF THE SLAVE ADDRESS.
                        422 *       ========================================================
                        423 *-------------------------------------------------------------------------
                        424 *             IF THE SLAVE SUBROUTINE IS PRESENT AND THE OWN SLAVE ADDRESS
                        425 *             IS RECEIVED, THEN CALL THE SLAVE SUBROUTINE, ELSE GENERATE
                        426 *             THE NINTH CLOCK PULSE AND RELEASE THE BUS.
                        427 *-------------------------------------------------------------------------
                        428         IF SLAVE=USED
                        429 MAL     MOV     R1,A            SAVE FLAGS.
                        430         MOV     A,R4            RECEIVED SLAVE ADDRESS TO ACCU.
                        431         ANL     A,#.NOT.RWB     MASK READ/WRITE BIT.
                        432         XRL     A,#OWNAD        COMPARE WITH OWN SLAVE ADDRESS.
                        433         JZ      MAL1            JUMP IF EQUAL.
                        434         CALL    MTAC            GENERATE NINTH CLOCK PULSE.
                        435         ORL     P1,#H'FF'       RESET SIO HARDWARE, RELEASE SDA.
                        436         ANL     P1,#.NOT.NXTBN  RELEASE SCL.
                        437         JMP     MAL2            PREPARE TO RETURN TO MAIN PROGRAM.
                        438 *
                        439 *
                        440 *
```

```
                        441 *
                        442 MAL1    CALL   SAR             CALL SLAVE SUBROUTINE.
                        443 MAL2    MOV    A,R1            COLLECT FLAGS.
                        444         EN     SI              ENABLE SERIAL INTERRUPT.
                        445         RET                    RETURN TO MAIN PROGRAM.
                        446         ENDIF
                        447 *-------------------------------------------------------------------------
                        448 *
                        449         IF SLAVE=.NOT.USED
                        450 MAL     EQU    $               MASTER ERROR ROUTINE IS ENTERED.
                        451         ENDIF
                        452 *-------------------------------------------------------------------------
                        453 *
                        454 *       MASTER ERROR ROUTINE.
                        455 *       ====================
                        456 *-------------------------------------------------------------------------
                        457 *             IF ARBITRATION IS LOST, THE ACKNOWLEDGE BIT IS
                        458 *             GENERATED.
                        459 *             IN CASE OF A NOT-ACKNOWLEDGE OR A SPURIOUS START
                        460 *             CONDITION, A STOP CONDITION IS GENERATED.
                        461 *             AT A SPURIOUS STOP CONDITION OR A BUS OBSTRUCTION, THE
                        462 *             HARDWARE IS RESET AND SDA AND SCL ARE RELEASED.
                        463 *-------------------------------------------------------------------------
                        464 *
0001BB A9       117     465 MSTER   MOV    R1,A            SAVE FLAGS.
0001BC F2C9     118     466         JB7    MSTER2          JUMP IF ARBITRATION LOST.
0001BE 5323     119     467 MSTER0  ANL    A,#ERRF3+NAACK+NDACK  MASK FLAGS.
0001C0 96CF     120     468         JNZ    MSTER3          JUMP IF STOP COND. MUST BE GENERATED.
0001C2 89FF     121     469 MSTER1  ORL    P1,#H'FF'       RESET SIO HARDWARE, RELEASE SDA.
0001C4 99FB     122     470         ANL    P1,#.NOT.NXTBN  RELEASE SCL.
0001C6 49       123     471         ORL    A,R1            COLLECT FLAGS.
0001C7 85       124     472         EN     SI              ENABLE SERIAL INTERRUPT.
0001C8 83       125     473         RET                    RETURN TO MAIN PROGRAM.
                        474 *
0001C9 546E     126     475 MSTER2  CALL   MTAC            GENERATE ACKNOWLEDGE BIT.
0001CB 49       127     476         ORL    A,R1            COLLECT FLAGS.
0001CC A9       128     477         MOV    R1,A            SAVE FLAGS.
0001CD 24BE     129     478         JMP    MSTER0          JUMP FOR STOP COND. OR RELEASE OF THE BUS.
                        479 *
0001CF 5474     130     480 MSTER3  CALL   MSSTP           OUTPUT STOP CONDITION.
0001D1 D2C2     131     481         JB6    MSTER1          JUMP IF SCL OR/AND SDA STAY LOW. BUS
                        482 *                              OBSTRUCTED.
0001D3 F9       132     483 MSTER4  MOV    A,R1            RESTORE FLAGS.
0001D4 85       133     484         EN     SI              ENABLE SERIAL INTERRUPT.
0001D5 83       134     485         RET                    RETURN TO MAIN PROGRAM.
```

```
                        486         EJECT
0001D6                  487         ORG    H'200'
                        488 *       ##########################################################
                        489 *       #     SINGLE BYTE, ACKNOWLEDGE AND STOP SUBROUTINES      #
                        490 *       ##########################################################
                        491 *
                        492 *       MASTER TRANSMITTER SINGLE DATA BYTE SUBROUTINE.
                        493 *       ==============================================
                        494 *-------------------------------------------------------------------------
                        495 *THIS SUBROUTINE OUTPUTS 8 BITS OF DATA IN MASTER TRANSMITTER MODE.
                        496 * ENTRY: ACCU CONTAINS DATA BYTE TO BE TRANSMITTED.
                        497 * EXIT: A = 0 AND C = 0; DATA IS TRANSMITTED CORRECTLY
                        498 *                        R4 CONTAINS THE DATA BYTE WHICH IS TRANSMITTED.
                        499 *       A = 0 AND C = 1; ARBITRATION LOST. BYTE HAS BEEN COMPLETED
                        500 *                        WITH CLOCK PULSES ON SCL AND SDA RELEASED.
                        501 *       A = STAF; START CONDITION OR STOP CONDITION FOLLOWED BY
                        502 *                 A START CONDITION IS DETECTED.
                        503 *       A = BBFN; STOP CONDITION DETECTED
                        504 *       A = ERRF2; SCL EXCEEDS TIME LIMIT
                        505 *-------------------------------------------------------------------------
000200 BB08      135    506 MTDB   MOV    R3,#8          SET BIT COUNTER
000202 AC        136    507        MOV    R4,A           SAVE DATA BYTE.
000203 BAFA      137    508 MTDB0  MOV    R2,#SCLC       LOAD SCLC TIME-OUT COUNTER.
000205 FC        138    509        MOV    A,R4           FETCH DATA BYTE.
                        510 *
000206 F7        139    511 MTDB1  RLC    A              SHIFT DATA BIT INTO CARRY.
000207 AC        140    512        MOV    R4,A           RESTORE SHIFTED DATA BYTE.
000208 E626      141    513        JNC    MTDB8          JUMP IF DATA BIT = 0.
                        514 *-------------------------------------------------------------------------
                        515 *       TRANSMISSION OF AN "ONE" = HIGH LEVEL ON SDA OUTPUT.
                        516 *-------------------------------------------------------------------------
00020A 8907      142    517 MTDB2  ORL    P1,#SDA+SCL+NXTBN   SET SDA OUTPUT HIGH,
                        518 *                                 RELEASE P11/SCL (BUT STILL STRETCHED).
00020C 99FB      143    519        ANL    P1,#.NOT.NXTBN      SET SCL OUTPUT HIGH.
                        520 *
00020E 09        144    521 MTDB3  IN     A,P1           FETCH STATUS.
00020F D221      145    522        JB6    MTDB7          JUMP IF SCL IS STILL LOW.
000211 99FD      146    523        ANL    P1,#.NOT.SCL   SET SCL OUTPUT TO LOW.
000213 09        147    524        IN     A,P1           FETCH STATUS.
000214 F218      148    525        JB7    MTDB4          JUMP IF "ONE" TRANSMITTED CORRECTLY.
000216 444C      149    526        JMP    MTABL          ARBITRATION LOST. JUMP TO BYTE COMPLETE ROUT.
000218 B21E      150    527 MTDB4  JB5    MTDB6          JUMP IF STA CONDITION RECEIVED.
00021A EB03      151    528        DJNZ   R3,MTDB0       DECR. AND TEST BIT COUNTER.
00021C 4433      152    529        JMP    MTDB10         LAST BIT HAS BEEN TRANSMITTED.
                        530 *
                        531 *
00021E 5330      153    532 MTDB6  ANL    A,#STAF+BBFN   MASK FLAGS.
000220 83        154    533        RET
                        534 *
000221 EA0E      155    535 MTDB7  DJNZ   R2,MTDB3       DECR. AND TEST TIME-OUT COUNTER.
000223 2340      156    536 SCLER  MOV    A,#ERRF2       SET SCL TIME-OUT ERROR FLAG.
000225 83        157    537        RET
```

```
                        538         EJECT
                        539 *
                        540 *-------------------------------------------------------------------------
                        541 *               TRANSMISSION OF A "ZERO"= LOW LEVEL ON THE SDA OUTPUT.
                        542 *               SPURIOUS START AND STOP CONDITIONS ARE NOT POSSIBLE.
                        543 *               NO BIT DISTURBANCE POSSIBLE.
                        544 *-------------------------------------------------------------------------
000226 8906      158    545 MTDB8   ORL     P1,#SCL+NXTBN  RELEASE P11/SCL (BUT STILL STRETCHED).
000228 99FE      159    546         ANL     P1,#.NOT.SDA   SET SDA OUTPUT TO LOW.
00022A 99FB      160    547         ANL     P1,#.NOT.NXTBN SET SCL OUTPUT TO HIGH.
                        548 *
00022C 09        161    549 MTDB9   IN      A,P1           FETCH STATUS.
00022D D239      162    550         JB6     MTDB11         JUMP IF SCL IS STILL LOW.
00022F 99FD      163    551         ANL     P1,#.NOT.SCL   SET SCL OUTPUT TO LOW.
000231 EB03      164    552         DJNZ    R3,MTDB0       DEC.AND TEST BIT COUNTER.
                        553 *
000233 FC        165    554 MTDB10  MOV     A,R4           SHIFT LAST TRANSMITTED BIT INTO DATA BYTE.
000234 F7        166    555         RLC     A                "    "       "       "    "    "    "
000235 AC        167    556         MOV     R4,A             "    "       "       "    "    "    "
000236 27        168    557         CLR     A              CLEAR FLAGS
000237 97        169    558         CLR     C                "    "
000238 83        170    559         RET
                        560 *
000239 EA2C      171    561 MTDB11  DJNZ    R2,MTDB9       DECR. AND TEST SCL TIME OUT COUNTER.
00023B 4423      172    562         JMP     SCLER          JUMP IF SCL STAYS LOW.
```

```
                                563         EJECT
                                564 *
                                565 *       MASTER RECEIVER SINGLE DATA BYTE SUBROUTINE.
                                566 *       ===========================================
                                567 *-------------------------------------------------------------------------------
                                568 *THIS SUBROUTINE INPUTS 8 BITS OF DATA IN MASTER RECEIVER MODE.
                                569 * EXIT: A = 0; DATA IS RECEIVED CORRECTLY
                                570 *               R4 CONTAINS THE DATA BYTE WHICH IS RECEIVED
                                571 *        A = STAF; START CONDITION OR STOP CONDITION FOLLOWED BY
                                572 *                  A START CONDITION IS DETECTED.
                                573 *        A = BBFN; STOP CONDITION DETECTED
                                574 *        A = ERRF2; SCL EXCEEDS TIME LIMIT
                                575 *-------------------------------------------------------------------------------
                                576 *
00023D BB08          173       577 MRDB    MOV     R3,#8           SET BIT COUNTER
00023F 8901          174       578 MRDB0   ORL     P1,#SDA         SET SDA OUTPUT TO HIGH.
000241 BAFA          175       579 MRDB1   MOV     R2,#SCLC        LOAD SCL TIME-OUT COUNTER.
000243 8906          176       580         ORL     P1,#SCL+NXTBN   RELEASE P11/SCL (BUT STILL STRETCHED).
000245 99FB          177       581         ANL     P1,#.NOT.NXTBN SET SCL OUTPUT TO HIGH.
                                582 *
000247 09            178       583 MRDB2   IN      A,P1            FETCH STATUS.
000248 D25A          179       584         JB6     MRDB3           JUMP IF SCL STILL LOW.
00024A 99FD          180       585         ANL     P1,#.NOT.SCL    SET SCL OUTPUT TO LOW.
                                586 *-------------------------------------------------------------------------------
                                587 *           FROM HERE THE BYTE COMPLETE ROUTINE (ARBITRATION LOST)
                                588 *           SHARES THE MST/REC DATA BYTE SUBROUTINE.
                                589 *-------------------------------------------------------------------------------
00024C F7            181       590 MTABL   RLC     A               SHIFT RECEIVED DATA BIT IN CARRY BIT.
00024D FC            182       591         MOV     A,R4            DATA BYTE TO ACCU.
00024E F7            183       592         RLC     A               SHIFT RECEIVED BIT INTO DATA BYTE.
00024F AC            184       593         MOV     R4,A            SAVE DATA BYTE.
000250 09            185       594         IN      A,P1            FETCH STATUS.
000251 5330          186       595         ANL     A,#STAF+BBFN    MASK FLAGS.
000253 965E          187       596         JNZ     MRDB4           JUMP IF START OR STOP CONDITION DETECTED.
000255 EB41          188       597         DJNZ    R3,MRDB1        DECR. AND TEST BIT COUNTER.
000257 97            189       598         CLR     C
000258 A7            190       599         CPL     C               SET CARRY TO INDICATE ARBITRATION LOST.
000259 83            191       600         RET
                                601 *
00025A EA47          192       602 MRDB3   DJNZ    R2,MRDB2        DECR. SCL TIME-OUT COUNTER AND TEST.
00025C 4423          193       603         JMP     SCLER           JUMP IF SCL STAYS LOW.
                                604 *
00025E 2C            194       605 MRDB4   XCH     A,R4            EXCHANGE FLAGS AND DATA BYTE.
00025F 67            195       606         RRC     A               DELETE LAST RECEIVED BIT FROM DATA BYTE.
000260 2C            196       607         XCH     A,R4            SAVE DATA BYTE IN R4.
000261 83            197       608         RET
```

```
                        609         EJECT
                        610 *       MASTER RECEIVER ACKNOWLEDGE SUBROUTINE.
                        611 *       ======================================
                        612 *-------------------------------------------------------------------------
                        613 *THIS SUBROUTINE OUTPUTS AN ACKNOWLEDGE (LOW) IN MASTER REC. MODE.
                        614 * EXIT: A = 00; ACKNOWLEDGE IS TRANSMITTED CORRECTLY.
                        615 *       A = ERRF2; SCL EXCEEDS TIME LIMIT
                        616 *-------------------------------------------------------------------------
000262 BB01      198    617 MRAC    MOV     R3,#1           SET BIT COUNTER.
000264 BAFA      199    618         MOV     R2,#SCLC        LOAD SCL TIME-OUT COUNTER.
000266 4426      200    619         JMP     MTDB8           JUMP TO MST/TRX MODE.
                        620 *
                        621 *
                        622 *       MASTER RECEIVER NOT-ACKNOWLEDGE SUBROUTINE.
                        623 *       ==========================================
                        624 *-------------------------------------------------------------------------
                        625 *THIS SUBROUTINE OUTPUTS A NOT-ACKNOWLEDGE (HIGH) IN MASTER REC. MODE
                        626 * EXIT: A = 00; NOT-ACKNOWLEDGE IS TRANSMITTED CORRECTLY.
                        627 *       A = STAF; START CONDITION OR STOP CONDITION FOLLOWED BY
                        628 *                 A START CONDITION IS DETECTED.
                        629 *       A = BBFN; STOP CONDITION DETECTED
                        630 *       A = ERRF2; SCL EXCEEDS TIME LIMIT
                        631 *-------------------------------------------------------------------------
000268 BB01      201    632 MRNAC   MOV     R3,#1           SET BIT COUNTER.
00026A BAFA      202    633         MOV     R2,#SCLC        LOAD SCL TIME-OUT COUNTER.
00026C 440A      203    634         JMP     MTDB2           JUMP TO MST/TRX MODE.
                        635 *
                        636 *
                        637 *       MASTER TRANSMITTER INPUT ACKNOWLEDGE SUBROUTINE.
                        638 *       ================================================
                        639 *-------------------------------------------------------------------------
                        640 * EXIT: A = 00; ACKNOWLEDGE BIT IS RECEIVED CORRECTLY
                        641 *               R4 = 0; ACKNOWLEDGE (LOW) RECEIVED
                        642 *               R4 = 1; NOT-ACKNOWLEDGE (HIGH) RECEIVED
                        643 *       A = STAF; START CONDITION OR STOP CONDITION FOLLOWED BY
                        644 *                 A START CONDITION IS DETECTED.
                        645 *       A = BBFN; STOP CONDITION DETECTED
                        646 *       A = ERRF2; SCL EXCEEDS TIME LIMIT
                        647 *-----------------------------------------------------------------
00026E BB01      204    648 MTAC    MOV     R3,#1           SET BIT COUNTER.
000270 BC00      205    649         MOV     R4,#0           CLEAR DATA REGISTER.
000272 443F      206    650         JMP     MRDB0           JUMP TO MST/REC MODE.
```

```
                        651         EJECT
                        652 *
                        653 *       MASTER STOP CONDITION SUBROUTINE.
                        654 *       ================================
                        655 *-------------------------------------------------------------------------------
                        656 * SUBROUTINE OUTPUTS STOP CONDITION IN MASTER MODE.
                        657 * EXIT: A = BBFN OR STAF; STOP CONDITION IS TRANSMITTED CORRECTLY.
                        658 *       A = ERRF2; SCL AND/OR SDA STAY LOW AND EXCEED THE TIME LIMIT.
                        659 *-------------------------------------------------------------------------------
000274 99FE      207    660 MSSTP   ANL     P1,#.NOT.SDA    SET SDA OUTPUT TO LOW
000276 890E      208    661         ORL     P1,#SCL+NXTBN+STCHN   RELEASE P11/SCL; DISABLE STRETCHING.
000278 99FB      209    662         ANL     P1,#.NOT.NXTBN SET SCL OUTPUT TO HIGH.
00027A BAFA      210    663         MOV     R2,#SCLC        LOAD SCL TIME-OUT COUNTER.
                        664 *
00027C 09        211    665 MSSTP1  IN      A,P1            FETCH STATUS.
00027D D286      212    666         JB6     MSSTP2          JUMP IF SCL STILL LOW.
00027F 8901      213    667         ORL     P1,#SDA         OUTPUT STOP CONDITION.
000281 09        214    668         IN      A,P1            FETCH STATUS.
000282 5330      215    669         ANL     A,#BBFN+STAF    MASK FLAGS.
000284 968C      216    670         JNZ     MSSTP3          JUMP IF A STOP COND. IS TRANSMITTED. A NEW
                        671 *                               START COND. MAY BE RECEIVED.
                        672 *
000286 EA7C      217    673 MSSTP2  DJNZ    R2,MSSTP1       DECR. AND TEST SCL TIME-OUT COUNTER.
000288 8901      218    674         ORL     P1,#SDA         RELEASE SDA.
00028A 2340      219    675         MOV     A,#ERRF2        SET ERROR FLAG; SCL AND/OR SDA STAY LOW.
00028C 83        220    676 MSSTP3  RET
                        677 *-------------------------------------------------------------------------------
```

```
                        678         EJECT
                        679 *
                        680         IF SLAVE=.NOT.USED
                        681 *
                        682 ********************************************************
                        683 *       THE NEXT SUBROUTINE SIMULATES THE SLAVE        *
                        684 *       FUNCTION. IT IS DELETED IF A SLAVE SUBROUTINE  *
                        685 *       IS PRESENT.                                    *
                        686 ********************************************************
                        687 *                                                      *
                        688 *       THE SUBROUTINE STARTS AT THE SERIAL INTERRUPT  *
                        689 *       VECTOR AND RELEASES THE SCL LINE IF IT IS      *
                        690 *       STRETCHED AFTER RECEPTION OF A START CONDITION *
                        691 *       FROM ANOTHER MASTER. THIS SUBROUTINE IS        *
                        692 *       ENTERED EITHER BY THE SERIAL INTERRUPT OR FROM *
                        693 *       THE MULTI-MASTER SUBROUTINE.                   *
                        694 *       THE LAST CASE HAPPENS IF AFTER THE SERIAL      *
                        695 *       INTERRUPT IS DISABLED AND BEFORE A START       *
                        696 *       CONDITION IS TRANSMITTED, ANOTHER MASTER       *
                        697 *       OCCUPIES THE BUS.                              *
                        698 *                                                      *
                        699 *--------------------------------------------------------*
00028D                  700         ORG     5               SERIAL INTERRUPT VECTOR. *
000005 0410      221    701 SIV     JMP     RSCL                                    *
000007                  702         ORG     H'10'                                   *
000010 89FF      222    703 RSCL    ORL     P1,#H'FF'       RESET SIO HARDWARE.     *
000012 99FB      223    704         ANL     P1,#.NOT.NXTBN FINISH STRETCHING SCL    *
000014 93        224    705         RETR                                            *
                        706 *--------------------------------------------------------*
                        707 ********************************************************
                        708 *
                        709         ENDIF
000015                  710         END
```

NO ERRORS DETECTED

Section 4 - 84CXXX Family and Derivatives

Single-chip 8-bit microcontroller family specification

PCF84CXXX

CONTENTS

Section

CONTENTS

Section

Single-chip 8-bit microcontroller family specification

PCF84CXXX

1 INTRODUCTION

This family data sheet describes the microcontroller core which is common for all members of the PCF84CXXX family. For complete information of a particular microcontroller, consult both the specific microcontroller data sheet and this family data sheet.

2 FEATURES

- 8-bit CPU, ROM, RAM, I/O in a single package
- Up to 8 K bytes ROM
- Up to 256 bytes RAM
- Over 80 instructions, all instructions 1 or 2 cycles
- 8 or more quasi bi-directional I/O port lines
- 8-bit programmable timer/event counter
- 3 single-level vectored interrupts (external, 8-bit programmable timer/counter, SIO/derivative)
- 2 Test inputs: T0 (may also be used as an interrupt) and T1 (may also be used as an input to an 8-bit counter)
- Serial I/O interface (not all devices)
- Power-on-reset
- 2 power reduction modes: Idle and Stop
- V_{DD} supply range: 2.5 V to 5.5 V
- Clock frequency range: 450 kHz to 10 MHz
- Operating temperature range: −40 °C to 85 °C
- Silicon gate CMOS fabrication process

3 GENERAL DESCRIPTION

The PCF84CXXX single-chip 8-bit microcontroller family consists of a wide range of derivatives containing up to 8 K bytes of on-chip mask programmable program ROM and up to 256 bytes RAM. All devices include flexible I/O ports, an 8-bit programmable timer/event counter and a choice of single-level vectored interrupts. The instruction set is based on that of the MAB8048.
A number of PCF84CXXX family members may be used as CMOS replacements for their NMOS counterparts, where lower power consumption and higher speed are required.

The family is well supported with:

- Cross assemblers
- In-circuit emulation tools
- Window debugger
- Piggy-back versions for prototyping.

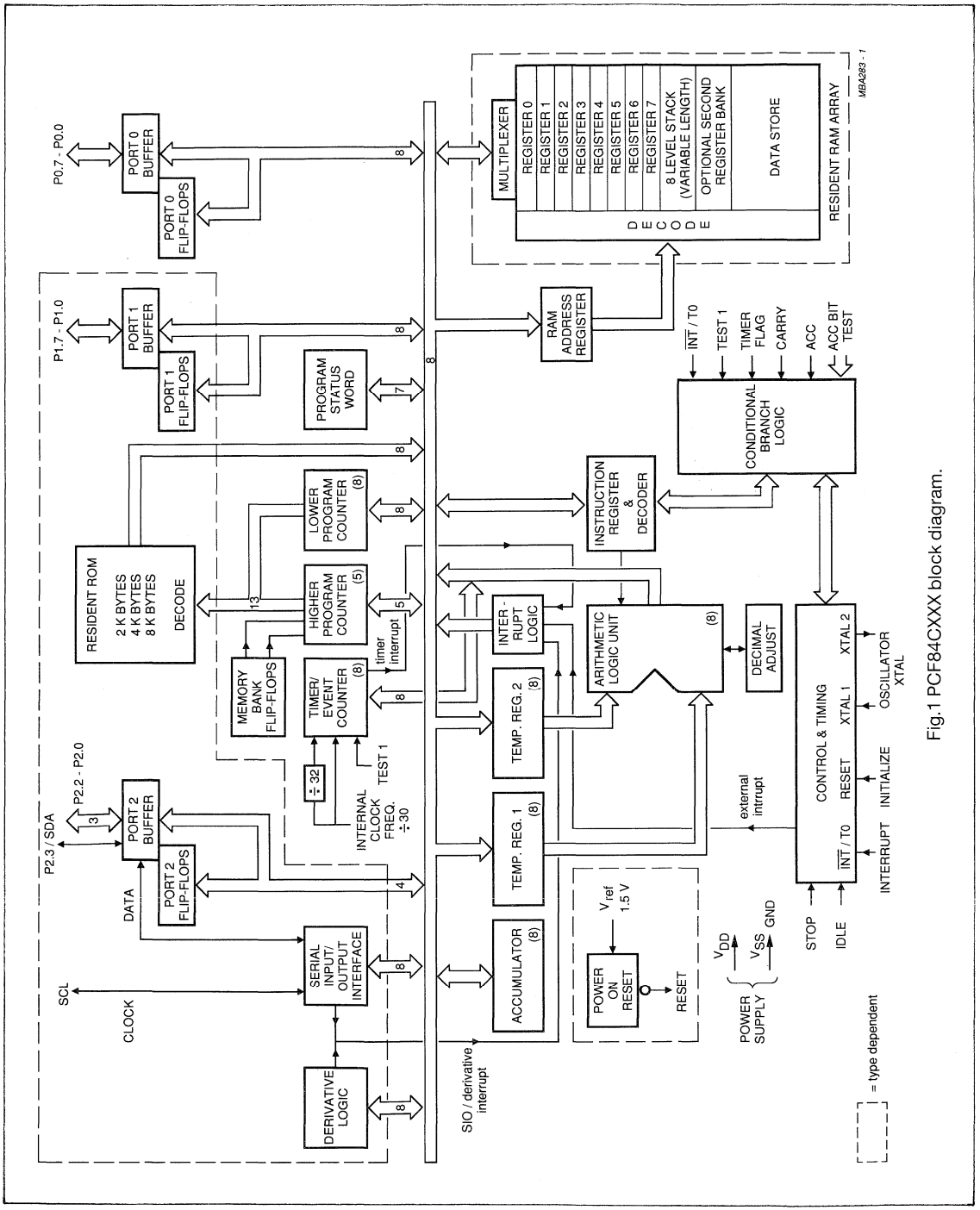


Fig.1 PCF84CXXX block diagram.

4 FUNCTIONAL DESCRIPTION

4.1 Program memory

Program memory consists of up to 8 K bytes of read-only memory (ROM). Each location is directly addressable by the program counter. The program memory is mask-programmed at the factory. Figure 2 shows the program memory map.

Four program memory locations are of special importance:

- Location 0: first instruction to be executed after the microcontroller is reset.
- Location 3: first instruction of an external interrupt ($\overline{INT}$/T0) service routine.
- Location 5: first instruction of an SIO/derivative interrupt service routine.
- Location 7: first instruction of a timer/event counter interrupt service routine.

Of the 13-bits in the program counter, only 11 function as a counter. The two most significant bits are preset by SEL MB instructions. Thus, program memory is arranged in banks of 2 K bytes. Memory bank boundaries can only be crossed using unconditional branches (JMP) or subroutine calls (CALL) after the appropriate memory bank has been selected by a SEL MB instruction.

Each program memory bank is further divided into 8 pages of 256 bytes. Indirect (JMPP) and conditional branches can't cross page boundaries.

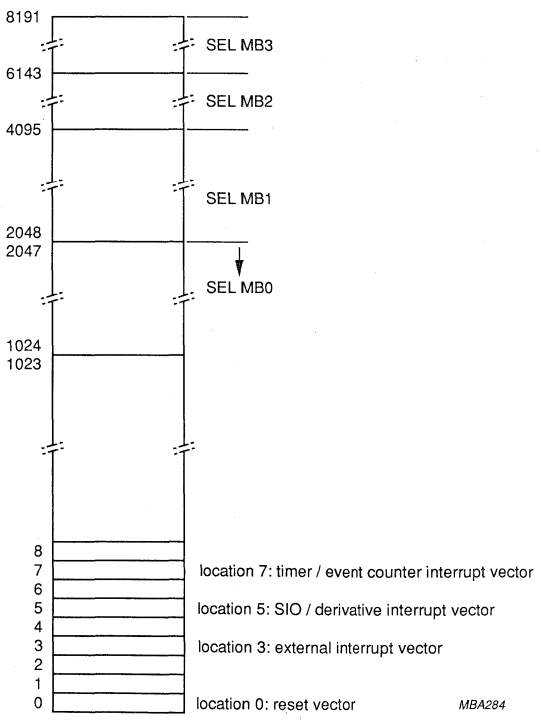


Fig.2 Program memory map.

4.2 Data memory

Data memory consists of up to 256 bytes of random access memory (RAM). All locations are indirectly addressable using RAM pointer registers. Up to 16 register locations are directly addressable. Memory also includes an 8-level program counter stack addressed by a 3-bit stack pointer. All RAM locations make efficient program loop counters when used with the 'decrement register and test' instruction DJNZ. Figure 3 shows the data memory map.

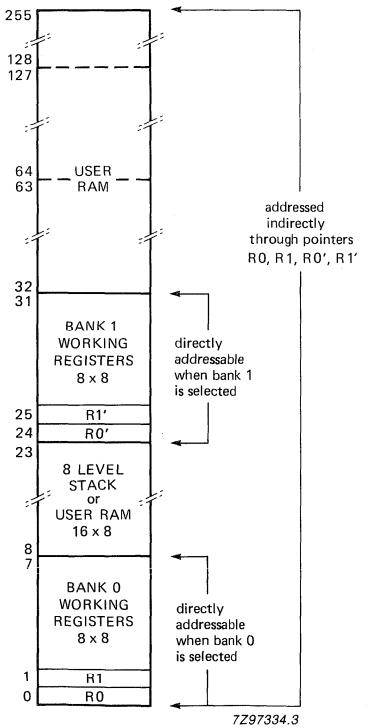


Fig.3 Data memory map.

4.2.1 Working registers

Locations 0 to 7 may be selected as working registers by the SEL RB0 instruction. These locations may then be accessed using efficient one byte, one cycle instructions. Ease of addressing, and a minimum requirement of instruction bytes to manipulate their contents make the working registers ideal for frequently accessed intermediate results.

Instead of locations 0 to 7, locations 24 to 31 may be selected as working registers by the SEL RB1 instruction. Register bank 1 may be used as an extension of register bank 0, as an alternative register bank for use by interrupt service routines, or as general purpose data memory.

The first two locations of each bank contain the RAM pointer registers (R0, R1, R0' and R1') which indirectly address all RAM locations.

4.2.2 Program counter stack

Locations 8 to 23 may be used as an 8-level program counter stack reserving 2 locations per level, or as general purpose RAM. The stack (Fig.4) saves return addresses and status during interrupt or subroutine servicing. Up to 8 levels of subroutine/interrupt nesting is possible.

A 3-bit stack pointer always points to the next free stack level. Following a device reset, the stack pointer points to level 0 (locations 8 and 9). On each subroutine call or interrupt, the contents of the 13-bit program counter and bits 4, 6 and 7 of the program status word are transferred to the level indicated by the stack pointer. The stack pointer increments and points to the next free level. Overflow from level 7 to level 0 occurs after nesting eight levels deep. Further subroutine calls and/or interrupts should be avoided since the contents of level 0 would be overwritten and lost.

The RETR instruction must be used to terminate an interrupt service routine. The RETR instruction decrements the stack pointer, and restores the program counter and status word. A subroutine should be terminated with the RET instruction. The RET instruction restores the program counter but does not restore the program status word.

As a general rule, the use of RETR in conjunction with a subroutine call is not recommended. RETR must not be used to terminate a subroutine which has been called within an interrupt routine since it would prematurely terminate the interrupt routine.

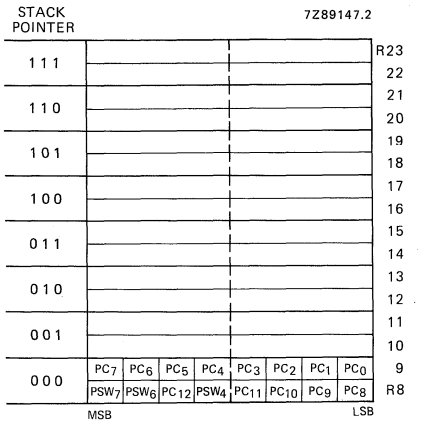


Fig.4 Program counter stack.

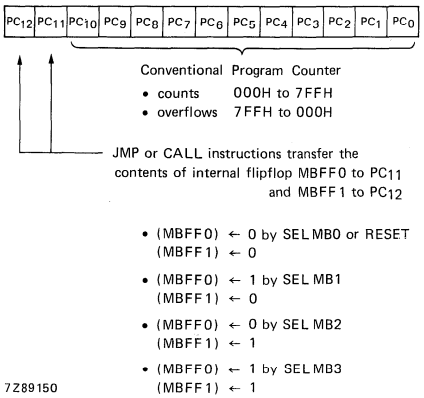


Fig.5 Program counter.

4.3 Program counter

The 13-bit program counter can address up to 8 K bytes of ROM (Fig.5). The least significant 11 bits (PC0 to PC10) are auto-incrementing. The two most significant bits, PC11 and PC12, are loaded with the contents of two internal flip flops (MBFF0 and MBFF1 respectively) when a JMP or CALL instruction is executed. The contents of these two internal flip-flops can be altered by the SEL MB instructions.

4.4 Program status word

The program status word (PSW) is an 8-bit register in the CPU which stores information about the current status of the microcontroller (Fig.6).

All bits can be read using the MOV A,PSW instruction. Only the PS bit can be written using the MOV PSW,A instruction. Table 1 describes the PSW bits and how they are affected.

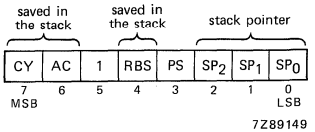


Fig.6 Program status word.

Table 1 Program status word.

BIT	NAME	FUNCTION	AFFECTED BY
7	CY	Carry, signals accumulator overflow	ADD, ADDC, DA, RLC, RRC, CLR C and CPL C instructions
6	AC	Auxiliary Carry, half carry	ADD and ADDC instructions
5	-	Not used, always 1 when read	
4	RBS	Register Bank Select 0: select register bank 0 1: select register bank 1	SEL RB instructions
3	PS	Timer Prescaler Select 0: prescaler selected (÷32) 1: prescaler not selected (÷1)	MOV PSW,A instruction
2 1 0	SP2 SP1 SP0	Stack Pointer bits 2-0	CALL, RET, RETR instructions and interrupts

Table 2 Interrupts.

INTERRUPT			INSTRUCTION	
PRIORITY	TYPE	VECTOR LOCATION	ENABLE	DISABLE
1 (highest)	external	003 (ROM)	EN I	DIS I
2	SIO/derivative interrupt	005 (ROM)	EN SI	DIS SI
3 (lowest)	timer/event counter	007 (ROM)	EN TCNTI	DIS TCNTI

4.5 Central processing unit

The PCF84CXXX instruction set provides arithmetic, logical, branching, input/output, and control facilities. The DA A, SWAP A, and XCHD instructions simplify BCD arithmetic and nibble handling. The MOVP A,@A instruction enables efficient table look-up within the current ROM page. The instruction set also has facilities for conditional branching and loop control (DJNZ).

4.6 Interrupts

The PCF84CXXX family handles external, SIO/derivative and timer/event counter interrupts. The interrupt mechanism is single level. An executing interrupt routine can only be interrupted by a hardware RESET; it can't be interrupted by other interrupts (which are latched).
If several interrupt requests are detected simultaneously, they are serviced in order of priority (see Table 2).

An interrupt request will only be serviced if the corresponding interrupt enable flag is set (Fig.7). When a request is serviced, the contents of the program counter, and bits 4, 6 and 7 of the program status word are saved on the program counter stack. The program counter is loaded with the appropriate interrupt vector which points to the start of the interrupt service routine. Since the accumulator is not automatically saved, it must be saved and restored by user software. The interrupt routine must be terminated by the RETR (return and restore) instruction. At least one instruction in the main program will then be executed before another interrupt routine is serviced.

To avoid erroneous real-time programs, a few words of caution:

- While the interrupt is in progress, the two most significant bits of the program counter are frozen at zero. Thus, interrupt routines and subroutines called from interrupt routines must reside (entirely) in memory bank 0.
- The SEL MB instruction should not be used within interrupt routines or in subroutines called within interrupt routines because altering the contents of MBFF0 and MBFF1 (Fig.5) may lead to erroneous JMP and CALL destinations after return from interrupt.
- Subroutines and nested subroutines called within an interrupt routine must all end with RET since RETR clears the IIP (interrupt in progress) flag (Fig.7 and Fig.8). Further pending interrupts would then interfere with the interrupt routine in progress.

4.6.1 External interrupt

A HIGH-to-LOW transition on the $\overline{INT}$/T0 pin is latched in the digital filter latch if the LOW state exceeds 7 clock periods after a HIGH state of more than 4 clock periods. If the external interrupt is enabled, then the external interrupt flag (EIF) will also be asserted, constituting a valid external interrupt request. As soon as the IIP is clear, indicating that no interrupt routine is in progress, the external interrupt is invoked by a forced CALL to location 3. The EIF flag is simultaneously cleared (Fig.7 and Fig.8). The interrupt routine may acknowledge the interrupt via port lines.

Execution of a DIS I (disable external interrupt) instruction cancels a stored interrupt request by clearing both the digital filter latch and the EIF.

EI	External Interrupt
SI	SIO/Derivative Interrupt
TI	Timer/event counter Interrupt
EIF	External Interrupt Flag
TIF	Timer Interrupt Flag
PIN	Pending Interrupt Not
IIP	Interrupt In Progress Flag

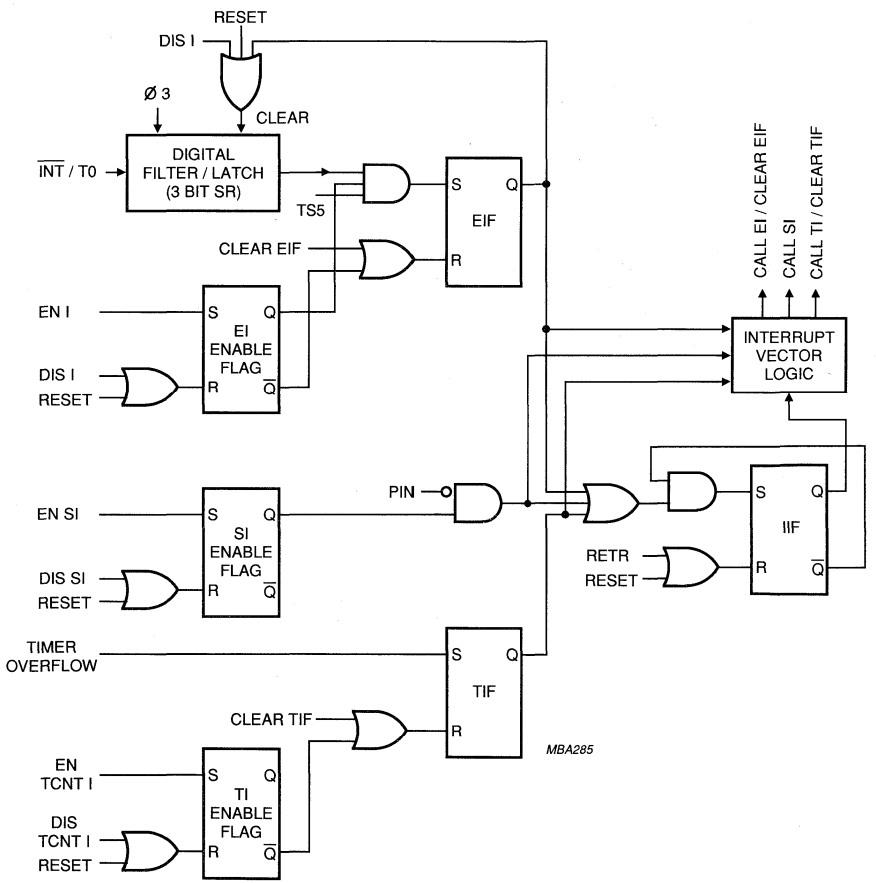
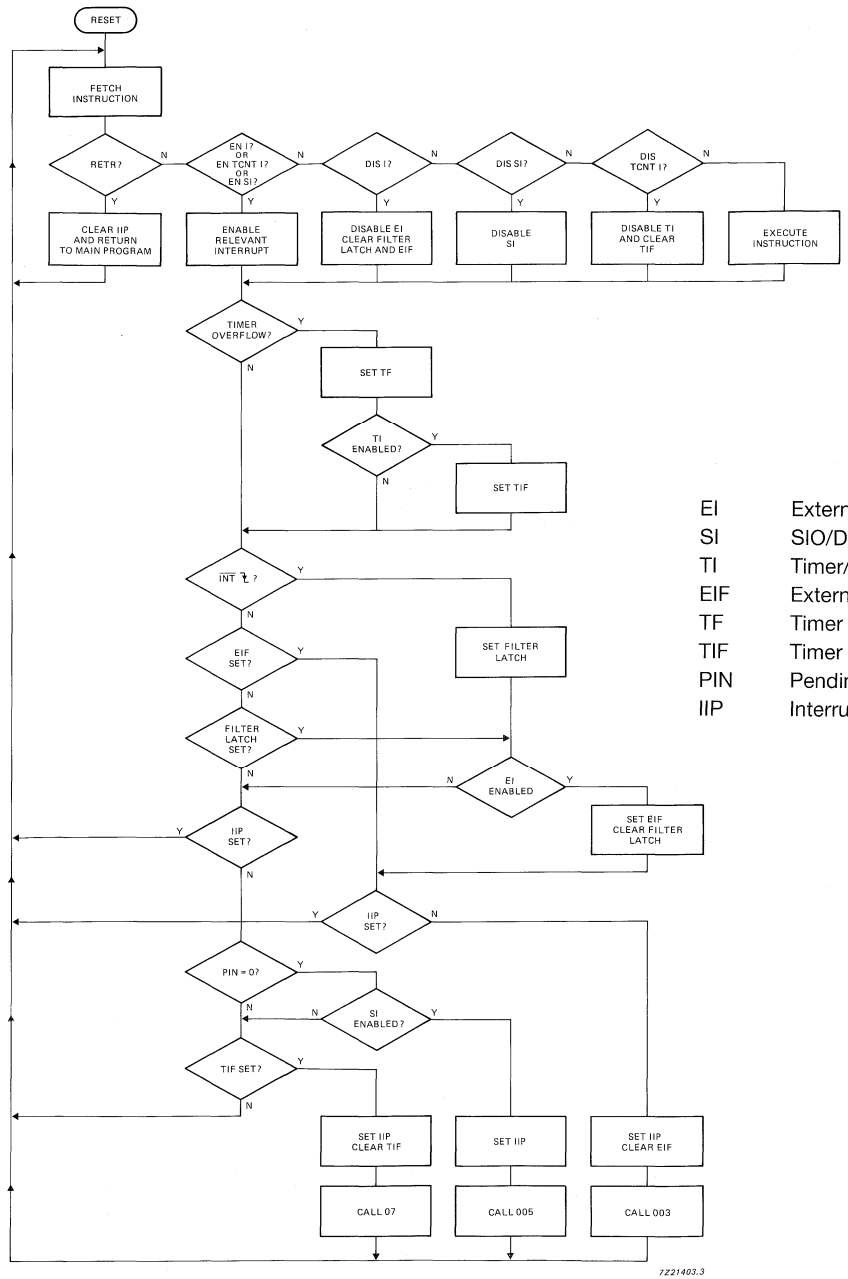


Fig.7 Simplified interrupt logic schematic (the R input overrules the S input for all flags).



EI	External Interrupt
SI	SIO/Derivative Interrupt
TI	Timer/event counter Interrupt
EIF	External Interrupt Flag
TF	Timer Flag
TIF	Timer Interrupt Flag
PIN	Pending Interrupt Not
IIP	Interrupt In Progress Flag

Fig.8 Flow chart illustrating CPU control in the presence of interrupts.

4.6.2 SIO/Derivative interrupt

The SIO/derivative interrupt is shared between the serial I/O interface (if available) and the derivative logic (if available). Software polling may be necessary to determine the origin of a request.

An interrupt request from the SIO or derivative logic will force the PIN flag to its active LOW state. This action is independent of the Enable SIO interrupt enable flag. If the SIO/derivative interrupt is enabled and no interrupt routine is in progress, the active LOW PIN flag will invoke the SIO/derivative interrupt routine by forcing a CALL to program memory location 5. Invoking the SIO/derivative interrupt routine does not automatically clear the PIN flag. PIN must be cleared to its inactive HIGH state by software during the interrupt routine.

More details on SIO interrupts are given in the section on the serial I/O interface. For specific information on derivative interrupts, consult the relevant data sheet.

4.6.3 Timer/event counter interrupt

When the timer/event counter interrupt is enabled, a timer/event counter overflow sets the Timer Interrupt Flag (TIF). As soon as the Interrupt in Progress (IIP) flag is clear (indicating that no interrupt routine is in progress), the timer/event counter interrupt routine is invoked by a forced CALL to program memory location 7, and the TIF flag is simultaneously cleared (Fig.7 and Fig.8).

Execution of a DIS TCNTI (disable timer/event counter interrupt) instruction cancels a stored interrupt request by clearing the TIF flag.

An additional external interrupt may be simulated by enabling the timer/event counter (EN TCNTI) and loading the counter with FFH (one less than overflow). If the event counter mode is enabled by executing the STRT CNT instruction, a rising edge on the T1 input will cause a counter overflow and set TIF.

4.7 Interrupt/Test 0 input ($\overline{INT}$/T0)

The $\overline{INT}$/T0 input may be used as:

- an external interrupt
- a test 0 input for branch instructions JT0 and JNT0.

When used as a test 0 input (external interrupt disabled):

- the conditional branch instruction JT0 will cause a branch if $\overline{INT}$/T0 = logic 1
- the conditional branch instruction JNT0 will cause a branch if $\overline{INT}$/T0 = logic 0.

There is no internal pull-up or pull-down resistor connected to the $\overline{INT}$/T0 input. When $\overline{INT}$/T0 is not used, it must be tied to V_{DD} or V_{SS}.

4.8 Timer/event counter

The internal 8-bit up-counter may be configured to count external events, modulo-32 machine cycles, or machine cycles directly. The MOV A,T and MOV T,A instructions can be used to read and preset the counter.

After a STRT T (start timer) instruction, the counter will increment either every machine cycle (30 oscillator periods) or every 32 machine cycles. If the PS bit in the program status word is set, the counter increments every machine cycle. If the PS bit is reset, it increments every 32 machine cycles. STRT T clears the prescaler which is not otherwise accessible (see Fig.9).

After a STRT CNT (start event counter) instruction, the counter will count each LOW-to-HIGH transition on pin T1 provided that the HIGH state exceeds 4 oscillator periods after a LOW state of more than 4 oscillator periods. The maximum count rate is one increment per machine cycle ($f_{XTAL}/30$).

The timer/event counter is inhibited after reset or by executing a STOP TCNT (stop timer/event counter) instruction (see Fig.9).

When a T1 overflow occurs:

- if the timer/event counter interrupt is enabled, the Timer Interrupt Flag (TIF) is set and an interrupt request is generated
- the Timer Flag (TF) is set. TF can be tested by conditional branch instructions JTF (jump if TF = logic 1) and JNTF (jump if TF = logic 0). When a JTF or JNTF instruction is executed, TF is reset. The only other way to clear TF is to reset the microcontroller.

Table 3 Timer/event counter control.

FUNCTION	TIMER MODE	COUNTER MODE
clear	MOV T,A (A) = 0 or RESET	MOV T,A (A) = 0 or RESET
preset	MOV T,A	MOV T,A
start	STRT T	STRT CNT
stop	STOP TCNT or RESET	STOP TCNT or RESET
test	JTF/JNTF	JTF/JNTF
read	MOV A,T	MOV A,T

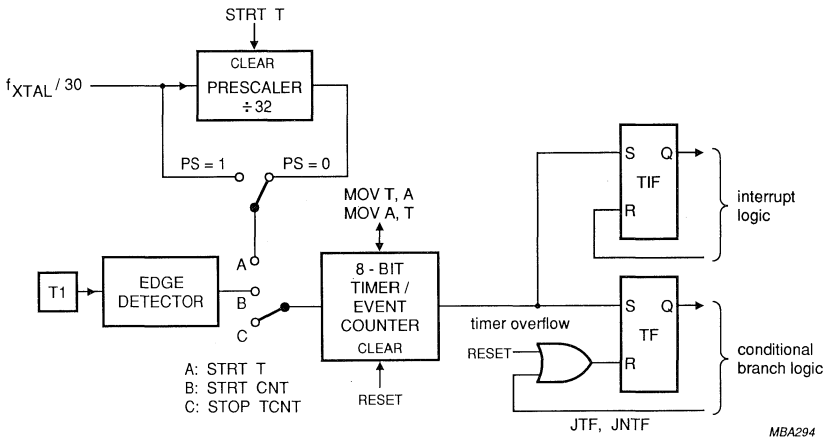


Fig.9 Timer/event counter.

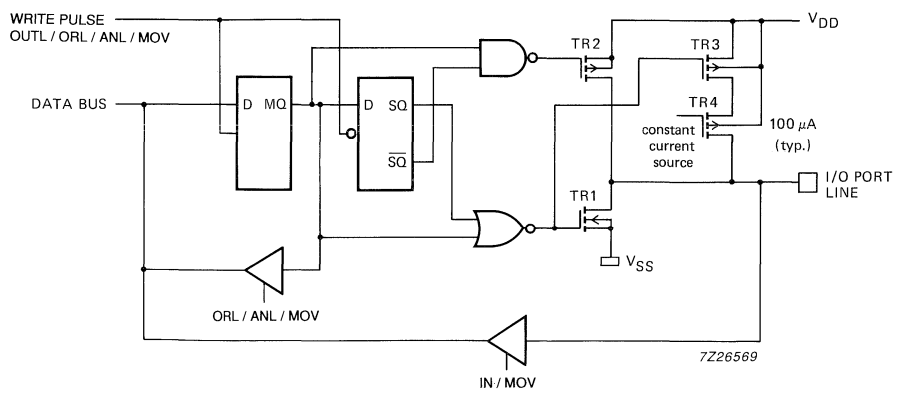


Fig.10 Standard port with switched pull-up current source (mask option 1).

4.9 Test 1/Count input (T1)

The T1 input may be used as:
- a count input to the 8-bit timer/event counter
- a test 1 input for branch instructions JT1 and JNT1.

When used as a test input:
- the conditional branch instruction JT1 will cause a branch if T1 = logic 1
- the conditional branch instruction JNT1 will cause a branch if T1 = logic 0.

There is no internal pull-up or pull-down resistor connected to the T1 input. When T1 is not used, it must be tied to V_{DD} or V_{SS}.

4.10 Parallel ports

The PCF84CXXX family provides up to three standard quasi-bidirectional I/O ports:
- Port 0: 8-bit parallel port (P0.0 to P0.7)
- Port 1: 8-bit parallel port (P1.0 to P1.7)
- Port 2: 4-bit parallel port (P2.0 to P2.2, P2.3/SDA)

Several members of the PCF84CXXX family provide all 20 I/O lines and all members contain port 0.

In addition to the standard ports, many PCF84CXXX microcontrollers provide a variety of derivative ports. For specific details, consult the relevant data sheet.

In general, all parallel ports lines can be individually configured as outputs or inputs. Output data written to a port is latched and remains unchanged until rewritten. Input data is not latched and must be stable when read by an input instruction.

The standard port configuration is shown in Fig.10. When a logic 0 is written to the master/slave flip-flop, TR2 and TR3 are turned off, turning off the constant current source. Current sinking is provided by TR1 which is simultaneously turned on, and the output is pulled LOW to V_{SS}.

When a logic 1 is written to the master/slave flip-flop for the first time (MQ = 1, SQ = 0), TR1 is turned off and TR3 is turned on. TR2 is also switched on for the duration of the internal write pulse (one oscillator period), driving the output rapidly to V_{DD}. TR3 turns on the constant current source TR4 which sources sufficient current for a TTL HIGH level; however, the port line can be pulled LOW by an external CMOS device, enabling the same pin to be used for both input and output. This arrangement also facilitates wired-OR applications. Subsequent writing of a logic 1 to the port line will not switch TR2 on. Booster transistor TR2 is only turned on for one oscillator period during a 0-to-1 transition at the output of the master/slave flip-flop.

To use the port line as an input, a logic one must written to the port line to turn TR1 off.

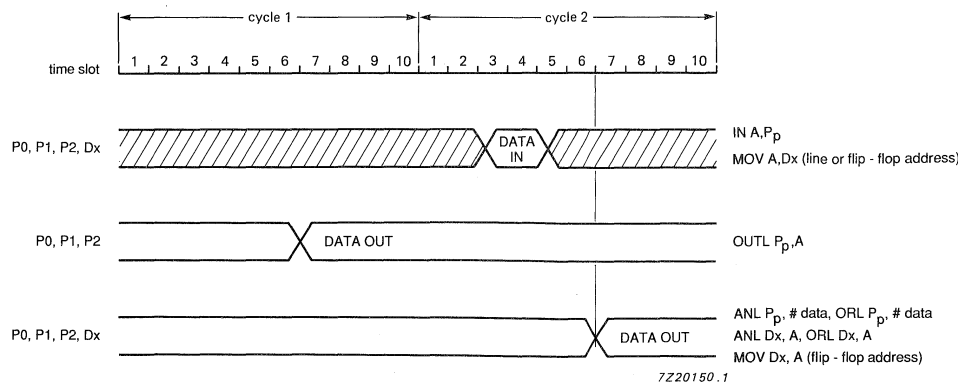


Fig.11 Input/output timing of standard and derivative ports.

Table 4 Derivative port addressing.

DERIVATIVE ADDRESS	TYPE	ACCESS
8-bit line address	R	derivative port line
8-bit flip-flop address	R/W	derivative port flip-flop

Ports 0, 1 and 2 are accessed using the parallel input/output instructions IN, OUTL, ANL and ORL. IN inputs port data to the accumulator. OUTL outputs accumulator data to the port. ANL and ORL are used to manipulate data in the port flip-flops.

Derivative ports are accessed by the derivative input/output instructions MOV, ANL and ORL. ANL and ORL are used to manipulate data in the port flip-flops. MOV is used for all data transfers between the port and the accumulator. When reading data into the accumulator, the data source can be a port line or the port flip-flop. Thus, two derivative addresses are provided per port (see Table 4).

All standard and derivative port accesses are performed by two-cycle instructions. Instruction timing is shown in Fig.11. For input operations, data is read during time slots 3 and 4 of machine cycle 2. For output operations, the data is written during time slot 7. For OUTL, data is written during machine cycle 1. For ANL, ORL and MOV Dx,A, data is transferred during machine cycle 2.

Three I/O mask options make it possible for every parallel I/O port line to be individually configured as follows:

Option 1 Standard Port: quasi-bidirectional I/O with switched pull-up current source (100 µA typ.) and p-channel booster transistor TR2. TR2 is only active for 1 clock cycle during 0-to-1 transitions (Fig.10).

Option 2 Open Drain: quasi-bidirectional I/O with only an N-channel open drain output. Application as an output requires the connection of an external pull-up resistor (Fig.12). If unused, an option 2 output should be tied to V_{SS} to prevent undesirable current flow through input stages.

Option 3 Push-Pull: outputs can sink or source 2 mA (typ.) at V_{DD} = 3 V. Push-pull lines may not be used as inputs (Fig.13).

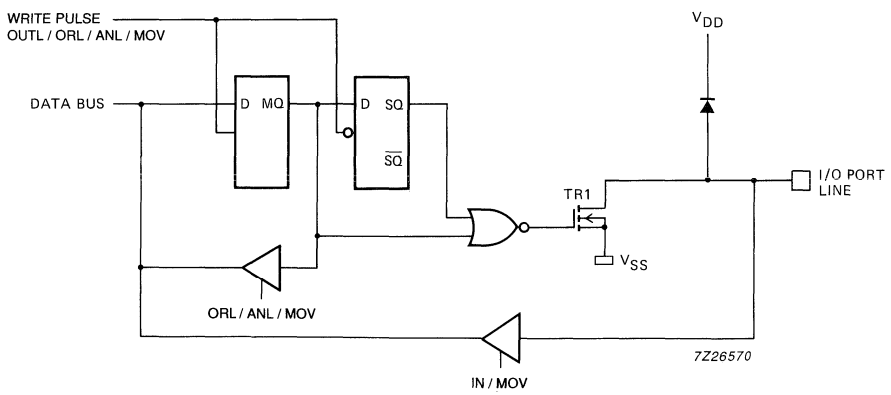


Fig.12 Open drain (mask option 2).

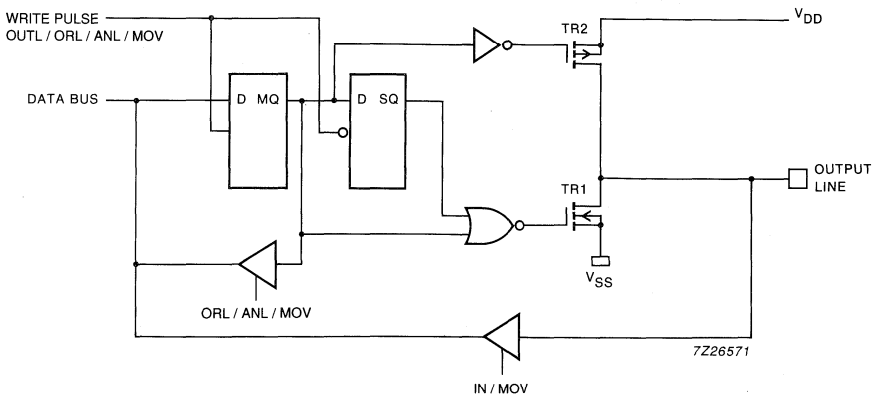


Fig.13 Push-pull (mask option 3).

For devices with a serial I/O interface, P2.3 becomes SDA and must be configured as an open drain (option 2) I/O line.

For the remaining port lines (P0.0 to P2.2), all three options are generally available. For some family members, option 3 on port 0 and port 1 lines may be restricted for emulation purposes. For specific information, refer to the relevant data sheet.

4.11 Serial I/O interface

Many members of the PCF84CXXX family have an on-chip serial I/O interface (I^2C). This two-line serial bus extends the microcontroller capabilities since it fully supports the PCF85XX (clips) family of I^2C-bus peripheral devices.

Microcontrollers that do not have dedicated serial I/O hardware can use port pins and software to simulate a serial interface. However, such microcontrollers must continuously monitor the serial bus, the data transfer rate is slower, and this approach may require significant processing and memory resources.

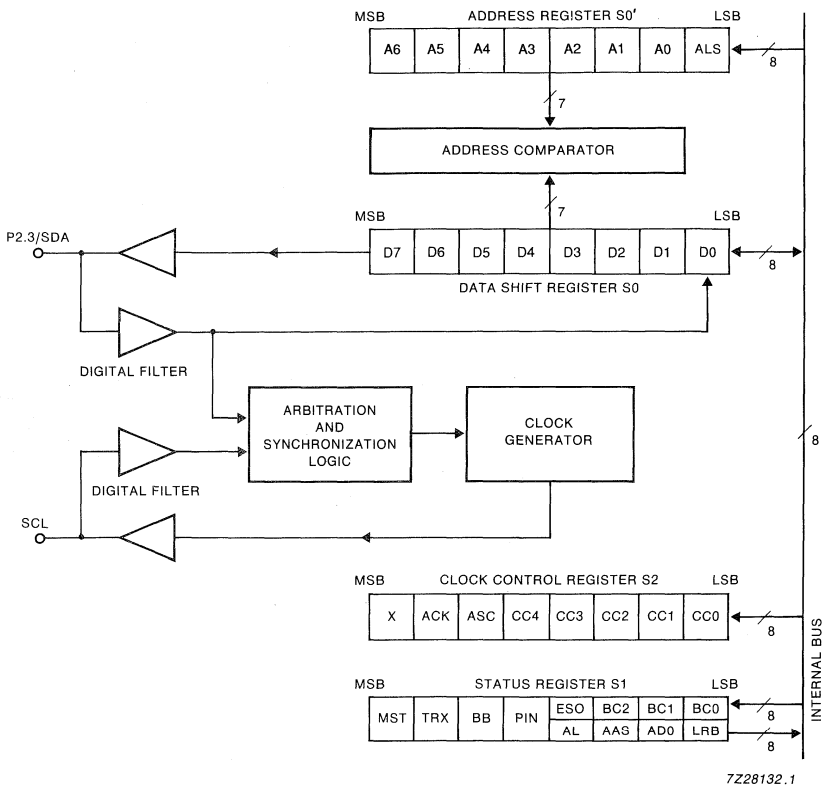
Each device on the I^2C-bus is allocated a 7-bit address. Address recognition is performed by the serial interface hardware and the microcontroller is interrupted only after a valid address (own address or general call address) has been recognized. The SIO hardware also transfers data serially and performs parallel-to serial and serial-to-parallel conversion without disrupting program execution. The microcontroller is interrupted only after a complete data byte has been transferred; the next data byte can then be written to or read from the serial I/O interface.

The serial I/O interface also facilitates the implementation of multimaster systems in which two or more microcontrollers communicate via the same I^2C-bus. An automatic arbitration procedure handles bus conflicts.

The I^2C-bus consists of a dedicated bidirectional clock line (SCL) and a bidirectional data line (P2.3/SDA) which also functions as parallel port line P2.3. When the serial I/O interface is enabled, P2.3 is disabled as a port line. Input signals on SCL and SDA are filtered for enhanced noise immunity. SCL and SDA are open drain and require external pull-up resistors if they are to be used as outputs. If unused, these two pins should be tied to V_{SS}.

The CPU and serial I/O interface communicate via the following four serial I/O interface registers (see Fig.14):

- Data Shift Register (S0)
- Address Register (S0')
- Clock Control Register (S2)
- Status Register (S1)



X = don't care

Fig.14 Serial I/O interface block diagram.

4.11.1 Data shift register (S0)

The data shift register converts serial data to parallel format and vice versa. The most significant bit is transferred first. An interrupt request is generated after a complete byte has been transferred or after a valid I²C-bus address has been detected. The MOV A,S0 instruction may be used to read S0; the MOV S0,A and MOV S0,#data instructions may be used to write to S0 if the ESO (enable serial I/O) bit in the status register (S1) is set.

4.11.2 Address register (S0')

The address register contains the device's 7-bit I²C-bus address and the ALS (always selected) bit. When ALS is zero (the recommended operating mode) bus transfers are ignored unless the START condition is immediately followed by the valid device address or the 'general call' address (00H). If ALS is set, any transfer on the bus is stored in the data shift register. The address register S0' is write-only. The MOV S0,A and MOV S0,#data instructions may be used to write to S0' if the ESO (enable serial I/O) bit in the status register (S1) is zero.

4.11.3 CLOCK CONTROL REGISTER (S2)

The clock control register is a write only register. Bits 0 to 4 of S2 define the serial clock frequency (f_{SCL}) as an integer multiple of the microcontroller clock frequency (Table 5).

Bit 5 (ASC) defines the asymmetrical clock duty cycle. If ASC is set, SCL has a duty cycle of approximately 75%. The asymmetrical clock limits the I²C-bus transmission rate to below 55 kHz. Divisors 39, 45 and 51 are not allowed if ASC is set. Resetting ASC (recommended operating mode) sets the SCL duty cycle to approximately 50%, allowing higher I²C-bus transmission rates of up to 100 kHz, and all of the divisors in Table 5 may be used.

For normal I²C-bus operation, bit 6 (ACK) must be set. After each byte transfer, an extra SCL pulse is generated during which the receiver may acknowledge reception. If ACK is reset, no acknowledge phase is available. This mode is used when a master receiver refuses to acknowledge a transfer in order to signal the 'end of transmission' to a slave transmitter.

Table 5 f_{SCL} as defined by the clock control register S2.

CC0 to CC4 (HEX)	f_{XTAL} DIVIDED BY	f_{SCL} (kHz) @ f_{XTAL} = 6 MHz	f_{SCL} (kHz) @ f_{XTAL} = 10 MHz
0	not allowed		
1	39	154*	256*
2	45	133*	222*
3	51	118*	196*
4	63	95	159*
5	75	80	133*
6	87	69	115*
7	99	61	101*
8	123	49	81
9	147	41	68
A	171	35	58
B	195	31	51
C	243	25	41
D	291	21	34
E	339	18	29
F	387	16	26
10	483	12	21
11	579	10	17
12	675	8.9	15
13	771	7.8	13.4
14	963	6.2	10.4
15	1155	5.2	8.7
16	1347	4.5	7.4
17	1539	3.9	6.5
18	1923	3.1	5.2
19	2307	2.6	4.3
1A	2691	2.2	3.7
1B	3075	2.0	3.3
1C	3843	1.6	2.6
1D	4611	1.3	2.2
1E	5379	1.1	1.9
1F	6147	1.0	1.6

* not permitted; f_{SCL} max. = 100 kHz in I²C systems

4.11.4 STATUS REGISTER (S1)

The status register controls the serial I/O interface and provides feedback about on-going bus transfers. Register S1 can be accessed by the MOV A,S1, MOV S1,A and MOV S1,#data instructions. The lower nibble of the status register is duplicated: control bits BC0-BC2 and ESO are write only, whereas feedback bits LRB, AD0, AAS and AL are read only. The status bits interact in intricate ways with each other. This must be kept in mind when developing an I²C-bus application. Table 6 describes the status bits.

4.11.4.1 Master bit (MST) and transmitter bit (TRX)

The MST and TRX bits determine the operating mode of the serial I/O interface. When not engaged in a bus transfer, MST and TRX should always be zero, placing the SIO in the slave receiver mode (see Fig.15). Return to the slave receiver mode is always performed by software. If the previous mode was a master mode, the transition (MOV S1,#D8H) involves a STOP condition, which automatically clears both MST and TRX.

The transition to the master transmitter mode is also performed by software. However, transitions to the master receiver and slave transmitter modes occur automatically if ALS = 0 (standard I²C-bus protocol). A slave receiver becomes a slave transmitter if the R/$\overline{W}$ bit in the valid address immediately following a START condition is set. A master transmitter becomes a master receiver if R/$\overline{W}$ is set in the transmitted address.

Table 6 Serial I/O status register (S1).

BIT	NAME	TYPE	DESCRIPTION
MST	Master	R/W	MST = 0: Slave (SCL input) MST = 1: Master (SCL output)
TRX	Transmitter	R/W	TRX = 0: Receiver (SDA/P2.3 input) TRX = 1: Transmitter (SDA/P2.3 output)
BB	Bus Busy	R/W	BB = 0: Bus inactive (R)/generates STOP condition (W) BB = 1: Bus busy (R)/generates START condition (W)
PIN	Pending Interrupt Not	R/W	PIN = 0: Serial interrupt pending (after byte transfer, valid address or lost arbitration); SCL forced to V_{SS} PIN = 1: No serial interrupt pending
ESO	Enable Serial I/O	W	ESO = 0: Serial I/O interface disabled; write access to S0' possible ES0 = 1: Serial I/O interface enabled; write access to S0 possible
BC0, BC1, BC2	Bit Counter	W	Preset of the Bit Counter for 1 up to 8 serial data bits. (001) = 1 bit, (010) = 2 bits, etc. (000) = complete byte (8 bits) = default value.
AL	Arbitration Lost	R	AL = 1: Arbitration Lost (bus conflict) AL = 0: When corresponding serial interrupt (PIN) is cancelled
AAS	Addressed As Slave	R	AAS = 1: Following a START condition if a valid address is detected (ALS = 0), or if the first byte is received (ALS = 1) AAS = 0: When corresponding serial interrupt (PIN) is cancelled
AD0	Address Zero	R	AD0 = 1: Following a START condition if the general call address (00H) is detected AD0 = 0: After a repeated START or a STOP condition
LRB	Last Received Bit	R	Set or reset depending on last bit transferred, acknowledgement bit if ACK = 1

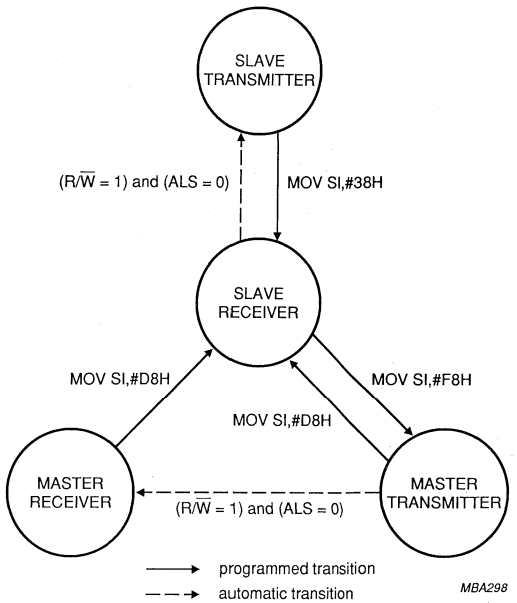


Fig.15 State diagram of the serial I/O interface.

4.11.4.2 Pending interrupt not bit (PIN)

If either MST or ALS is set, PIN is reset to zero (activated) after every byte transfer. If MST and ALS are both reset, PIN becomes zero (generating a serial interrupt request) when the valid address is detected and after each byte of the following transfer. Clock synchronization is implemented as follows: the SCL line is pulled to V_{SS} as long as PIN is zero enabling a slave to delay a master in order to read the data register (in the case of a slave receiver) or to write to the data register (in the case of a slave transmitter). PIN is automatically deactivated when S0 is accessed; it may also be deactivated by explicitly setting PIN to logic 1.

If the SIO/derivative interrupt is disabled, the serial I/O interface may be serviced by testing PIN directly in user software.

4.11.4.3 Bus busy bit (BB)

The status of the BB bit is controlled by the serial I/O interface or by bus master software. When a master clears BB (MOV S1,D8H), the serial I/O interface automatically clears MST and TRX, returning to the slave receiver mode (see Fig.15). If BB is set, write access to S1 other than accesses by the master or an addressed slave are inhibited. If BB is inadvertently set by excessive noise on the bus, the deadlock can be resolved by executing two consecutive MOV S1,18H instructions, the first of which just clears BB.

When a slave transmitter detects an end of transmission (signalled by the absence of an acknowledgment from the master receiver), it has to access S1 to inactivate PIN and become a slave receiver. However, BB should remain set. This is reflected by the MOV S1,38H instruction in Fig.15. When PIN = logic 1, clock synchronization terminates, enabling the master to generate the STOP condition.

A START condition must only be generated when BB = logic 0. Otherwise, the serial I/O interface responds as if bus arbitration had been lost.

4.11.4.4 Arbitration lost bit (AL)

The AL bit is set by the serial I/O interface when it loses bus arbitration in the master transmitter mode. Simultaneously, MST and TRX are cleared to enable the interface (now in the slave receiver mode) to determine if it is validly addressed by the device that won the arbitration. PIN is activated when the byte transfer is complete. AL will be cleared when the serial interrupt is terminated.

4.11.4.5 Addressed as slave bit (AAS)

AAS is set by the serial I/O interface following a START condition when the valid address is detected (ALS = logic 0 in register S0') or when the first byte is received (ALS = logic 1 in register S0'). AAS is cleared when the serial interrupt is terminated.

4.11.4.6 Address zero bit (AD0)

AD0 is set by the serial I/O interface independently of ALS when the 'general call' address (00H) is detected following a START condition. AD0 is cleared after a repeated START or a STOP condition.

4.11.4.7 Last received bit (LRB)

LRB contains the last bit transferred. If ACK = logic 1, LRB contains the acknowledgement bit. It remains valid as long as PIN = logic 0.

4.11.4.8 Enable serial I/O bit (ESO)

The ESO bit enables/disables the serial I/O interface. When ESO = logic 0, access to register S0' is enabled, the SCL pin is in a high impedance state and the P2.3/SDA pin is made available as a normal I/O port line.

When ESO = logic 1, the serial I/O interface is enabled and access to register S0 is possible. The remaining S1 bits can only be altered when ESO is set. The SCL and P2.3/SDA pins are enabled as serial clock and data lines respectively.

To avoid bus deadlock, ESO must be reset before a STOP instruction is executed.

4.11.4.9 Bit counter bits (BC0, BC1 and BC2)

BC0, BC1 and BC2 should all be zero for normal I^2C-bus operation. The bit counter is always cleared by a START condition; thus all eight bits of the first byte are transferred.

If a non-zero bit counter value is chosen, it is only valid for one S0 transfer since the counter decrements to zero. The bit counter is useful when a master receiver signals an end of transmission by sending a negative acknowledge after the last byte received; the last byte is received with ACK = logic 0 in register S2. The negative acknowledge is then issued by setting the bit counter to one and 'receiving' one bit from the HIGH level on the SDA line. The slave transmitter interprets the same signals as a negative acknowledgement.

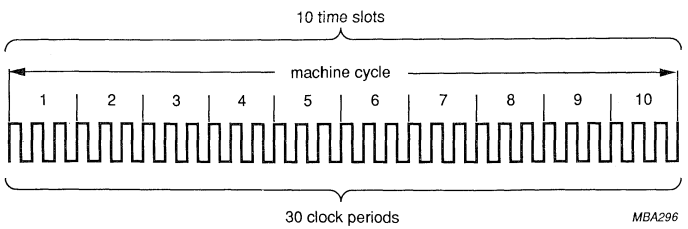


Fig.16 Machine cycle timing.

4.12 Timing

Every machine cycle consists of 10 time slots. Each time slot consists of 3 clock periods (see Fig.16).

The clock frequencies range is from 100 kHz to a maximum which is a function of supply voltage. When $V_{DD} \geq 4.5$ V, operation at 10 MHz is guaranteed.

The clock signal may be internally generated by the on-chip oscillator and an external crystal. Alternatively, an external clock may be applied to the XTAL1 pin.

4.13 Oscillator

The on-chip oscillator consists of an invertor stage including a feedback resistor and load capacitors (see Fig.17). A quartz crystal is usually connected between XTAL1 and XTAL2. Alternatively, a ceramic resonator or an inductor may be used as the timing element; however, external load capacitors should then be added for good frequency stability.

When the supply voltage drops below the power-on-reset level, the oscillator is inhibited. The internal oscillator may also be inhibited by the STOP instruction.

Oscillator start-up time depends on the external timing element. The start-up time of a quartz crystal is several milliseconds due to the narrow crystal bandwidth.

For proper oscillator start-up, the transconductance (gm) of the invertor stage must satisfy the following relationship (see Fig.17 and Fig.18):

$$4.2[R_1\omega^2(C_L + C_0 + C_f)^2 + 1/R_f] < gm < C_1C_2/[R_1(C_0 + C_f)^2 + 1/\omega^2R_f]$$

where:

R_1 = resonator series resistance
C_0 = static resonator capacitance
R_f = feedback resistor
C_L = $C_1C_2/[C_1 + C_2]$
C_f = parasitic feedback capacitance (typically 2 pF on-chip, external value depends on PC board wiring)
ω = $2\pi f_{XTAL}$

For more information on crystal oscillators and start-up conditions see the application note 'Crystal Oscillators for CMOS Circuits'.

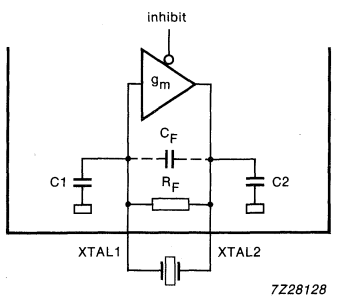


Fig.17 Oscillator with integrated elements.

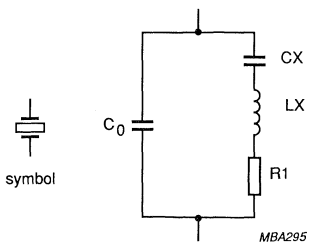


Fig.18 Crystal unit equivalent circuit.

4.14 Reset

After a falling edge on the internal reset, 1866 clock cycles are required to initialize the microcontroller to a defined state. The first instruction is then executed.

4.14.1 Passive reset

A passive reset is generated by the RC circuit shown in Fig.19. As V_{DD} rises, the discharged C_{reset} pulls the RESET pin close to V_{DD}. When V_{DD} crosses the power-on reference level V_{ref} (typically 1.5 V), the device generates a reset pulse of approximately 50 μs which helps to pull the RESET pin to V_{DD}. To ensure a correct reset, the RESET voltage should reach at least 70% of the final value of V_{DD} before C_{reset} charges through TR2 and R_{reset}. If the RESET voltage and V_{DD} rise exponentially, this requirement is satisfied when the time constant τ ($C_{reset}R_{reset}$) of the RESET pulse is greater than eight times the time constant of V_{DD}. If V_{DD} rises linearly, the requirement is satisfied when the time constant of the RESET pulse is greater than twice the time constant of V_{DD}.

In the event of a drop in the supply voltage, the diode rapidly discharges C_{reset}, ensuring reliable power-on-reset even after short supply voltage interruptions.

In battery-powered systems where V_{DD} quickly reaches its minimum operating value, passive reset can be performed without external components since the 50 μs reset pulse guarantees proper initialization.

4.14.2 Active reset

An active reset can be generated by driving the RESET pin HIGH with external logic. This pulse should be present until V_{DD} has reached its minimum operating value.

4.14.3 Reset state

After a reset, the microcontroller is initialized as follows:

The program counter points to 00H. Memory bank 0, register bank 0, and stack pointer 0 (locations 8 and 9) are selected. All interrupts are disabled. The timer/event counter is stopped and cleared, the timer flag is cleared, and the timer prescaler is set to modulo 32 (PS = 0). All port flip-flops (except P2.3/SDA) are set to logic 1. P2.3/SDA and SCL are high impedance 30 clock pulses (max.) after the end of the internal reset pulse. The serial I/O interface is disabled (ESO = 0) and in the slave receiver mode 30 clock pulses (max.) after the end of internal reset pulse (S0, S0', S1 and S2 are cleared except when PIN = logic 1). The Idle and Stop modes are terminated.

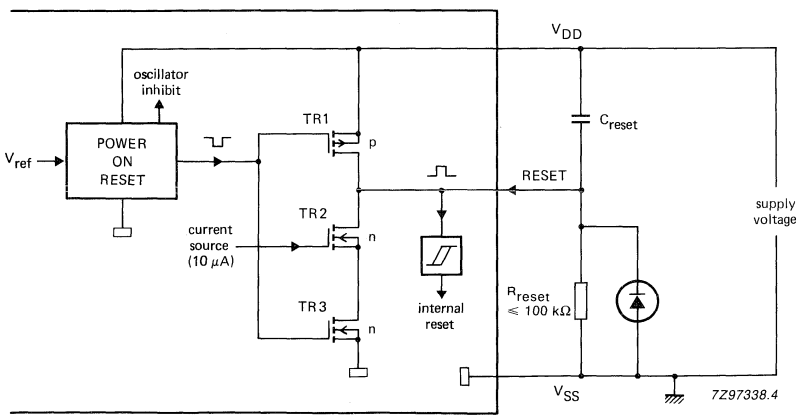


Fig.19 Passive power-on-reset.

4.15 Idle mode

The Idle mode is very useful in low power applications. When all computational tasks are completed, the device can be placed in the Idle mode instead of a power consuming waiting loop.

When the microcontroller enters the Idle mode by executing an IDLE instruction, all activity is halted except for the oscillator, the timer/event counter and the serial I/O interface (if available).

The Idle mode is terminated when an enabled interrupt (or reset) occurs. The interrupt routine is executed and program execution resumes at the instruction immediately following the IDLE instruction.

For timer/event counter interrupts and SIO/derivative interrupts, termination of the Idle mode is straightforward. However, care must be taken when the Idle mode is terminated by the external interrupt since $\overline{INT}$/T0 is negative-edge triggered. If $\overline{INT}$/T0 was LOW prior to entering the Idle mode, it must go HIGH before a negative edge can be generated. Fig.20 shows the exact timing for Idle mode termination with an external interrupt $\overline{INT}$/T0.

If no interrupt is enabled, the Idle mode can only be terminated by an active signal on the RESET pin. A normal reset sequence is executed (Fig.20).

4.16 Stop mode

The Stop mode enables very low power operation. When all computational tasks are completed, the device can be virtually shut down by stopping the oscillator.

When the microcontroller enters the Stop mode by executing a STOP instruction, the oscillator is switched off. All internal states (CPU status, RAM) and I/O levels are maintained.

The Stop mode is terminated by a LOW level on the $\overline{INT}$/T0 pin or a reset. In the latter case, a normal reset sequence is executed (see Fig.21).

Unlike the Idle mode, the microcontroller responds to a LOW level on the $\overline{INT}$/T0 pin (i.e. not to a negative edge). If the $\overline{INT}$/T0 pin is LOW when the STOP instruction is executed, the Stop mode will not be entered.

A negative edge on $\overline{INT}$/T0 causes program execution to continue after a delay of 1866 clock cycles. If the external interrupt is enabled, the microcontroller executes the instruction immediately following the STOP instruction before executing the interrupt routine. If the external interrupt is disabled, program execution continues with the instruction following the STOP instruction (see Fig.21).

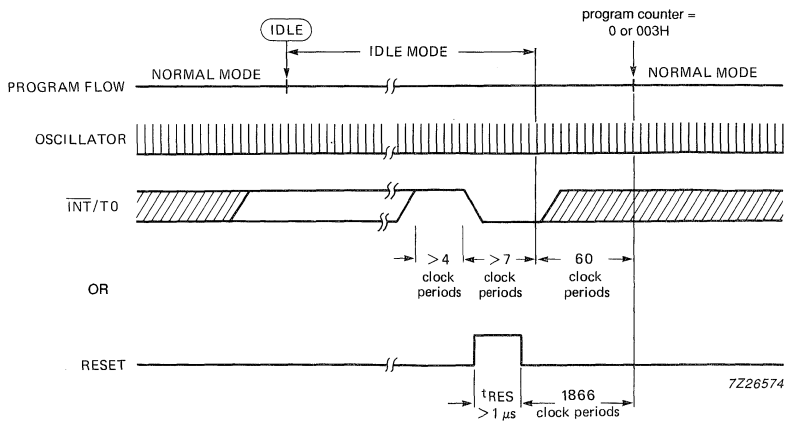


Fig.20 Entering and leaving the Idle mode.

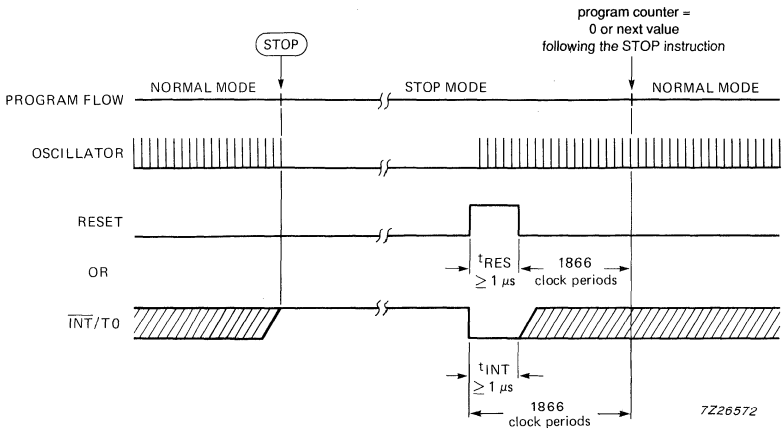


Fig.21 Entering and leaving the Stop mode.

4.17 Derivative logic

Several members of the PCF84CXXX family contain derivative logic. For specific information on a particular device, refer to the relevant data sheet.

The derivative registers are write only, read only or read/write (see Fig.22). They may be accessed internally via the derivative address register using the derivative input/output instructions (MOV A,Dx, MOV Dx,A, ANL Dx,A and ORL Dx,A).

Derivative interrupts share the PIN flag with the SIO interrupt (if available). When the derivative interrupt routine is executed, the PIN flag must be deactivated by software.

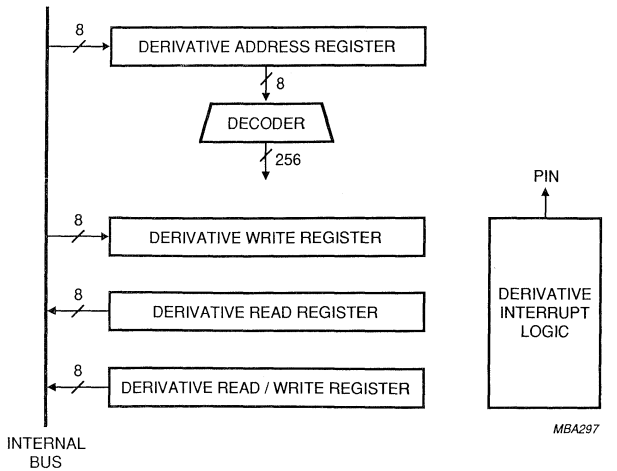


Fig.22 Block diagram of derivative logic.

5 INSTRUCTION SET

The PCF84CXXX family instruction set consists of over 80 one and two byte instructions. Program code efficiency is high because all RAM locations and all ROM locations on a 256 byte page require only a single byte address.

Table 8 contains the instruction set of the PCF84CXXX. Figure 23 shows the instruction map and Table 7 describes the symbols that are used.

Table 7 Symbols and definitions.

SYMBOL	DESCRIPTION
A	accumulator
addr	program memory address
Bb	bit designation (b = 0 to 7)
C	carry bit (bit CY)
CNT	event counter
Dx	mnemonic derivative register (x = 0 to 255)
direct	8-bit derivative register address
data	8-bit immediate data
I	interrupt
MBn	memory bank (n = 0 to 3)
MBFFn	memory bank flip-flop (n = 0 or 1)
P	mnemonic for 'in-page' operation
PC	program counter
Pp	port designation (p = 0, 1, or 2)
PS	Timer prescaler select
PSW	program status word
RB	register bank
RBS	register bank select flag
@Rr	8-bit address register (r = 0, 1)
Rr	8-bit register (r = 0 to 7)
Sn	serial I/O register (n = 0, 1, or 2)
SP	stack pointer
T	timer
TCNT	timer/event counter
TF	timer flag
T0, T1	test 0 and test 1 inputs
#	immediate data prefix
@	indirect address prefix
(X)	contents of X
((X))	contents of location addressed by X
←	is replaced by
↔	is exchanged with
< >	represents a hex digit

PCF84CXXX PCD33XX

first hexadecimal character of opcode ↓	second hexadecimal character of opcode →															
	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
0	NOP	IDLE		ADD A, #data	JMP page 0	EN I	JNTF addr	DEC A	IN A,Pp 0	IN A,Pp 1	IN A,Pp 2		MOV A,Sn 0	MOV A,Sn 1		
1	INC @ Rr 0	INC @ Rr 1	JB0 addr	ADDC A,#data	CALL page 0	DIS I	JTF addr	INC A	INC Rr 0	INC Rr 1	INC Rr 2	INC Rr 3	INC Rr 4	INC Rr 5	INC Rr 6	INC Rr 7
2	XCH A, @Rr 0	XCH A, @Rr 1	STOP	MOV A, #data	JMP page 1	EN TCNTI	JNT0 addr	CLR A	XCH A,Rr 0	XCH A,Rr 1	XCH A,Rr 2	XCH A,Rr 3	XCH A,Rr 4	XCH A,Rr 5	XCH A,Rr 6	XCH A,Rr 7
3	XCHD A, @Rr 0	XCHD A, @Rr 1	JB1 addr		CALL page 1	DIS TCNTI	JT0 addr	CPL A	OUTL Pp,A 0	OUTL Pp,A 1	OUTL Pp,A 2		MOV Sn,A 0	MOV Sn,A 1	MOV Sn,A 2	
4	ORL A, @Rr 0	ORL A, @Rr 1	MOV A, T	ORL A, #data	JMP page 2	STRT CNT	JNT1 addr	SWAP A	ORL A,Rr 0	ORL A,Rr 1	ORL A,Rr 2	ORL A,Rr 3	ORL A,Rr 4	ORL A,Rr 5	ORL A,Rr 6	ORL A,Rr 7
5	ANL A, @Rr 0	ANL A, @Rr 1	JB2 addr	ANL A, #data	CALL page 2	STRT T	JT1 addr	DA A	ANL A,Rr 0	ANL A,Rr 1	ANL A,Rr 2	ANL A,Rr 3	ANL A,Rr 4	ANL A,Rr 5	ANL A,Rr 6	ANL A,Rr 7
6	ADD A, @Rr 0	ADD A, @Rr 1	MOV T, A		JMP page 3	STOP TCNT		RRC A	ADD A,Rr 0	ADD A,Rr 1	ADD A,Rr 2	ADD A,Rr 3	ADD A,Rr 4	ADD A,Rr 5	ADD A,Rr 6	ADD A,Rr 7
7	ADDC A, @Rr 0	ADDC A, @Rr 1	JB3 addr		CALL page 3			RR A	ADDC A,Rr 0	ADDC A,Rr 1	ADDC A,Rr 2	ADDC A,Rr 3	ADDC A,Rr 4	ADDC A,Rr 5	ADDC A,Rr 6	ADDC A,Rr 7
8				RET	JMP page 4	EN SI			ORL Pp,#data 0	ORL Pp,#data 1	ORL Pp,#data 2		MOV A,Dx	MOV Dx,A	ANL Dx,A	ORL Dx,A
9			JB4 addr	RETR	CALL page 4	DIS SI	JNZ addr	CLR C	ANL Pp,#data 0	ANL Pp,#data 1	ANL Pp,#data 2		MOV Sn,#data 0	MOV Sn,#data 1	MOV Sn,#data 2	
A	MOV @ Rr,A 0	MOV @ Rr,A 1		MOVP A,@A	JMP page 5	SEL MB2		CPL C	MOV Rr,A 0	MOV Rr,A 1	MOV Rr,A 2	MOV Rr,A 3	MOV Rr,A 4	MOV Rr,A 5	MOV Rr,A 6	MOV Rr,A 7
B	MOV @Rr, #data 0	MOV @Rr, #data 1	JB5 addr	JMPP @A	CALL page 5	SEL MB3			MOV Rr,#data 0	MOV Rr,#data 1	MOV Rr,#data 2	MOV Rr,#data 3	MOV Rr,#data 4	MOV Rr,#data 5	MOV Rr,#data 6	MOV Rr,#data 7
C	DEC @Rr 0	DEC @Rr 1			JMP page 6	SEL RB0	JZ addr	MOV A,PSW	DEC Rr 0	DEC Rr 1	DEC Rr 2	DEC Rr 3	DEC Rr 4	DEC Rr 5	DEC Rr 6	DEC Rr 7
D	XRL A, @Rr 0	XRL A, @Rr 1	JB6 addr	XRL A,#data	CALL page 6	SEL RB1		MOV PSW,A	XRL A,Rr 0	XRL A,Rr 1	XRL A,Rr 2	XRL A,Rr 3	XRL A,Rr 4	XRL A,Rr 5	XRL A,Rr 6	XRL A,Rr 7
E	DJNZ @ Rr,addr 0	DJNZ @ Rr,addr 1			JMP page 7	SEL MB0	JNC addr	RL A	DJNZ Rr,addr 0	DJNZ Rr,addr 1	DJNZ Rr,addr 2	DJNZ Rr,addr 3	DJNZ Rr,addr 4	DJNZ Rr,addr 5	DJNZ Rr,addr 6	DJNZ Rr,addr 7
F	MOV A, @Rr 0	MOV A, @Rr 1	JB7 addr		CALL page 7	SEL MB1	JC addr	RLC A	MOV A,Rr 0	MOV A,Rr 1	MOV A,Rr 2	MOV A,Rr 3	MOV A,Rr 4	MOV A,Rr 5	MOV A,Rr 6	MOV A,Rr 7

MBA281

Fig.23 PCF84CXXX instruction map.

Table 8 PCF84CXXX family instruction set.

MNEMONIC	OPCODE (HEX)	BYTES/ CYCLES	DESCRIPTION	FUNCTION		NOTES
ACCUMULATOR						
ADD A, Rr	6<8 + r>	1/1	Add register contents to A	(A) ← (A) + (Rr)	r = 0 to 7	1
ADD A, @Rr	6r	1/1	Add RAM data, addressed by Rr, to A	(A) ← (A) + ((Rr))	r = 0, 1	1
ADD A, #data	03 data	2/2	Add immediate data to A	(A) ← (A) + data		1
ADDC A, Rr	7<8 + r>	1/1	Add carry and register contents to A	(A) ← (A) + (Rr) + (C)	r = 0 to 7	1
ADDC A, @Rr	7r	1/1	Add carry and RAM data, addressed by Rr, to A	(A) ← (A) + ((Rr)) + (C)	r = 0, 1	1
ADDC A, #data	13 data	2/2	Add carry and immediate data to A	(A) ← (A) + data + (C)		1
ANL A, Rr	5<8 + r>	1/1	'AND' Rr with A	(A) ← (A) AND (Rr)	r = 0 to 7	
ANL A, @Rr	5r	1/1	'AND' RAM data, addressed by Rr, with A	(A) ← (A) AND ((Rr))	r = 0, 1	
ANL A, #data	53 data	2/2	'AND' immediate data with A	(A) ← (A) AND data		
ORL A, Rr	4<8 + r>	1/1	'OR' Rr with A	(A) ← (A) OR (Rr)	r = 0 to 7	
ORL A, @Rr	4r	1/1	'OR' RAM data, addressed by Rr, with A	(A) ← (A) OR ((Rr))	r = 0, 1	
ORL A, #data	43 data	2/2	'OR' immediate data with A	(A) ← (A) OR data		
XRL A, Rr	D<8 + r>	1/1	'XOR' Rr with A	(A) ← (A) XOR (Rr)	r = 0 to 7	
XRL A, @Rr	Dr	1/1	'XOR' RAM data, addressed by Rr, with A	(A) ← (A) XOR ((Rr))	r = 0, 1	
XRL A, #data	D3 data	2/2	'XOR' immediate data with A	(A) ← (A) XOR data		
INC A	17	1/1	Increment A by 1	(A) ← (A) + 1		
DEC A	07	1/1	Decrement A by 1	(A) ← (A) − 1		
CLR A	27	1/1	Clear A to zero	(A) ← 0		
CPL A	37	1/1	One's complement A	(A) ← NOT(A)		
RL A	E7	1/1	Rotate A left	$(A_{n+1}) \leftarrow (A_n)$ $(A_0) \leftarrow (A_7)$	n = 0 to 6	
RLC A	F7	1/1	Rotate A left through carry	$(A_{n+1}) \leftarrow A_n$ $(A_0) \leftarrow (C), (C) \leftarrow (A_7)$	n = 0 to 6	2
RR A	77	1/1	Rotate A right	$(A_n) \leftarrow (A_{n+1})$ $(A_7) \leftarrow (A_0)$	n = 0 to 6	
RRC A	67	1/1	Rotate A right through carry	$(A_n) \leftarrow (A_{n+1})$ $(A_7) \leftarrow (C), (C) \leftarrow (A_0)$	n = 0 to 6	2
DA A	57	1/1	Decimal adjust A	(A) ← (A) + 06H if AC = 1 or $(A_{0-3}) > 9$; (A) ← (A) + 60H if $(A_{4-7}) > 9$		2
SWAP A	47	1/1	Swap nibbles of A	$(A_{4-7}) \leftrightarrow (A_{0-3})$		2

MNEMONIC	OPCODE (HEX)	BYTES/ CYCLES	DESCRIPTION	FUNCTION		NOTES
DATA MOVES						
MOV A, Rr	F<8 + r>	1/1	Move register contents to A	(A) ← (Rr)	r = 0 to 7	
MOV A, @Rr	Fr	1/1	Move RAM data, addressed by Rr, to A	(A) ← ((Rr))	r = 0, 1	
MOV A, #data	23 data	2/2	Move immediate data to A	(A) ← data		
MOV Rr, A	A<8 + r>	1/1	Move accumulator contents to register	(Rr) ← (A)	r = 0 to 7	
MOV @Rr, A	Ar	1/1	Move accumulator contents to RAM location addressed by Rr	((Rr)) ← (A)	r = 0, 1	
MOV Rr, #data	B<8 + r> data	2/2	Move immediate data to Rr	(Rr) ← data	r = 0 to 7	
MOV @Rr, #data	Br data	2/2	Move immediate data to RAM location addressed by Rr	((R0)) ← data	r = 0, 1	
XCH A, Rr	2<8 + r>	1/1	Exchange accumulator contents with Rr	(A) ‹ › (Rr)	r = 0 to 7	
XCH A, @Rr	2r	1/1	Exchange accumulator contents with RAM data addressed by Rr	(A) ↔ ((Rr))	r = 0, 1	
XCHD A, @Rr	3r	1/1	Exchange lower nibbles of A and RAM data addressed by Rr	$(A_{0-3}) \leftrightarrow ((Rr_{0-3}))$	r = 0, 1	
MOV A, PSW	C7	1/1	Move PSW contents to accumulator	(A) ← (PSW)		
MOV PSW, A	D7	1/1	Move accumulator bit 3 to PSW_3 (PS)	(PS) ← (A_3)		3
MOVP A, @A	A3	1/2	Move indirectly addressed data in current page to A	$(PC_{0-7}) \leftarrow (A), (A) \leftarrow ((PC))$		
CARRY FLAG						
CLR C	97	1/1	Clear carry bit	(C) ← 0		2
CPL C	A7	1/1	Complement carry bit	(C) ← NOT(C)		2
REGISTER						
INC Rr	1<8 + r>	1/1	Increment register by 1	(Rr) ← (Rr) + 1	r = 0 to 7	
INC @Rr	1r	1/1	Increment RAM data, addressed by Rr, by 1	((Rr)) ← ((Rr)) + 1	r = 0, 1	
DEC Rr	C<8 + r>	1/1	Decrement register by 1	(Rr) ← (Rr) – 1	r = 0 to 7	
DEC @Rr	Cr	1/1	Decrement RAM data, addressed by Rr, by 1	((Rr)) ← ((Rr)) – 1	r = 0, 1	

MNEMONIC	OPCODE (HEX)	BYTES/ CYCLES	DESCRIPTION	FUNCTION		NOTES
BRANCH						
JMP addr	<2n>4 addr	2/2	Unconditional jump within a 2 K bank	(PC_{8-10}) ← n (PC_{0-7}) ← addr (PC_{11-12}) ← (MBFF 0-1)	n = 0 to 7	
JMPP @A	B3	1/2	Indirect jump within a page	(PC_{0-7}) ← ((A))		
DJZN Rr, addr	E<8 + r> addr	2/2	Decrement Rr by 1 and jump if not zero to addr	(Rr) ← (Rr) – 1; if (Rr) not zero, then (PC_{0-7}) ← addr	r = 0 to 7	
DJNZ @Rr, addr	Er	2/2	Decrement RAM data, addressed by Rr, by 1 and jump if not zero to addr	((Rr)) ← ((Rr)) – 1; if ((Rr)) not zero, then (PC_{0-7}) ← addr	r = 0, 1	
JBb addr	<2b + 1>2 addr	2/2	Jump to addr if Accumulator bit b = 1	If (A_b) = 1, then (PC_{0-7}) ← addr	b = 0 to 7	
JC addr	F6 addr	2/2	Jump to addr if C = 1	If (C) = 1, then (PC_{0-7}) ← addr		
JNC addr	E6 addr	2/2	Jump to addr if C = 0	If (C) = 0, then (PC_{0-7}) ← addr		
JZ addr	C6 addr	2/2	Jump to addr if A = 0	If (A) = 0, then (PC_{0-7}) ← addr		
JNZ addr	96 addr	2/2	Jump to addr if A is NOT zero	If (A) ≠ 0, then (PC_{0-7}) ← addr		
JTO addr	36 addr	2/2	Jump to addr if T0 = 1	If T0 = 1, then (PC_{0-7}) ← addr		
JNT0 addr	26 addr	2/2	Jump to addr if T0 = 0	If T0 = 0: (PC_{0-7}) ← addr		
JT1 addr	56 addr	2/2	Jump to addr if T1 = 1	If T1 = 1: (PC_{0-7}) ← addr		
JNT1 addr	46 addr	2/2	Jump to addr if T1 = 0	If T1 = 0: (PC_{0-7}) ← addr		
JTF addr	16 addr	2/2	Jump to addr if Timer Flag = 1	If TF = 1: (PC_{0-7}) ← addr		4
JNTF addr	06 addr	2/2	Jump to addr if Timer Flag = 0	If TF = 0: (PC_{0-7}) ← addr		4
TIMER/EVENT COUNTER						
MOV A, T	42	1/1	Move timer/event counter contents to accumulator	(A) ← (T)		
MOV T, A	62	1/1	Move accumulator contents to timer/event counter	(T) ← (A)		
STRT CNT	45	1/1	Start event counter			
STRT T	55	1/1	Start timer			
STOP TCNT	65	1/1	Stop timer/event counter			
EN TCNTI	25	1/1	Enable timer/event counter interrupt			
DIS TCNTI	35	1/1	Disable timer/event counter interrupt			
CONTROL						
EN I	05	1/1	Enable external interrupt			
DIS I	15	1/1	Disable external interrupt			
SEL RB0	C5	1/1	Select register bank 0	(RBS) ← 0		5
SEL RB1	D5	1/1	Select register bank 1	(RBS) ← 1		5
SEL MB0	E5	1/1	Select program memory bank 0	(MBFF0) ← 0, (MBFF1) ← 0		10
SEL MB1	F5	1/1	Select program memory bank 1	(MBFF0) ← 1, (MBFF1) ← 0		10
SEL MB2	A5	1/1	Select program memory bank 2	(MBFF0) ← 0, (MBFF1) ← 1		10
SEL MB3	B5	1/1	Select program memory bank 3	(MBFF0) ← 1, (MBFF1) ← 1		10
STOP	22	1/1	Enter Stop mode			
IDLE	01	1/1	Enter Idle mode			

MNEMONIC	OPCODE (HEX)	BYTES/ CYCLES	DESCRIPTION	FUNCTION		NOTES
SUBROUTINE						
CALL addr	<2n + 1>4 addr	2/2	Jump to subroutine	((SP)) ← (PC), ($PSW_{4, 6, 7}$) (SP) ← (SP) + 1 (PC_{8-10}) ← n (PC_{0-7}) ← addr (PC_{11-12}) ← (MBFF0-1)	n = 0 to 7	6
RET	83	1/2	Return from subroutine	(SP) ← (SP) - 1 (PC) ← ((SP))		6
RETR	93	1/2	Return from interrupt and restore bits 4, 6, 7 of PSW	(SP) ← (SP) - 1 ($PSW_{4, 6, 7}$) + (PC) ← ((SP))		6
PARALLEL INPUT/OUTPUT						
IN A, Pp	08 09 0A	1/2	Input port p data to accumulator	(A) ← (P0) (A) ← (P1) (A) ← (P2)		7
OUTL Pp, A	38 39 3A	1/2	Output accumulator data to port p	(P0) ← (A) (P1) ← (A) (P2) ← (A)		
ANL Pp, #data	98 data 99 data 9A data	2/2	AND port p with immediate data	(P0) ← (P0) AND data (P1) ← (P1) AND data (P2) ← (P2) AND data		
ORL Pp, #data	88 data 89 data 8A data	2/2	OR port p data with immediate data	(P0) ← (P0) OR data (P1) ← (P1) OR data (P2) ← (P2) OR data		
DERIVATIVE INPUT/OUTPUT						
MOV A, Dx	8C direct	2/2	Move derivative register contents to accumulator	(A) ← (Dx)	x = 0 to 255	8
MOV Dx, A	8D direct	2/2	Move accumulator contents to derivative register	(Dx) ← (A)	x = 0 to 255	8
ANL Dx, A	8E direct	2/2	AND derivative register with accumulator	(Dx) ← (Dx) AND (A)	x = 0 to 255	8
ORL Dx, A	8F direct	2/2	OR derivative register with accumulator	(Dx) ← (Dx) OR (A)	x = 0 to 255	8
SERIAL INPUT/OUTPUT						
MOV A, S_n	0C 0D	1/2	Move serial I/O register contents to accumulator	(A) ← (S0) (A) ← (S1)	n = 0, 1	
MOV S_n, A	3C 3D 3E	1/2	Move accumulator contents to serial I/O register	(S0) ← (A) (S1) ← (A) (S2) ← (A)	n = 0, 1, 2	
MOV S_n, #data	9C data 9D data 9E data	2/2	Move immediate data to serial I/O register	(S0) ← data (S1) ← data (S2) ← data	n = 0, 1, 2	
EN SI	85	1/1	Enable serial I/O interrupt			
DIS SI	95	1/1	Disable serial I/O interrupt			
NOP	00	1/1	No operation	(PC_{0-10}) ← (PC_{0-10}) + 1		

Notes

1. PSW CY, AC affected.
2. PSW CY affected.
3. PSW PS affected.
4. Execution of a JTF or JNTF instruction resets the Timer Flag (TF).
5. PSW RBS affected.
6. PSW SP_0, SP_1, SP_2 affected.
7. (A) = 0000, P2.3, P2.2, P2.1, P2.0
8. For more information on the derivative input/output instructions of a particular microcontroller, consult the specific microcontroller data sheet.
9. (S1) has a different meaning for read and write operations. See section 4.11.4.
10. SEL MB instructions may not be used within interrupt routines.

6 LIMITING VALUES

Limiting values in accordance with the Absolute Maximum System (IEC 134).

SYMBOL	PARAMETER	MIN.	MAX.	UNIT
V_{DD}	Supply voltage range	–0.8	8.0	V
V_I	All input voltages	–0.5	V_{DD} + 0.5	V
I_I, I_O	DC current into any input or output	-	10	mA
P_{tot}	Total power dissipation	-	125	mW
T_{stg}	Storage temperature range	–65	150	°C
T_{amb}	Operating ambient temperature range (if $P_{tot} \leq 100$ mW)	–40	70	°C
T_{amb}	Operating ambient temperature range (if $P_{tot} \leq 30$ mW)	–40	85	°C
T_j	Operating junction temperature	-	90	°C

7 HANDLING

Inputs and outputs are protected against electrostatic discharge in normal handling. However, it is good practice to take normal precautions appropriate to handling MOS devices (see 'Handling MOS devices').

8 DC CHARACTERISTICS

V_{DD} = 2.5 V to 5.5 V; V_{SS} = 0 V; T_{amb} = −40 to 85 °C; all voltages with respect to V_{SS} unless otherwise specified.

SYMBOL	PARAMETER	CONDITIONS	MIN.	TYP.	MAX.	UNIT
Supply						
V_{DD}	Operating supply voltage range	See Fig.24.	2.5	-	5.5	V
I_{DD}	Operating supply current	See Fig.26, Fig.27, and note 1; at V_{DD} = 5 V; f_{XTAL} = 10 MHz at V_{DD} = 5 V; f_{XTAL} = 6 MHz at V_{DD} = 3 V; f_{XTAL} = 3.58 MHz	- - -	1.6 1.0 0.3	3.2 2.0 0.8	mA mA mA
I_{DD}	Idle mode supply current	See Fig.28, Fig.29, and note 1; at V_{DD} = 5 V; f_{XTAL} = 10 MHz at V_{DD} = 5 V; f_{XTAL} = 6 MHz at V_{DD} = 3 V; f_{XTAL} = 3.58 MHz	- - -	0.8 0.5 0.15	1.6 1.0 0.4	mA mA mA
IDD	Stop mode supply current	See Fig.25, note 1, and note 2; at V_{DD} = 2.5 V; T_{amb} = 25 °C at V_{DD} = 2.5 V; T_{amb} = 85 °C	- -	1.2 -	2.5 10.0	µA µA
Inputs						
V_{IL}	Input voltage LOW		0	-	$0.3V_{DD}$	V
V_{IH}	Input voltage HIGH		$0.7V_{DD}$	-	V_{DD}	V
$\pm I_{IL}$	Input leakage current	$V_{SS} \leq V_I \leq V_{DD}$	-	-	1	µA
Outputs						
I_{OL}	Output sink current	All outputs except SCL, P2.3/SDA; see Fig.30. at V_{DD} = 5 V ± 10%; V_O = 0.4 V	1.6	3.0	-	mA
I_{OL}	Output sink current	SCL, P2.3/SDA; see Fig.31. at V_{DD} = 5 V ± 10%; V_O = 0.4 V	3.0	5.5	-	mA
$-I_{OH}$	Output source current	See Fig.32. at V_{DD} = 5 V ± 10%; V_O = $0.7V_{DD}$ at V_{DD} = 5 V ± 10%; V_O = V_{SS}	40 -	- -	- 400	µA µA
$-I_{OH}$	Output source current	See Fig.33. at V_{DD} = 5 V ± 10%; V_O = V_{DD} − 0.4 V	1.6	3.0	-	mA

9 AC CHARACTERISTICS

V_{DD} = 2.5 to 5.5 V; V_{SS} = 0 V; T_{amb} = −40 to 85 °C. All voltages with respect to V_{SS} unless otherwise specified.

SYMBOL	PARAMETER	CONDITIONS	MIN.	TYP.	MAX.	UNIT
t_r	Rise time all outputs	See note 3.	-	30	-	ns
t_f	Fall time all outputs	See note 3.	-	30	-	ns
f_{XTAL}	Clock frequency	See Fig.24.	0.45	-	10.0	MHz

10 SERIAL I/O INTERFACE CHARACTERISTICS

See Fig.34, Fig.35 and note 4.

SYMBOL	PARAMETER	CONDITIONS	SCL INPUT	SCL OUTPUT
$t_{HD;STA}$	START condition hold time		$\geq 14/f_{XTAL}$	$(DF+9)/(2f_{XTAL})$
t_{LOW}	SCL LOW time	Note 5.	$\geq 17/f_{XTAL}$	$(DF-3)/(2f_{XTAL})$
t_{HIGH}	SCL HIGH time	Note 5.	$\geq 17/f_{XTAL}$	$(DF+3)/(2f_{XTAL})$
t_{RC}	SCL rise time	Note 6.	≤ 1 µs	≤ 1 µs
t_{FC}	SCL fall time	Note 7.	≤ 0.3 µs	≤ 0.1 µs
SYMBOL	**PARAMETER**	**CONDITIONS**	**P2.3/SDA INPUT**	**P2.3/SDA OUTPUT**
t_{BUF}	Bus free time	Note 8.	$\geq 14/f_{XTAL}$	≥ 4.7 µs
$t_{SU;DAT}$	Data set-up time	Note 9.	≥ 250 ns	$\geq 15/f_{XTAL}$
$t_{HD,DAT}$	Data hold time		> 0	$\geq 9/f_{XTAL}$
t_{RD}	SDA rise time	Note 6.	≤ 1 µs	≤ 1 µs
t_{FD}	SDA fall time	Note 7.	≤ 0.3 µs	≤ 0.1 µs
$t_{SU;STO}$	Stop condition set-up time		$\geq 14/f_{XTAL}$	$(DF-3)/(2f_{XTAL})$

Notes

1. $V_{IL} = 0$; $V_{IH} = V_{DD}$; open drain outputs connected to V_{SS}; all other outputs open.
2. Crystal connected between XTAL1 and XTAL2; pin T1 at V_{SS}; pin $\overline{INT}$/T0 at V_{DD}.
3. $V_{DD} = 5$ V; $T_{amb} = 25$ °C; $C_L = 50$ pF.
4. DF is the f_{XTAL} divisor (see Table 2).
5. Values given for ASC = 0; for ASC = 1, $t_{LOW} = (DF-3)/4f_{XTAL}$, $t_{HIGH} = 3(DF+1)/4f_{XTAL}$.
6. Determined by the I^2C-bus capacitance (C_b) and the external pull-up resistor.
7. At maximum allowed I^2C-bus capacitance $C_b = 400$ pF.
8. Determined by program.
9. Independently of ASC, if $t_{LOW} \leq 24/f_{XTAL}$, $t_{SU;DAT} \geq t_{LOW} - 9/f_{XTAL}$.

11 CHARACTERISTIC CURVES

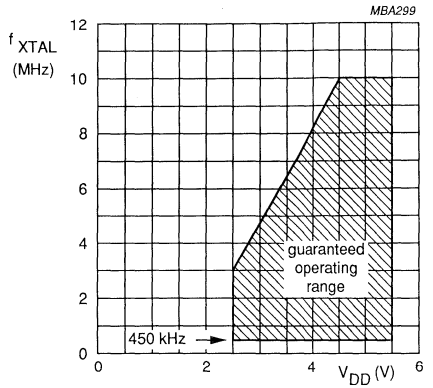
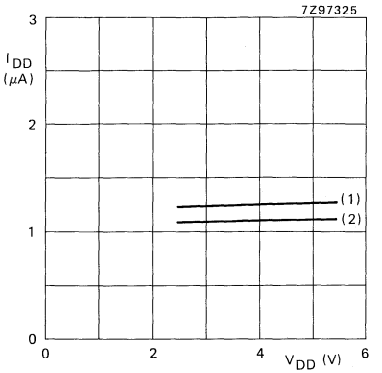
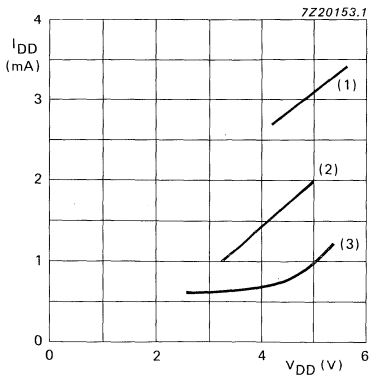


Fig.24 Maximum clock frequency (f_{XTAL}) as a function of supply voltage (V_{DD}).



(1) T_{amb} = 85 °C
(2) T_{amb} = 25 °C

Fig.25 Typical supply current (I_{DD}) in Stop mode as a function of supply voltage (V_{DD}).



(1) f_{XTAL} = 10.0 MHz
(2) f_{XTAL} = 6.0 MHz
(3) f_{XTAL} = 3.58 MHz

Fig.26 Maximum operating supply current (I_{DD}) as a function of supply voltage (V_{DD}).

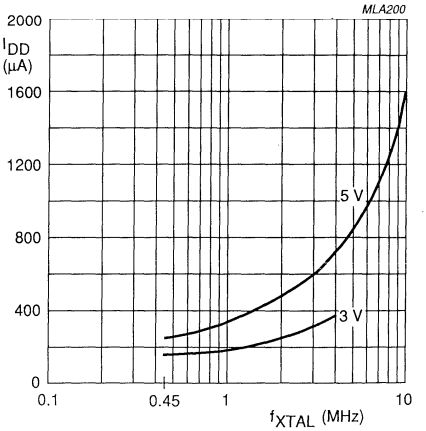
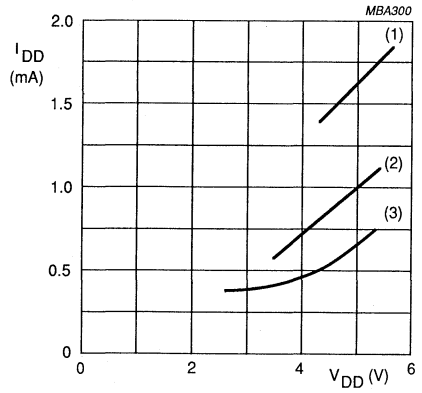


Fig.27 Typical operating supply current (I_{DD}) as a function of frequency at V_{DD} = 3 V and V_{DD} = 5 V.



(1) f_{XTAL} = 10.0 MHz
(2) f_{XTAL} = 6.0 MHz
(3) f_{XTAL} = 3.58 MHz

Fig.28 Maximum supply current (I_{DD}) in Idle mode as a function of supply voltage (V_{DD}).

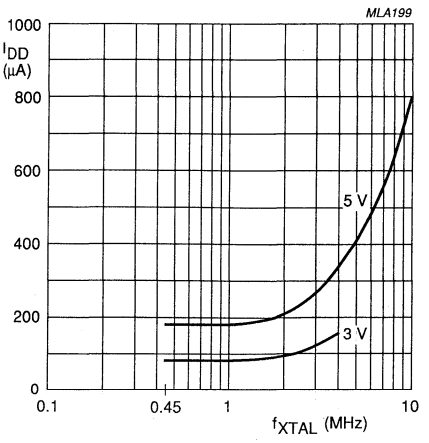


Fig.29 Typical supply current (I_{DD}) in Idle mode as a function of frequency at V_{DD} = 3 V and V_{DD} = 5 V.

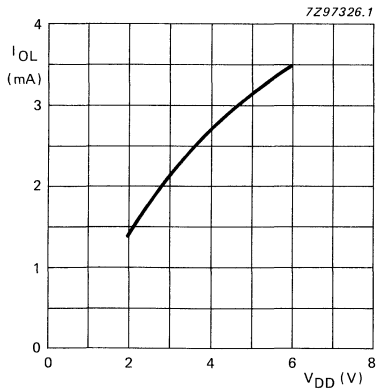


Fig.30 Typical output sink current (I_{OL}) as a function of supply voltage (V_{DD}); V_O = 0.4 V.

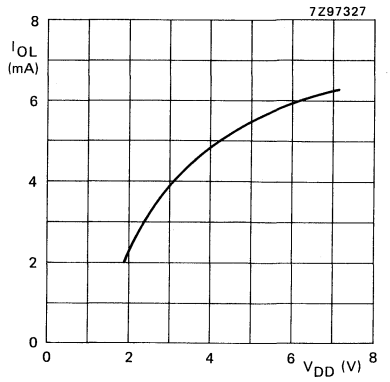
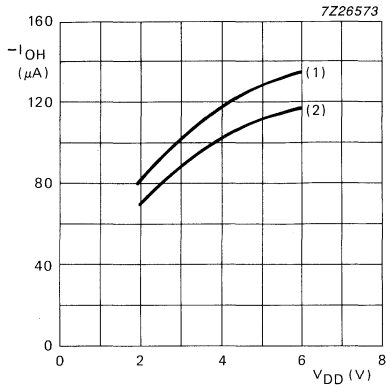


Fig.31 Typical output sink current (I_{OL}), outputs P2.3/SDA and SCL as a function of supply voltage (V_{DD}); V_O = 0.4 V.



(1) $V_O = V_{SS}$
(2) $V_O = 0.7V_{DD}$

Fig.32 Typical output source current ($-I_{OH}$) as a function of supply voltage (V_{DD}).

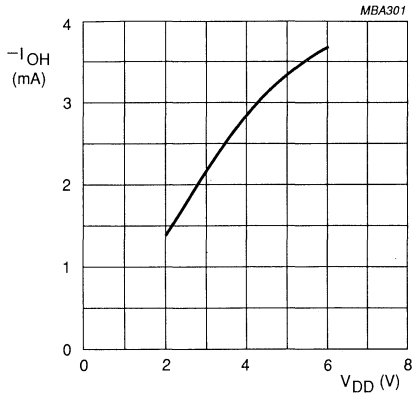


Fig.33 Typical push-pull output source current ($-I_{OH}$) as a function of supply voltage (V_{DD}); $V_O = V_{DD} - 0.4$ V.

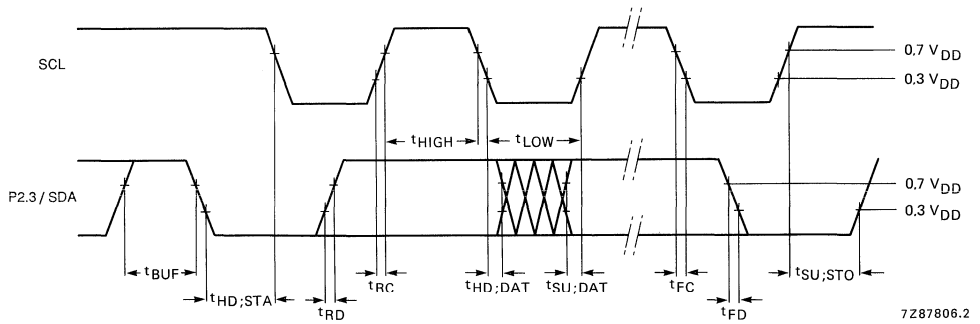


Fig.34 Slave SCL and receiver P2.3/SDA timing (SCL and P2.3/SDA input).

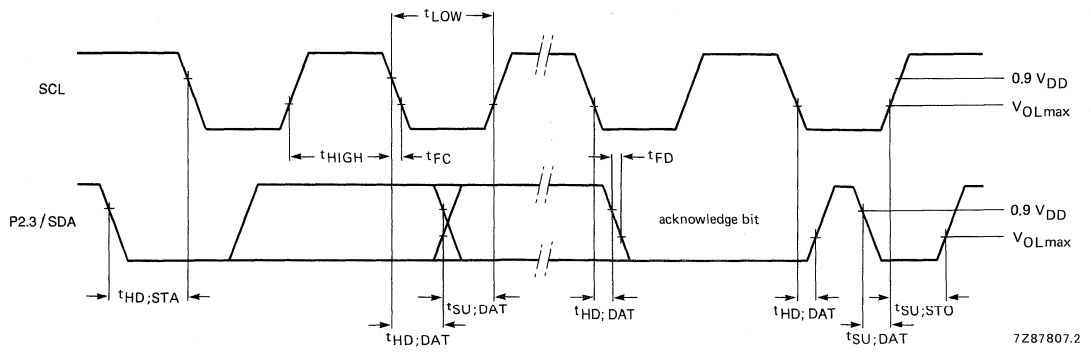


Fig.35 Master SCL and transmitter P2.3/SDA timing (SCL and P2.3/SDA output).

SINGLE-CHIP 8-BIT MICROCONTROLLERS WITH I²C-BUS INTERFACE

DESCRIPTION

An advanced CMOS process is used to manufacture the PCF84C00, PCF84C21/C, PCF84C41/C and PCF84C81/C microcontrollers. The PCF84C21C, PCF84C41C and PCF84C81C operate at a higher clock frequency. Each device has 20 quasi-bidirectional I/O port lines, a serial I/O interface, a single-level vectored interrupt structure, an 8-bit timer/event counter and on-chip clock oscillator and clock circuits. On-chip RAM and ROM content is as follows:

- PCF84C00 – 256 x 8 RAM, external program memory
- PCF84C21 – 64 x 8 RAM, 2 K x 8 ROM
- PCF84C41 – 128 x 8 RAM, 4 K x 8 ROM
- PCF84C81 – 256 x 8 RAM, 8 K x 8 ROM

These efficient controllers also perform well as arithmetic processors. They have facilities for both binary and BCD arithmetic plus bit-handling capabilities. The instruction set is similar to that of the MAB8048.

These microcontrollers are members of the 84CXXX family. For detailed information, consult the 84CXXX family specification.

Features

- 8-bit CPU, ROM, RAM, I/O in a single 28-lead DIL or SO package
- 2K, 4 K or 8 K x ROM; also a ROM-less version
- 64, 128 or 256 x 8 RAM
- 20 quasi-bidirectional I/O port lines
- Two test inputs, one of which is also the external interrupt input
- Single-level vectored interrupts: external, timer/event counter and serial I/O
- I²C hardware interface for serial data transfer on two lines (serial I/O data via an existing port line and clock via a dedicated line)
- 8-bit programmable timer/event counter
- Clock frequency range: 100 kHz to 10 MHz ; C versions: 1 MHz to 12 MHz
- Over 80 instructions (similar to those of the MAB8048) all of 1 or 2 cycles
- Single supply voltage (2,5 to 5,5 V)
- STOP and IDLE modes
- Power-on reset circuit
- Operating temperature range: −40 to +85 °C
- High current on Port 1: I_{OL} = 10 mA at V_{OL} = 1,2 V (all versions except the PCF84C00).

For following sections see 84CXXX family specification

Program memory
Data memory
Program counter stack
IDLE and STOP modes
I/O facilities
Serial I/O
Interrupts
Oscillator
Timer/event counter
Program status word

Program counter
Central processing unit
Conditional branch logic
Test input T1

Power-on reset
Instruction set

PACKAGE OUTLINES

PCF84C21/41/81P: 28-lead DIL; plastic (SOT117).
PCF84C21/41/81T: 28-lead mini-pack; plastic (SO28; SOT136A).
PCF84C00B : 28-lead 'piggy-back' package (supports up to 28-pin EPROM).
PCF84C00T : 56-lead mini-pack; plastic (VSO56; SOT190).

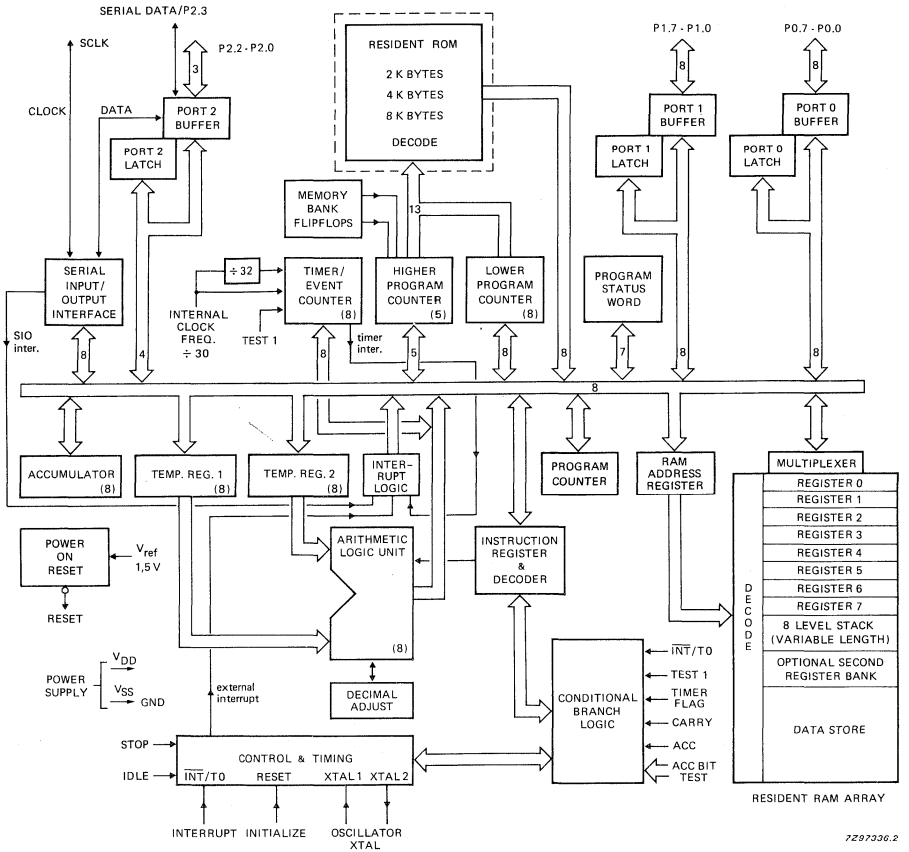


Fig. 1 Block diagram.

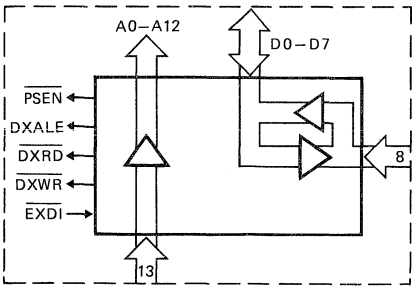


Fig. 1a Replacement of dotted section in Fig. 1, for the PCF84C00T ROM-less version.

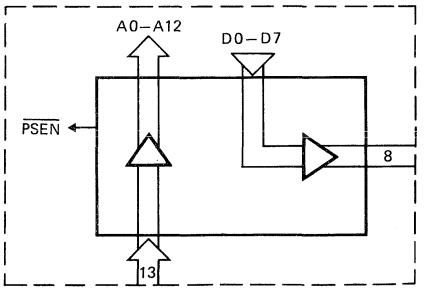


Fig. 1b Replacement of dotted section in Fig. 1, for the PCF84C00B 'piggy-back' version.

PINNING

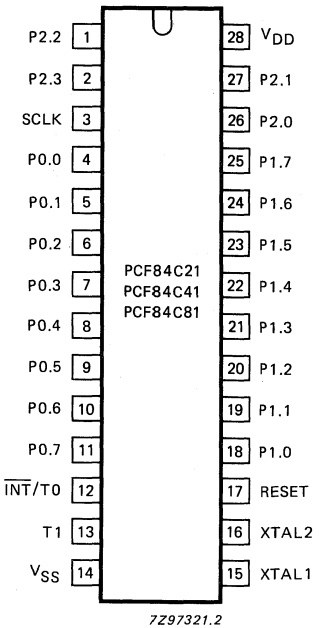


Fig. 2 Pinning diagram.

PIN DESIGNATION

Pin	*Symbol*	*Type*	*Function*
3	SCLK	I/O	**Clock:** bidirectional clock for serial I/O.
4-11	P0.0-P0.7	I/O	**Port 0:** 8-bit quasi-bidirectional I/O port.
12	$\overline{\text{INT}}$/T0	I	**Interrupt/Test 0:** external interrupt input (negative edge triggered)/ test input pin; when used as a test input, this pin is directly tested by conditional branch instructions JT0 and JNT0.
13	T1	I	**Test 1:** test input pin, directly tested by conditional branch instructions JT1 and JNT1. T1 may also be selected as an input to the 8-bit timer/event counter via the STRT CNT instruction.
14	V_{SS}	I	**Ground:** circuit earth potential.
15	XTAL 1	I	**Oscillator input:** input from a crystal which determines the internal oscillator frequency or an external clock generator.
16	XTAL 2	I/O	**Oscillator output:** output of the inverting amplifier.
17	RESET	I/O	**Reset input:** used to initialize the microcontroller (active HIGH); also output of power-on-reset circuit.
18-25	P1.0-P1.7	I/O	**Port 1:** 8-bit quasi-bidirectional I/O port.
26, 27, 1, 2	P2.0-P2.3	I/O	**Port 2:** 4-bit quasi-bidirectional I/O port. P2.3 is the serial data input/output in serial I/O mode.
28	V_{DD}	I	**Power supply:** 2,5 V to 5,5 V.

PINNING (continued)

Pin diagram of the PCF84C00B 'piggy-back' version

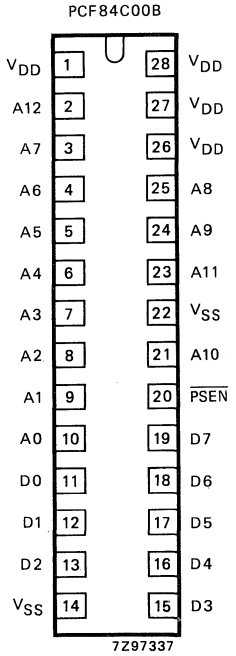


Fig. 3(a) Pinning diagram (top pins) of the PCF84C00B 'piggy-back' version.

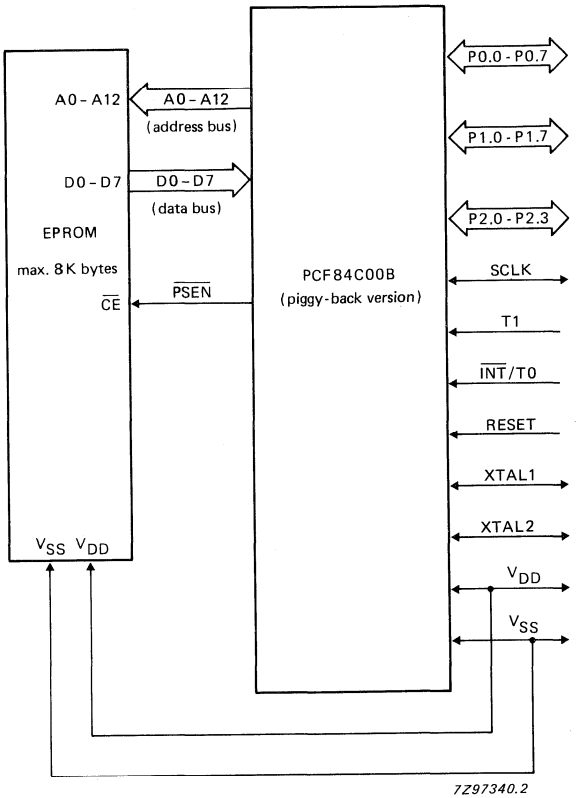


Fig. 3(b) Connection of EPROM to the PCF84C00B 'piggy-back' version.

The PCF84C00B is mainly used for prototyping and for low volume production applications. This device is packaged in a 'piggy-back' package. i.e. a 4 K or 8 K byte EPROM (2732 or 2764) may be mounted in the 24/28 pin socket on top of the package.

Notes

1. The PCF84C00B has 256 bytes of on-chip RAM.
2. Access time for ROMS/EPROMS must be less than 7 x $1/f_{XTAL}$.
3. Bottom pinning is identical to that of Fig. 2.

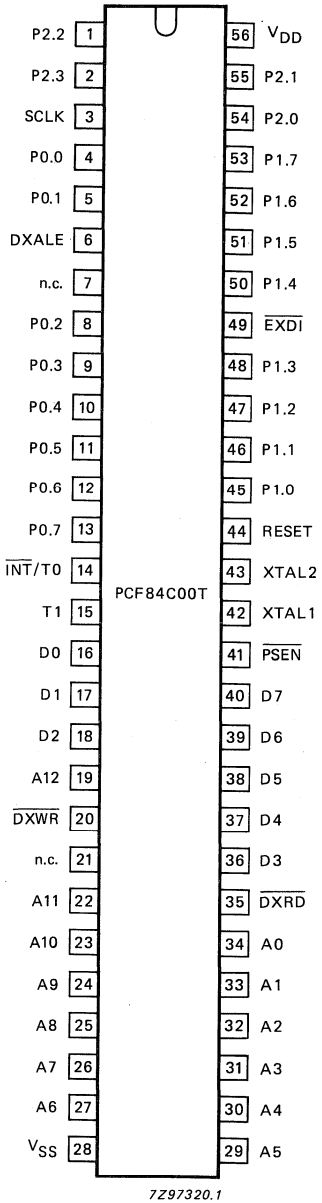


Fig. 4 Pinning diagram; ROM-less version PCF84C00T.

The PCF84C00T may be used for prototyping future derivatives of the PCF84CXX family or for low volume production applications. This device is packaged in a 56-lead VSO outline. Additional signals are available (see pinning information following) to control external program memory and derivative functions.

PIN FUNCTION (continued)

Pin	*Symbol*	*Type*	*Function*
34-29, 27-22, 19	A00-A12	O	**Address bus.** For external memory and pheripherals.
16-18, 36-40	D0-D7	I/O	**Data bus.** For external memory and pheripherals. The specified STOP mode supply current is valid only if external pull-ups are connected to all data lines.
41	$\overline{\text{PSEN}}$	O	**Program store enable** (active LOW). $\overline{\text{PSEN}}$ is used to enable external program memory and is active during TS9 and TS10 of each machine cycle and TS1 of each following cycle. $\overline{\text{PSEN}}$ is HIGH during the STOP mode.
6	DXALE	O	**Address latch enable.** On the falling edge of DXALE, the Dx address can be latched in an external latch. This signal occurs only during execution of the MOV Dx,A, MOV A,Dx, ANL Dx,A and ORL Dx,A instructions, with x = 0 to 255. It is active during TS10 of cycle 1 and the first half of TS1 of cycle 2.
35	$\overline{\text{DXRD}}$	O	**Read strobe** (active LOW). When this signal is active, external registers emulating Dx registers can be read by the data bus. This signal occurs only during execution of MOV A,Dx, ANL Dx,A and ORL Dx,A instructions, with x = 0 to 255. It is active during TS3 and TS4 of cycle 2.
20	$\overline{\text{DXWR}}$	O	**Write strobe** (active LOW). On the rising edge, data on D0-D7 may be written to external registers. This signal occurs only during MOV Dx,A, ANL Dx,A and ORL Dx,A instructions, with x = 0 to 255. It is active during TS7 of cycle 2.
49	$\overline{\text{EXDI}}$	I	**External derivative interrupt** (active LOW). $\overline{\text{EXDI}}$ is 'OR-ed' with the internal serial interrupt and can be used to initiate an interrupt from external hardware emulating derivative functions. $\overline{\text{EXDI}}$ is pulled HIGH internally. The derivative interrupt is polled during time slot TS6*, and is only accepted if an EN SI instruction has been executed and the device is not already executing an interrupt routine. Derivative interrupts are not latched in the PCF84C00.

* The interrupt signal must remain active until the vector address (05 H) is present on the address bus.

FUNCTIONAL DESCRIPTION

ROM-less version PCF84C00T

The PCF84C00T microcontroller contains no on-chip ROM, but has all address and data lines brought out to access an external ROM or EPROM. This version has more pins than the PCF84CXXX with on-chip ROM (see Fig. 1a). The PCF84C00T can address up to 8 K bytes of external program memory, and has 256 bytes of internal data RAM.

'Piggy-back' version PCF84C00B

The PCF84C00B package has standard pinning on the bottom to facilitate insertion as a mask-programmed device. An EPROM can be mounted on top in an additional socket. The total package height is greater than the height of a standard DIL package. The PCF84C00B can address up to 8 K bytes of external ROM/RAM, and has 256 bytes of internal data RAM.

Program memory

The program memory consists of 2, 4 or 8 K bytes of read-only memory (ROM). Each location is directly addressable by the program counter. The ROM is mask-programmed at the factory.

Data memory

Data memory consists of 64, 128 or 256 bytes of random-access memory (RAM). All locations are indirectly addressable using RAM pointer registers; up to 16 designated locations are directly addressable. Memory also includes an 8-level program counter stack addressed by a 3-bit stack pointer.

I/O facilities

Each device has 23 I/O lines arranged as follows:

- Port 0 8-bit parallel port (P0.0-P0.7)
- Port 1 8-bit parallel port (P1.0-P1.7)
- Port 2 4-bit parallel port (P2.0-P2.3)
- $\overline{\text{SCLK}}$ serial I/O clock line
- $\overline{\text{INT}}$/T0 external interrupt and test input. When used as a test input, it can be directly tested by conditional branch instructions JT0 and JNT0
- T1 test input which can alter program sequences when tested by conditional jump instructions JT1 and JNT1. T1 also functions as an input to the 8-bit timer/event counter

Reset

A positive-going signal on the RESET input/output:

- Sets the program counter to zero
- Selects location 0 of memory bank 0 and register bank 0
- Sets the stack pointer to zero (000); pointing to RAM address 8
- Disables the interrupts (external, timer and serial I/O)
- Stops the timer/event counter, then sets it to zero
- Sets the timer prescaler to divide by 32
- Resets the timer flag
- Sets all ports except P2.3 to input mode
- Sets the serial I/O to slave receiver mode and disables the serial I/O
- Cancels IDLE and STOP mode

A negative-going signal on the RESET input/output:

- Sets P2.3/SDA and SCLK to HIGH after a maximum of 30 clock pulses
- Sets the serial I/O to slave receiver mode and disables the serial I/O after a maximum of 30 clock pulses
- Starts program execution after 1866 clock pulses

INSTRUCTION SET

The instruction set consists of over 80 one and two byte instructions. Program code efficiency is high because all RAM locations and all ROM locations on a 256 byte page require only a single byte address.

Table 3 contains the instruction set. Table 2 shows the instruction map and Table 1 details the symbols that are used.

Table 1 Symbols and definitions used in Table 3

symbol	description
A	accumulator
addr	program memory address
Bb	bit designation (b = 0-7)
RBS	register bank select
C	carry bit (bit CY)
CNT	event counter
Dx	mnemonic derivative register (x = 0 ... 255)
direct	8-bit derivative register address
data	8-bit number or expression
I	interrupt
MB	memory bank
MBFF	memory bank flip-flop
P	mnemonic for 'in-page' operation
PC	program counter
Pp	port designation (p = 0, 1 or 2)
PSW	program status word
RB	register bank
Rr	register designation (r = 0-7)
Sn	serial I/O register
SP	stack pointer
T	timer
TF	timer flag
T0, T1	test 0 and 1 inputs
#	immediate data prefix
@	indirect address prefix
(X)	contents of X
((X))	contents of location addressed by X
←	is replaced by
↔	is exchanged with

INSTRUCTION SET (continued)

Table 2 Instruction map

first hexadecimal character of opcode ↓	second hexadecimal character of opcode															
	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
0	NOP	IDLE		ADD A, # data	JMP page 0	EN I	JNTF addr	DEC A	IN A,Pp 0	IN A,Pp 1	IN A,Pp 2		MOV A,Sn 0	MOV A,Sn 1		
1	INC @Rr 0	INC @Rr 1	JB0 addr	ADDC A, # data	CALL page 0	DIS I	JTF addr	INC A	INC Rr 0	INC Rr 1	INC Rr 2	INC Rr 3	INC Rr 4	INC Rr 5	INC Rr 6	INC Rr 7
2	XCH A,@Rr 0	XCH A,@Rr 1	STOP	MOV A, # data	JMP page 1	EN TCNTI	JNT0 addr	CLR A	XCH A,Rr 0	XCH A,Rr 1	XCH A,Rr 2	XCH A,Rr 3	XCH A,Rr 4	XCH A,Rr 5	XCH A,Rr 6	XCH A,Rr 7
3	XCHD A,@Rr 0	XCHD A,@Rr 1	JB1 addr		CALL page 1	DIS TCNTI	JT0 addr	CPL A	OUTL Pp,A 0	OUTL Pp,A 1	OUTL Pp,A 2		MOV Sn,A 0	MOV Sn,A 1	MOV Sn,A 2	
4	ORL A,@Rr 0	ORL A,@Rr 1	MOV A, T	ORL A, # data	JMP page 2	STRT CNT	JNT1 addr	SWAP A	ORL A,Rr 0	ORL A,Rr 1	ORL A,Rr 2	ORL A,Rr 3	ORL A,Rr 4	ORL A,Rr 5	ORL A,Rr 6	ORL A,Rr 7
5	ANL A,@Rr 0	ANL A,@Rr 1	JB2 addr	ANL A, # data	CALL page 2	STRT T	JT1 addr	DA A	ANL A,Rr 0	ANL A,Rr 1	ANL A,Rr 2	ANL A,Rr 3	ANL A,Rr 4	ANL A,Rr 5	ANL A,Rr 6	ANL A,Rr 7
6	ADD A,@Rr 0	ADD A,@Rr 1	MOV T, A		JMP page 3	STOP TCNT		RRC A	ADD A,Rr 0	ADD A,Rr 1	ADD A,Rr 2	ADD A,Rr 3	ADD A,Rr 4	ADD A,Rr 5	ADD A,Rr 6	ADD A,Rr 7
7	ADDC A,@Rr 0	ADDC A,@Rr 1	JB3 addr		CALL page 3			RR A	ADDC A,Rr 0	ADDC A,Rr 1	ADDC A,Rr 2	ADDC A,Rr 3	ADDC A,Rr 4	ADDC A,Rr 5	ADDC A,Rr 6	ADDC A,Rr 7
8				RET	JMP page 4	EN SI			ORL Pp, # data 0	ORL Pp, # data 1	ORL Pp, # data 2		MOV A, Dx	MOV Dx, A	ANL Dx, A	ORL Dx, A
9			JB4 addr	RETR	CALL page 4	DIS SI	JNZ addr	CLR C	ANL Pp,# data 0	ANL Pp,# data 1	ANL Pp,# data 2		MOV Sn,# data 0	MOV Sn,# data 1	MOV Sn,# data 2	
A	MOV @Rr,A 0	MOV @Rr,A 1		MOVP A,@A	JMP page 5	SEL MB2		CPL C	MOV Rr,A 0	MOV Rr,A 1	MOV Rr,A 2	MOV Rr,A 3	MOV Rr,A 4	MOV Rr,A 5	MOV Rr,A 6	MOV Rr,A 7
B	MOV @Rr, # data 0	MOV @Rr, # data 1	JB5 addr	JMPP @A	CALL page 5	SEL MB3			MOV Rr, # data 0	MOV Rr, # data 1	MOV Rr, # data 2	MOV Rr, # data 3	MOV Rr, # data 4	MOV Rr, # data 5	MOV Rr, # data 6	MOV Rr, # data 7
C	DEC @Rr 0	DEC @Rr 1			JMP page 6	SEL RB0	JZ addr	MOV A, PSW	DEC Rr 0	DEC Rr 1	DEC Rr 2	DEC Rr 3	DEC Rr 4	DEC Rr 5	DEC Rr 6	DEC Rr 7
D	XRL A,@Rr 0	XRL A,@Rr 1	JB6 addr	XRL A, # data	CALL page 6	SEL RB1		MOV PSW, A	XRL A,Rr 0	XRL A,Rr 1	XRL A,Rr 2	XRL A,Rr 3	XRL A,Rr 4	XRL A,Rr 5	XRL A,Rr 6	XRL A,Rr 7
E	DJNX @Rr, addr 0	DJNX @Rr, addr 1			JMP page 7	SEL MB0	JNC addr	RL A	DJNZ Rr, addr 0	DJNZ Rr, addr 1	DJNZ Rr, addr 2	DJNZ Rr, addr 3	DJNZ Rr, addr 4	DJNZ Rr, addr 5	DJNZ Rr, addr 6	DJNZ Rr, addr 7
F	MOV A,@Rr 0	MOV A,@Rr 1	JB7 addr		CALL page 7	SEL MB1	JC addr	RLC A	MOV A,Rr 0	MOV A,Rr 1	MOV A,Rr 2	MOV A,Rr 3	MOV A,Rr 4	MOV A,Rr 5	MOV A,Rr 6	MOV A,Rr 7

Table 3 Instruction set

	mnemonic	opcode (hex.)	bytes/ cycles	description	function		notes
ACCUMULATOR	ADD A, Rr	6*	1/1	Add register contents to A	(A)←(A) + (Rr)	r = 0–7	1
	ADD A, @Rr	60 61	1/1	Add RAM data, addressed by Rr, to A	(A)←(A) + ((R0)) (A)←(A) + ((R1))		1
	ADD A, #data	03 data	2/2	Add immediate data to A	(A)←(A) + data		1
	ADDC A, Rr	7*	1/1	Add carry and register contents to A	(A)←(A) + (Rr) + (C)	r = 0–7	1
	ADDC A, @Rr	70 71	1/1	Add carry and RAM data, addressed by Rr, to A	(A)←(A) + ((R0)) + (C) (A)←(A) + ((R1)) + (C)		1
	ADDC A, #data	13 data	2/2	Add carry and immediate data to A	(A)←(A) + data + (C)		1
	ANL A, Rr	5*	1/1	'AND' Rr with A	(A)←(A) AND (Rr)	r = 0–7	
	ANL A, @Rr	50 51	1/1	'AND' RAM data, addressed by Rr, with A	(A)←(A) AND ((R0)) (A)←(A) AND ((R1))		
	ANL A, #data	53 data	2/2	'AND' immediate data with A	(A)←(A) AND data		
	ORL A, Rr	4*	1/1	'OR' Rr with A	(A)←(A) OR (Rr)	r = 0–7	
	ORL A, @Rr	40 41	1/1	'OR' RAM data, addressed by Rr, with A	(A)←(A) OR ((R0)) (A)←(A) OR ((R1))		
	ORL A, #data	43 data	2/2	'OR' immediate data with A	(A)←(A) OR data		
	XRL A, Rr	D*	1/1	'XOR' Rr with A	(A)←(A) XOR (Rr)	r = 0–7	
	XRL A, @Rr	D0 D1	1/1	'XOR' RAM, addressed by Rr, with A	(A)←(A) XOR ((R0)) (A)←(A) XOR ((R1))		
	XRL A, #data	D3 data	2/2	'XOR' immediate data with A	(A)←(A) XOR data		
	INC A	17	1/1	increment A by 1	(A)←(A) + 1		
	DEC A	07	1/1	decrement A by 1	(A)←(A) −1		
	CLR A	27	1/1	clear A to zero	(A)←0		
	CPL A	37	1/1	one's complement A	(A)←NOT(A)		
	RL A	E7	1/1	rotate A left	$(A_{n+1})\leftarrow(A_n)$ $(A_0)\leftarrow(A_7)$	n = 0–6	

ACCUMULATOR (cont.)	RLC A	F7	1/1	rotate A left through carry	$(A_{n+1})\leftarrow A_n$ $(A_0)\leftarrow(C)$, $(C)\leftarrow(A_7)$	n = 0–6	2
	RR A	77	1/1	rotate A right	$(A_n)\leftarrow(A_{n+1})$ $(A_7)\leftarrow(A_0)$	n = 0–6	
	RRC A	67	1/1	rotate A right through carry	$(A_n)\leftarrow(A_{n+1})$ $(A_7)\leftarrow(C)$, $(C)\leftarrow(A_0)$	n = 0–6	2
	DA A	57	1/1	decimal adjust A			2
	SWAP A	47	1/1	swap nibbles of A	$(A_{4-7})\leftrightarrow(A_{0-3})$		
DATA MOVES	MOV A, Rr	F*	1/1	move register contents to A	(A)←(Rr)	r = 0–7	
	MOV A, @Rr	F0 F1	1/1	move RAM data, addressed by Rr, to A	(A)←((R0)) (A)←((R1))		
	MOV A, #data	23 data	2/2	move immediate data to A	(A)←data		
	MOV Rr, A	A*	1/1	move accumulator contents to register	(Rr)←(A)	r = 0–7	
	MOV @Rr, A	A0 A1	1/1	move accumulator contents to RAM location addressed by Rr	((R0))←(A) ((R1))←(A)		
	MOV Rr, #data	B* data	2/2	move immediate data to Rr	(Rr)←data		
	MOV @Rr, #data	B0 data B1 data	2/2	move immediate data to RAM location addressed by Rr	((R0))←data ((R1))←data		
	XCH A, Rr	2*	1/1	exchange accumulator contents with Rr	(A)↔(Rr)	r = 0–7	
	XCH A, @Rr	20 21	1/1	exchange accumulator contents with RAM data addressed by Rr	(A)↔((R0)) (A)↔((R1))		
	XCHD A, @Rr	30 31	1/1	exchange lower nibbles of A and RAM data addressed by Rr	$(A_{0-3})\leftrightarrow((R0_{0-3}))$ $(A_{0-3})\leftrightarrow((R1_{0-3}))$		
	MOV A, PSW	C7	1/1	move PSW contents to accumulator	(A)←(PSW)		
	MOV PSW, A	D7	1/1	move accumulator bit 3 to PSW_3	$(PSW_3)\leftarrow(A_3)$		3
	MOVP A, @A	A3	1/2	move indirectly addressed data in current page to A	$(PC_{0-7})\leftarrow(A)$, $(A)\leftarrow((PC))$		
FLAGS	CLR C	97	1/1	clear carry bit	(C)←0		2
	CPL C	A7	1/1	complement carry bit	(C)←NOT(C)		2

INSTRUCTION SET (continued)

	mnemonic	opcode (hex.)	bytes/ cycles	description	function	notes
REGISTER	INC Rr	1*	1/1	increment register by 1	(Rr)←(Rr) + 1 r = 0–7	
	INC @Rr	10 11	1/1	increment RAM data, addressed by Rr, by 1	((R0))←((R0)) + 1 ((R1))←((R1)) + 1	
	DEC Rr	C*	1/1	decrement register by 1	(Rr)←(Rr) − 1 r = 0–7	
	DEC @Rr	C0 C1	1/1	decrement RAM data, addressed by Rr, by 1	((R0))←((R0)) − 1 ((R1))←((R1)) − 1	
BRANCH	JMP addr	● 4 addr	2/2	unconditional jump within a 2 K bank	(PC_{8-10})←$addr_{8-10}$ (PC_{0-7})←$addr_{0-7}$ (PC_{11-12})←MBFF 0–1	
	JMPP @A	B3	1/2	indirect jump within a page	(PC_{0-7})←((A))	
	DJNZ Rr, addr	E* addr	2/2	decrement Rr by 1 and jump if not zero to addr	(Rr)←(Rr) − 1 r = 0–7 if (Rr) not zero (PC_{0-7})←addr	
	DJNZ @Rr, addr	E0	2/2	decrement RAM data, addressed by Rr by 1 and jump if not zero to addr	((R0))←((R0)) − 1 if ((R0)) not zero (PC_{0-7})←addr	
		E1			((R1))←((R1)) − 1 if ((R1)) not zero (PC_{0-7})←addr	
	JBb addr	▲ 2 addr	2/2	jump to addr if Acc. bit b = 1	if b = 1 : (PC_{0-7})←addr b = 0–7	
	JC addr	F6 addr	2/2	jump to addr if C = 1	if C = 1 : (PC_{0-7})←addr	
	JNC addr	E6 addr	2/2	jump to addr if C = 0	if C = 0 : (PC_{0-7})←addr	
	JZ addr	C6 addr	2/2	jump to addr if A = 0	if A = 0 : (PC_{0-7})←addr	
	JNZ addr	96 addr	2/2	jump to addr if A is NOT zero	if A≠0 : (PC_{0-7})←addr	
	JT0 addr	36 addr	2/2	jump to addr if T0 = 1	if T0 = 1: (PC_{0-7})←addr	
	JNT0 addr	26 addr	2/2	jump to addr if T0 = 0	if T0 = 0: (PC_{0-7})←addr	
	JT1 addr	56 addr	2/2	jump to addr if T1 = 1	if T1 = 1: (PC_{0-7})←addr	
	JNT1 addr	46 addr	2/2	jump to addr if T1 = 0	if T1 = 0: (PC_{0-7})←addr	
	JTF addr	16 addr	2/2	jump to addr if Timer Flag = 1	if TF = 1: (PC_{0-7})←addr	4
	JNTF addr	06 addr	2/2	jump to addr if Timer Flag = 0	if TF = 0: (PC_{0-7})←addr	

TIMER/EVENT COUNTER	MOV A, T	42	1/1	move timer/event counter contents to accumulator	(A)←(T)	
	MOV T, A	62	1/1	move accumulator contents to timer/event counter	(T)←(A)	
	STRT CNT	45	1/1	start event counter		
	STRT T	55	1/1	start timer		
	STOP TCNT	65	1/1	stop timer/event counter		
	EN TCNTI	25	1/1	enable timer/event counter interrupt		
	DIS TCNTI	35	1/1	disable timer/event counter interrupt		
CONTROL	EN I	05	1/1	enable external interrupt		
	DIS I	15	1/1	disable external interrupt		
	SEL RB0	C5	1/1	select register bank 0	(RBS)←0	5
	SEL RB1	D5	1/1	select register bank 1	(RBS)←1	5
	SEL MB0	E5	1/1	select program memory bank 0	(MBFF0)←0, (MBFF1)←0	
	SEL MB1	F5	1/1	select program memory bank 1	(MBFF0)←1, (MBFF1)←0	10
	SEL MB2	A5	1/1	select program memory bank 2	(MBFF0)←0, (MBFF1)←1	10
	SEL MB3	B5	1/1	select program memory bank 3	(MBFF0)←1, (MBFF1)←1	10
	STOP	22	1/1	enter STOP mode		
	IDLE	01	1/1	enter IDLE mode		
SUBROUTINE	CALL addr	▲ 4 addr	2/2	jump to subroutine	((SP))←(PC), ($PSW_{4, 6, 7}$) (SP)←(SP) + 1 (PC_{8-10})←$addr_{8-10}$ (PC_{0-7})←$addr_{0-7}$ (PC_{11-12})←MBFF 0–1	6
	RET	83	1/2	return from subroutine	(SP)←(SP) – 1 (PC)←((SP))	6
	RETR	93	1/2	return from interrupt and restore bits 4, 6, 7 of PSW	(SP)←(SP) – 1 ($PSW_{4, 6, 7}$) + (PC)←((SP))	6

INSTRUCTION SET (continued)

	mnemonic	opcode (hex.)	bytes/ cycles	description	function	notes
PARALLEL INPUT/OUTPUT	IN A, Pp	08 09 0A	1/2	input port p data to accumulator	(A)←(P0) (A)←(P1) (A)←(P2)	 7
	OUTL Pp, A	38 39 3A	1/2	output accumulator data to port p	(P0)←(A) (P1)←(A) (P2)←(A)	
	ANL Pp, #data	98 data 99 data 9A data	2/2	AND port p data with immediate data	(P0)←(P0) AND data (P1)←(P1) AND data (P2)←(P2) AND data	
	ORL Pp, #data	88 data 89 data 8A data	2/2	OR port p data with immediate data	(P0)←(P0) OR data (P1)←(P1) OR data (P2)←(P2) OR data	
DERIVATIVE INPUT/OUTPUT	MOV A, Dx	8C direct	2/2	move derivative register contents to accumulator	(A)←(Dx) x = 0 to 255	8
	MOV Dx, A	8D direct	2/2	move accumulator contents to derivative register	(Dx)←(A) x = 0 to 255	8
	ANL Dx, A	8E direct	2/2	AND derivative register with accumulator	(Dx)←(Dx) AND (A) x = 0 to 255	8
	ORL Dx, A	8F direct	2/2	OR derivative register with accumulator	(Dx)←(Dx) OR (A) x = 0 to 255	8

	mnemonic	opcode (hex.)	bytes/ cycles	description	function	notes
SERIAL INPUT/OUTPUT	MOV A, S_n	0C 0D	1/2	move serial I/O register contents to accumulator	(A)←(S0) (A)←(S1)	9
	MOV S_n, A	3C 3D 3E	1/2	move accumulator contents to serial I/O register	(S0)←(A) (S1)←(A) (S2)←(A)	
	MOV S_n, #data	9C data 9D data 9E data	2/2	move immediate data to serial I/O register	(S0)←data (S1)←data (S2)←data	
	EN SI	85	1/1	enable serial I/O interrupt		
	DIS SI	95	1/1	disable serial I/O interrupt		
	NOP	00	1/1	no operation		

Notes to Table 3

1. PSW CY, AC affected
2. PSW CY affected
3. PSW PS affected
4. Execution of a JTF or JNTF instruction resets the Timer Flag (TF).

* : 8, 9, A, B, C, D, E, F
● : 0, 2, 4, 5, 6, A, C, E
▲ : 1, 3, 5, 7, 9, B, D, F

5. PSW RBS affected
6. PSW SP_0, SP_1, SP_2 affected
7. (A) = 0000, P2.3, P2.2, P2.1, P2.0.
8. Instructions for PCF84C00T only.
9. (S1) has a different meaning for read and write operation, see serial I/O interface.
10. SEL MB1, SEL MB2 and SEL MB3 may not be used within interrupt routines.

RATINGS

Limiting values in accordance with the Absolute Maximum System (IEC 134)

Supply voltage	V_{DD}	−0,8 to +8	V
All input voltages	V_I	−0,5 to V_{DD} +0,5	V
DC current into any input or output	$\pm I_I$, $\pm I_O$	max. 10	mA
Total power dissipation (see note)	P_{tot}	max. 125	mW
Storage temperature range	T_{stg}	−65 to +150	°C
Operating ambient temperature range (if P_{tot} max. = 100 mW)	T_{amb}	−40 to +70	°C
Operating ambient temperature range (if P_{tot} max. = 30 mW)	T_{amb}	−40 to +85	°C
Operating junction temperature	T_j	max. 90	°C

HANDLING

Inputs and outputs are protected against electrostatic discharge in normal handling. However, it is good practice to take normal precautions appropriate to handling MOS devices (see 'Handling MOS devices').

Note

Thermal resistance (junction to ambient)

for SOT117	$R_{th\ j\text{-}a}$	max. 120	K/W
for SOT136A	$R_{th\ j\text{-}a}$	max. 150	K/W
for SOT190	$R_{th\ j\text{-}a}$	max. 110	K/W

DC CHARACTERISTICS

V_{DD} = 2,5 to 5,5 V; V_{SS} = 0 V; T_{amb} = −40 to +85 °C; all voltages with respect to V_{SS}; unless otherwise specified.

parameter	symbol	min.	typ.	max.	unit
Supply voltage					
operating (see Fig. 9)	V_{DD}	2,5	–	5,5	V
Supply current					
operating (see Fig. 10; not valid for PCF84C00)					
at V_{DD} = 5 V; f_{XTAL} = 10 MHz (note 2)	I_{DD}	–	1,6	3,2	mA
at V_{DD} = 5 V; f_{XTAL} = 6 MHz (note 2)	I_{DD}	–	1	2	mA
at V_{DD} = 3 V; f_{XTAL} = 3,58 MHz (note 2)	I_{DD}	–	0,3	0,6	mA
IDLE mode (see Fig. 11; not valid for PCF84C00)					
at V_{DD} = 5 V; f_{XTAL} = 10 MHz (note 2)	I_{DD}	–	0,8	1,6	mA
at V_{DD} = 5 V; f_{XTAL} = 6 MHz (note 2)	I_{DD}	–	0,5	1	mA
at V_{DD} = 3 V; f_{XTAL} = 3,58 MHz (note 2)	I_{DD}	–	0,15	0,4	mA
STOP mode (see Fig. 17, note 1 and note 2)					
at V_{DD} = 2,5 V; T_{amb} = 25 °C	I_{DD}	–	1,2	2,5	μA
at V_{DD} = 2,5 V; T_{amb} = 85 °C	I_{DD}	–	–	10	μA
Inputs					
Input voltage LOW	V_{IL}	0	–	$0,3V_{DD}$	V
Input voltage HIGH	V_{IH}	$0,7V_{DD}$	–	V_{DD}	V
Input leakage current					
at $V_{SS} < V_I < V_{DD}$	$\pm I_{IL}$	–	–	1	μA
Outputs					
Output sink current LOW					
at V_{DD} = 5 V ± 10%; V_O = 0,4 V					
except P2.3/SDA, SCLK (see Fig. 13) and port 1	I_{OL}	1,6	3	–	mA
P2.3/SDA, SCLK (see Fig. 14)	I_{OL}	3	–	–	mA
P1.0-P1.7 (not PCF84C00) at V_{OL} = 1,2 V	I_{OL}	10	–	–	mA
Pull-up output source current HIGH (see Fig. 15)					
at V_{DD} = 5 V ± 10%; $V_O = 0,7V_{DD}$	$-I_{OH}$	40	–	–	μA
at V_{DD} = 5 V ± 10%; $V_O = V_{SS}$	$-I_{OH}$	–	–	400	μA
Push-pull output source current HIGH					
at V_{DD} = 5 V ± 10%; $V_O = V_{DD}$−0,4 V	$-I_{OH}$	1,6	3	–	mA

Note 1: Crystal connected between XTAL1 and XTAL2; T1 = V_{SS}; $\overline{INT}$ = V_{DD}.

Note 2: V_{IL} = V_{SS}; V_{IH} = V_{DD}; all outputs unloaded; all open drain outputs connected to V_{SS}.

AC CHARACTERISTICS

V_{DD} = 2,5 to 5,5 V; V_{SS} = 0 V; T_{amb} = −40 to +85 °C. All voltages with respect to V_{SS} unless otherwise specified.

parameter	symbol	min.	typ.	max.	unit
Rise time all outputs (note 1)	t_R	–	30	–	ns
Fall time all outputs (note 1)	t_F	–	30	–	ns
Cycle time (= 30 CP; note 2)	t_{CY}	3	–	30	μs
PCF84C00/non-standard pins:					
Control pulse width	t_{CC}	–	9	–	CP
Address to $\overline{PSEN}$ set-up	t_{AS}	–	1,5	–	CP
Data to $\overline{PSEN}$ set-up	t_{DS}	–	2	–	CP
Data hold time	t_{DR}	0	–	–	ns
Data out to $\overline{DWXR}$ set-up	t_{SDO}	–	2	–	CP
Data out to $\overline{DXWR}$ hold	t_{HDO}	–	1	–	CP
Time from DXALE to $\overline{PSEN}$	t_{SLPH}	–	1,5	–	CP
Data-in to $\overline{DXRD}$ set-up	t_{DS1}	–	2,5	–	CP
Data-in to $\overline{DXRD}$ hold	t_{DR1}	0	–	–	ns
HIGH time of DXALE	t_{DXALE}	–	4,5	–	CP
LOW time of $\overline{DXRD}$	t_{DXRD}	–	6	–	CP
LOW time of $\overline{DXWR}$	t_{DXWR}	–	3	–	CP

Notes:

1. At V_{DD} = 5 V; T_{amb} = + 25 °C; C_L = 50 pF.
2. 1 Time slot (TS) = 3 CP, 1 clock pulse (CP) = $1/f_{XTAL}$.

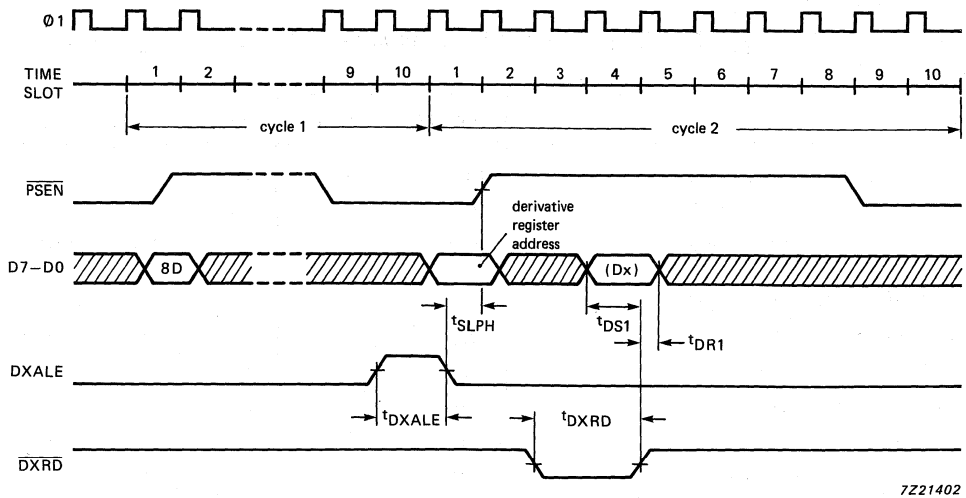


Fig. 5 MOV A,Dx timing (PCF84C00T only).

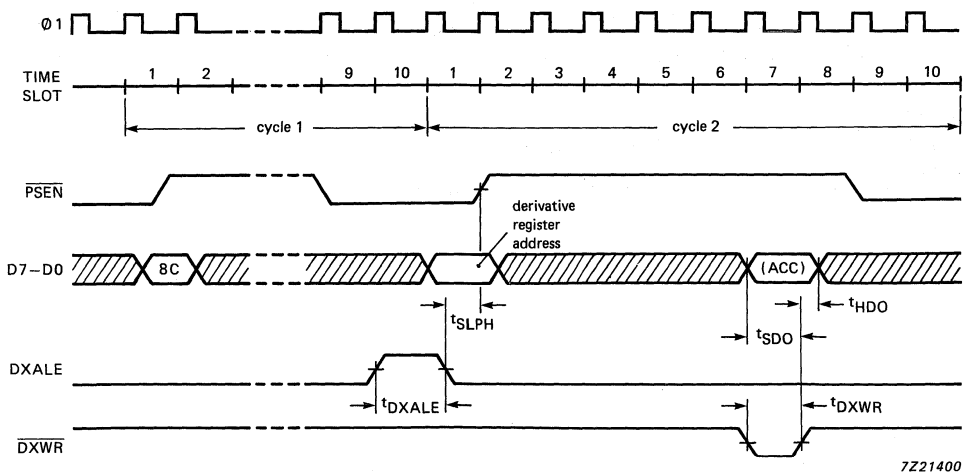


Fig. 6 MOV Dx,A timing (PCF84C00T only).

AC CHARACTERISTICS (continued)

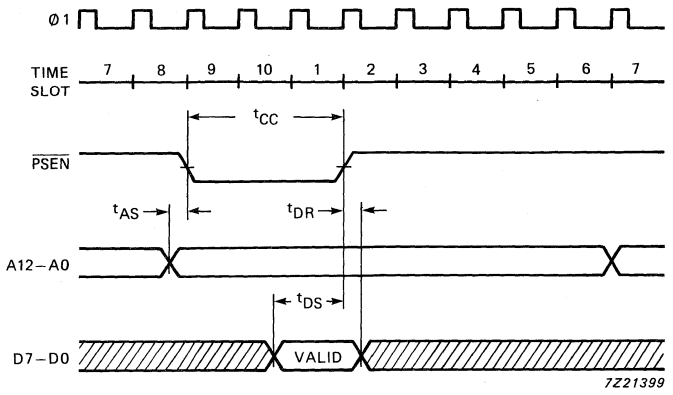


Fig. 7 External memory access timing (PCF84C00T and PCF8400B).

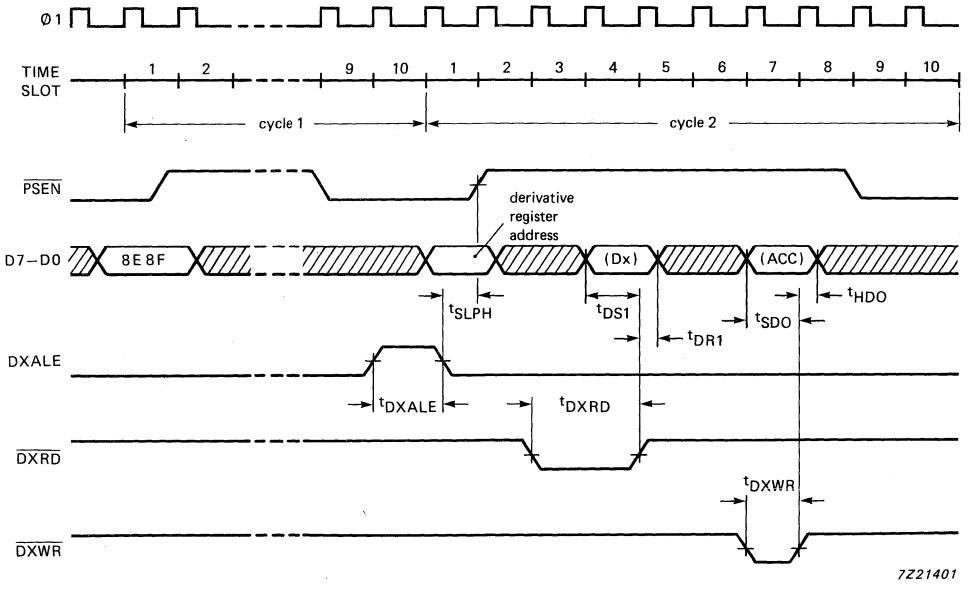


Fig. 8 ANL/ORL derivative interface timing (PCF84C00T only).

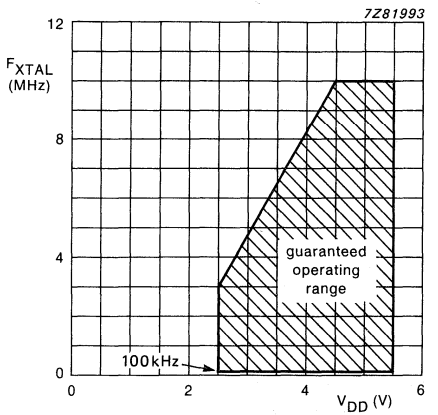


Fig. 9(a) Maximum clock frequency (f_{XTAL}) as a function of the supply voltage (PCF84C00, PCF84C21, PCF84C41 and PCF84C81).

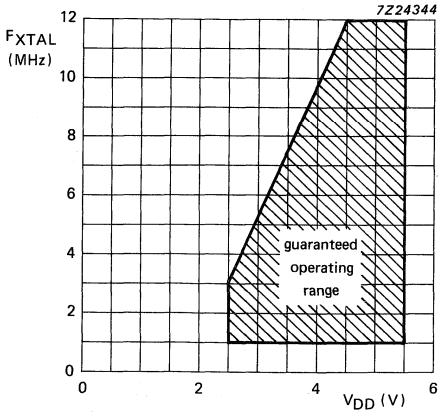
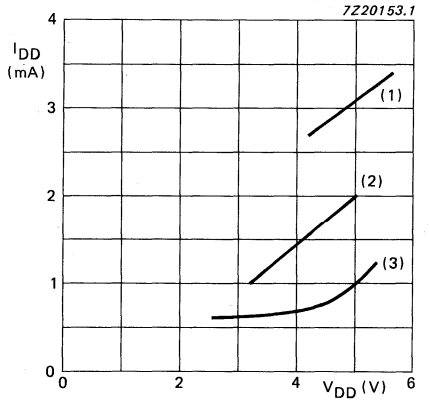
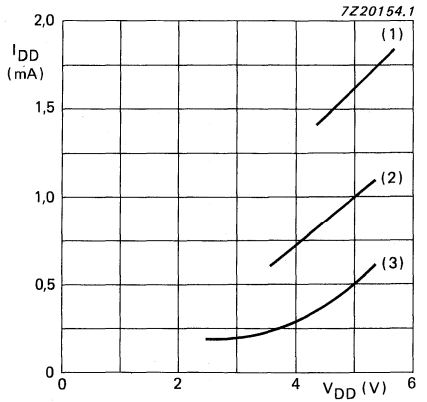


Fig. 9(b) Maximum clock frequency (f_{XTAL}) as a function of the supply voltage (PCF84C21C, PCF84C41C and PCF84C81C).



(1) clock frequency = 10 MHz
(2) clock frequency = 6 MHz
(3) clock frequency = 3,58 MHz

Fig. 10 Maximum supply current (I_{DD}) in operation mode as a function of the supply voltage (V_{DD}).



(1) clock frequency = 10 MHz
(2) clock frequency = 6 MHz
(3) clock frequency = 3,58 MHz

Fig. 11 Maximum supply current (I_{DD}) in IDLE mode as a function of the supply voltage (V_{DD}).

AC CHARACTERISTICS (continued)

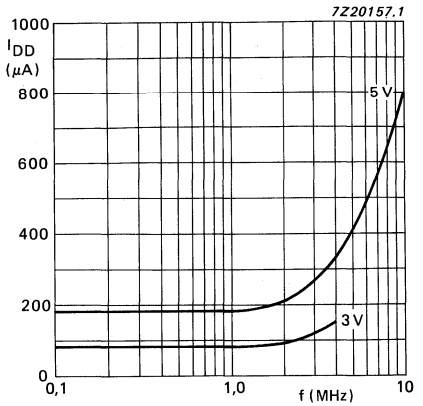


Fig. 12 Typical supply current during IDLE mode as a function of frequency at V_{DD} = 3 V and V_{DD} = 5 V.

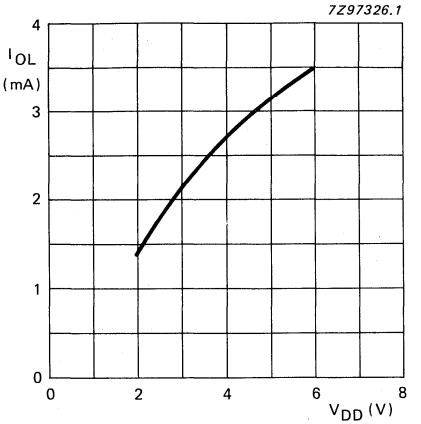


Fig. 13 Typical output sink current (I_{OL}), outputs P0.0 to P0.7 and P2.0 to P2.2, as a function of the supply voltage (V_{DD}); V_O = 0,4 V.

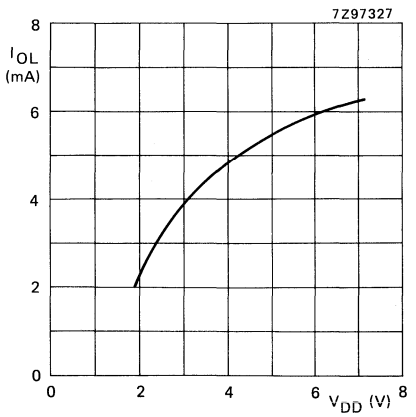


Fig. 14(a) Typical output sink current (I_{OL}), outputs P2.3/SDA and SCLK, as a function of the supply voltage (V_{DD}); V_O = 0,4 V.

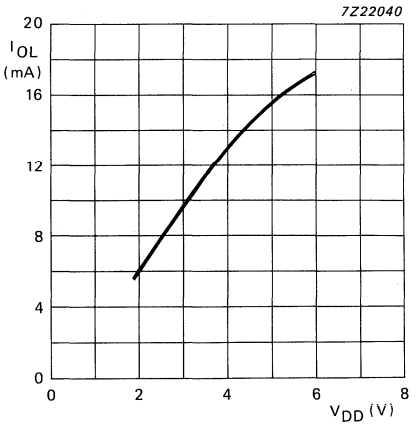
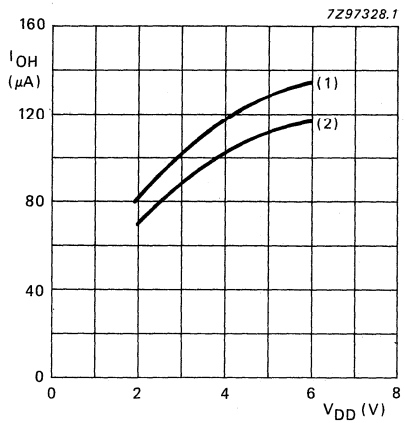


Fig. 14(b) Typical output sink current (I_{OL}), outputs P1.0 to P1.7, as a function of the supply voltage (V_{DD}); V_O = 1,2 V.



(1) $V_O = V_{SS}$
(2) $V_O = 0{,}7\ V_{DD}$

Fig. 15 Typical output source current ($-I_{OH}$) as a function of the supply voltage (V_{DD}).

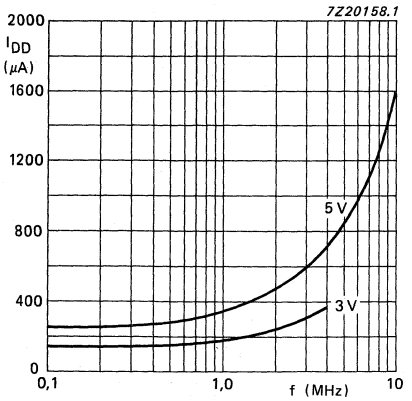
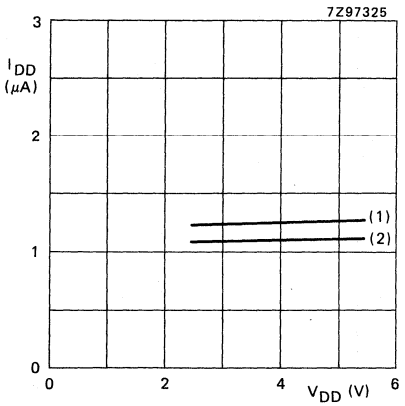


Fig. 16 Typical supply current during operating mode as a function of frequency at V_{DD} = 3 V and V_{DD} = 5 V.



(1) $T_{amb} = 85\ ^{o}C$
(2) $T_{amb} = 25\ ^{o}C$

Fig. 17 Typical supply current (I_{DD}) in STOP mode as a function of the supply voltage (V_{DD}).

Table 4 Input timing shown in Fig. 18

symbol	timing
t_{BUF}	$\geqslant 14t_{XTAL}$
$t_{HD;STA}$	$\geqslant 14t_{XTAL}$
t_{HIGH}	$\geqslant 17t_{XTAL}$
t_{LOW}	$\geqslant 17t_{XTAL}$
$t_{SU;STO}$	$\geqslant 14t_{XTAL}$
$t_{HD;DAT}$	> 0
$t_{SU;DAT}$	$\geqslant 250$ ns
t_{RD}	$\leqslant 1\ \mu s$
t_{RC}	$\leqslant 1\ \mu s$
t_{FD}	$\leqslant 1\ \mu s$
t_{FC}	$\leqslant 0{,}3\ \mu s$

Notes to Table 4

t_{XTAL} = one period of the XTAL input frequency (f_{XTAL})
= 167 ns for f_{XTAL} = 6 MHz
These figures apply to all modes.

AC CHARACTERISTICS (continued)

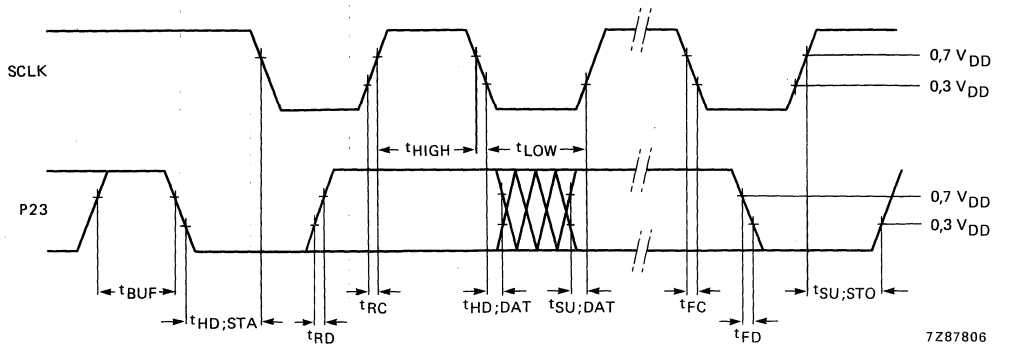


Fig. 18 Timing requirements for the P2.3 and SCLK *input* signals.

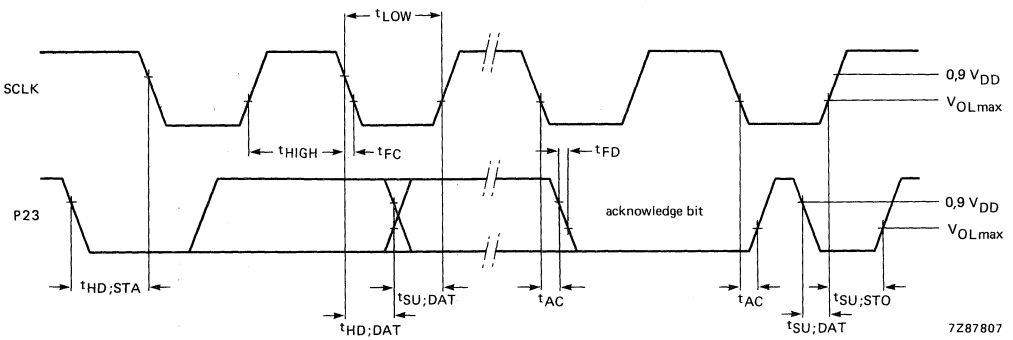


Fig. 19 Timing requirements for the P2.3 and SCLK *output* signals.

Table 5 Output timing shown in Fig. 19

symbol	timing	
	normal mode (ASC in S2 = 0)	low-speed mode (ASC in S2 = 1)
$t_{HD;STA}$	$\frac{1}{2}$ (DF + 9) t_{XTAL}	$\frac{3}{4}$ (DF + 9) t_{XTAL}
t_{HIGH}	$\frac{1}{2}$ (DF) t_{XTAL}	$\frac{3}{4}$ (DF) t_{XTAL}
t_{LOW}	$\frac{1}{2}$ (DF) t_{XTAL}	$\frac{1}{4}$ (DF) t_{XTAL}
$t_{SU;STO}$	$\frac{1}{2}$ (DF − 3) t_{XTAL}	$\frac{1}{4}$ (DF − 3) t_{XTAL}
$t_{HD;DAT}$ (slave transmitter) any DF	$\geqslant 9t_{XTAL}$ $\leqslant 12t_{XTAL}$	$\geqslant 9t_{XTAL}$ $\leqslant 12t_{XTAL}$
$t_{HD;DAT}$ (master transmitter) for DF $\leqslant 51$	$\geqslant 9t_{XTAL}$ $\leqslant 12t_{XTAL}$	– –
for DF $\leqslant 99$	– –	$\geqslant 9t_{XTAL}$ $\leqslant 12t_{XTAL}$
$t_{SU;DAT}$ (master transmitter) for DF > 51	$\geqslant 15t_{XTAL}$ $\leqslant 24t_{XTAL}$	– –
for DF > 99	– –	$\geqslant 15t_{XTAL}$ $\leqslant 24t_{XTAL}$
t_{AC}	$\geqslant 9t_{XTAL}$ $\leqslant 12t_{XTAL}$	$\geqslant 9t_{XTAL}$ $\leqslant 12t_{XTAL}$
t_{FD}, t_{FC}	$\leqslant$ 100 ns at C_b = 400 pF	$\leqslant$ 100 ns at C_b = 400 pF

Notes to Table 5

t_{XTAL} = one period of the XTAL input frequency (f_{XTAL})
= 167 ns for f_{XTAL} = 6 MHz
DF = divisor (see Table 2 Serial I/O section).
C_b = the maximum bus capacitance for each line.

AC CHARACTERISTICS (continued)

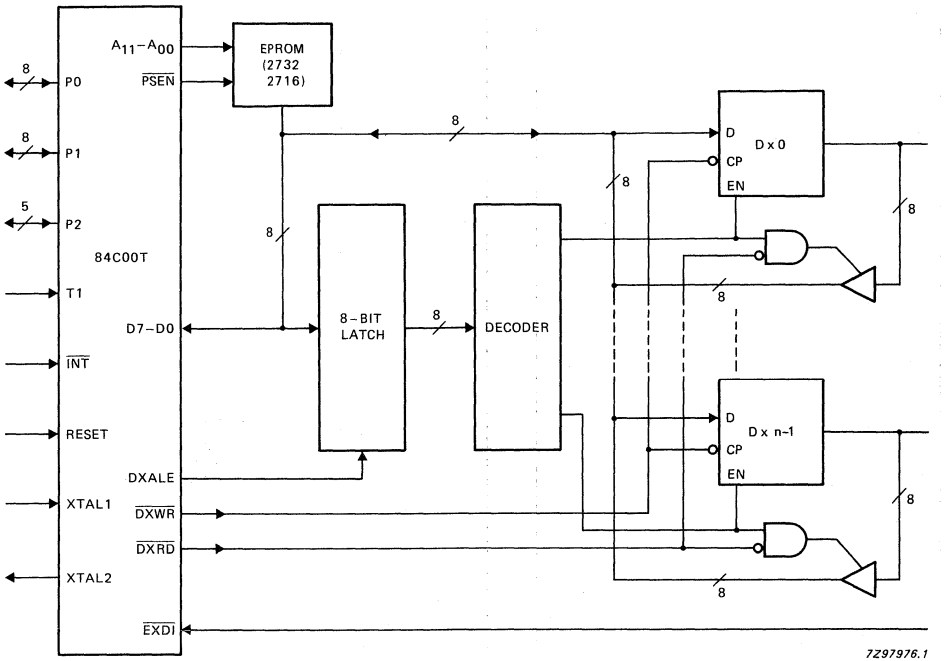


Fig. 20 Block diagram of the external Dx register interface.
The Dx interface can only be used with the PCF84C00T.

Purchase of Philips' I²C components conveys a license under the Philips' I²C patent to use the components in the I²C-system provided the system conforms to the I²C specifications defined by Philips.

SINGLE-CHIP 8-BIT MICROCONTROLLERS

DESCRIPTION

An advanced CMOS process is used to manufacture the PCF84C12, PCF84C22 and PCF84C42 microcontrollers. Each device has 13 quasi-bidirectional I/O port lines, a single-level vectored interrupt structure, an 8-bit timer and on-chip clock oscillator and clock circuits. On-chip RAM and ROM content is as follows:

- PCF84C12 – 64 x 8 RAM, 1 K x 8 ROM
- PCF84C22 – 64 x 8 RAM, 2 K x 8 ROM
- PCF84C42 – 64 x 8 RAM, 4 K x 8 ROM

These efficient microcontrollers also perform well as arithmetic processors. The instruction set is similar to that of the MAB8048. They have bit handling abilities and facilities for both binary and BCD arithmetic.

These microcontrollers are members of the 84CXXX family. For detailed information, consult the 84CXXX family specification.

Features

- 8-bit CPU, ROM, RAM, I/O in a single 20-lead DIL or SO package
- 1 K, 2 K or 4 K x 8 ROM
- 64 x 8 RAM
- 2 timers (8-bit programmable)
- 13 quasi-bidirectional I/O port lines
- Two test inputs: one of which is also the external interrupt input
- Single-level, vectored interrupts: external and timer/event counter
- 8-bit programmable timer/event counter
- Clock frequency range: 100 kHz to 10 MHz
- Over 80 instructions (similar to those of the MAB8048) all of 1 or 2 cycles
- Single supply voltage (2.5 V to 5.5 V)
- STOP and IDLE modes
- Power-on-reset circuit
- Operating temperature range: −40 to + 85 °C

PACKAGE OUTLINES

PCF84C12/22/42P: 20-lead DIL; plastic (SOT146).
PCF84C12/22/42T: 20-lead mini-pack; plastic (SO20, SOT163A).

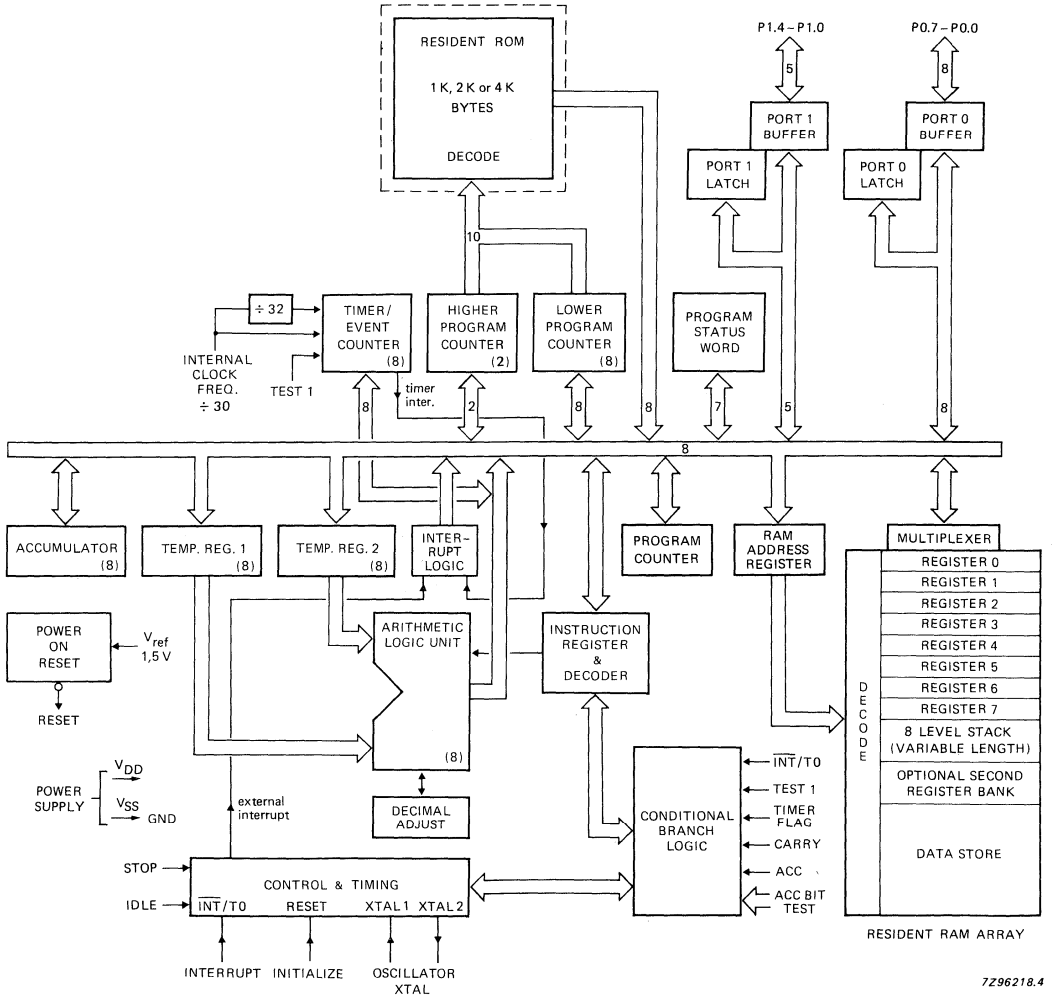


Fig. 1 Block diagram.

PINNING

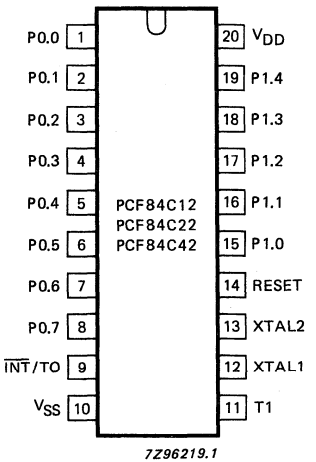


Fig. 2 Pinning diagram.

DEVELOPMENT DATA

PIN DESIGNATION

1-8	P0.0-P0.7	**Port 0:** 8-bit quasi-bidirectional I/O port.
9	$\overline{INT}$/T0	**Interrupt/Test 0:** external interrupt input (negative edge triggered)/test input pin; when used as a test input, this pin is directly tested by conditional branch instructions JT0 and JNT0.
10	V_{SS}	**Ground:** circuit earth potential.
11	T1	**Test 1:** test input pin, directly tested by conditional branch instructions JT1 and JNT1. T1 may also be selected as an input to the 8-bit timer/event counter via the STRT CNT instruction.
12	XTAL 1	**Oscillator input:** input from a crystal which determines the internal oscillator frequency, or an external clock generator.
13	XTAL 2	**Oscillator output:** output of the inverting amplifier.
14	RESET	**Reset input:** used to initialize the microcontroller (active HIGH); also output of power-on-reset circuit.
15-19	P1.0-P1.4	**Port 1:** 8-bit quasi-bidirectional I/O port.
20	V_{DD}	**Power supply:** 2.5 V to 5.5 V.

FUNCTIONAL DESCRIPTION

Program memory

The program memory consists of 1 K, 2 K or 4 K bytes of read-only memory (ROM). Each location is directly addressable by the program counter. The ROM is mask-programmed at the factory.

Data memory

Data memory consists of 64 bytes, random-access data memory (RAM). All locations are indirectly addressable using RAM pointer registers; up to 16 designated locations are directly addressable. Memory also includes an 8-level program counter stack addressed by a 3-bit stack pointer.

I/O facilities

Each device has 13 I/O lines arranged as follows:

- Port 0 — 8-bit parallel port (P0.0 to P0.7)
- Port 1 — 5-bit parallel port (P1.0 to P1.4)
- $\overline{\text{INT}}$/T0 — external interrupt and test input. When used as a test input can be directly tested by conditional branch instructions JT0 and JNT0
- T1 — test input which can alter program sequences when tested by conditional jump instructions JT1 and JNT1. T1 also functions as an input to the 8-bit timer/event counter.

Reset (pin 14)

A positive-going signal on the RESET input:

- Sets the program counter to zero
- Selects location 0 of memory bank 0 and register bank 0
- Sets the stack pointer to zero (000); pointing to RAM address 8
- Disables the interrupts (external and timer)
- Stops the timer/event counter, then sets it to zero
- Sets the timer prescaler to divide by 32
- Resets the timer flag
- Sets all ports to input mode
- Cancels IDLE and STOP mode

INSTRUCTION SET

The instruction set consists of over 80 one and two byte instructions. Program code efficiency is high because all RAM locations and all ROM locations on a 256 byte page require only a single byte address.

Table 3 gives the instruction set. Table 2 shows the instruction map and Table 1 details the symbols that are used.

Table 1 Symbols used in Table 3

symbol	description
A	accumulator
addr	program memory address
Bb	bit designation (b = 0-7)
RBS	register bank select
C	carry bit (bit CY)
CNT	event counter
D	mnemonic for 4-bit digit (nibble)
data	8-bit number or expression
I	interrupt
P	mnemonic for 'in-page' operation
PC	program counter
Pp	port designation (p = 0 or 1)
PSW	program status word
RB	register bank
Rr	register designation (r = 0-7)
SP	stack pointer
T	timer
TF	timer flag
T0, T1	test 0 and 1 inputs
#	immediate data prefix
@	indirect address prefix
(X)	contents of X
((X))	contents of location addressed by X
←	is replaced by
↔	is exchanged with

INSTRUCTION SET (continued)

Table 2 Instruction map

first hexadecimal character of opcode

second hexadecimal character of opcode

	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
0	NOP	IDLE		ADD A,#data	JMP page 0	EN I	JNTF addr	DEC A	IN A,Pp							
									0	1						
1	INC @Rr		JBO addr	ADDC A,#data	CALL page 0	DIS I	JTF addr	INC A	INC Rr							
	0	1							0	1	2	3	4	5	6	7
2	XCH A,@Rr		STOP	MOV A,#data	JMP page 1	EN TCNTI	JNTO addr	CLR A	XCH A,Rr							
	0	1							0	1	2	3	4	5	6	7
3	XCHD A,@Rr		JB1 addr		CALL page 1	DIS TCNTI	JTO addr	CPL A	OUTL Pp,A							
	0	1							0	1						
4	ORL A,@Rr		MOV A,T	ORL A,#data	JMP page 2	STRT CNT	JNT1 addr	SWAP A	ORL A,Rr							
	0	1							0	1	2	3	4	5	6	7
5	ANL A,@Rr		JB2 addr	ANL A,#data	CALL page 2	STRT T	JT1 addr	DA A	ANL A,Rr							
	0	1							0	1	2	3	4	5	6	7
6	ADD A,@Rr		MOV T,A		JMP page 3	STOP TCNT		RRC A	ADD A,Rr							
	0	1							0	1	2	3	4	5	6	7
7	ADDC A,@Rr		JB3 addr		CALL page 3			RR A	ADDC A,Rr							
	0	1							0	1	2	3	4	5	6	7
8				RET	JMP page 4				ORL Pp,#data							
									0	1						
9			JB4 addr	RETR	CALL page 4		JNZ addr	CLR C	ANL Pp,#data							
									0	1						
A	MOV @Rr,A			MOVP A,@A	JMP page 5			CPL C	MOV Rr,A							
	0	1							0	1	2	3	4	5	6	7
B	MOV @Rr,#data		JB5 addr	JMPP @A	CALL page 5				MOV Rr,#data							
	0	1							0	1	2	3	4	5	6	7
C	DEC @Rr				JMP page 6	SEL RBO	JZ addr	MOV A,PSW	DEC Rr							
	0	1							0	1	2	3	4	5	6	7
D	XRL A,@Rr		JB6 addr	XRL A,#data	CALL page 6	SEL RB1		MOV PSW,A	XRL A,Rr							
	0	1							0	1	2	3	4	5	6	7
E	DJNZ @Rr,addr				JMP page 7		JNC addr	RL A	DJNZ Rr, addr							
	0	1							0	1	2	3	4	5	6	7
F	MOV A,@Rr		JB7 addr		CALL page 7		JC addr	RLC A	MOV A,Rr							
	0	1							0	1	2	3	4	5	6	7

Table 3 Instruction set

	mnemonic	opcode (hex.)	bytes/ cycles	description	function		notes
ACCUMULATOR	ADD A, Rr	6*	1/1	Add register contents to A	(A)←(A) + (Rr)	r = 0–7	1
	ADD A, @Rr	60 61	1/1	Add RAM data, addressed by Rr, to A	(A)←(A) + ((R0)) (A)←(A) + ((R1))		1
	ADD A, #data	03 data	2/2	Add immediate data to A	(A)←(A) + data		1
	ADDC A, Rr	7*	1/1	Add carry and register contents to A	(A)←(A) + (Rr) + (C)	r = 0–7	1
	ADDC A, @Rr	70 71	1/1	Add carry and RAM data, addressed by Rr, to A	(A)←(A) + ((R0)) + (C) (A)←(A) + ((R1)) + (C)		1
	ADDC A, #data	13 data	2/2	Add carry and immediate data to A	(A)←(A) + data + (C)		1
	ANL A, Rr	5*	1/1	'AND' Rr with A	(A)←(A) AND (Rr)	r = 0–7	
	ANL A, @Rr	50 51	1/1	'AND' RAM data, addressed by Rr, with A	(A)←(A) AND ((R0)) (A)←(A) AND ((R1))		
	ANL A, #data	53 data	2/2	'AND' immediate data with A	(A)←(A) AND data		
	ORL A, Rr	4*	1/1	'OR' Rr with A	(A)←(A) OR (Rr)	r = 0–7	
	ORL A, @Rr	40 41	1/1	'OR' RAM data, addressed by Rr, with A	(A)←(A) OR ((R0)) (A)←(A) OR ((R1))		
	ORL A, #data	43 data	2/2	'OR' immediate data with A	(A)←(A) OR data		
	XRL A, Rr	D*	1/1	'XOR' Rr with A	(A)←(A) XOR (Rr)	r = 0–7	
	XRL A, @Rr	D0 D1	1/1	'XOR' RAM, addressed by Rr, with A	(A)←(A) XOR ((R0)) (A)←(A) XOR ((R1))		
	XRL A, #data	D3 data	2/2	'XOR' immediate data with A	(A)←(A) XOR data		
	INC A	17	1/1	increment A by 1	(A)←(A) + 1		
	DEC A	07	1/1	decrement A by 1	(A)←(A) −1		
	CLR A	27	1/1	clear A to zero	(A)←0		
	CPL A	37	1/1	one's complement A	(A)←NOT(A)		
	RL A	E7	1/1	rotate A left	$(A_{n+1})\leftarrow(A_n)$ $(A_0)\leftarrow(A_7)$	n = 0–6	

INSTRUCTION SET (continued)

ACCUMULATOR (cont.)	RLC A	F7	1/1	rotate A left through carry	$(A_{n+1}) \leftarrow A_n$ $(A_0) \leftarrow (C), (C) \leftarrow (A_7)$	n = 0–6	2
	RR A	77	1/1	rotate A right	$(A_n) \leftarrow (A_{n+1})$ $(A_7) \leftarrow (A_0)$	n = 0–6	
	RRC A	67	1/1	rotate A right through carry	$(A_n) \leftarrow (A_{n+1})$ $(A_7) \leftarrow (C), (C) \leftarrow (A_0)$	n = 0–6	2
	DA A	57	1/1	decimal adjust A			2
	SWAP A	47	1/1	swap nibbles of A	$(A_{4-7}) \leftrightarrow (A_{0-3})$		
DATA MOVES	MOV A, Rr	F*	1/1	move register contents to A	(A)←(Rr)	r = 0–7	
	MOV A, @Rr	F0 F1	1/1	move RAM data, addressed by Rr, to A	(A)←((R0)) (A)←((R1))		
	MOV A, #data	23 data	2/2	move immediate data to A	(A)←data		
	MOV Rr, A	A*	1/1	move accumulator contents to register	(Rr)←(A)	r = 0–7	
	MOV @Rr, A	A0 A1	1/1	move accumulator contents to RAM location addressed by Rr	((R0))←(A) ((R1))←(A)		
	MOV Rr, #data	B* data	2/2	move immediate data to Rr	(Rr)←data		
	MOV @Rr, #data	B0 data B1 data	2/2	move immediate data to RAM location addressed by Rr	((R0))←data ((R1))←data		
	XCH A, Rr	2*	1/1	exchange accumulator contents with Rr	(A)↔(Rr)	r = 0–7	
	XCH A, @Rr	20 21	1/1	exchange accumulator contents with RAM data addressed by Rr	(A)↔((R0)) (A)↔((R1))		
	XCHD A, @Rr	30 31	1/1	exchange lower nibbles of A and RAM data addressed by Rr	$(A_{0-3}) \leftrightarrow ((R0_{0-3}))$ $(A_{0-3}) \leftrightarrow ((R1_{0-3}))$		
	MOV A, PSW	C7	1/1	move PSW contents to accumulator	(A)←(PSW)		
	MOV PSW, A	D7	1/1	move accumulator bit 3 to PSW_3	$(PSW_3) \leftarrow (A_3)$		3
	MOVP A, @A	A3	1/2	move indirectly addressed data in current page to A	$(PC_{0-7}) \leftarrow (A), (A) \leftarrow ((PC))$		
FLAGS	CLR C	97	1/1	clear carry bit	(C)←0		2
	CPL C	A7	1/1	complement carry bit	(C)←NOT(C)		2

	mnemonic	opcode (hex.)	bytes/ cycles	description	function	notes
REGISTER	INC Rr	1*	1/1	increment register by 1	(Rr)←(Rr) + 1 r = 0–7	
	INC @Rr	10 11	1/1	increment RAM data, addressed by Rr, by 1	((R0))←((R0)) + 1 ((R1))←((R1)) + 1	
	DEC Rr	C*	1/1	decrement register by 1	(Rr)←(Rr) – 1 r = 0–7	
	DEC @Rr	C0 C1	1/1	decrement RAM data, addressed by Rr, by 1	((R0))←((R0)) – 1 ((R1))←((R1)) – 1	
BRANCH	JMP addr	● 4 address	2/2	unconditional jump within a 2 K bank	(PC_{8-10})←$addr_{8-10}$ (PC_{0-7})←$addr_{0-7}$	
	JMPP @A	B3	1/2	indirect jump within a page	(PC_{0-7})←((A))	
	DJNZ Rr, addr	E* address	2/2	decrement Rr by 1 and jump if not zero to addr	(Rr)←(Rr) – 1 r = 0–7 if (Rr) not zero (PC_{0-7})←addr	
	DJNZ @Rr, addr	E0	2/2	decrement RAM data, addressed by Rr by 1 and jump if not zero to addr	((R0))←((R0)) – 1 if ((R0)) not zero (PC_{0-7})←addr	
		E1			((R1))←((R1)) – 1 if ((R1)) not zero (PC_{0-7})←addr	
	JBb addr	▲ 2 address	2/2	jump to addr if Acc. bit b = 1	if b = 1 : (PC_{0-7})←addr b = 0–7	
	JC addr	F6 address	2/2	jump to addr if C = 1	if C = 1 : (PC_{0-7})←addr	
	JNC addr	E6 address	2/2	jump to addr if C = 0	if C = 0 : (PC_{0-7})←addr	
	JZ addr	C6 address	2/2	jump to addr if A = 0	if A = 0 : (PC_{0-7})←addr	
	JNZ addr	96 address	2/2	jump to addr if A is NOT zero	if A≠0 : (PC_{0-7})←addr	
	JT0 addr	36 address	2/2	jump to addr if T0 = 1	if T0 = 1: (PC_{0-7})←addr	
	JNT0 addr	26 address	2/2	jump to addr if T0 = 0	if T0 = 0: (PC_{0-7})←addr	
	JT1 addr	56 address	2/2	jump to addr if T1 = 1	if T1 = 1: (PC_{0-7})←addr	
	JNT1 addr	46 address	2/2	jump to addr if T1 = 0	if T1 = 0: (PC_{0-7})←addr	
	JTF addr	16 address	2/2	jump to addr if Timer Flag = 1	if TF = 1: (PC_{0-7})←addr	4
	JNTF addr	06 address	2/2	jump to addr if Timer Flag = 0	if TF = 0: (PC_{0-7})←addr	

INSTRUCTION SET (continued)

TIMER/EVENT COUNTER	MOV A, T	42	1/1	move timer/event counter contents to accumulator	(A)←(T)	
	MOV T, A	62	1/1	move accumulator contents to timer/event counter	(T)←(A)	
	STRT CNT	45	1/1	start event counter		
	STRT T	55	1/1	start timer		
	STOP TCNT	65	1/1	stop timer/event counter		
	EN TCNTI	25	1/1	enable timer/event counter interrupt		
	DIS TCNTI	35	1/1	disable timer/event counter interrupt		
CONTROL	EN I	05	1/1	enable external interrupt		
	DIS I	15	1/1	disable external interrupt		
	SEL RB0	C5	1/1	select register bank 0	(RBS)←0	5
	SEL RB1	D5	1/1	select register bank 1	(RBS)←1	5
	STOP	22	1/1	enter STOP mode		
	IDLE	01	1/1	enter IDLE mode		
SUBROUTINE	CALL addr	▲ 4 address	2/2	jump to subroutine	((SP))←(PC), ($PSW_{4,6,7}$) 6 (SP)←(SP) + 1 (PC_{8-10})←$addr_{8-10}$ (PC_{0-7})←$addr_{0-7}$	6
	RET	83	1/2	return from subroutine	(SP)←(SP) − 1 (PC)←((SP))	6
	RETR	93	1/2	return from interrupt and restore bits 4, 6, 7 of PSW	(SP)←(SP) − 1 ($PSW_{4,6,7}$) + (PC)←((SP))	6

	mnemonic	opcode (hex.)	bytes/ cycles	description	function	notes
PARALLEL INPUT/OUTPUT	IN A, Pp	08 09	1/2	input port p data to accumulator	(A)←(P0) (A)←(P1)	
	OUTL Pp, A	38 39	1/2	output accumulator data to port p	(P0)←(A) (P1)←(A)	
	ANL Pp, #data	98 99	2/2	AND port p data with immediate data	(P0)←(P0) AND data (P1)←(P1) AND data	
	ORL Pp, #data	88 89	2/2	OR port p data with immediate data	(P0)←(P0) OR data (P1)←(P1) OR data	
	NOP	00	1/1	no operation		

Notes to Table 3

1. PSW CY, AC affected
2. PSW CY affected
3. PSW PS affected
4. Execution of JTF and JNTF instructions resets the Timer Flag (TF).
5. PSW RBS affected
6. PSW SP_0, SP_1, SP_2 affected

* : 8, 9, A, B, C, D, E, F
● : 0, 2, 4, 6, 8, A, C, E
▲ : 1, 3, 5, 7, 9, B, D, F

RATINGS

Limiting values in accordance with the Absolute Maximum System (IEC 134)

Supply voltage (pin 20)	V_{DD}		−0.8 to + 8	V
All input voltages	V_I		−0.8 to V_{DD} + 0.8	V
DC current into any input or output	$\pm I_I, \pm I_O$	max.	10	mA
Total power dissipation (see note 1)	P_{tot}	max.	500	mW
Power dissipation per output	P_O	max.	50	mW
Storage temperature range	T_{stg}		−65 to + 150	°C
Operating ambient temperature range	T_{amb}		−40 to +85	°C
Operating junction temperature	T_j	max.	125	°C

HANDLING

Inputs and outputs are protected against electrostatic discharge in normal handling. However, it is good practice to take normal precautions appropriate to handling MOS devices (see 'Handling MOS devices').

Notes to the ratings

1. Thermal resistance (junction to ambient)

for SOT146	$R_{th\ j\text{-}a}$	max.	120	K/W
for SOT163A	$R_{th\ j\text{-}a}$	max.	150	K/W

DC CHARACTERISTICS

V_{DD} = 2.5 to 5.5 V; V_{SS} = 0 V; T_{amb} = −40 to + 85 °C; all voltages with respect to V_{SS}; unless otherwise specified

parameter	symbol	min.	typ.	max.	unit
Supply voltage					
operating (see Fig. 3)	V_{DD}	2.5	–	5.5	V
Supply current					
operating (see Fig. 4 and note 2)					
at V_{DD} = 5 V; f_{XTAL} = 10 MHz	I_{DD}	–	1.6	3.2	mA
at V_{DD} = 5 V; f_{XTAL} = 6 MHz	I_{DD}	–	1	2	mA
at V_{DD} = 3 V; f_{XTAL} = 3.58 MHz	I_{DD}	–	0.3	0.6	mA
IDLE mode (see Fig. 5 and note 2)					
at V_{DD} = 5 V; f_{XTAL} = 10 MHz	I_{DD}	–	0.8	1.6	mA
at V_{DD} = 5 V; f_{XTAL} = 6 MHz	I_{DD}	–	0.5	1	mA
at V_{DD} = 3 V; f_{XTAL} = 3.58 MHz	I_{DD}	–	0.15	0.4	mA
STOP mode (see Fig. 8, note 1 and note 2)					
at V_{DD} = 2.5 V; T_{amb} = 25 °C	I_{DD}	–	1.2	2.5	μA
at V_{DD} = 2.5 V; T_{amb} = 85 °C	I_{DD}	–	–	10	μA
Inputs					
Input voltage LOW	V_{IL}	0	–	$0.3V_{DD}$	V
Input voltage HIGH	V_{IH}	$0.7V_{DD}$	–	V_{DD}	V
Input leakage current					
as $V_{SS} < V_I < V_{DD}$	$\pm I_{IL}$	–	–	1	μA
Outputs					
Output sink current LOW					
at V_{DD} = 5 V ± 10%; V_O = 0.4 V	I_{OL}	1.6	3	–	mA
Pull-up output source current HIGH					
at V_{DD} = 5 V ± 10%; V_O = 0.7 V_{DD}	$-I_{OH}$	40	–	–	μA
at V_{DD} = 5 V ± 10%; V_O = V_{SS}	$-I_{OH}$	–	–	400	μA
Push-pull output source current HIGH					
at V_{DD} = 5 V ± 10%; V_O = V_{DD}−0.4 V	$-I_{OH}$	1.6	3	–	mA

Notes to the DC characteristics

1. Crystal connected between XTAL 1 and XTAL 2; T1 = V_{SS}; $\overline{INT}$ = V_{DD}.
2. V_{IL} = V_{SS}; V_{IH} = V_{DD}; all outputs unloaded; all open drain outputs connected to V_{SS}.

DEVELOPMENT DATA

DC CHARACTERISTICS (continued)

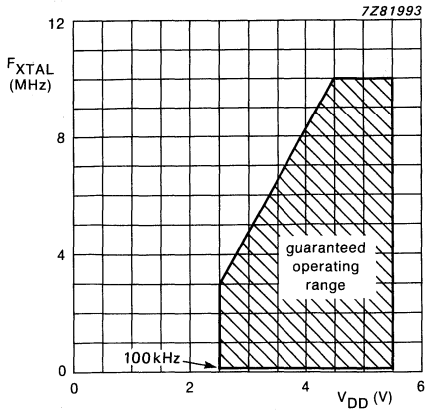
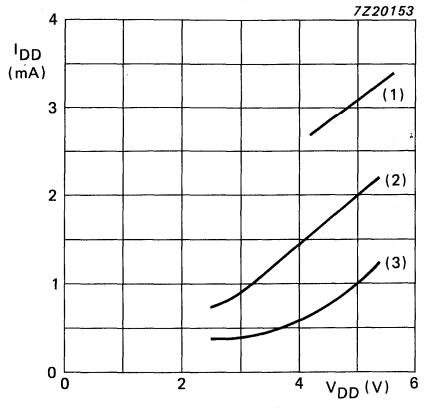
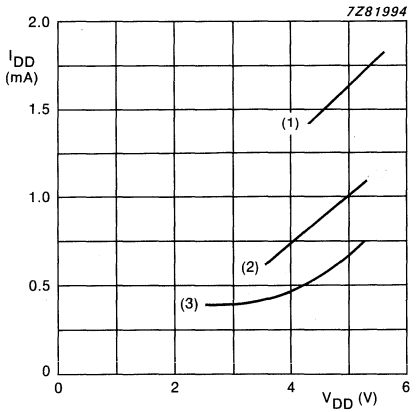


Fig. 3 Maximum clock frequency (f_{XTAL}) as a function of supply voltage (V_{DD}).



(1) f_{OSC} = 10 MHz
(2) f_{OSC} = 6 MHz
(3) f_{OSC} = 3.58 MHz

Fig. 4 Maximum supply current (I_{DD}) in operating mode as a function of supply voltage (V_{DD}).



(1) f_{OSC} = 10 MHz
(2) f_{OSC} = 6 MHz
(3) f_{OSC} = 3.58 MHz

Fig. 5 Maximum supply current (I_{DD}) in IDLE mode as a function of supply voltage (V_{DD}).

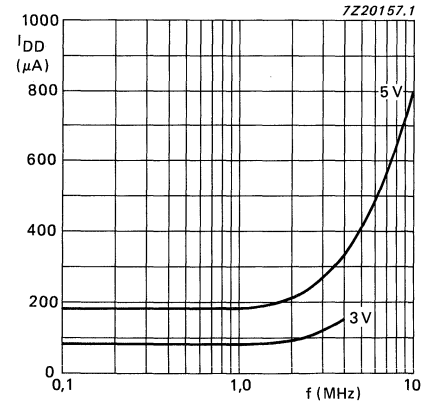


Fig. 6 Typical supply current during IDLE mode as a function of frequency at V_{DD} = 3 V and V_{DD} = 5 V.

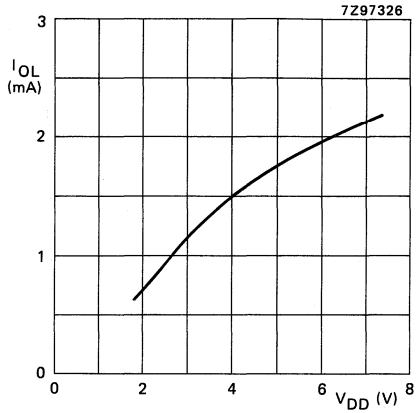
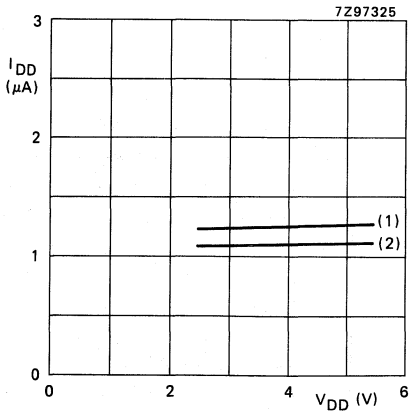
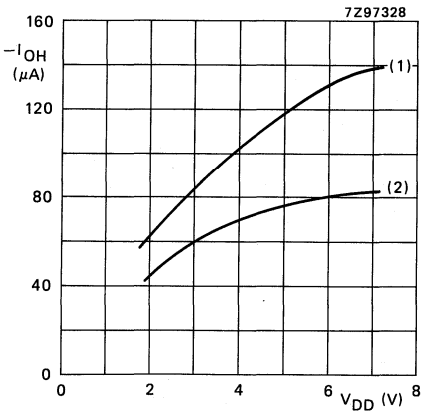


Fig. 7 Typical output sink current (I_{OL}), as a function of supply voltage (V_{DD}); V_O = 0.4 V.



(1) T_{amb} = 85 °C
(2) T_{amb} = 25 °C

Fig. 8 Typical supply current (I_{DD}) in STOP mode as a function of the supply voltage (V_{DD}).



(1) $V_O = V_{SS}$
(2) $V_O = 0.7\ V_{DD}$

Fig. 9 Typical output source current ($-I_{OH}$) as a function of supply voltage (V_{DD}).

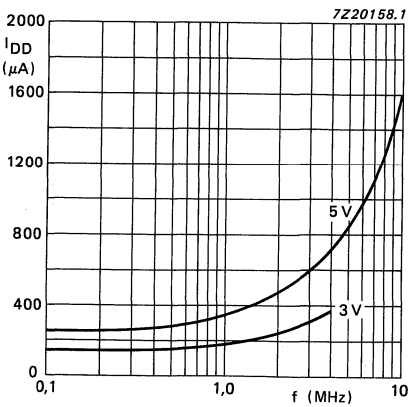


Fig. 10 Typical supply current during operating mode as a function of frequency at V_{DD} = 3 V and V_{DD} = 5 V.

DEVELOPMENT DATA

SINGLE-CHIP 8-BIT MICROCONTROLLER WITH 32 I/O LINES

DESCRIPTION

The PCF84C85 microcontroller is manufactured in CMOS, and is designed to be an efficient controller as well as an arithmetic processor. The instruction set is based on that of the MAB8048 and is software compatible with the 84CXXX family. The PCF84C85 has two additional derivative ports and the microcontroller has bit handling abilities and facilities for both binary and BCD arithmetic.

For detailed information see the 84CXXX family specification.

Features

- 8-bit CPU, ROM, RAM, I/O in a single 40-lead DIL or mini-pack package
- 8 K ROM
- 256 RAM bytes
- 32 quasi-bidirectional I/O port lines
- Two test inputs: one of which is also the external interrupt input
- Single-level vectored interrupts: external, timer/event counter, serial I/O
- I^2C hardware interface for two-line serial data transfer
  (serial I/O data via an existing port line and clock via a dedicated line)
- 8-bit programmable timer/event counter
- Clock frequency 100 kHz to 10 MHz
- Over 80 instructions (based on MAB8048) all of 1 or 2 cycles
- Single supply voltage from 2,5 V to 5,5 V
- STOP and IDLE mode
- Power-on-reset circuit
- Operating temperature range: −40 to +85 °C

PACKAGE OUTLINES

PCF84C85P: 40-lead DIL; plastic (SOT129).
PCF84C85T: 40-lead; mini-pack (VSO40; SOT158A).

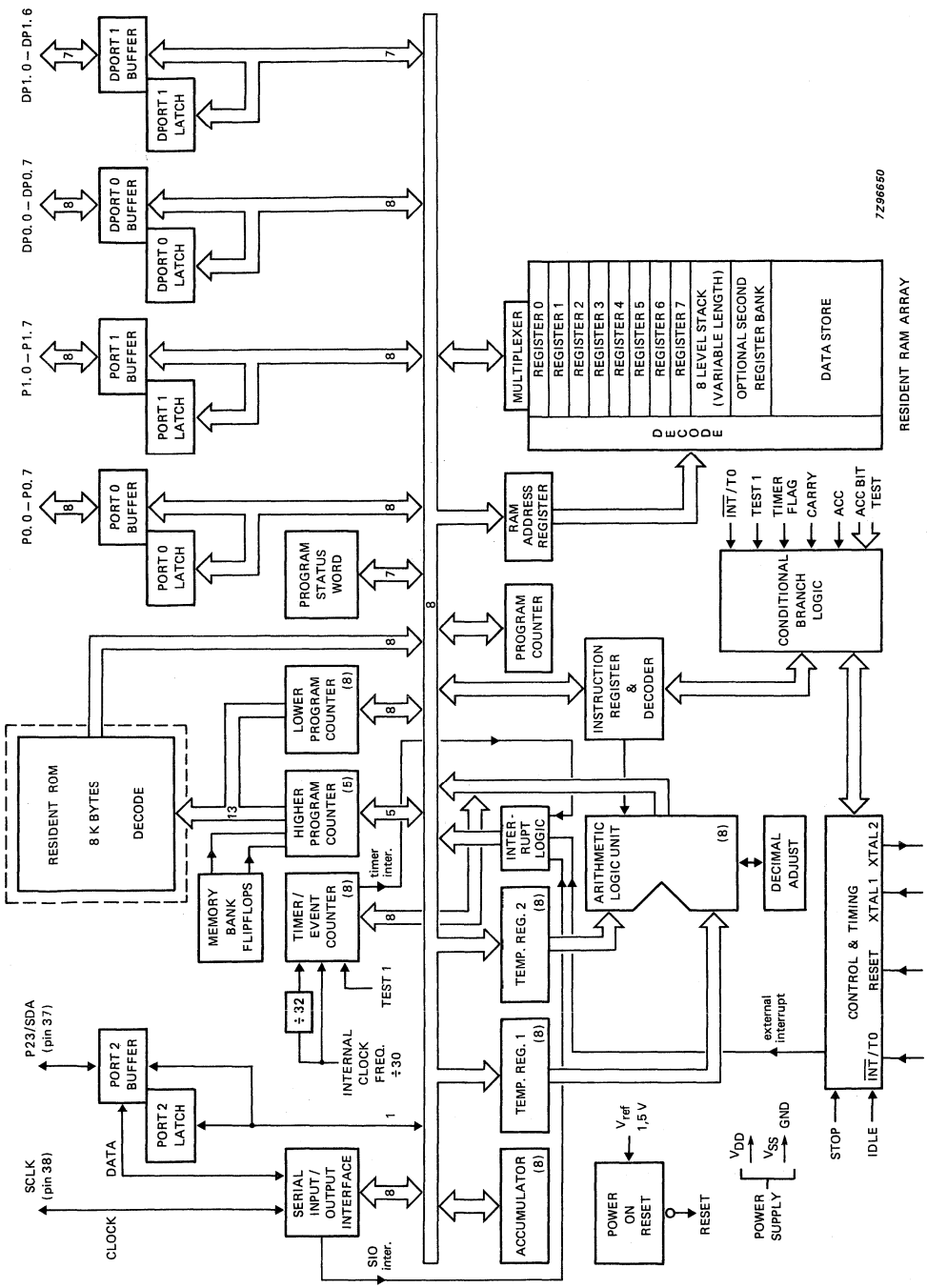


Fig. 1 Block diagram.

DEVELOPMENT DATA

PINNING

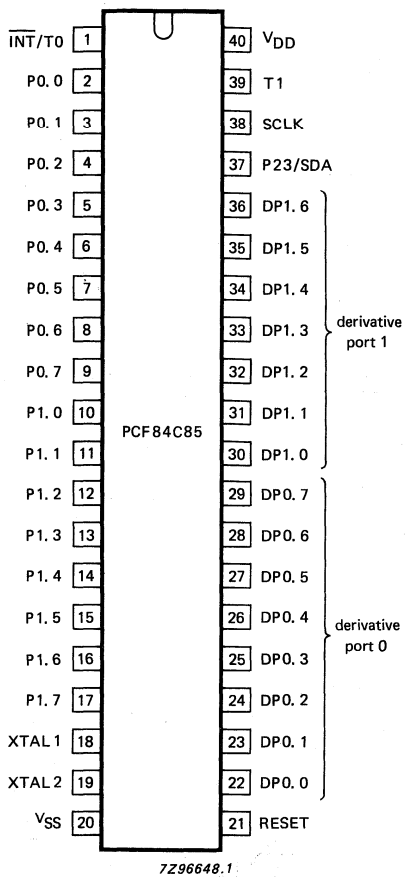


Fig. 2 Pinning diagram.

PIN DESIGNATION

38	SCLK	**Clock:** bidirectional clock for serial I/O.
1	$\overline{\text{INT}}$/T0	**Interrupt/Test 0:** external interrupt input (sensitive to negative-going edge)/test input pin; when used as a test input directly tested by conditional branch instructions JT0 and JNT0.
39	T1	**Test 1:** test input pin, directly tested by conditional branch instructions JT1 and JNT1. T1 also functions as an input to the 8-bit timer/event counter, using the STRT CNT instruction.
18	XTAL 1	**Crystal input:** connection to timing component (crystal) which determines the frequency of the internal oscillator; also the input for an external clock source.
19	XTAL 2	Connection to the other side of the timing component.
21	RESET	**Reset input:** used to initialize the processor (active HIGH), or output of power-on-reset circuit.

PIN DESIGNATION (continued)

2-9	P0.0-P0.7	**Port 0:** 8-bit quasi-bidirectional I/O port.
10-17	P1.0-P1.7	**Port 1:** 8-bit quasi-bidirectional I/O port.
37	P23/SDA	**Port 2:** 1-bit quasi-bidirectional I/O port or serial data input/output in serial I/O mode.
22-29	DP0.0-DP0.7	**Derivative port 0:** 8-bit quasi-bidirectional I/O port.
30-36	DP1.0-DP1.6	**Derivative port 1:** 7-bit quasi-bidirectional I/O port.
20	V_{SS}	**Ground:** circuit earth potentional.
40	V_{DD}	**Power supply:** 2,5 V to 5,5 V.

FUNCTIONAL DESCRIPTION

Program memory

The program memory consists of 8 K bytes, in a read-only memory (ROM). Each location is directly addressable by the program counter. The memory is mask-programmed at the factory. Figure 3 shows the program memory map.

Four program memory locations are of special importance:

- Location 0; contains the first instruction to be executed after the processor is initialized (RESET),
- Location 3; contains the first byte of an external interrupt service subroutine;
- Location 5; contains the first byte of a serial I/O interrupt service subroutine.
- Location 7; contains the first byte of a timer/event counter interrupt service subroutine.

Program memory is arranged in banks of 2 K bytes, which are selected by SEL MB instructions. The program memory is further divided into location 'pages', each of 256 bytes. This latter division applies only for conditional branches. Memory bank boundaries can be crossed only by using unconditional branch instructions after the appropriate memory bank has been selected. A CALL instruction can transfer control to a subroutine on any 'page'; RET and RETR instructions can transfer control from a subroutine back to the main program.

Data memory

Data memory consists of 256 bytes, random-access data memory (RAM). All locations are indirectly addressable using RAM pointer registers; up to 16 designated locations are directly addressable. Memory also includes an 8-level program counter stack addressed by a 3-bit stack pointer. Figure 4 shows the data memory map.

Working registers

Locations 0 to 7 are designated as working registers, directly addressable by the direct register instructions. Ease of addressing, and a minimum requirement of instruction bytes to manipulate their contents, makes these locations suitable for storing frequently addressed intermediate results. This bank of registers can be selected by the SEL RB0 instruction.

Executing the select register bank instruction SEL RB1, designates locations 24 to 31 as working registers, instead of locations 0 to 7, and these are then directly addressable. This second bank of working registers may be used as an extension of the first or reserved for use during interrupt service subroutines saving the first bank for use in the main program. If the second bank is not used, locations 24 to 31 may serve as general purpose RAM.

The first locations of each bank contain the RAM pointer registers R0, R1, R0' and R1', which indirectly address all RAM locations.

All RAM locations make efficient program loop counters when used with the decrement register and test instruction DJNZ.

DEVELOPMENT DATA

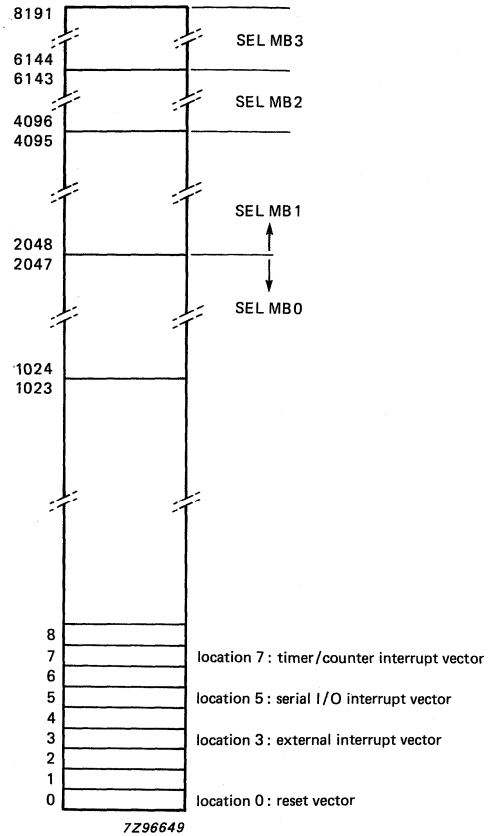


Fig. 3 Program memory map.

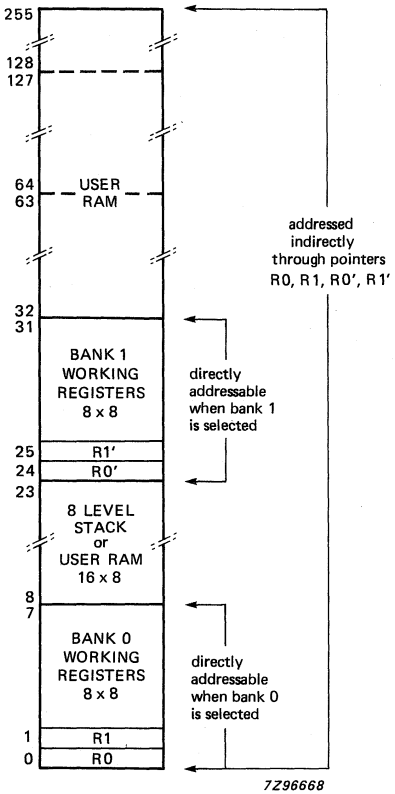


Fig. 4 Data memory map.

Program counter stack

Locations 8 to 23 may be designated as an 8-level program counter stack (2 locations per level), or as general purpose RAM. The program counter stack (Fig.5) enables the processor to keep track of the return addresses and status generated by interrupts or CALL instructions by storing the contents of the program counter prior to servicing the subroutine. A 3-bit stack pointer determines which of the eight register pairs of the program counter stack will be loaded with the next generated return address.

The stack pointer, when initialized to 000 by RESET, points to RAM locations 8 and 9. On the first subroutine CALL or interrupt, the contents of the program counter and bits 4, 6 and 7 of the program status word (PSW) are transferred to locations 8 and 9. The stack pointer increments by one and points to locations 10 and 11 ready for another CALL. Because an address may be up to 13 bits long, two bytes must be used to store each address.

At the end of a subroutine, which is signalled by a return instruction (RET or RETR), the stack pointer decrements by one and the contents of the register pair on top of the stack are transferred to the program counter. The saved PSW bits are transferred to the PSW only by the RETR instruction.

If not all 8 levels of subroutine and interrupt nesting are used, the unused portion of the stack may be used as any other indirectly addressable RAM locations.

Nesting of subroutines within subroutines can continue up to 8 times without overflowing the stack. If overflow does occur the deepest address stored (locations 8 and 9) will be overwritten and lost since the stack pointer overflows from 111 to 000. It also underflows from 000 to 111.

FUNCTIONAL DESCRIPTION (continued)

The value of the saved contents of the program counter is different for an interrupt CALL compared to a normal CALL to subroutine. With an interrupt CALL, the program counter return address is saved; with a subroutine CALL, the saved program counter value is one less than the program counter return address.

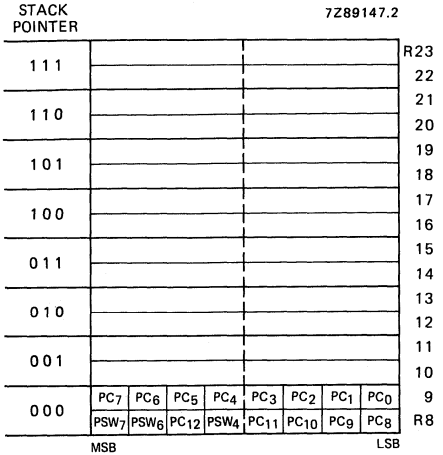


Fig. 5 Program counter stack.

IDLE and STOP modes

IDLE mode

When the microcontroller enters the IDLE mode via the IDLE instruction (01 H) the oscillator, timer/counter and serial I/O are kept running. The microcontroller exits from the IDLE mode by one of three interrupts if they are enabled or by activating a RESET. If the interrupt is not enabled the processor will remain in the IDLE mode. An active signal on the RESET pin restarts the microcontroller and a normal RESET sequence is executed (see Fig. 6).

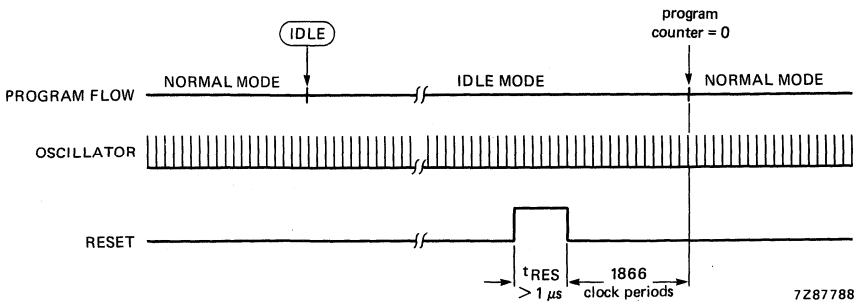


Fig. 6 Exit from IDLE mode via a RESET.

An active signal coming from an enabled interrupt causes the execution of the normal interrupt routine since normal interrupt scanning is still being carried out. A HIGH-to-LOW transition on the external interrupt pin ($\overline{INT}$/T0) reactivates the microcontroller. A LOW level applied to $\overline{INT}$/T0 will reactivate the microcontroller only in the STOP mode. Thus,if $\overline{INT}$/T0 was LOW before the microcontroller entered the IDLE mode, it must go HIGH before the microcontroller can be reactivated (see Fig. 7).

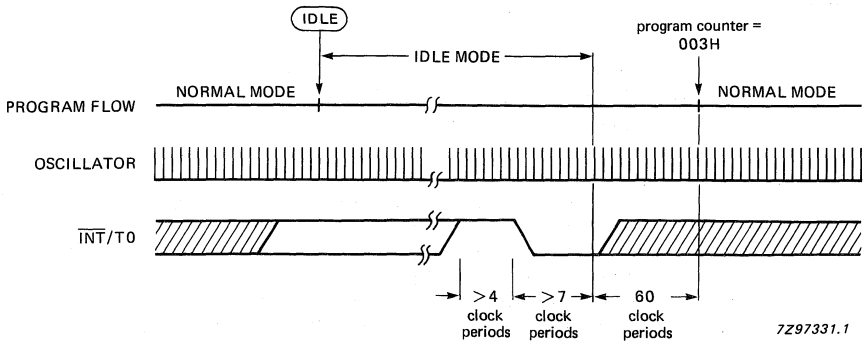


Fig. 7 Exit from IDLE mode via an interrupt.

Wake-up from the IDLE mode is ensured when $\overline{INT}$/T0 is HIGH for at least 4 CP (clock periods) followed by a LOW for 7 CP. After the initial forced CALL (*4H) operation (60 CP) the program continues with the external interrupt service routine.

* 1, 3, 5, 7, 9, B, D and F.

STOP mode

The microcontroller enters the STOP mode by the STOP instruction (22 H). The oscillator is switched off. The internal status of the CPU, RAM contents and the state of I/O ports are not affected. The microcontroller can be brought-out of the STOP mode by an active signal at the external interrupt input or by an external RESET signal. When one of these two signals is applied an internal delay of 1866 CP is provided to ensure that all internal clocks are operating correctly before restart (see Fig. 8). Note; the start-up time of a crystal oscillator is measured in milliseconds, and the 1866 CP count begins after this start-up time.

If the microcontroller exits from the STOP mode by activating RESET, a normal RESET sequence is executed.

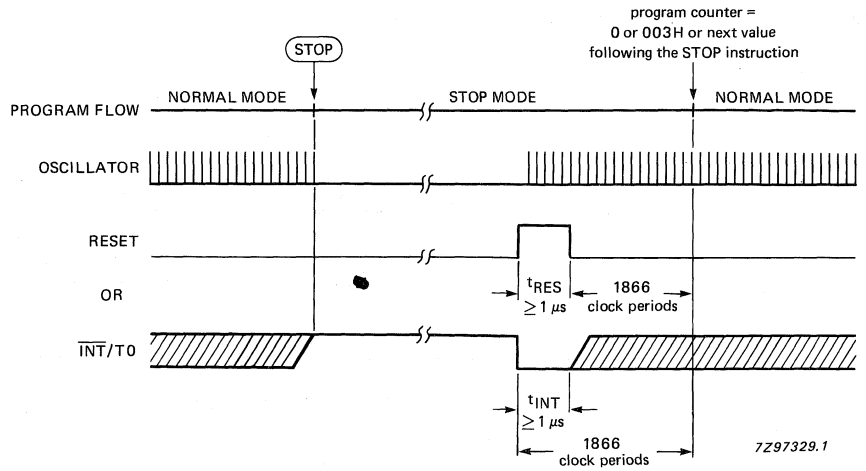


Fig. 8 Entering and exiting the STOP mode.

FUNCTIONAL DESCRIPTION (continued)

If the microcontroller exits the STOP mode by pulling the external interrupt input pin LOW, an interrupt sequence is executed only if the external interrupt is enabled. In this event the microcontroller resumes the normal program sequence after returning from the interrupt routine, as in the normal mode. If the interrupt is not enabled, it continues the normal program sequence, executing the instruction following the STOP instruction.

The microcontroller is restarted by a LOW level applied at the $\overline{INT}$/T0 pin, and not by a HIGH-to-LOW transition as in a normal interrupt mechanism.
Note: when leaving the STOP mode with an interrupt, a further instruction in the main program series is executed prior to entering the interrupt routine.

When the $\overline{INT}$/T0 level is active during the STOP instruction then no STOP is executed.

A LOW level on the external interrupt input of at least 1 μs will cause the microcontroller to exit the STOP mode.

I/O facilities

The PCF84C85 family has 32 I/O lines arranged as:

- Port 0 parallel port of 8 lines (P0.0 to P0.7)
- Port 1 parallel port of 8 lines (P1.0 to P1.7)
- Port 2 parallel port of 1 line (P2.3)
- D port 0 parallel port of 8 lines (DP0.0-DP.7)
- D port 1 parallel port of 7 lines (DP1.0-DP1.6)

In addition to these the PCF84C43 also comprises four specialized I/O lines:

- SCLK I^2C-bus serial clock line
- SDA I^2C-bus serial data line (shared with P2.3)
- $\overline{INT}$/T0 external interrupt and test input. When used as a test input T0 can be directly tested by conditional branch instructions JT0 and JNT0
- T1 test input which can alter program sequences when tested by conditional jump instructions JT1 and JTN1. T1 also functions as an input to the 8-bit timer/event counter.

Parallel ports

All parallel ports can be used as outputs or inputs, their structure is quasi-bidirectional.
Output data written to a port is latched and remains unchanged until rewritten.
Input data is not latched and so must be present until read by an input instruction.
Input lines are fully CMOS compatible, output lines can drive one TTL or CMOS load.

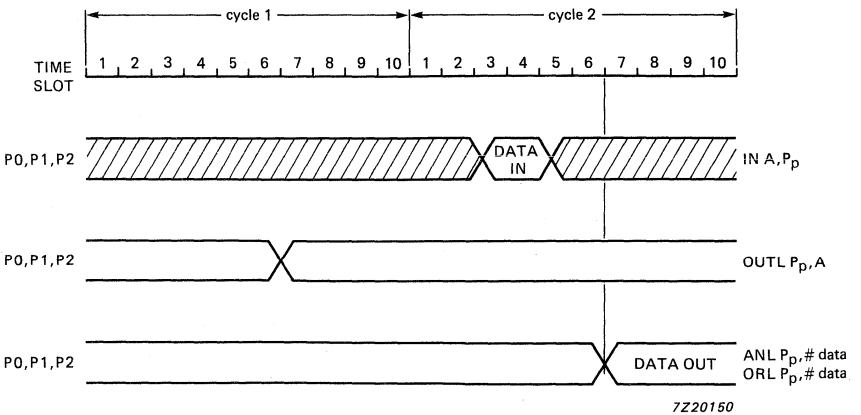


Fig. 9 Shows the timing diagram for all ports using IN, OUTL, ANL and ORL instructions. For the OUTL instruction data changes on time slot 7 of cycle 1. For the MOV, ANL and ORL instructions, the ports change on time slot 7 of cycle 2.

Fig. 10 shows the quasi-bidirectional I/O interface with push-pull output and switched pull-up current source. Each line is pulled up to V_{DD} via a constant current source (TR4), which is enabled via TR3 whenever one of the two output latches contains a logic 1. This current source is sufficient for a TTL HIGH level, yet can be pulled LOW by an external CMOS device, thus allowing the same pin to be used for both input and output.

When a logic 1 is written to the line for the first time (MQ = 1, SQ = 0), TR2 is switched on for the duration of the internal write pulse (one oscillator period), to provide a fast transition from logic 0 to logic 1. Subsequent writing of a logic 1 to the port lines will not switch TR2 on. This prevents unnecessary current through external components connected to the port lines of the same port which might be in the input mode and also connected to ground.

When a logic 0 is written to the line, TR3 switches off the current source. Current sinking capability is provided by TR1, which is now switched on. When used as an input, a logic 1 must first be written to the line, otherwise TR1 will remain low impedance.

The PCF84C85 family offers the possibility to select individually 31 of the 32 parallel port pins (not P23), by the following mask options:

Option 1 – STANDARD PORT; quasi-bidirectional I/O with switched pull-up current source of 100 μA (typ.) and P-channel booster transistor TR2.
TR2 is only active during 1 clock cycle (Fig. 10).

Option 2 – OPEN DRAIN; quasi-bidirectional I/O with only an N-channel open drain output. Application as an output requires connection of an external pull-up resistor (Fig. 11).

Option 3 – PUSH-PULL OUTPUT; drive capability of the output will be 1,6 mA (min.) at V_{DD} = 5 V in both polarities. To avoid a large current flowing through the output transistors during the input mode, these push-pull pins must only be used as outputs (Fig. 12).

DEVELOPMENT DATA

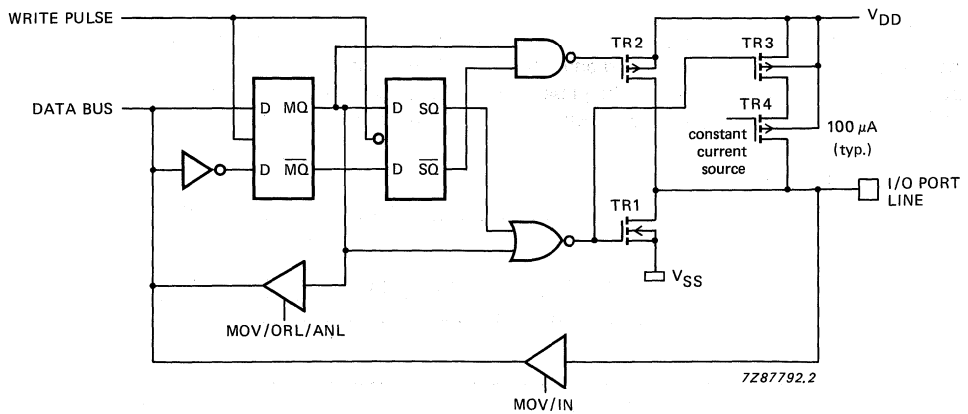


Fig. 10 Standard output with switched pull-up current source.

FUNCTIONAL DESCRIPTION (continued)

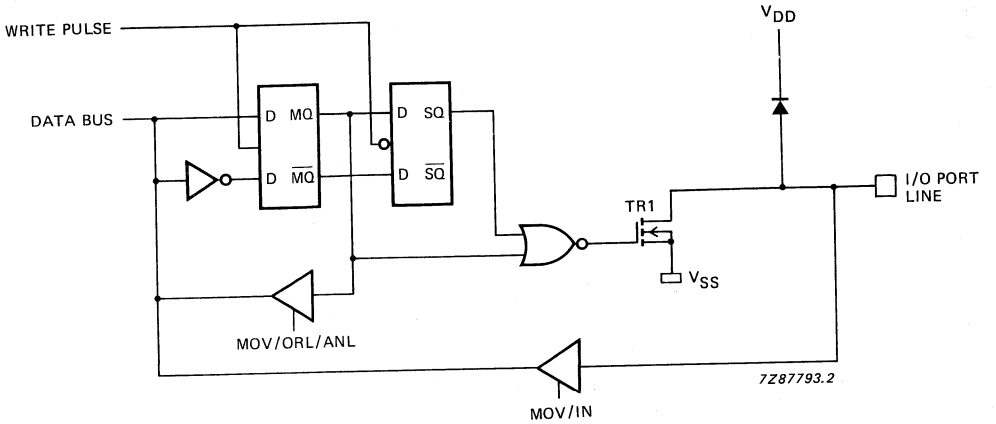


Fig. 11 Open drain output.

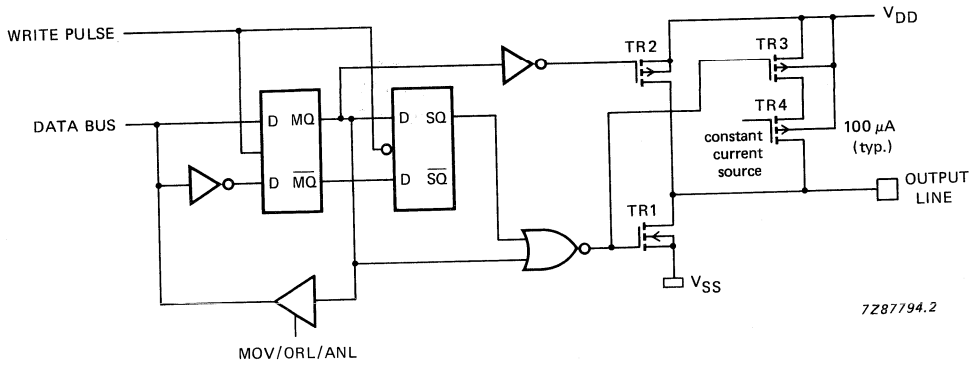


Fig. 12 Push-pull output.

Serial I/O (SIO)

The PCF84C85 has an on-chip serial I/O interface.

Whereas a normal microcontroller must regularly monitor the serial data bus for the presence of data, the serial I/O interface detects, receives and converts the serial data stream into parallel format without interrupting the execution of the current program. An interrupt is sent to the PCF84C85 only when a complete byte is received. It then reads the data byte in one instruction. Likewise during transmission the serial I/O interface performs parallel to serial conversion and subsequent serial output of the data. The microcontroller is only interrupted in the execution of its programmed tasks when a complete byte has been transmitted.

The design of the PCF84C85 serial I/O system allows any number of devices from PCF85XX family (clips) to be connected via the two-line serial bus. The ability of any devices to communicate, without interrupting the operation of any other devices on the bus, is an outstanding attribute of the system. This is achieved by allocating a specific 7-bit address to each device and providing a system whereby a device reacts only to a message prefixed with its own address or the 'general CALL' address. Address recognition is performed by the interface hardware so that operation of the microcontroller need only be interrupted when a valid address has been received. This saves significant processing time and memory space compared with a conventional microcontroller employing a software serial interface. When the addressing facility is not required, for instance in a system with only two microcontrollers, direct data transfer without addressing can be performed. In multi-master systems, an automatically invoked arbitration procedure prevents two or more devices from continuing simultaneous transmission.

In NORMAL (running) and IDLE mode, the serial I/O logic remains active; its internal system clock will be switched off when there is no activity on the serial bus.

After execution of the STOP instruction, the oscillator of the PCF84C85 is switched off. This means that the serial I/O logic will remain in the state it was at the occurance of the STOP instruction. To avoid "bus block" problems and to assure correct start-up of the bus after exit from the STOP mode, the user should disable the serial logic (ESO = 0) prior to the execution of the STOP instruction. This must be carried out only when the PCF84C85 has finished a serial data transfer.

DEVELOPMENT DATA

Serial I/O interface

Figure 13 shows the serial I/O interface. The clock line of the serial bus has exclusive use of pin 38 (SCLK) while the data line shares pin 37 (serial data) with the I/O line P23 of port 2. When the serial I/O is enabled, P23 is disabled as a parallel port line; (P23 and SCLK only open drain).

The microcontroller and interface communicate via the internal microcontroller bus and the Serial Interrupt Request line. Data and information controlling the operation of the interface are stored in four registers:

- Data shift register (S0)
- Serial I/O interface status word (S1)
- Serial clock control word (S2)
- Address register

FUNCTIONAL DESCRIPTION (continued)

Data shift register (S0)

Register S0 converts serial data to parallel format and vice versa. A pending interrupt is generated only after a complete byte has been transmitted, or after a complete data byte, specific address or 'general CALL' address has been received. The most significant bit is transmitted first.

Serial I/O interface status word (S1)

Register S1 provides information concerning the state of the interface and stores information from the microcontroller. Bits 0 to 3 are duplicated: control bits in these positions can only be written by the microcontroller, while interface bits can only be read.

MST and TRX (see Table 1)

These bits determine the operating mode of the serial I/O interface.

Table 1 Operating modes of the serial I/O interface

MST	TRX	operating mode
0	0	slave receiver
1	0	master receiver
0	1	slave transmitter
1	1	master transmitter

BB: Bus Busy.

This is the flag which indicates the status of the bus.

PIN: Pending Interrupt Not

PIN = '0' indicates the presence of a pending interrupt, which will cause a Serial Interrupt Request when the serial interrupt mechanism is enabled.

ESO: Enable Serial output

The ESO flag enables/disables the serial I/O interface: ESO = '1' enables, ESO = '0' disables. ESO can only be written by software.

BC0, BC1 and BC2

Bits BC0, BC1 and BC2 indicate the number of bits received or transmitted in a data stream. These bits can only be written by software.

AL: Arbitration Lost

The arbitration lost flag is set by hardware when the serial I/O interface, as master transmitter, loses a bus arbitration procedure.

AAS: Addressed As Slave

This flag is set by hardware when the interface detects either its own specific address or the 'general CALL' address as the first byte of a transfer and the interface has been programmed to operate in the address recognition mode.

AD0: Address Zero

This flag is set by hardware after detection of the 'general CALL' address when the interface is operating in the address recognition mode.

LRB: Last Received Bit

This contains either the last data bit received or, for a transmitting device in the acknowledgement mode, the acknowledgement signal from the receiving device.

Bits AL, AAS, AD0 and LRB can only be read by software.

Serial clock control word (S2)

Bits 0 to 4 of the clock control register S2 are used to set the frequency of the serial clock signal. When a 6 MHz crystal is used, the frequency of the serial clock can be varied between 154 kHz and 1 kHz (see Table 2). An asymmetrical clock with a HIGH-to-LOW ratio of 3 : 1 can be generated using bit 5. The asymmetrical clock allows a microcontroller more time per clock period for sampling the data line, making the timing of this action less critical. Bit 6 can be used to activate the acknowledge mode of the serial I/O. S2 is a write only register.

Address register

The address register contains the 7-bit address back-up latches and the bit (ALS) used to enable/disable the address recognition mode. The address register can be written using the MOV S0, A and MOV S0, # data instructions, but only when ES0 = '0'.

Serial I/O interrupt logic

An EN SI instruction enables and a DIS SI instruction disables the interrupt logic. When the logic is enabled, a pending interrupt results in a serial I/O interrupt to the processor, causing a CALL to location 5 in the ROM. When disabled, the presence of an interrupt is still indicated by PIN in S1, allowing the interrupt to be serviced. However, vectored interrupt will not occur.

FUNCTIONAL DESCRIPTION (continued)

Table 2 SIO clock pulse frequency control when using 6 MHz crystal

hexadecimal S20-S24 code	divisor	f_{XTAL}(6 MHz) f_{SCLK} (kHz)	f_{XTAL} (10 MHz) f_{SCLK} (kHz)
0	not allowed		
1	39	*154	*256
2	45	*133	*222
3	51	*118	*196
4	63	95	*159
5	75	80	*133
6	87	69	*115
7	99	61	*101
8	123	49	81
9	147	41	68
A	171	35	58
B	195	31	51
C	243	25	41
D	291	21	34
E	339	18	29
F	387	16	26
10	483	12	21
11	579	10	17
12	675	8,9	15
13	771	7,8	13,4
14	963	6,2	10,4
15	1155	5,2	8,7
16	1347	4,5	7,4
17	1539	3,9	6,5
18	1923	3,1	5,2
19	2307	2,6	4,3
1A	2691	2,2	3,7
1B	3075	2,0	3,3
1C	3843	1,6	2,6
1D	4611	1,3	2,2
1E	5379	1,1	1,9
1F	6147	1,0	1,6

* Not permitted for I^2C operation; the maximum clock frequency in the I^2C systems is 100 kHz.

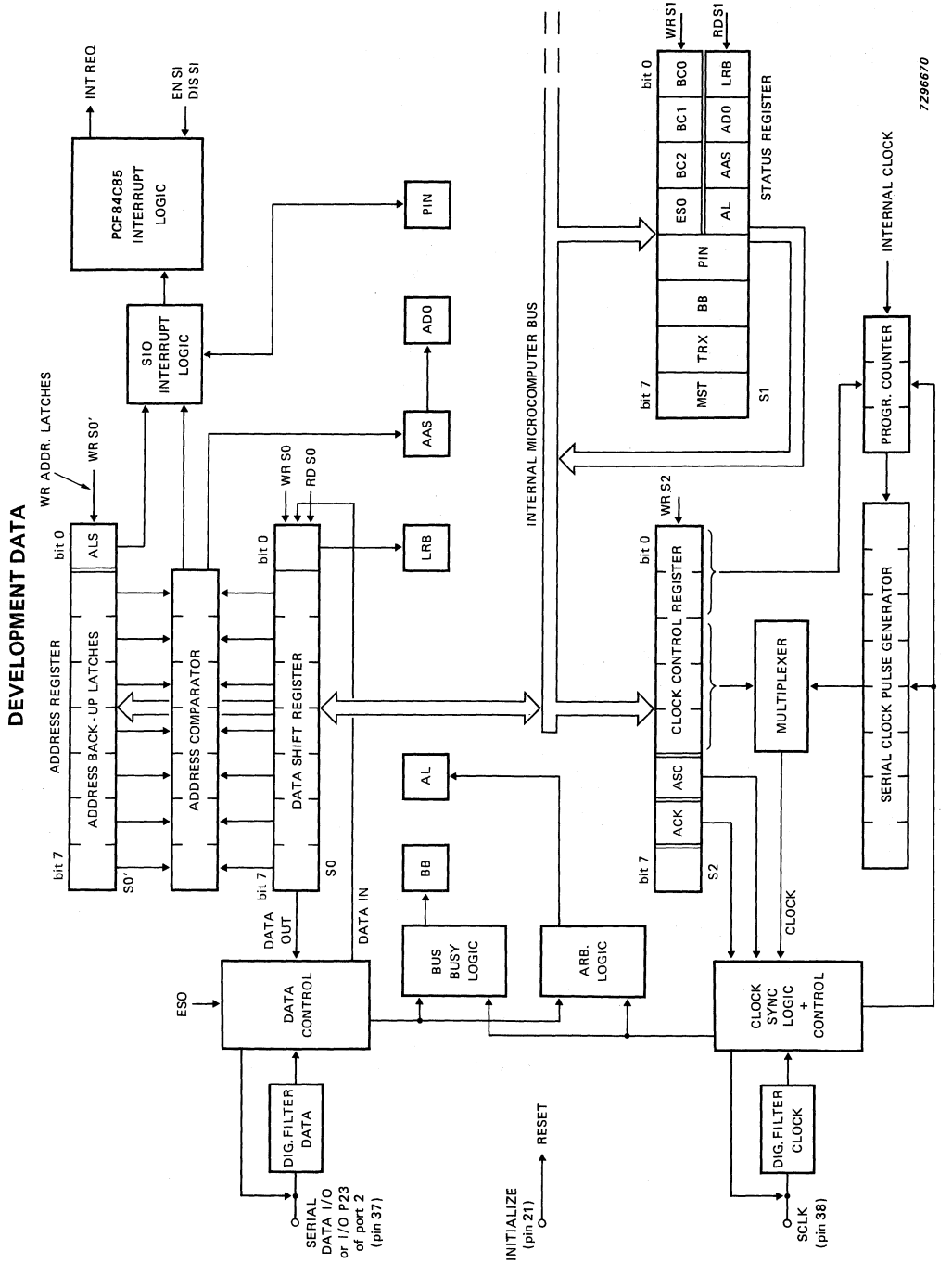


Fig. 13 Serial I/O interface.

FUNCTIONAL DESCRIPTION (continued)

Interrupts (see Fig. 14)

When the external interrupt is enabled, a HIGH-to-LOW transition on the $\overline{INT}$/T0 input initiates an external interrupt subroutine which causes a CALL to program memory location 3 following completion of the current instruction.

Serial I/O interrupt, when enabled, causes a CALL to location 5, and a timer/event counter overflow forces a CALL to location 7 when the timer interrupt is enabled.

When an interrupt subroutine starts, the contents of the program counter and bits 4, 6 and 7 of the PSW have been saved in the program counter stack. Accumulator contents have to be saved by software. Interrupt acknowledgement can be carried out by software via port pins. All interrupt subroutines must reside in memory bank 0.

Since the interrupt system is single level, once an interrupt is detected, all further interrupt requests are latched, but ignored, pending a RETR instruction to re-enable the interrupt input logic. After executing RETR, the program continues in the main part; this is independent of the occurance of a second interrupt during the running of the first routine. If 2 or 3 interrupts occur simultaneously, their priority is:

- (1) external
- (2) serial I/O
- (3) timer/event counter

An external interrupt can be generated by using the timer/counter in the event counter mode. The counter is first preset to (FFH), then EN TCNTI instruction is executed. A LOW-to-HIGH transition of the T1 input will then initiate an interrupt subroutine and cause a CALL to location 7.

On execution of a DIS I instruction, the PCF84C85 always clears the digital filter/latch and the External Interrupt Flag.

The Timer Flag (TF) is reset only when the JTF or JNTF instruction is executed or after RESET.

The Timer Interrupt Flag is set when timer overflow occurs, only if the timer interrupt is enabled.

The microcontroller will exit the IDLE mode when any one of the following three interrupts is enabled:

- External
- Serial I/O
- Timer/event counter

There is no internal pull-up or pull-down device connected to the external interrupt input (pin 1). If required pin 1 must be externally connected to a resistor ($R \leqslant 100\ k\Omega$). When the external interrupt is not used pin 1 must be connected to V_{DD}.

DEVELOPMENT DATA

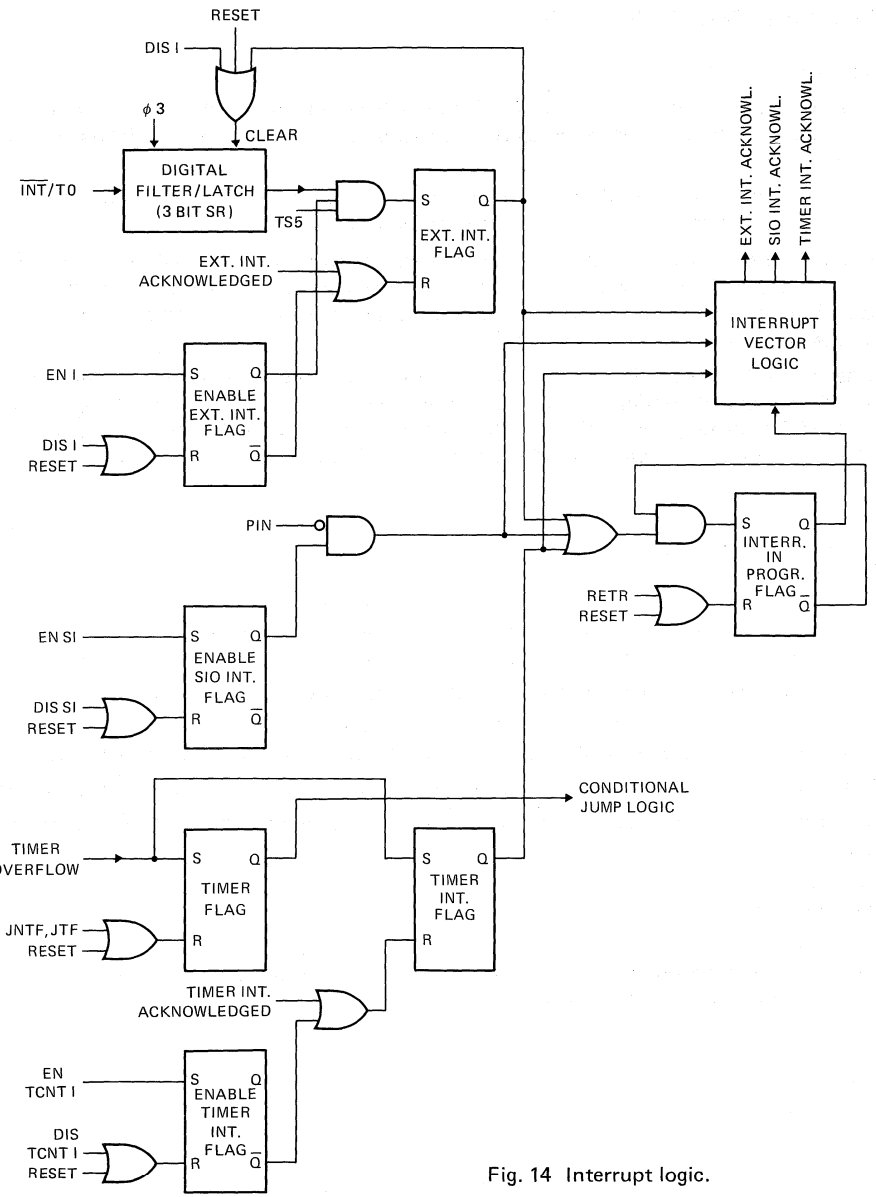


Fig. 14 Interrupt logic.

Notes to figure 14

1. $\overline{INT}$/T0 negative edge is always latched in the digital filter/latch.
2. Correct interrupt timing is ensured when INT/T0 is HIGH for $>$ 4 CP followed by a LOW for $>$ 7 CP.
3. When the interrupt in progress flag is set, further interrupts are latched but ignored, until RETR is executed.
4. A DIS I instruction always clears a pending external interrupt.
5. For all flip-flops, RESET overrules SET.

FUNCTIONAL DESCRIPTION (continued)

Oscillator (see Fig. 15)

The oscillator can be inhibited by the STOP instruction under software control. It is also inhibited when a low-supply voltage condition is present to prevent discharge of a weak back-up battery. Provided the supply voltage is within the operating range the oscillator will be restarted after a STOP instruction by a LOW level at the $\overline{INT}$/T0 pin or a HIGH level at the RESET pin.

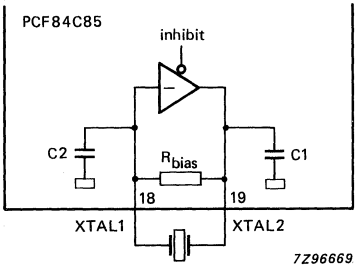


Fig. 15 Oscillator with integrated elements.

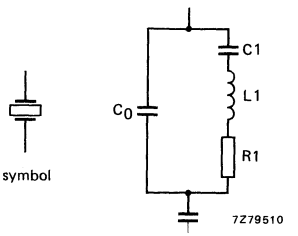


Fig. 16 Crystal unit equivalent circuit.

The values of crystal series resistance R1 and the crystal's total load capacitance C_L (C_0 + wiring + external capacitors) must not be above the curve (Fig. 17) for the corresponding frequency.
Note; if external capacitors are connected to XTAL 1 and XTAL2 they must be of equal value.

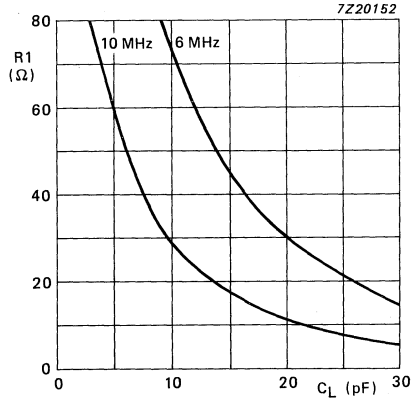


Fig. 17 Crystal circuit criteria.

The oscillator has the output drive capability via pin 19 (XTAL2). An external clock can be applied to pin 18 (XTAL1). A machine cycle consists of 10 time slots, each time slot being 3 oscillator periods.

Timer/event counter (see Fig. 18)

An internal 8-bit up-counter is provided. This can count external events, modulo-32 machine cycles, or machine cycles directly. Table 3 gives the instructions that control the counter and the prescaler, and the functions performed.

When used as a timer, the input to the counter is either the overflow or input of a 5-bit prescaler. When used as an event counter, LOW-to-HIGH transitions on pin 39 (T1) are counted. The maximum rate at which the counter may be incremented is once every machine cycle. When the counter overflows, the timer flag is set. The flag can be tested and reset using the JTF (jump is timer flag = 1) or JNTF instruction. Overflow also generates an interrupt to the processor via setting of the Timer Interrupt Flag when the timer/event counter interrupt is enabled.

Table 3 Timer/event counter control

function	timer mode modulo-1, modulo-32*	counter mode
CLEAR	MOV T,A (A) = 0 or RESET	MOV T,A (A) = 0 or RESET
PRESET	MOV T,A	MOV T,A
START	STRT T	STRT CNT
STOP	STOP TCNT or RESET	STOP TCNT or RESET
TEST	JTF/JNTF	JTF/JNTF
READ**	MOV A,T	MOV A,T

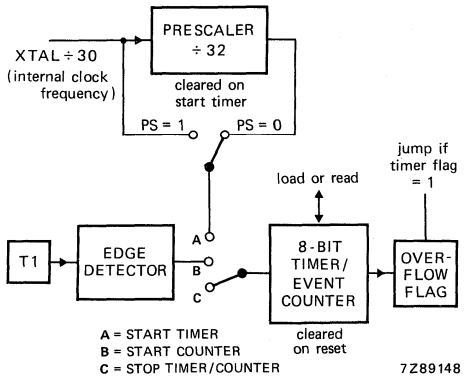


Fig. 18 Timer/event counter.

Program status word (see Fig. 19)

The program status word (PSW) is an 8-bit word (1 byte) in the CPU which stores information about the current status of the microcontroller.
The PSW bits are:

- Bits 0 to 2 — stack pointer bits (SP_0, SP_1, SP_2)
- Bit 3 — prescaler select (PS);
  0 = modulo-32; 1 = modulo-1 (no prescaling)
- Bit 4 — working register bank select (RBS);
  0 = register bank 0; 1 = register bank 1
- Bit 5 — not used (1)
- Bit 6 — auxiliary carry (AC); half-carry bit generated by an ADD instruction and used by the decimal adjust instruction DA A
- Bit 7 — carry (CY); the carry flag indicates that previous operation has resulted in an overflow of the accumulator.

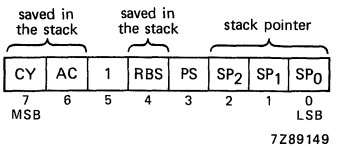


Fig. 19 Program status word.

* With prescaler select, PS = 0, the timer counts modulo-32 machine cycles, with PS = 1 it counts modulo-1 cycles (prescaler not used); prescaler cleared with STRT T, prescaler not readable.

** READ does not disturb the counting process.

DEVELOPMENT DATA

FUNCTIONAL DESCRIPTION (continued)

Program status word (continued)

All bits can be read using the MOV A, PSW instruction. Bits 7 and 6 are set and cleared by CPU operation. Bit 4 can be changed by a SEL RB instruction, bit 3 by the MOV PSW, A instruction, and bits 0, 1 and 2 by the CALL, RET or RETR instructions in the event of an interrupt. Bits 7, 6 and 4 are stored in the program counter stack during subroutine and interrupt calls. These bits are restored in the PSW with a RETR (return and restore) instruction which must be used at the end of an interrupt and can be used at the end of a normal subroutine. The RET instruction has no restore feature and cannot be used at the end of an interrupt.

Program counter (see Fig. 20)

The 13-bit program counter is able to address 8 K bytes of ROM. The arrangement of the bits is shown in figure 20. During an interrupt subroutine PC_{11} and PC_{12} are forced to logic 0. All 13 bits are saved in the stack during CALL and interrupt routines.

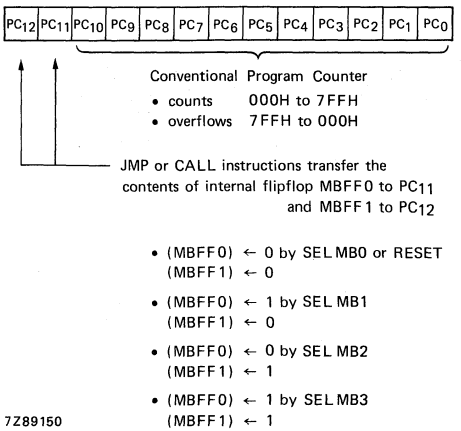


Fig. 20 Program counter.

Central processing unit

The PCF84C85 has arithmetic, logical and branching capabilities. The DA A, SWAP A and XCHD instructions simplify BCD arithmetic and the handling of nibbles. The MOVP A,@A instruction permits efficient table look-up from the current ROM page.

Conditional branch logic

The conditional branch logic within the processor enables several conditions, internal and external to the processor, to be tested by the user's program. Table 4 lists the conditional jump instructions used to change the program sequence. The DJNZ instruction decrements a designated register or data memory location and branches if the contents are not zero. This instruction is useful for looping control. The JMPP@A instruction allows multiway branches to destinations determined by the contents of the accumulator.

Table 4 Conditional branches

test	jump condition	jump instruction
accumulator	all bits zero	JZ
	any bit non-zero	JNZ
accumulator bit test	1	JB0 to JB7
carry flag	1	JC
	0	JNC
timer overflow flag	1	JTF
	0	JNTF
test input T0	1	JT0
	0	JNT0
test input T1	1	JT1
	0	JNT1
register	non-zero	DJNZ

Test input T1 (pin 39)

The T1 input line can be used as:

- A test input for branch instructions JT1 and JNT1
- An external input to the event counter

When used as a test input:

- JT1 instruction tests for logic 1 level
- JNT1 instruction tests for logic 0 level

When used as an input to the event counter, T1 must be LOW for > 4 CP, followed by a HIGH for > 4 CP. The transition can be recognized with a repetition rate of once per 30 oscillator clock periods (1 machine cycle).

There is no internal pull-up or pull-down resistor connected to the T1 input. If required it must be externally connected to a resistor ($R = \leqslant 100$ kΩ). When T1 is not used pin 39 must be connected to V_{DD} or V_{SS}.

Reset (pin 21)

A positive-going signal on the RESET input

- Sets the program counter to zero
- Selects location 0 of memory bank 0 and register bank 0
- Sets the stack pointer to zero (000); pointing to RAM address 8
- Disables the interrupts (external, timer and serial I/O)
- Stops the timer/event counter, then sets it to zero
- Sets the timer prescaler to modulo-32
- Resets the timer flag
- Sets all ports to input mode
- Sets the serial I/O to slave receiver mode and disables the serial I/O
- Cancels IDLE and STOP mode

FUNCTIONAL DESCRIPTION (continued)

Power-on-reset

The internal power-on reset circuit monitors the PCF84C85 supply voltage V_{DD}. For as long as the supply voltage remains below the internal reference level V_{ref} (typically 1,5 V) the oscillator is inhibited and RESET (pin 21) has an undefined level. When V_{DD} rises above the internal reference level, the oscillator is released and RESET is pulled high to V_{DD} by TR1 for a period t_D (typically 50 μs).

N.B. Because of the narrow bandwidth of the crystal, the start-up time of the oscillator is typically 10 ms.

Three modes of power-on reset are possible:

1. If V_{DD} can be switched on with fast rise time i.e. V_{DD} reaches its minimum operating value (corresponding to the selected oscillator frequency) before the RESET signal (t_D) has finished, then no extra components are required (see Fig. 21 and 22). Note that the first instruction is executed after the oscillator start-up time plus 1866 clock periods have elapsed.

2. If V_{DD} has a slow rise time then the RESET signal should be stretched by an external RC circuit (see Fig. 23 and 24). In the event of a short drop in the supply voltage, the diode path rapidly discharges the capacitor to ensure a reliable power-on reset. To ensure a correct reset, the RESET signal should reach at least 70% of the final value of V_{DD}. Given that the RESET voltage and V_{DD} rise exponentially, the above requirement is satisfied when the time constant τ of the RESET pulse is >8 times the time constant of V_{DD}. If V_{DD} rises linearly, then a RESET time constant >2 times the rise time of V_{DD} is required.

   When a reset is completed (RESET goes LOW) before the oscillator has started up, program execution begins after the oscillator start-up time plus 1866 clock periods have elapsed (see Fig. 24). If the oscillator is started-up prior to the completion of RESET, then program execution begins 1866 clock periods after RESET goes LOW.

3. Figure 25 shows an external reset to the PCF84C85 during power-on. The external reset signal must remain HIGH until V_{DD} has reached its minimum operating value corresponding to the selected oscillator frequency. When a reset is completed (RESET goes LOW) before the oscillator has started up, program execution begins after the oscillator start-up time plus 1866 clock periods have elapsed (see Fig. 26). If the oscillator is started-up prior to the completion of RESET, then program execution begins 1866 clock periods after RESET goes LOW.

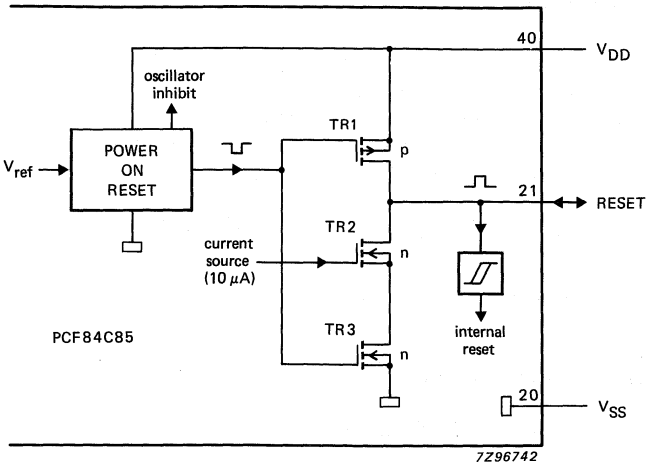


Fig. 21 Power-on-reset configuration.

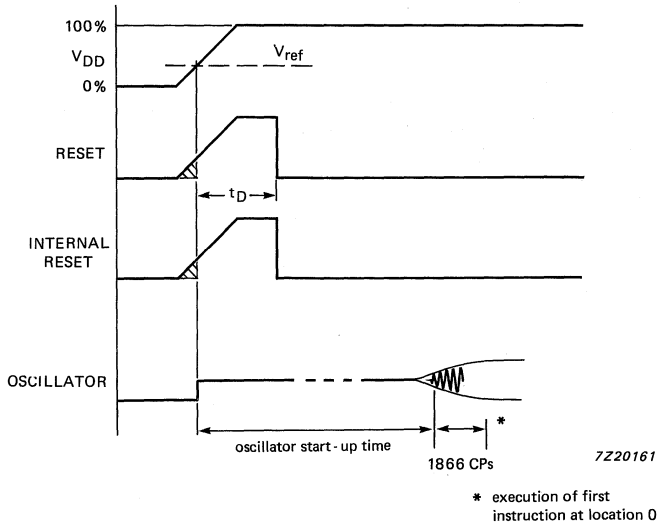


Fig. 22 Timing of power-on-reset with fast rise time.

FUNCTIONAL DESCRIPTION (continued)

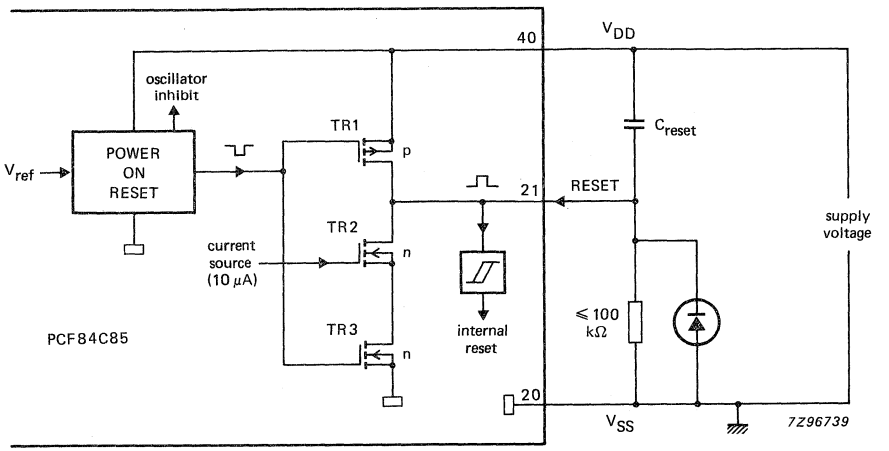


Fig. 23 Stretched power-on-reset with external components.

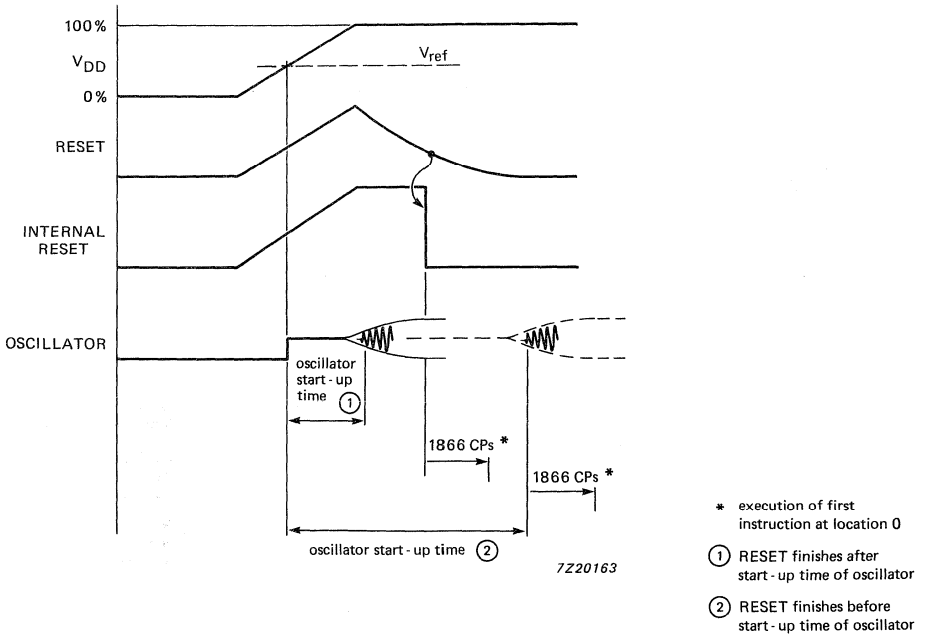


Fig. 24 Timing of power-on-reset with a slowly rising V_{DD} and a stretched RESET pulse.

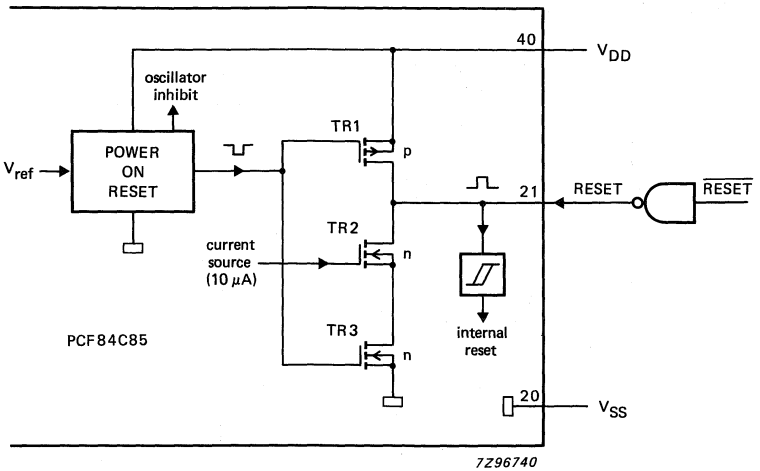


Fig. 25 External power-on-reset configuration.

DEVELOPMENT DATA

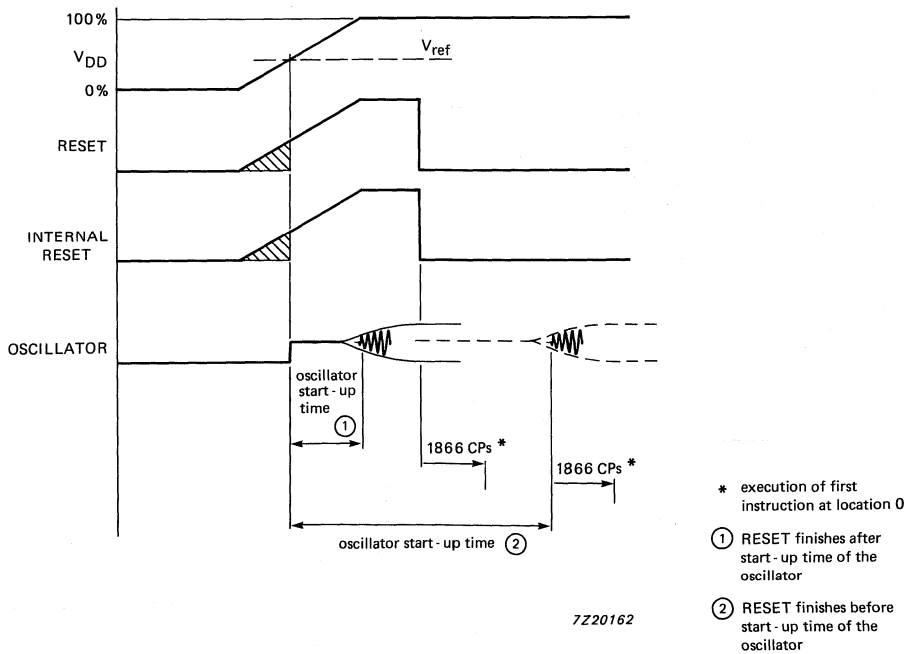


Fig. 26 Timing of external power-on-reset.

INSTRUCTION SET

The PCF84C85 instruction set consists of over 80 one and two byte instructions and is identical to the MAB8400 instruction set. New instructions are added for STOP and IDLE mode. Program code efficiency is high because all RAM locations and all ROM locations on a 256 byte page require only a single byte address.

Table 7 gives the instruction set of the PCF84C85. Table 6 shows the instruction map and Table 5 details the symbols and definition descriptions that are used.

Table 5 Symbols and definitions used in Table 7

symbol	definition description
A	accumulator
addr	program memory address
Bb	bit designation (b = 0-7)
RBS	register bank select
C	carry bit (bit CY)
CNT	event counter
Dx	Derivative register designation (x = 0,1,2 or 3)
data	8-bit number or expression
I	interrupt
MB	memory bank
MBFF	memory bank flip-flop
P	mnemonic for 'in-page' operation
PC	program counter
Pp	port designation (p = 0, 1 or 2)
PSW	program status word
RB	register bank
Rr	register designation (r = 0-7)
Sn	serial I/O register
SP	stack pointer
T	timer
TF	timer flag
T0, T1	test 0 and 1 inputs
#	immediate data prefix
@	indirect address prefix
(X)	contents of X
((X))	contents of location addressed by X
←	is replaced by
↔	is exchanged with

Table 6 PCF84C85 instruction map

first hexadecimal character of opcode ↓ \ second hexadecimal character of opcode →	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
0	NOP	IDLE		ADD A,#data	JMP page 0	EN I	JNTF addr	DEC A	IN A,Pp				MOV A,Sn			
									0	1	2		0	1		
1	INC @Rr		JBO addr	ADDC A,#data	CALL page 0	DIS I	JTF addr	INC A	INC Rr							
	0	1							0	1	2	3	4	5	6	7
2	XCH A,@Rr		STOP	MOV A,#data	JMP page 1	EN TCNTI	JNTO addr	CLR A	XCH A,Rr							
	0	1							0	1	2	3	4	5	6	7
3	XCHD A,@Rr		JB1 addr		CALL page 1	DIS TCNTI	JTO addr	CPL A	OUTL Pp,A				MOV Sn,A			
	0	1							0	1	2		0	1	2	
4	ORL A,@Rr		MOV A,T	ORL A,#data	JMP page 2	STRT CNT	JNT1 addr	SWAP A	ORL A,Rr							
	0	1							0	1	2	3	4	5	6	7
5	ANL A,@Rr		JB2 addr	ANL A,#data	CALL page 2	STRT T	JT1 addr	DA A	ANL A,Rr							
	0	1							0	1	2	3	4	5	6	7
6	ADD A,@Rr		MOV T,A		JMP page 3	STOP TCNT		RRC A	ADD A,Rr							
	0	1							0	1	2	3	4	5	6	7
7	ADDC A,@Rr		JB3 addr		CALL page 3			RR A	ADDC A,Rr							
	0	1							0	1	2	3	4	5	6	7
8				RET	JMP page 4	EN SI			ORL Pp,#data				MOV A,Dx	MOV Dx,A	ANL Dx,A	ORL Dx,A
									0	1	2					
9			JB4 addr	RETR	CALL page 4	DIS SI	JNZ addr	CLR C	ANL Pp,#data				MOV Sn,#data			
									0	1	2		0	1	2	
A	MOV @Rr,A			MOVP A,@A	JMP page 5	SEL MB2		CPL C	MOV Rr,A							
	0	1							0	1	2	3	4	5	6	7
B	MOV @Rr,#data		JB5 addr	JMPP @A	CALL page 5	SEL MB3			MOV Rr,#data							
	0	1							0	1	2	3	4	5	6	7
C	DEC @Rr				JMP page 6	SEL RB0	JZ addr	MOV A,PSW	DEC Rr							
	0	1							0	1	2	3	4	5	6	7
D	XRL A,@Rr		JB6 addr	XRL A,#data	CALL page 6	SEL RB1		MOV PSW,A	XRL A,Rr							
	0	1							0	1	2	3	4	5	6	7
E	DJNZ @Rr,addr				JMP page 7	SEL MB0	JNC addr	RL A	DJNZ Rr, addr							
	0	1							0	1	2	3	4	5	6	7
F	MOV A,@Rr		JB7 addr		CALL page 7	SEL MB1	JC addr	RLC A	MOV A,Rr							
	0	1							0	1	2	3	4	5	6	7

INSTRUCTION SET (continued)

Table 7 Instruction set

	mnemonic	opcode (hex.)	bytes/ cycles	description	function		notes
ACCUMULATOR	ADD A, Rr	6*	1/1	Add register contents to A	(A)←(A) + (Rr)	r = 0–7	1
	ADD A, @Rr	60 61	1/1	Add RAM data, addressed by Rr, to A	(A)←(A) + ((R0)) (A)←(A) + ((R1))		1
	ADD A, #data	03 data	2/2	Add immediate data to A	(A)←(A) + data		1
	ADDC A, Rr	7*	1/1	Add carry and register contents to A	(A)←(A) + (Rr) + (C)	r = 0–7	1
	ADDC A, @Rr	70 71	1/1	Add carry and RAM data, addressed by Rr, to A	(A)←(A) + ((R0)) + (C) (A)←(A) + ((R1)) + (C)		1
	ADDC A, #data	13 data	2/2	Add carry and immediate data to A	(A)←(A) + data + (C)		1
	ANL A, Rr	5*	1/1	'AND' Rr with A	(A)←(A) AND (Rr)	r = 0–7	
	ANL A, @Rr	50 51	1/1	'AND' RAM data, addressed by Rr, with A	(A)←(A) AND ((R0)) (A)←(A) AND ((R1))		
	ANL A, #data	53 data	2/2	'AND' immediate data with A	(A)←(A) AND data		
	ORL A, Rr	4*	1/1	'OR' Rr with A	(A)←(A) OR (Rr)	r = 0–7	
	ORL A, @Rr	40 41	1/1	'OR' RAM data, addressed by Rr, with A	(A)←(A) OR ((R0)) (A)←(A) OR ((R1))		
	ORL A, #data	43 data	2/2	'OR' immediate data with A	(A)←(A) OR data		
	XRL A, Rr	D*	1/1	'XOR' Rr with A	(A)←(A) XOR (Rr)	r = 0–7	
	XRL A, @Rr	D0 D1	1/1	'XOR' RAM, addressed by Rr, with A	(A)←(A) XOR ((R0)) (A)←(A) XOR ((R1))		
	XRL A, #data	D3 data	2/2	'XOR' immediate data with A	(A)←(A) XOR data		
	INC A	17	1/1	increment A by 1	(A)←(A) + 1		
	DEC A	07	1/1	decrement A by 1	(A)←(A) −1		
	CLR A	27	1/1	clear A to zero	(A)←0		
	CPL A	37	1/1	one's complement A	(A)←NOT(A)		
	RL A	E7	1/1	rotate A left	$(A_{n+1})\leftarrow(A_n)$ $(A_0)\leftarrow(A_7)$	n = 0–6	

ACCUMULATOR (cont.)	RLC A	F7	1/1	rotate A left through carry	$(A_{n+1})\leftarrow A_n$ $(A_0)\leftarrow(C)$, $(C)\leftarrow(A_7)$	n = 0–6	2
	RR A	77	1/1	rotate A right	$(A_n)\leftarrow(A_{n+1})$ $(A_7)\leftarrow(A_0)$	n = 0–6	
	RRC A	67	1/1	rotate A right through carry	$(A_n)\leftarrow(A_{n+1})$ $(A_7)\leftarrow(C)$, $(C)\leftarrow(A_0)$	n = 0–6	2
	DA A	57	1/1	decimal adjust A			2
	SWAP A	47	1/1	swap nibbles of A	$(A_{4-7})\leftrightarrow(A_{0-3})$		
DATA MOVES	MOV A, Rr	F*	1/1	move register contents to A	(A)←(Rr)	r = 0–7	
	MOV A, @Rr	F0 F1	1/1	move RAM data, addressed by Rr, to A	(A)←((R0)) (A)←((R1))		
	MOV A, #data	23 data	2/2	move immediate data to A	(A)←data		
	MOV Rr, A	A*	1/1	move accumulator contents to register	(Rr)←(A)	r = 0–7	
	MOV @Rr, A	A0 A1	1/1	move accumulator contents to RAM location addressed by Rr	((R0))←(A) ((R1))←(A)		
	MOV Rr, #data	B* data	2/2	move immediate data to Rr	(Rr)←data		
	MOV @Rr, #data	B0 data B1 data	2/2	move immediate data to RAM location addressed by Rr	((R0))←data ((R1))←data		
	XCH A, Rr	2*	1/1	exchange accumulator contents with Rr	(A)↔(Rr)	r = 0–7	
	XCH A, @Rr	20 21	1/1	exchange accumulator contents with RAM data addressed by Rr	(A)↔((R0)) (A)↔((R1))		
	XCHD A, @Rr	30 31	1/1	exchange lower nibbles of A and RAM data addressed by Rr	$(A_{0-3})\leftrightarrow((R0_{0-3}))$ $(A_{0-3})\leftrightarrow((R1_{0-3}))$		
	MOV A, PSW	C7	1/1	move PSW contents to accumulator	(A)←(PSW)		
	MOV PSW, A	D7	1/1	move accumulator bit 3 to PSW_3	$(PSW_3)\leftarrow(A_3)$		3
	MOVP A, @A	A3	1/2	move indirectly addressed data in current page to A	$(PC_{0-7})\leftarrow(A)$, $(A)\leftarrow((PC))$		
FLAGS	CLR C	97	1/1	clear carry bit	(C)←0		2
	CPL C	A7	1/1	complement carry bit	(C)←NOT(C)		2

INSTRUCTION SET (continued)

	mnemonic	opcode (hex.)	bytes/ cycles	description	function	notes
REGISTER	INC Rr	1*	1/1	increment register by 1	(Rr)←(Rr) + 1 r = 0–7	
	INC @Rr	10 11	1/1	increment RAM data, addressed by Rr, by 1	((R0))←((R0)) + 1 ((R1))←((R1)) + 1	
	DEC Rr	C*	1/1	decrement register by 1	(Rr)←(Rr) – 1 r = 0–7	
	DEC @Rr	C0 C1	1/1	decrement RAM data, addressed by Rr, by 1	((R0))←((R0)) – 1 ((R1))←((R1)) – 1	
BRANCH	JMP addr	• 4 address	2/2	unconditional jump within a 2 K bank	(PC_{8-10})←$addr_{8-10}$ (PC_{0-7})←$addr_{0-7}$ (PC_{11-12})←MBFF 0–1	
	JMPP @A	B3	1/2	indirect jump within a page	(PC_{0-7})←((A))	
	DJNZ Rr, addr	E* address	2/2	decrement Rr by 1 and jump if not zero to addr	(Rr)←(Rr) – 1 r = 0–7 if (Rr) not zero (PC_{0-7})←addr	
	DJNZ @Rr, addr	E0 address	2/2	decrement RAM data, addressed by Rr by 1 and jump if not zero to addr	((R0))←((R0)) – 1 if ((R0)) not zero (PC_{0-7})←addr	
		E1 address			((R1))←((R1)) – 1 if ((R1)) not zero (PC_{0-7})←addr	
	JBb addr	▲ 2 address	2/2	jump to addr if Acc. bit b = 1	if b = 1 : (PC_{0-7})←addr b = 0–7	
	JC addr	F6 address	2/2	jump to addr if C = 1	if C = 1 : (PC_{0-7})←addr	
	JNC addr	E6 address	2/2	jump to addr if C = 0	if C = 0 : (PC_{0-7})←addr	
	JZ addr	C6 address	2/2	jump to addr if A = 0	if A = 0 : (PC_{0-7})←addr	
	JNZ addr	96 address	2/2	jump to addr if A is NOT zero	if A≠0 : (PC_{0-7})←addr	
	JT0 addr	36 address	2/2	jump to addr if T0 = 1	if T0 = 1: (PC_{0-7})←addr	
	JNT0 addr	26 address	2/2	jump to addr if T0 = 0	if T0 = 0: (PC_{0-7})←addr	
	JT1 addr	56 address	2/2	jump to addr if T1 = 1	if T1 = 1: (PC_{0-7})←addr	
	JNT1 addr	46 address	2/2	jump to addr if T1 = 0	if T1 = 0: (PC_{0-7})←addr	
	JTF addr	16 address	2/2	jump to addr if Timer Flag = 1	if TF = 1: (PC_{0-7})←addr	4
	JNTF addr	06 address	2/2	jump to addr if Timer Flag = 0	if TF = 0: (PC_{0-7})←addr	

TIMER/EVENT COUNTER	MOV A, T	42	1/1	move timer/event counter contents to accumulator	(A)←(T)	
	MOV T, A	62	1/1	move accumulator contents to timer/event counter	(T)←(A)	
	STRT CNT	45	1/1	start event counter		
	STRT T	55	1/1	start timer		
	STOP TCNT	65	1/1	stop timer/event counter		
	EN TCNTI	25	1/1	enable timer/event counter interrupt		
	DIS TCNTI	35	1/1	disable timer/event counter interrupt		
CONTROL	EN I	05	1/1	enable external interrupt		
	DIS I	15	1/1	disable external interrupt		
	SEL RB0	C5	1/1	select register bank 0	(RBS)←0	5
	SEL RB1	D5	1/1	select register bank 1	(RBS)←1	5
	SEL MB0	E5	1/1	select program memory bank 0	(MBFF0)←0, (MBFF1)←0	
	SEL MB1	F5	1/1	select program memory bank 1	(MBFF0)←1, (MBFF1)←0	
	SEL MB2	A5	1/1	select program memory bank 2	(MBFF0)←0, (MBFF1)←1	
	SEL MB3	B5	1/1	select program memory bank 3	(MBFF0)←1, (MBFF1)←1	
	STOP	22	1/1	enter STOP mode		
	IDLE	01	1/1	enter IDLE mode		
SUBROUTINE	CALL addr	▲ 4 address	2/2	jump to subroutine	$((SP))\leftarrow(PC), (PSW_{4,6,7})$ 6 $(SP)\leftarrow(SP)+1$ $(PC_{8-10})\leftarrow addr_{8-10}$ $(PC_{0-7})\leftarrow addr_{0-7}$ $(PC_{11-12})\leftarrow$MBFF 0–1	6
	RET	83	1/2	return from subroutine	$(SP)\leftarrow(SP)-1$ $(PC)\leftarrow((SP))$	6
	RETR	93	1/2	return from interrupt and restore bits 4, 6, 7 of PSW	$(SP)\leftarrow(SP)-1$ $(PSW_{4,6,7})+(PC)\leftarrow((SP))$	6

INSTRUCTION SET (continued)

	mnemonic	opcode (hex.)	bytes/ cycles	description	function	notes
PARALLEL INPUT/OUTPUT	IN A, Pp	08 09 0A	1/2	input port p data to accumulator	(A)←(P0) (A)←(P1) (A)←(P2)	 7
	OUTL Pp, A	38 39 3A	1/2	output accumulator data to port p	(P0)←(A) (P1)←(A) (P2)←(A)	
	ANL Pp, #data	98 data 99 data 9A data	2/2	AND port p data with immediate data	(P0)←(P0) AND data (P1)←(P1) AND data (P2)←(P2) AND data	
	ORL Pp, #data	88 data 89 data 8A data	2/2	OR port p data with immediate data	(P0)←(P0) OR data (P1)←(P1) OR data (P2)←(P2) OR data	
DERIVATIVE INPUT/OUTPUT	MOV A, Dx	8C 00 8C 01	2/2 2/2	input pin data of port DP0, DP1 to accumulator	(A)←(D0) (A)←(D1)	 8
	MOV Dx, A	8D 02 8D 03	2/2 2/2	move contents of accumulator to latch of port DP0, DP1	(D2)←(A) (D3)←(A)	
	ANL Dx, A	8E 02 8E 03	2/2 2/2	AND contents of DP0, DP1 latch with accumulator	(D2)←(D2) AND (A) (D3)←(D3) AND (A)	
	ORL Dx, A	8F 02 8F 03	2/2 2/2	OR contents of DP0, DP1 latch with accumulator	(D2)←(D2) OR (A) (D3)←(D3) OR (A)	
	MOV A, Dx	8C 02 8C 03	2/2 2/2	move contents of DP0, DP1 latch to accumulator	(A)←(D2) (A)←(D3)	 8

	mnemonic	opcode (hex.)	bytes/ cycles	description	function	notes
SERIAL INPUT/OUTPUT	MOV A, S_n	0C 0D	1/2	move serial I/O register contents to accumulator	(A)←(S0) (A)←(S1)	9
	MOV S_n, A	3C 3D 3E	1/2	move accumulator contents to serial I/O register	(S0)←(A) (S1)←(A) (S2)←(A)	
	MOV S_n, #data	9C data 9D data 9E data	2/2	move immediate data to serial I/O register	(S0)←data (S1)←data (S2)←data	
	EN SI	85	1/1	enable serial I/O interrupt		
	DIS SI	95	1/1	disable serial I/O interrupt		
	NOP	00	1/1	no operation		

Notes to Table 8

1. PSW CY, AC affected
2. PSW CY affected
3 PSW PS affected
4. Execution of JTF and JNTF instructions resets the Timer Flag (TF).

* : 8,9,A,B,C,D,E,F
● : 0,2,4,6,8,A,C,E
▲ : 0,3,5,7,9,B,D,F

5. PSW RBS affected
6. PSW SP_0, SP_1, SP_2 affected
7. (A) = 0000 (P23) 111
8. The MSB of A becomes a logic 0
9. (S1) has a different function in read and write operations, see serial I/O interface.

RATINGS

Limiting values in accordance with the Absolute Maximum System (IEC 134)

Supply voltage (pin 40)	V_{DD}		−0,8 to +8	V
All input voltages	V_I		−0,8 to V_{DD} +0,8	V
D.C. current into any input or output	$\pm I_I$, $\pm I_O$	max.	10	mA
Total power dissipation				
Power dissipation per output				
except P23, SCLK	P_O	max.	50	mW
P23, SCLK	P_O	max.	180	mW
Storage temperature range	T_{stg}		−65 to +150	°C
Operating ambient temperature range	T_{amb}		−40 to +85	°C
Operating junction temperature	T_j	max.	125	°C

HANDLING

Inputs and outputs are protected against electrostatic discharge in normal handling. However, to be totally safe, it is desirable to take normal precautions appropriate to handling MOS devices (see 'Handling MOS devices').

DEVELOPMENT DATA

D.C. CHARACTERISTICS

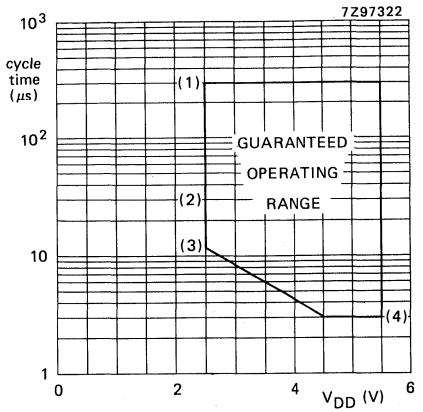
V_{DD} = 2,5 to 5,5 V; V_{SS} = 0 V; T_{amb} = −40 to + 85 °C; all voltages with respect to V_{SS}; unless otherwise specified.

parameter	symbol	min.	typ.	max.	unit
Supply voltage					
operating (see Fig. 27)	V_{DD}	2,5	–	5,5	V
Supply current					
operating (see Fig. 28)					
at V_{DD} = 5 V; f_{XTAL} = 10 MHz	I_{DD}	–	1,6	3,2	mA
at V_{DD} = 5 V; f_{XTAL} = 6 MHz	I_{DD}	–	1	2	mA
at V_{DD} = 3 V; f_{XTAL} = 3,58 MHz	I_{DD}	–	0,5	0,6	mA
IDLE mode (see Fig. 29)					
at V_{DD} = 5 V; f_{XTAL} = 10 MHz	I_{DD}	–	0,8	1,6	mA
at V_{DD} = 5 V; f_{XTAL} = 6 MHz	I_{DD}	–	1	1	mA
at V_{DD} = 3 V; f_{XTAL} = 3,58 MHz	I_{DD}	–	0,25	0,4	mA
STOP mode (see Fig. 35 and note 1)					
at V_{DD} = 2,5 V; T_{amb} = 25 °C	I_{DD}	–	1,2	2,5	μA
at V_{DD} = 2,5 V; T_{amb} = 70 °C	I_{DD}	–	–	10	μA
Inputs					
Input voltage LOW	V_{IL}	0	–	$0{,}3V_{DD}$	V
Input voltage HIGH	V_{IH}	$0{,}7V_{DD}$	–	V_{DD}	V
Input leakage current					
as $V_{SS} < V_I < V_{DD}$	$\pm I_{IL}$	–	–	1	μA
Outputs					
Output voltage LOW					
at V_I = V_{SS} or V_{DD}; $\lvert I_O \rvert < 1$ μA	V_{OL}	–	–	0,05	V
Output sink current LOW					
at V_{DD} = 5 V ± 10%; V_O = 0,4 V					
except P23/SDA, SCLK (see Fig. 31)	I_{OL}	1,6	3	–	mA
P23/SDA, SCLK (see Fig. 32)	I_{OL}	3	–	–	mA
Pull-up output source current HIGH (see Fig. 33)					
at V_{DD} = 5 V ± 10%; V_O = $0{,}7V_{DD}$	$-I_{OH}$	40	–	–	μA
at V_{DD} = 5 V ± 10%; V_O = V_{SS}	$-I_{OH}$	–	–	400	μA
Push-pull output source current HIGH					
at V_{DD} = 5 V ± 10%; V_O = V_{DD}−0,4 V	$-I_{OH}$	1,6	3	–	mA

Note 1

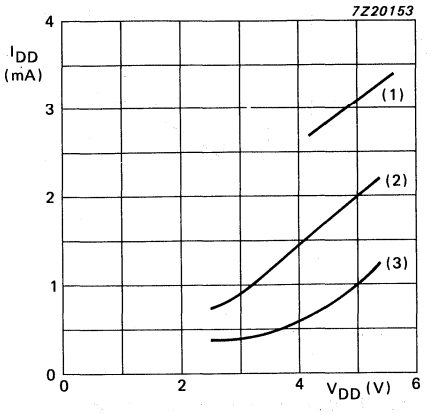
Crystal connected between XTAL1 and XTAL2; SCL and SDA pulled to V_{DD} via 5,6 kΩ resistor; T1 at V_{SS}, $\overline{INT}$ at V_{DD}.

A.C. CHARACTERISTICS (continued)



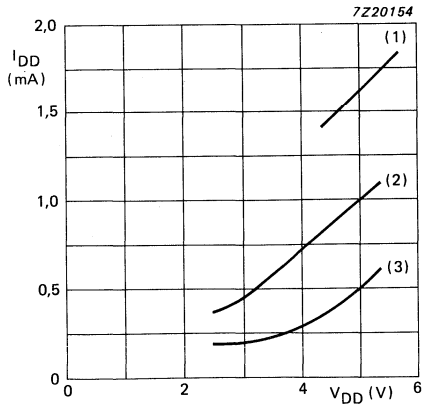
(1) clock frequency = 100 kHz
(2) clock frequency = 1 MHz
(3) clock frequency = 3 MHz
(4) clock frequency = 10 MHz

Fig. 27 Maximum clock frequency (f_{XTAL}) as a function of the supply voltage (V_{DD}).



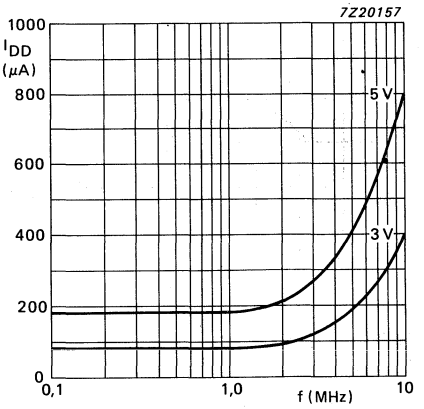
(1) clock frequency = 10 MHz
(2) clock frequency = 6 MHz
(3) clock frequency = 3,58 MHz

Fig. 28 Maximum supply current (I_{DD}) in operating mode as a function of the supply voltage.



(1) clock frequency = 10 MHz
(2) clock frequency = 6 MHz
(3) clock frequency = 3,58 MHz

Fig. 29 Maximum supply current (I_{DD}) in IDLE mode as a function of the supply voltage (V_{DD}).



(1) V_{DD} = 3 V
(2) V_{DD} = 5 V

Fig. 30 Typical supply current during IDLE mode as a function of frequency.

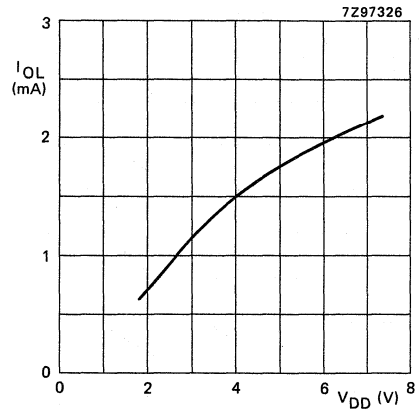


Fig. 31 Output sink current LOW (I_{OL}), except outputs P23/SDA and SCLK, as a function of supply voltage (V_{DD}); V_O = 0,4 V.

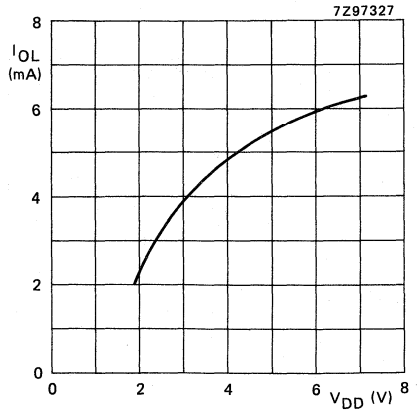
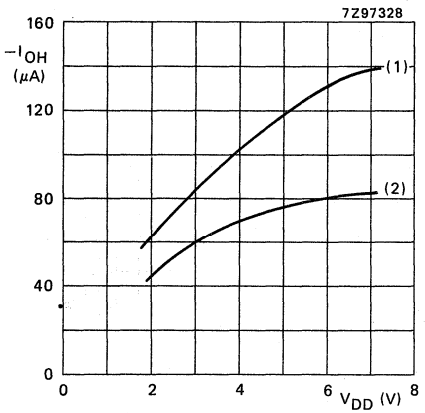


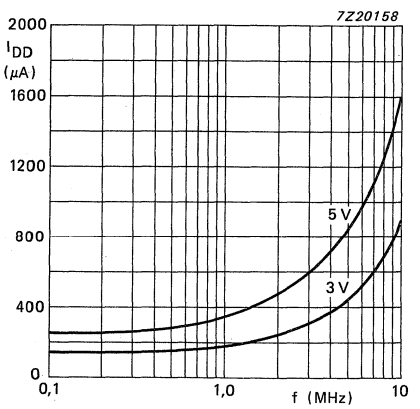
Fig. 32 Output current LOW (I_{OL}), outputs P23/SDA and SCLK, as a function of supply voltage (V_{DD}); V_O = 0,4 V.

DEVELOPMENT DATA



(1) $V_O = V_{SS}$
(2) $V_O = 0{,}7\ V_{DD}$

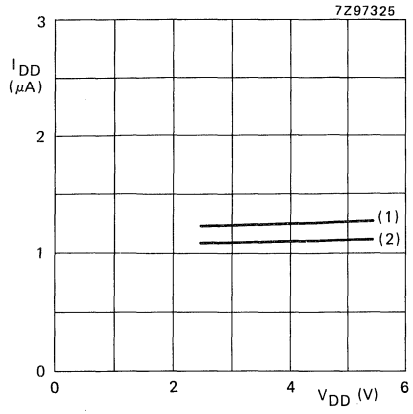
Fig. 33 Output source current HIGH ($-I_{OH}$) as a function of supply voltage (V_{DD}).



(1) V_{DD} = 3 V
(2) V_{DD} = 5 V

Fig. 34 Typical supply current during operating mode as a function of frequency.

A.C. CHARACTERISTICS (continued)



(1) T_{amb} = 85 °C
(2) T_{amb} = 25 °C

Fig. 35 Typical supply current (I_{DD}) in STOP mode as a function of the supply voltage (V_{DD}).

Table 8 Input timing shown in figure 36.

symbol	timing
t_{BUF}	$\geqslant 14t_{XTAL}$
$t_{HD;STA}$	$\geqslant 14t_{XTAL}$
t_{HIGH}	$\geqslant 17t_{XTAL}$
t_{LOW}	$\geqslant 17t_{XTAL}$
$t_{SU;STO}$	$\geqslant 14t_{XTAL}$
$t_{HD;DAT}$	> 0
$t_{SU;DAT}$	$\geqslant 250$ ns
t_{RD}	$\leqslant 1$ μs
t_{RC}	$\leqslant 1$ μs
t_{FD}	$\leqslant 1$ μs
t_{FC}	$\leqslant 0{,}3$ μs

Notes to Table 8

t_{XTAL} = one period of the XTAL input frequency (f_{XTAL})
= 167 ns for f_{XTAL} = 6 MHz.
These figures apply to all modes.

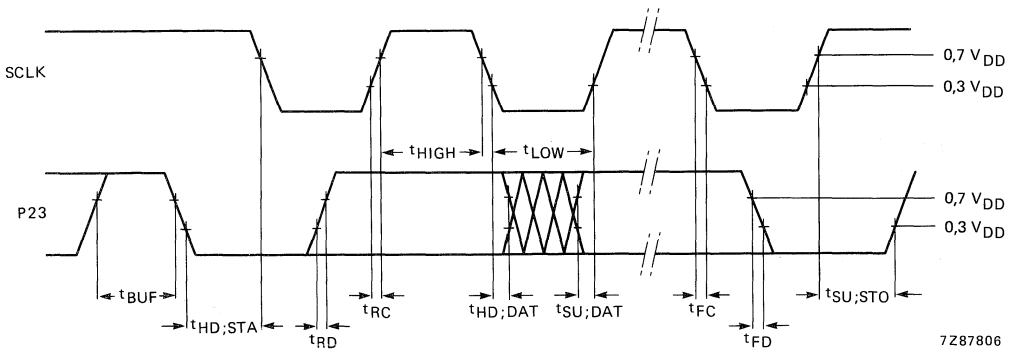


Fig. 36 PCF84C85 timing requirements for the P23 and SCLK *input* signals.

Purchase of Philips' I²C components conveys a license under the Philips' I²C patent to use the components in the I²C-system provided the system conforms to the I²C specifications defined by Philips.

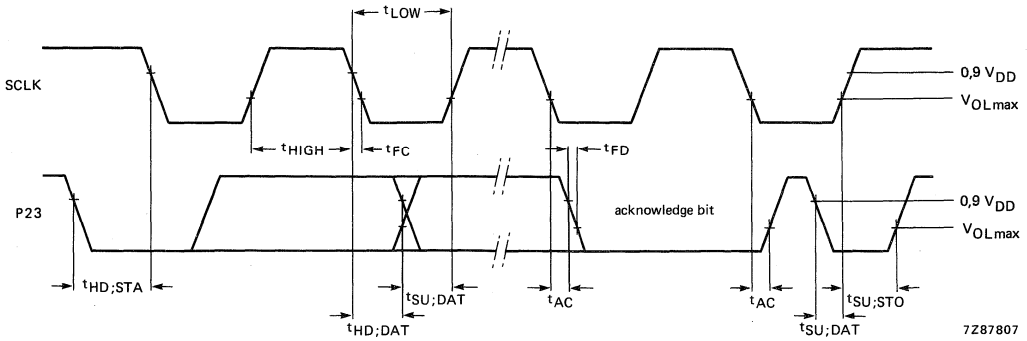


Fig. 37 PCF84C85 timing requirements for the P23 and SCLK *output* signals.

Table 9 Output timing shown in Figure 37

symbol	timing	
	normal mode (ASC in S2 = 0)	low-speed mode (ASC in S2 = 1)
$t_{HD;STA}$	½ (DF + 9) t_{XTAL}	¾ (DF + 9) t_{XTAL}
t_{HIGH}	½ (DF) t_{XTAL}	¾ (DF) t_{XTAL}
t_{LOW}	½ (DF) t_{XTAL}	¼ (DF) t_{XTAL}
$t_{SU;STO}$	½ (DF − 3) t_{XTAL}	¼ (DF − 3) t_{XTAL}
$t_{HD;DAT}$ (slave transmitter)		
any DF	$\geqslant 9t_{XTAL}$ $\leqslant 12t_{XTAL}$	$\geqslant 9t_{XTAL}$ $\leqslant 12t_{XTAL}$
$t_{HD;DAT}$ (master transmitter)		
for DF $\leqslant$ 51	$\geqslant 9t_{XTAL}$ $\leqslant 12t_{XTAL}$	– –
for DF $\leqslant$ 99	– –	$\geqslant 9t_{XTAL}$ $\leqslant 12t_{XTAL}$
$t_{SU;DAT}$ (master transmitter)		
for DF > 51	$\geqslant 15t_{XTAL}$ $\leqslant 24t_{XTAL}$	– –
for DF > 99	– –	$\geqslant 15t_{XTAL}$ $\leqslant 24t_{XTAL}$
t_{AC}	$\geqslant 9t_{XTAL}$ $\leqslant 12t_{XTAL}$	$\geqslant 9t_{XTAL}$ $\leqslant 12t_{XTAL}$
t_{FD}, t_{FC}	$\leqslant$ 100 ns at C_b = 400 pF	$\leqslant$ 100 ns at C_b = 400 pF

Notes to Table 9

t_{XTAL} = one period of the XTAL input frequency (f_{XTAL})
= 167 ns for f_{XTAL} = 6 MHz.
DF = divisor (see Table 2 Serial I/O section).
C_b = the maximum bus capacitance for each line.

DEVELOPMENT DATA

SINGLE-CHIP 8-BIT MICROCONTROLLER WITH 8 BYTES EEPROM

DESCRIPTION

An advanced CMOS process is used to manufacture the PCF84C121. The PCF84C121 has 13 quasi-bidirectional I/O port lines, three single-level vectored interrupts (one external and two timer), 8 bytes of EEPROM, two 8-bit timer counters and on-chip clock oscillator and clock circuits.

This efficient microcontroller also performs well as an arithmetic processor. The PCF84C121 is pin- and instruction set compatible with the PCF84C12. The PCF84C121 has bit handling abilities and facilities for both binary and BCD arithmetic.

This microcontroller is a member of the 84CXXX family. For detailed information, consult the 84CXXX family specification.

Features

- 8-bit CPU, ROM, RAM, EEPROM, I/O in a single 20-lead DIL or SO package
- 1 K x 8 ROM
- 64 x 8 RAM
- 8 x 8 EEPROM, designed for 10000 erase/write cycles per byte minimum
- 2 timers (8-bit programmable)
- 13 quasi-bidirectional I/O port lines
- 3 single-level, vectored interrupts: external, Timer 1 and Timer 2
- Two test inputs: one of which is also the external interrupt input
- 8-bit programmable timer/event counter
- Clock frequency range: 450 kHz to 10 MHz
- Over 80 instructions (similar to those of the MAB8048) all of 1 or 2 cycles
- Single supply voltage
- STOP and IDLE modes
- Power-on-reset circuit
- Operating temperature range: −40 to +85 °C; 0 to +55 °C for programming

PACKAGE OUTLINES

PCF84C121P: 20-lead DIL; plastic (SOT146).
PCF84C121T: 20-lead mini-pack; plastic (SO20; SOT163A).

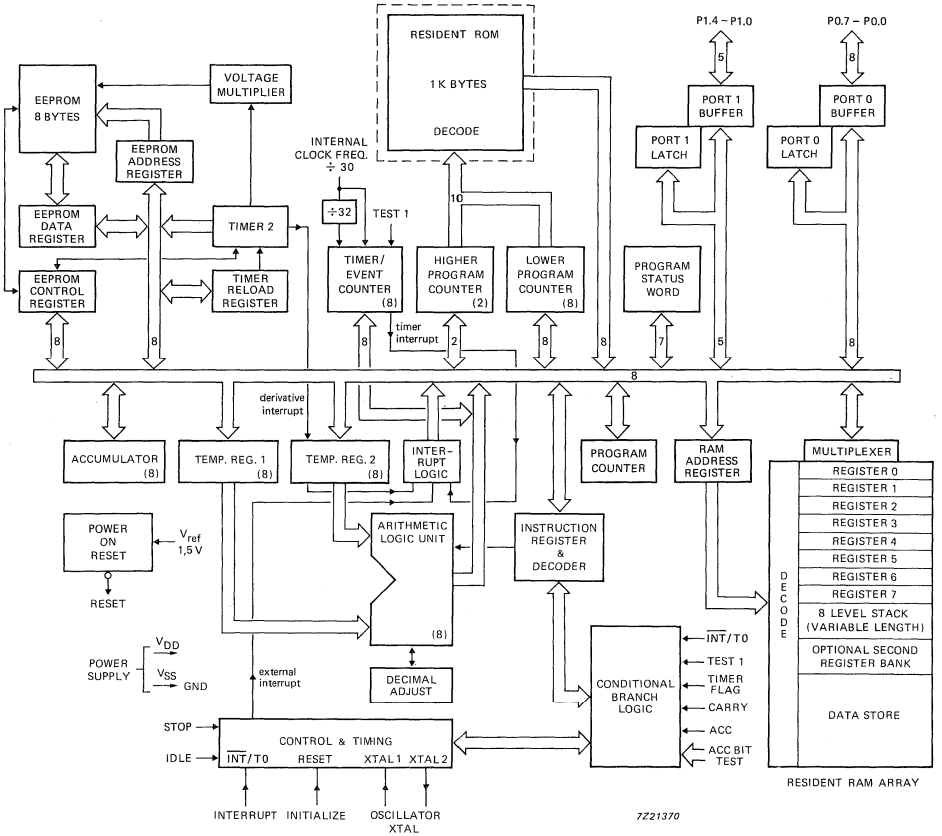


Fig. 1 Block diagram.

PINNING

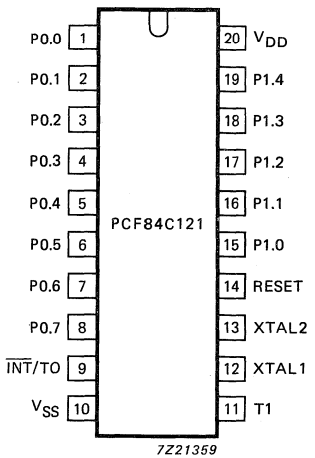


Fig. 2 Pinning diagram.

DEVELOPMENT DATA

Pin function

Pin	*Symbol*	*Function*
1-8	P0.0-P0.7	**Port 0:** 8-bit quasi-bidirectional I/O port.
9	$\overline{INT}$/T0	**Interrupt/Test 0:** external interrupt input (negative edge triggered)/test input pin; when used as a test input, this pin is directly tested by conditional branch instructions JT0 and JNT0.
10	V_{SS}	**Ground:** circuit earth potential.
11	T1	**Test 1:** test input pin, directly tested by conditional branch instructions JT1 and JNT1. T1 may also be selected as an input to the 8-bit timer/event counter via the STRT CNT instruction.
12	XTAL 1	**Oscillator input:** input from a crystal which determines the internal oscillator frequency or an external clock generator.
13	XTAL 2	**Oscillator output**
14	RESET	**Reset input:** used to initialize the microcontroller (active HIGH); also output of power-on-reset circuit.
15-19	P1.0-P1.4	**Port 1:** 5-bit quasi-bidirectional parallel I/O port.
20	V_{DD}	**Power supply:** 2.5 to 5.5 V

FUNCTIONAL DESCRIPTION

Program memory

The program memory consists of 1 K bytes of read-only memory (ROM). Each location is directly addressable by the program counter. The ROM is mask-programmed at the factory.

Data memory

The data memory consists of 64 bytes of random-access data memory (RAM). All locations are indirectly addressable using RAM pointer registers; up to 16 designated locations are directly addressable. Memory also includes an 8-level program counter stack addressed by a 3-bit stack pointer.

Derivative registers

The PCF84C121 contains 5 derivative registers (see Table 1). These registers are described below.

Table 1 Derivative registers

name	derivative address	function
ADDR	01H	EEPROM array address (0. . . 7)
DATR	03H	EEPROM read or write data
EPCR	04H	EEPROM control register (status)
RELR	05H	Timer 2 reload register
T2	06H	Timer 2 register

	MSB							LSB
ADDR (01H)	–	–	–	–	–	ADDR.2	ADDR.1	ADDR.0

Bits ADDR.7 to ADDR.3 are not used. ADDR.2, ADDR.1 and ADDR.0 are used to address one of the 8 EEPROM bytes. ADDR.2 is the most significant bit of the three bit EEPROM address.

DATR (03H)

The EEPROM data register (DATR) contains the last byte which was transferred to or from the EEPROM. When an EEPROM byte is read, it is transferred to the DATR register and then to the accumulator. The byte remains in DATR until the EEPROM is accessed again.

I/O facilities

The PCF84C121 has 13 I/O lines arranged as:

- Port 0 — 8-bit parallel port (P0.0 to P0.7)
- Port 1 — 5-bit parallel port (P1.0 to P1.4)
- $\overline{INT}$/T0 — external interrupt and test input. When used as a test input, it can be directly tested by conditional branch instructions JT0 and JNT0
- T1 — test input which can alter program sequences when tested by conditional jump instructions JT1 and JNT1. T1 also functions as an input to the 8-bit timer/event counter.

Timer 1

The maximum rate at which Timer 1 can be incremented is once every machine cycle. The timer flag is set when the counter overflows and can be tested and reset using the JTF (jump if timer flag is set) and JNTF (jump if timer flag is not set) instructions. An overflow also generates an interrupt by setting the Timer Interrupt Flag (TIF) when the timer/event counter interrupt is enabled.

Timer 2

Timer 2 may be used to program the EEPROM or as a general purpose timer. It is started by setting the STT2 bit in the EPCR register. When it is started, Timer 2 is loaded with the contents of register RELR. When an overflow occurs, an interrupt flag is set and Timer 2 is automatically reloaded. Timer 2 can be read 'on the fly' by accessing the T2 derivative register.

Timer 2 may be used as a second timer as follows.

MOV A, #time	load A with the time to be down-counted
MOV RELR, A	load derivative register RELR
EN SI	enable special interrupt
MOV A, #STT2 + EIT2	start timer and enable timer flag T2
ORL EPCR, A	load derivative control register with status

When a timer overflow occurs, a CALL to ROM address 5 is generated. The contents of the timer can be read 'on the fly' by a MOV A, T2 instruttion. Execution of this instruction takes two machine cycles and it is possible for the timer to be incremented while this instruction is being executed.

DEVELOPMENT DATA

Reset (pin 14)

A positive-going signal on the RESET input:

- Sets the program counter to zero
- Selects register bank 0
- Sets the stack pointer to zero (000 points to RAM address 8)
- Disables the interrupts (external, Timer 1 and Timer 2)
- Stops the timer/event counter, then sets it to zero
- Sets the Timer 1 prescaler to modulo-32
- Resets the timer flags
- Sets all ports to input mode
- Cancels IDLE and STOP mode
- Clears all derivative registers and the Timer 2 prescaler

Programming the EEPROM

All operations on the EEPROM are performed via the EEPROM Control Register (EPCR). EPCR is a derivative register which contains the status of the EEPROM and Timer 2.

EEPROM Control Register EPCR

	7	6	5	4	3	2	1	0
EPCR	STT2	EIT2	TF2	EWP	MC3	MC2	MC1	MC0

The EPCR bits are described below:

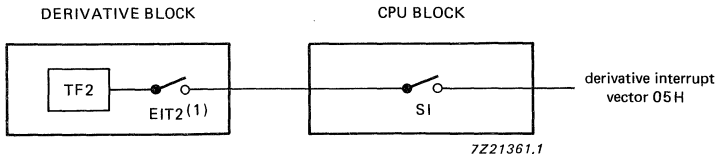
STT2, Start Timer 2

Setting this bit clears the prescaler and initiates Timer 2 as a general purpose timer without affecting the EEPROM. Initially, Timer 2 is loaded with the contents of RELR and it is subsequently reloaded every time a timer overflow occurs. Resetting STT2 stops the timer, reloads it and clears the prescaler.

FUNCTIONAL DESCRIPTION (continued)

EIT2, Enable T2 Interrupt flag

When EIT2 is set and the special interrupt (SI) is enabled, the special interrupt will be requested when TF2 is set. If EIT2 is not set, no interrupt will be requested when TF2 is set (see Fig. 3).



(1) EIT2 SET: Hardware
RESET: Software

Fig. 3 Organization of the special interrupt (SI).

TF2, Timer 2 Flag

This bit is set by hardware at the end of an erase/write or bulk erase cycle and when Timer 2 overflows while configured as a general purpose timer. A special interrupt is requested if enabled. TF2 must be cleared by software.

EWP, Erase/write cycle in progress flag

This bit must be set by user software before an erase/write or bulk erase cycle can commence. It is cleared by hardware when the erase/write or bulk erase cycle is complete.

Mode bits, MC0 to MC3

The mode bits and EWP determine the EEPROM operating mode as shown in Table 2.

Table 2 EEPROM operating modes.

mode	EWP	MC3	MC2	MC1	MC0	time (ms)
byte READ	0	0	0	0	0	–
byte ERASE/WRITE	1	1	0	0	1	20
bulk ERASE	1	1	0	1	0	10
byte WRITE	1	0	1	1	0	10

Other bit combinations have no effect on the EEPROM of the PCF84C121. MC0 to MC3 can be read and written by software and are automatically cleared by hardware when an erase/write or bulk erase cycle is complete.

The erase/write sequence

Before an erase/write sequence can commence, the RELR register must be loaded with the correct erase/write time (see Table 3). An erase/write sequence can be started if no erase or write sequence is in progress (EWP = logic 0). Six instructions are required to perform an erase/write sequence. The following performs a write operation:

```
MOV A,#data          load accumulator with EEPROM byte address
MOV ADDR, A          load ADDR with EEPROM byte address
MOV A, Rr            load accumulator with byte to be stored
MOV DATR, A          write byte to EEPROM
MOV A, #'0X011001'   erase/write cycle mode
MOV EPCR, A          commence erase/write cycle
```

When the microcontroller enters the IDLE mode, an erase/write cycle in progress will be completed. The interrupt request at the end of the erase/write cycle will cause the microcontroller to exit the IDLE mode if the interrupt is enabled. After the erase/write cycle has been completed, the EPCR register must be cleared by software.

Before entering the STOP mode, bit EWP in register EPCR should be tested. If an EEPROM write or erase cycle is in progress, this bit will be set and the STOP mode must not be entered.

The bulk erase sequence

Before a bulk erase sequence can commence, the RELR register must be loaded with the correct erase time (see Table 3). An erase or write sequence must not already be in progress (EWP = logic 0). The following clears all EEPROM bytes:

```
MOV A, #'0X011010'    bulk erase mode
MOV EPCR, A           commence bulk erase cycle
```

The read sequence

The read mode is automatically entered when an erase/write or bulk erase cycle is complete or after a RESET. The following is an example of a read operation:

```
MOV A, address    load accumulator with EEPROM address to be read
MOV ADDR, A       send address to ADDR derivative register
MOV A, DATR       load accumulator with EEPROM
MOV Rr, A         store data into register
```

The next byte may be read by repeating the above. An EEPROM byte can not be read while an erase/write or bulk erase operation is in progress.

Adjusting the erase/write time

Timer 2 determines the erase and write times during an erase/write cycle. Both times are equal and are of constant duration. Since the oscillator frequency may vary for different applications, a programmable timer and reload register (RELR) is required.

Timer 2 is clocked via a prescaler whose input is f_{osc} /30. RELR is a derivative register and can be read or written. When an erase/write cycle starts, Timer 2 is loaded with the contents of RELR and started. When the first underflow occurs, the erase process is complete, Timer 2 is reloaded and the write process commences. When the second overflow occurs, the write process is complete, the timer is halted and an interrupt flag is set. The 8-bit timer and 4-bit prescaler can generate the correct erase and write durations for an oscillator frequency range of 450 kHz to 10 MHz.

Table 3 Reload values for various oscillator frequencies

f_{XTAL}	reload value (Hex)
450 kHz	09H
1 MHz	15H
2 MHz	2AH
3.58 MHz	4BH
6 MHz	7DH
10 MHz	DϕH

Since T2 is incremented once every 480 clock cycles, the veloed value is calcilated as follows:

$$\frac{10\ ms}{480} \cdot f_{XTAL} = \text{reload value}$$

DEVELOPMENT DATA

INSTRUCTION SET

The PCF84C121 instruction set consists of over 80 one and two byte instructions. Program code efficiency is high because all RAM locations and all ROM locations on a 256 byte page require only a single byte address.

Table 6 gives the instruction set of the PCF84C121. Table 5 shows the instruction map and Table 4 details the symbols that are used.

Table 4 Symbols used in Table 6

symbol	description
A	accumulator
addr	program memory address
Bb	bit designation (b = 0 to 7)
RBS	register bank select
C	carry bit (bit CY)
CNT	event counter
Dx	derivative register (x = 0 to 5)
data	8-bit number of expression
I	interrupt
P	mnemonic for 'in-page' operation
PC	program counter
Pp	port designation (p = 0 or 1)
PSW	program status word
RB	register bank
Rr	register designation (r = 0 to 7)
SP	stack pointer
T	Timer 1
TF	Timer 1 flag
T0, T1	test 0 and 1 inputs
#	immediate data prefix
@	indirect address prefix
(X)	contents of X
((X))	contents of location addressed by X
←	is replaced by
↔	is exchanged with
STT2	start Timer 2
TF2	Timer 2 flag
EIT2	enable Timer 2 interrupt flag
EWP	erase/write cycle in progress
EPCR	EEPROM control register
SI	special interrupt (Timer 2)

Table 5 PCF84C121 instruction map

first hexadecimal character of opcode	second hexadecimal character of opcode															
	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
0	NOP	IDLE		ADD A, # data	JMP page 0	EN I	JNTF addr	DEC A	IN A,P$_p$ 0	1						
1	INC ∂Rr 0	1	JB0 addr	ADDC A, # data	CALL page 0	DIS I	JTF addr	INC A	0	1	2	3	INC Rr 4	5	6	7
2	XCH A,∂Rr 0	1	STOP	MOV A, # data	JMP page 1	EN TCNTI	JNT0 addr	CLR A	0	1	2	3	XCH A,Rr 4	5	6	7
3	XCHD A,∂Rr 0	1	JB1 addr		CALL page 1	DIS TCNTI	JT0 addr	CPL A	OUTL P$_p$,A 0	1						
4	ORL A,∂Rr 0	1	MOV A, T	ORL A, # data	JMP page 2	STRT CNT	JNT1 addr	SWAP A	0	1	2	3	ORL A,Rr 4	5	6	7
5	ANL A,∂Rr 0	1	JB2 addr	ANL A, # data	CALL page 2	STRT T	JT1 addr	DA A	0	1	2	3	ANL A,Rr 4	5	6	7
6	ADD A,∂Rr 0	1	MOV T, A		JMP page 3	STOP TCNT		RRC A	0	1	2	3	ADD A,Rr 4	5	6	7
7	ADDC A,∂Rr 0	1	JB3 addr		CALL page 3			RR A	0	1	2	3	ADDC A,Rr 4	5	6	7
8				RET	JMP page 4	EN SI			ORL P$_p$, #data 0	1			MOV A, Dx	MOV Dx, A	ANL Dx, A	ORL Dx, A
9			JB4 addr	RETR	CALL page 4	DIS SI	JNZ addr	CLR C	ANL P$_p$, #data 0	1						
A	MOV ∂Rr,A 0	1		MOVP A, ∂A	JMP page 5			CPL C	0	1	2	3	MOV Rr,A 4	5	6	7
B	MOV ∂Rr, # data 0	1	JB5 addr	JMPP ∂A	CALL page 5				0	1	2	3	MOV Rr, # data 4	5	6	7
C	DEC ∂Rr 0	1			JMP page 6	SEL RB0	JZ addr	MOV A,PSW	0	1	2	3	DEC Rr 4	5	6	7
D	XRL A,∂Rr 0	1	JB6 addr	XRL A, # data	CALL page 6	SEL RB1		MOV PSW,A	0	1	2	3	XRL A,Rr 4	5	6	7
E	DJNZ ∂Rr, addr 0	1			JMP page 7		JNC addr	RL A	0	1	2	3	DJNZ Rr, addr 4	5	6	7
F	MOV A, ∂Rr 0	1	JB7 addr		CALL page 7		JC addr	RLC A	0	1	2	3	MOV A, Rr 4	5	6	7

INSTRUCTION SET (continued)

Table 6 Instruction set

	mnemonic	opcode (hex.)	bytes/ cycles	description	function		notes
ACCUMULATOR	ADD A, Rr	6*	1/1	Add register contents to A	(A)←(A) + (Rr)	r = 0–7	1
	ADD A, @Rr	60 61	1/1	Add RAM data, addressed by Rr, to A	(A)←(A) + ((R0)) (A)←(A) + ((R1))		1
	ADD A, #data	03 data	2/2	Add immediate data to A	(A)←(A) + data		1
	ADDC A, Rr	7*	1/1	Add carry and register contents to A	(A)←(A) + (Rr) + (C)	r = 0–7	1
	ADDC A, @Rr	70 71	1/1	Add carry and RAM data, addressed by Rr, to A	(A)←(A) + ((R0)) + (C) (A)←(A) + ((R1)) + (C)		1
	ADDC A, #data	13 data	2/2	Add carry and immediate data to A	(A)←(A) + data + (C)		1
	ANL A, Rr	5*	1/1	'AND' Rr with A	(A)←(A) AND (Rr)	r = 0–7	
	ANL A, @Rr	50 51	1/1	'AND' RAM data, addressed by Rr, with A	(A)←(A) AND ((R0)) (A)←(A) AND ((R1))		
	ANL A, #data	53 data	2/2	'AND' immediate data with A	(A)←(A) AND data		
	ORL A, Rr	4*	1/1	'OR' Rr with A	(A)←(A) OR (Rr)	r = 0–7	
	ORL A, @Rr	40 41	1/1	'OR' RAM data, addressed by Rr, with A	(A)←(A) OR ((R0)) (A)←(A) OR ((R1))		
	ORL A, #data	43 data	2/2	'OR' immediate data with A	(A)←(A) OR data		
	XRL A, Rr	D*	1/1	'XOR' Rr with A	(A)←(A) XOR (Rr)	r = 0–7	
	XRL A, @Rr	D0 D1	1/1	'XOR' RAM, addressed by Rr, with A	(A)←(A) XOR ((R0)) (A)←(A) XOR ((R1))		
	XRL A, #data	D3 data	2/2	'XOR' immediate data with A	(A)←(A) XOR data		
	INC A	17	1/1	increment A by 1	(A)←(A) + 1		
	DEC A	07	1/1	decrement A by 1	(A)←(A) −1		
	CLR A	27	1/1	clear A to zero	(A)←0		
	CPL A	37	1/1	one's complement A	(A)←NOT(A)		
	RL A	E7	1/1	rotate A left	$(A_{n+1})\leftarrow(A_n)$ $(A_0)\leftarrow(A_7)$	n = 0–6	

DEVELOPMENT DATA

ACCUMULATOR (cont.)	RLC A	F7	1/1	rotate A left through carry	$(A_{n+1})\leftarrow A_n$ $(A_0)\leftarrow(C), (C)\leftarrow(A_7)$	n = 0–6	2
	RR A	77	1/1	rotate A right	$(A_n)\leftarrow(A_{n+1})$ $(A_7)\leftarrow(A_0)$	n = 0–6	
	RRC A	67	1/1	rotate A right through carry	$(A_n)\leftarrow(A_{n+1})$ $(A_7)\leftarrow(C), (C)\leftarrow(A_0)$	n = 0–6	2
	DA A	57	1/1	decimal adjust A			2
	SWAP A	47	1/1	swap nibbles of A	$(A_{4-7})\leftrightarrow(A_{0-3})$		
DATA MOVES	MOV A, Rr	F*	1/1	move register contents to A	(A)←(Rr)	r = 0–7	
	MOV A, @Rr	F0 F1	1/1	move RAM data, addressed by Rr, to A	(A)←((R0)) (A)←((R1))		
	MOV A, #data	23 data	2/2	move immediate data to A	(A)←data		
	MOV Rr, A	A*	1/1	move accumulator contents to register	(Rr)←(A)	r = 0–7	
	MOV @Rr, A	A0 A1	1/1	move accumulator contents to RAM location addressed by Rr	((R0))←(A) ((R1))←(A)		
	MOV Rr, #data	B* data	2/2	move immediate data to Rr	(Rr)←data		
	MOV @Rr, #data	B0 data B1 data	2/2	move immediate data to RAM location addressed by Rr	((R0))←data ((R1))←data		
	XCH A, Rr	2*	1/1	exchange accumulator contents with Rr	$(A)\leftrightarrow(Rr)$	r = 0–7	
	XCH A, @Rr	20 21	1/1	exchange accumulator contents with RAM data addressed by Rr	$(A)\leftrightarrow((R0))$ $(A)\leftrightarrow((R1))$		
	XCHD A, @Rr	30 31	1/1	exchange lower nibbles of A and RAM data addressed by Rr	$(A_{0-3})\leftrightarrow((R0_{0-3}))$ $(A_{0-3})\leftrightarrow((R1_{0-3}))$		
	MOV A, PSW	C7	1/1	move PSW contents to accumulator	(A)←(PSW)		
	MOV PSW, A	D7	1/1	move accumulator bit 3 to PSW_3	$(PSW_3)\leftarrow(A_3)$		3
	MOVP A, @A	A3	1/2	move indirectly addressed data in current page to A	$(PC_{0-7})\leftarrow(A), (A)\leftarrow((PC))$		
FLAGS	CLR C	97	1/1	clear carry bit	(C)←0		2
	CPL C	A7	1/1	complement carry bit	(C)←NOT(C)		2

INSTRUCTION SET (continued)

	mnemonic	opcode (hex.)	bytes/ cycles	description	function	notes
REGISTER	INC Rr	1*	1/1	increment register by 1	(Rr)←(Rr) + 1 r = 0–7	
	INC @Rr	10 11	1/1	increment RAM data, addressed by Rr, by 1	((R0))←((R0)) + 1 ((R1))←((R1)) + 1	
	DEC Rr	C*	1/1	decrement register by 1	(Rr)←(Rr) – 1 r = 0–7	
	DEC @Rr	C0 C1	1/1	decrement RAM data, addressed by Rr, by 1	((R0))←((R0)) – 1 ((R1))←((R1)) – 1	
BRANCH	JMP addr	• 4 address	2/2	unconditional jump within a 2 K bank	(PC_{8-10})←$addr_{8-10}$ (PC_{0-7})←$addr_{0-7}$	
	JMPP @A	B3	1/2	indirect jump within a page	(PC_{0-7})←((A))	
	DJNZ Rr, addr	E* address	2/2	decrement Rr by 1 and jump if not zero to addr	(Rr)←(Rr) – 1 r = 0–7 if (Rr) not zero (PC_{0-7})←addr	
	DJNZ @Rr, addr	E0	2/2	decrement RAM data, addressed by Rr by 1 and jump if not zero to addr	((R0))←((R0)) – 1 if ((R0)) not zero (PC_{0-7})←addr	
		E1			((R1))←((R1)) – 1 if ((R1)) not zero (PC_{0-7})←addr	
	JBb addr	▲ 2 address	2/2	jump to addr if Acc. bit b = 1	if b = 1 : (PC_{0-7})←addr b = 0–7	
	JC addr	F6 address	2/2	jump to addr if C = 1	if C = 1 : (PC_{0-7})←addr	
	JNC addr	E6 address	2/2	jump to addr if C = 0	if C = 0 : (PC_{0-7})←addr	
	JZ addr	C6 address	2/2	jump to addr if A = 0	if A = 0 : (PC_{0-7})←addr	
	JNZ addr	96 address	2/2	jump to addr if A is NOT zero	if A≠0 : (PC_{0-7})←addr	
	JT0 addr	36 address	2/2	jump to addr if T0 = 1	if T0 = 1: (PC_{0-7})←addr	
	JNT0 addr	26 address	2/2	jump to addr if T0 = 0	if T0 = 0: (PC_{0-7})←addr	
	JT1 addr	56 address	2/2	jump to addr if T1 = 1	if T1 = 1: (PC_{0-7})←addr	
	JNT1 addr	46 address	2/2	jump to addr if T1 = 0	if T1 = 0: (PC_{0-7})←addr	
	JTF addr	16 address	2/2	jump to addr if Timer Flag = 1	if TF = 1: (PC_{0-7})←addr	4
	JNTF addr	06 address	2/2	jump to addr if Timer Flag = 0	if TF = 0: (PC_{0-7})←addr	

TIMER/EVENT COUNTER	MOV A, T	42	1/1	move timer/event counter contents to accumulator	(A)←(T)	
	MOV T, A	62	1/1	move accumulator contents to timer/event counter	(T)←(A)	
	STRT CNT	45	1/1	start event counter		
	STRT T	55	1/1	start timer		
	STOP TCNT	65	1/1	stop timer/event counter		
	EN TCNTI	25	1/1	enable timer/event counter interrupt		
	DIS TCNTI	35	1/1	disable timer/event counter interrupt		
CONTROL	EN I	05	1/1	enable external interrupt		
	DIS I	15	1/1	disable external interrupt		
	SEL RB0	C5	1/1	select register bank 0	(RBS)←0	5
	SEL RB1	D5	1/1	select register bank 1	(RBS)←1	5
	STOP	22	1/1	enter STOP mode		
	IDLE	01	1/1	enter IDLE mode		
SUBROUTINE	CALL addr	▲ 4 address	2/2	jump to subroutine	((SP))←(PC), ($PSW_{4, 6, 7}$) 6 (SP)←(SP) + 1 (PC_{8-10})←$addr_{8-10}$ (PC_{0-7})←$addr_{0-7}$	6
	RET	83	1/2	return from subroutine	(SP)←(SP) − 1 (PC)←((SP))	6
	RETR	93	1/2	return from interrupt and restore bits 4, 6, 7 of PSW	(SP)←(SP) − 1 ($PSW_{4, 6, 7}$) + (PC)←((SP))	6

INSTRUCTION SET (continued)

	mnemonic	opcode (hex.)	bytes/ cycles	description	function		notes
PARALLEL INPUT/OUTPUT	IN A, Pp	08 09	1/2	input port p data to accumulator	(A)←(P0) (A)←(P1)		
	OUTL Pp, A	38 39	1/2	output accumulator data to port p	(P0)←(A) (P1)←(A)		
	ANL Pp, #data	98 99	2/2	AND port p data with immediate data	(P0)←(P0) AND data (P1)←(P1) AND data		
	ORL Pp, #data	88 89	2/2	OR port p data with immediate data	(P0)←(P0) OR data (P1)←(P1) OR data		
	NOP	00	1/1	no operation			
DERIVATIVE INPUT/OUTPUT	MOV A, Dx	8C	2/2	move derivative register contents to accumulator	(A)←(Dx)	x = 0 to 5	
	MOV Dx, A	8D	2/2	move accumulator contents to derivative register	(Dx)←(A)	x = 0 to 5	
	ANL Dx, A	8E	2/2	AND derivative register with accumulator	(Dx)←(Dx) AND (A)	x = 0 to 5	
	ORL Dx, A	8F	2/2	OR derivative register with accumulator	(Dx)←(Dx) OR (A)	x = 0 to 5	
	EN SI	85	1/1	enable special interrupt			
	DIS SI	95	1/1	disable special interrupt			

Notes to Table 6

1. PSW CY, AC affected
2. PSW CY affected
3. PSW PS affected
4. Execution of JTF and JNTF instructions resets the Timer Flag (TF).
5. PSW RBS affected
6. PSW SP_0, SP_1, SP_2 affected

* : 8, 9, A, B, C, D, E, F
● : 0, 2, 4, 6, 8, A, C, E
▲ : 1, 3, 5, 7, 9, B, D, F

RATINGS

Limiting values in accordance with the Absolute Maximum System (IEC 134)

Supply voltage (pin 20)	V_{DD}		−0.8 to + 8	V
All input voltages	V_I		0.8 to V_{DD} + 0.8	V
DC current into any input or output	$\pm I_I, \pm I_O$	max.	10	mA
Total power dissipation (see note 1)	P_{tot}	max.	125	mW
Power dissipation per output	P_O	max.	50	mW
Storage temperature range	T_{stg}		−65 to + 150	°C
Operating junction temperature	T_j	max.	125	°C

HANDLING

Inputs and outputs are protected against electrostatic discharge in normal handling. However, it is good practice to take normal precautions appropriate to handling MOS devices (see 'Handling MOS devices').

Notes to the ratings

1. Thermal resistance (junction to ambient)

for SOT146	$R_{th\ j\text{-}a}$	max.	75	K/W
for SOT163A	$R_{th\ j\text{-}a}$	max.	135	K/W

DEVELOPMENT DATA

LIFE SUPPORT APPLICATIONS

This product is not designed for use in life support appliances, devices, or systems where malfunction of this product can reasonably be expected to result in personal injury. Philips customers using or selling this product for use in such applications do so at their own risk and agree to fully indemnify Philips for any damages resulting from such improper use or sale.

DC CHARACTERISTICS

V_{DD} = 2.5 to 5.5 V; V_{SS} = 0 V; T_{amb} = −40 to + 85 °C; all voltages with respect to V_{SS}; unless otherwise specified

parameter	symbol	min.	typ.	max.	unit
Supply voltage					
operating (see Fig. 4)	V_{DD}	2.5	–	5,5	V
Supply current					
operating (see Fig. 5 and note 2)					
at V_{DD} = 5 V; f_{XTAL} = 10 MHz	I_{DD}	–	2.0	3.5	mA
at V_{DD} = 5 V; f_{XTAL} = 6 MHz	I_{DD}	–	1.2	2.4	mA
at V_{DD} = 3 V; f_{XTAL} = 3.58 MHz	I_{DD}	–	0.4	0.8	mA
IDLE mode (see Fig. 6 and note 2)					
at V_{DD} = 5 V; f_{XTAL} = 10 MHz	I_{DD}	–	1.0	2.0	mA
at V_{DD} = 5 V; f_{XTAL} = 6 MHz	I_{DD}	–	0.7	1.4	mA
at V_{DD} = 3 V; f_{XTAL} = 3.58 MHz	I_{DD}	–	0.25	0.6	mA
STOP mode (see Fig. 9, note 1 and note 2)					
at V_{DD} = 2.5 V; T_{amb} = 85 °C	I_{DD}	–	–	10	μA
Erase/write cycle					
time (see note 3)	$t_{e/w}$	–	10	–	ms
data retention	Dr	10	–	–	years
Inputs					
Input voltage LOW	V_{IL}	0	–	$0.3V_{DD}$	V
Input voltage HIGH	V_{IH}	$0.7V_{DD}$	–	V_{DD}	V
Input leakage current					
as $V_{SS} < V_I < V_{DD}$	$\pm I_{IL}$	–	–	1	μA
Outputs					
Output sink current LOW					
at V_{DD} = 5 V ± 10%; V_O = 0.4 V	I_{OL}	1.6	3	–	mA
Pull-up output source current HIGH					
at V_{DD} = 5 V ± 10%; V_O = 0.7 V_{DD}	$-I_{OH}$	40	–	–	μA
at V_{DD} = 5 V ± 10%; V_O = V_{SS}	$-I_{OH}$	–	–	400	μA
Push-pull output source current HIGH					
at V_{DD} = 5 V ± 10%; V_O = V_{DD}−0.4 V	$-I_{OH}$	1.6	3	–	mA

Notes to the DC characteristics

1. Crystal connected between XTAL 1 and XTAL 2; T1 = V_{SS}; $\overline{INT}$ = V_{DD}.
2. V_{IL} = V_{SS}; V_{IH} = V_{DD}; all outputs unloaded; all open-drain outputs connected to V_{SS}.
3. Select reload value (Table 3) for $t_{e/w}$ = 10 ms. No shorter nor longer time is recommended.

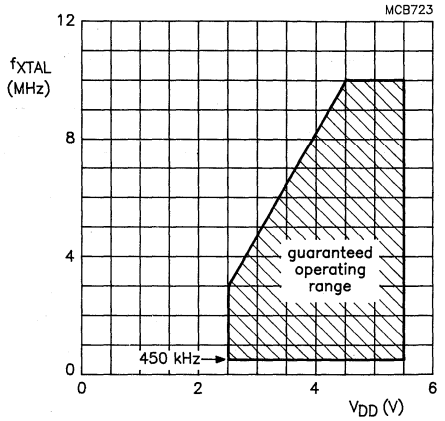
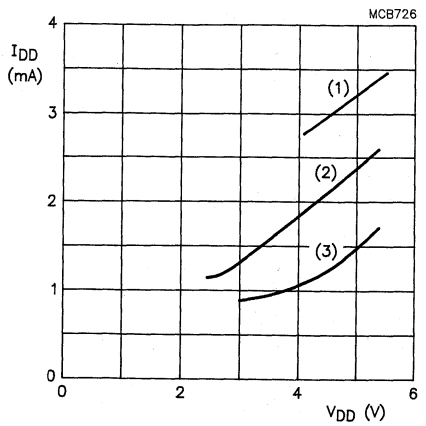


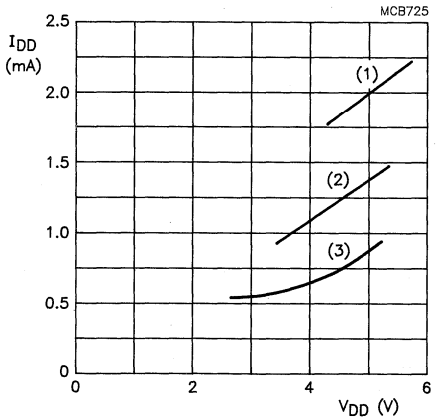
Fig. 4 Maximum clock frequency (f_{XTAL}) as a function of supply voltage (V_{DD}).



(1) f_{XTAL} = 10 MHz
(2) f_{XTAL} = 6 MHz
(3) f_{XTAL} = 3.58 MHz

Fig. 5 Maximum supply current (I_{DD}) in operating mode as a function of the supply voltage.

DEVELOPMENT DATA



(1) f_{XTAL} = 10 MHz
(2) f_{XTAL} = 6 MHz
(3) f_{XTAL} = 3.58 MHz

Fig. 6 Maximum supply current (I_{DD}) in IDLE mode as a function of supply voltage (V_{DD}).

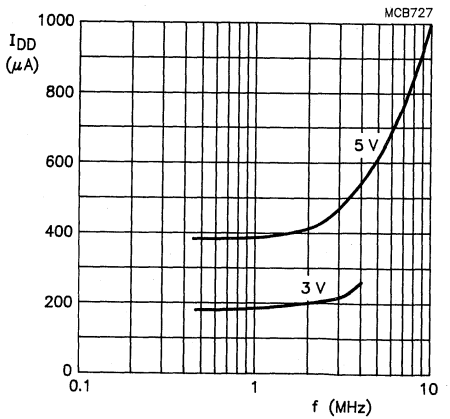


Fig. 7 Typical supply current during IDLE mode as a function of frequency at V_{DD} = 3 V and V_{DD} = 5 V.

DC CHARACTERISTICS (continued)

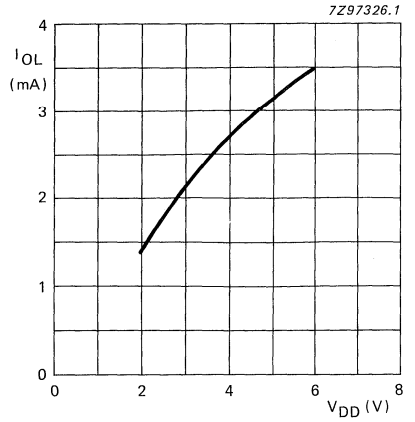
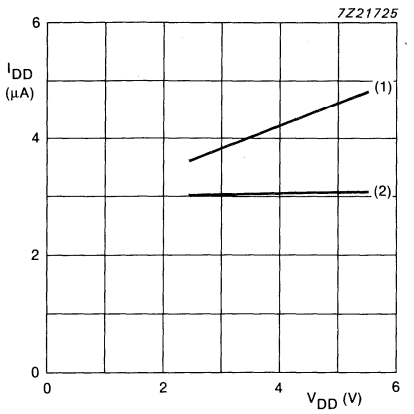
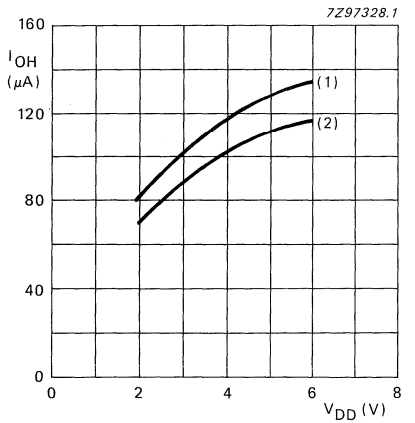


Fig. 8 Typical output sink current (I_{OL}), as a function of supply voltage (V_{DD}); V_O = 0.4 V.



(1) T_{amb} = 85 °C
(2) T_{amb} = 25 °C

Fig. 9 Typical supply current (I_{DD}) in STOP mode as a function of the supply voltage (V_{DD}).



(1) $V_O = V_{SS}$
(2) $V_O = 0.7\ V_{DD}$

Fig. 10 Typical output source current ($-I_{OH}$) as a function of supply voltage (V_{DD}).

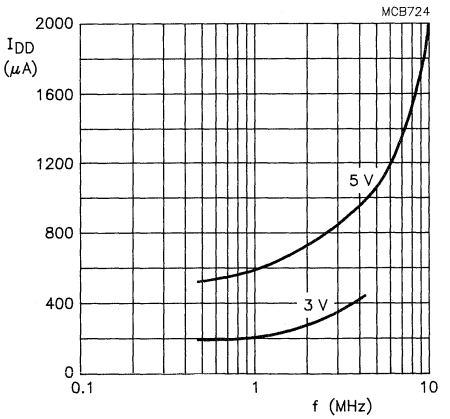


Fig. 11 Typical supply current during operating mode as a function of frequency at V_{DD} = 3 V and V_{DD} = 5 V.

SINGLE-CHIP 8-BIT MICROCONTROLLER WITH LCD DRIVER

GENERAL DESCRIPTION

The PCF84C230 is a single-chip 8-bit microcontroller manufactured in CMOS technology, and is a member of the 84CXXX family. For detailed information see the 84CXXX family specification.

The PCF84C230 provides 12 general purpose quasi-bidirectional I/O port lines, a line that is directly testable (T1), one external interrupt line, and an LCD driver for up to 64 graphic elements. The IC is mask-programmable and is designed for control in small systems with LCD displays.

Features

- 8-bit CPU, ROM, RAM, I/O in a single 40-lead DIL package
- 2 K ROM bytes
- 64 RAM bytes
- Over 80 instructions (based on MAB8048) all of 1 or 2 cycles
- 12 quasi-bidirectional I/O port lines
- Configuration of I/O lines can be individually selected by mask (pull-up, open drain, push-pull)
- LCD drive circuit with 16 segment drivers and selectable backplane drive configuration: static or 2/3/4 multiplex, to drive up to 64 graphic elements
- LCD possible during STOP mode
- Single-level vectored interrupts: external and timer/event counter
- Power-on reset and low voltage detector
- Single supply voltage from 2.5 V to 5.5 V
- STOP and IDLE modes
- Clock frequency 100 kHz to 10 MHz
- Operating ambient temperature range: −40 to + 85 °C

PACKAGE OUTLINES

40-lead DIL; plastic (SOT129).
40-lead mini-pack; plastic (VS040; SOT158A).

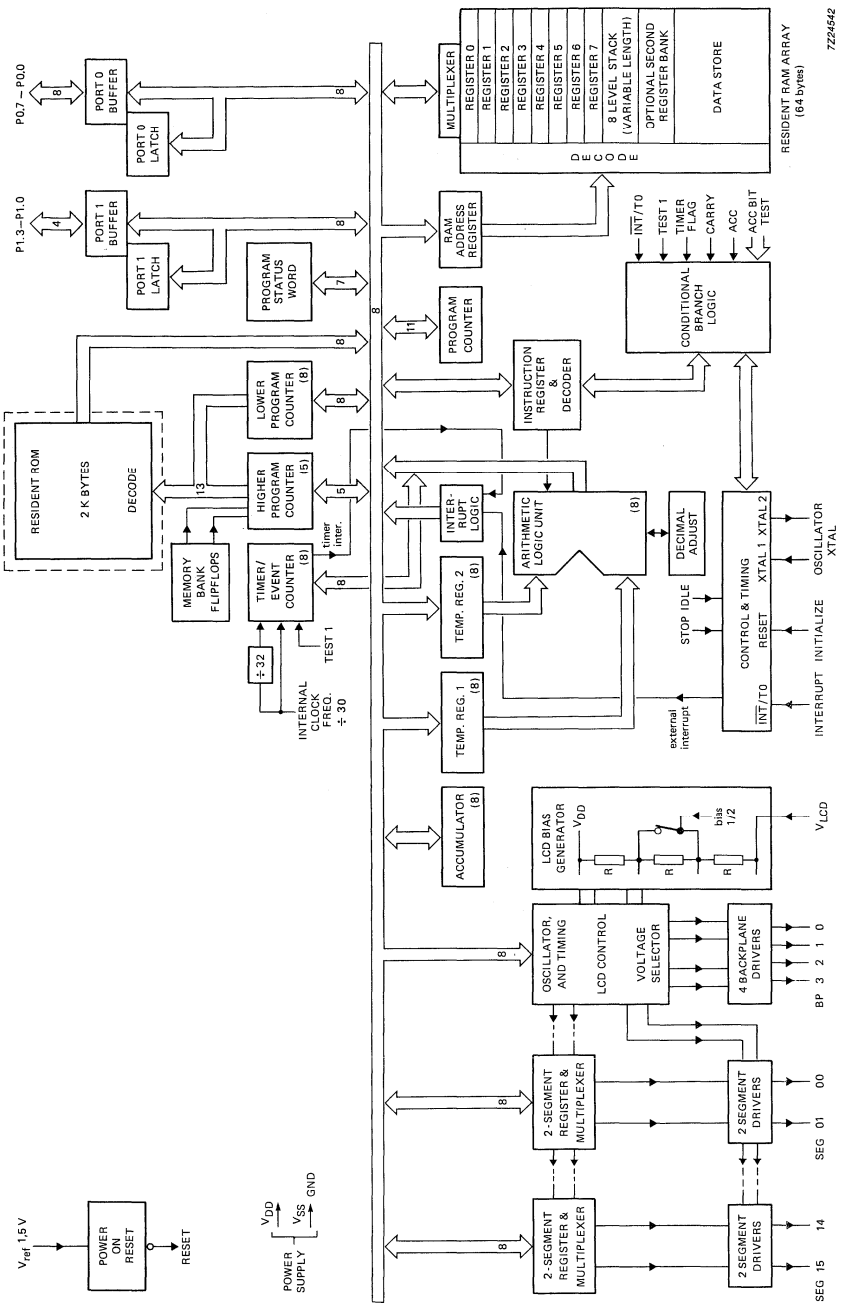


Fig.1 Block diagram.

DEVELOPMENT DATA

PINNING

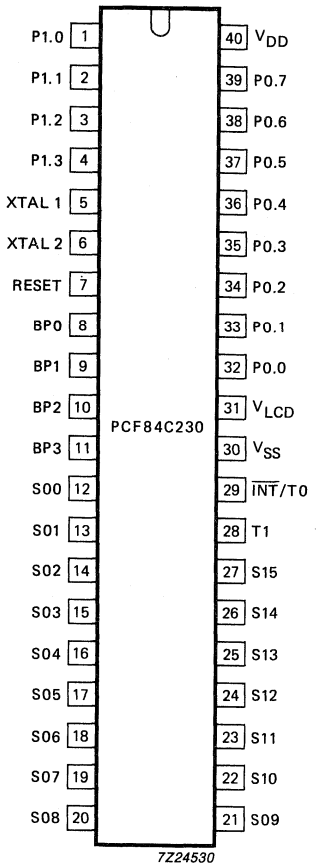


Fig.2 Pinning diagram.

Pin functions

pin no.	mnemonic	description
1 to 4	P1.0 to P1.3	**Port 1**: 4-bit quasi-bidirectional I/O port. This port can be mask programmed to either standard I/O, open drain or push-pull.
5	XTAL 1	**Crystal input**: connected to the timing component which determines the frequency of the internal oscillator; also used as an external clock source.
6	XTAL 2	Connection to the other side of the timing component.
7	RESET	**Reset**: bidirectional reset, used to initialize the processor or output of power-on reset circuit.
8 to 11	BP0 to BP3	**LCD**: backplane outputs.
12 to 27	S00 to S15	**LCD**: segment driver outputs.
28	T1	**Test 1**: test input pin, directly tested by conditional branch instructions JT1 and JNT1. T1 also functions as an input to the 8-bit timer/event counter, using the STRT CNT instruction.
29	$\overline{INT}$/T0	**Interrupt/Test 0**: external interrupt input, sensitive to negative going edge. Also shared with test input 0; when used as a test input directly tested by conditional branch instructions JT0 and JNT0.
30	V_{SS}	**Ground**: circuit earth potential.
31	V_{LCD}	**LCD supply**: LCD supply voltage.
32 to 39	P0.0 to P0.7	**Port 0**: 8-bit quasi-bidirectional I/O port. This port can be mask programmed to either standard I/O, open drain or push-pull.
40	V_{DD}	**Power supply**: 2.5 V to 5.5 V.

FUNCTIONAL DESCRIPTION

Program memory

The program memory consists of 2048 bytes (8-bit words) in a read-only memory (ROM). Each location is directly addressable by the program counter. The memory is mask-programmed at the factory. Figure 3 shows the program memory map.

Three program memory locations are of special importance:

- Location 0; first instruction to be executed after the processor is reset
- Location 3; first instruction of an external interrupt service ($\overline{INT}$/T0) subroutine
- Location 7; first instruction of a timer/event counter interrupt service subroutine

Program memory is divided into 8 'pages', each of 256 bytes. This division applies only for conditional branches. Conditional branches are performed within a page. Page boundaries can be crossed only by using the unconditional branch instructions. A CALL instruction can transfer control to a subroutine on any 'page'; RET and RETR instructions transfer control from a subroutine back to the main program.

Data memory

Data memory consists of 64 bytes of random-access memory (RAM). All locations are indirectly addressable using RAM pointer registers; up to 16 designated locations are directly addressable. Memory also includes an 8-level program counter stack addressed by a 3-bit stack pointer. Figure 4 shows the data memory map.

Working registers

Locations 0 to 7 are designated as working registers, directly addressable by the direct register instructions. Ease of addressing, and a minimum requirement of instruction bytes to manipulate their contents, makes these locations suitable for storing frequently addressed intermediate results. This bank of registers can be selected by the SEL RB0 instruction.

Executing the select register bank instruction SEL RB1, designates locations 24 to 31 as working registers, instead of locations 0 to 7, and these are then directly addressable. This second bank of working registers may be used as an extension of the first or reserved for use during interrupt service subroutines saving the first bank for use in the main program. If the second bank is not used, locations 24 to 31 may serve as general purpose RAM.

The first locations of each bank contain the RAM pointer registers R0, R1, R0' and R1', which indirectly address all RAM locations.

All RAM locations make efficient program loop counters when used with the decrement register and test instruction DJNZ.

FUNCTIONAL DESCRIPTION (continued)

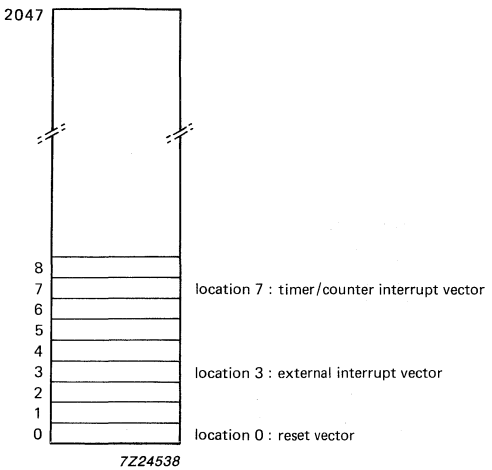


Fig.3 Program memory map.

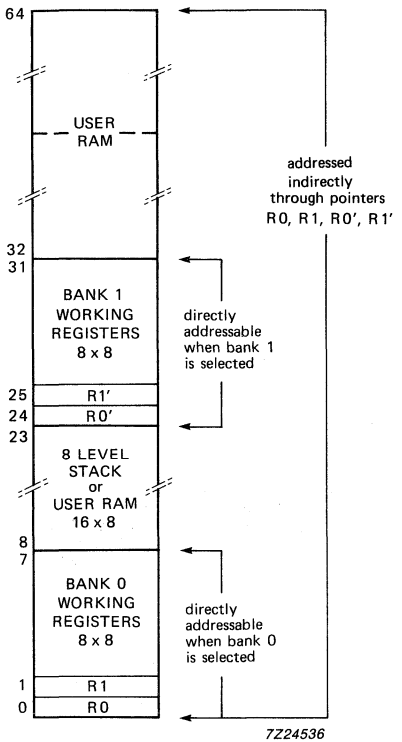


Fig.4 Data memory map.

Program counter stack

Locations 8 to 23 may be designated as an 8-level program counter stack (2 locations per level), or as general purpose RAM. The program counter stack (Fig.5) enables the processor to keep track of the return addresses and status generated by interrupts or CALL instructions by storing the contents of the program counter prior to servicing the subroutine. A 3-bit stack pointer determines which of the eight register pairs of the program counter stack will be loaded with the next generated return address.

The stack pointer, when initialized to 000 by RESET, points to RAM locations 8 and 9. On the first subroutine CALL or interrupt, the contents of the program counter and bits 4, 6 and 7 of the program status word (PSW) are transferred to locations 8 and 9. The stack pointer increments by one and points to locations 10 and 11 ready for another CALL. Because an address may be up to 13 bits long, two bytes must be used to store each address.

At the end of a subroutine, which is signalled by a return instruction (RET or RETR), the stack pointer decrements by one and the contents of the register pair on top of the stack are transferred to the program counter. The saved PSW bits are transferred to the PSW only by the RETR instruction.

If not all 8 levels of subroutine and interrupt nesting are used, the unused portion of the stack may be used as any other indirectly addressable RAM locations; for storage of program variables or data.

Nesting of subroutines within subroutines can continue up to 8 times without overflowing the stack. If overflow does occur the deepest address stored (locations 8 and 9) will be overwritten and lost since the stack pointer overflows from 111 to 000. It also underflows from 000 to 111.

The value of the saved contents of the program counter is different for an interrupt CALL compared to a normal CALL to subroutine. With an interrupt CALL, the program counter return address is saved; with a subroutine CALL, the saved program counter value is one less than the program counter return address.

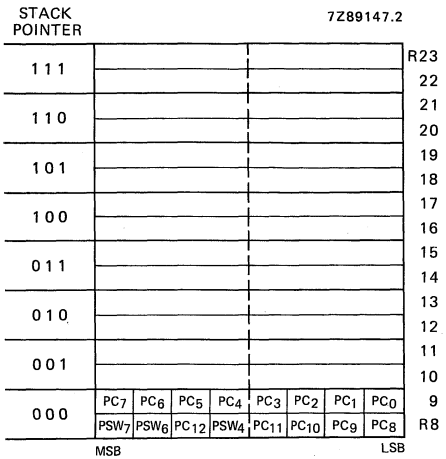


Fig.5 Program counter stack.

IDLE and STOP modes

IDLE mode

When the microcontroller enters the IDLE mode via the IDLE instruction (01 H) the oscillator, timer/counter and the LCD display are kept running.

The microcontroller exists from the IDLE mode by one of two interrupts ($\overline{INT}$/T0, TIMER/EVENT COUNTER interrupt) if they are enabled, or by activating a RESET. If the interrupt is not enabled the processor will remain in the IDLE mode. An active signal on the RESET pin restarts the microcontroller and a normal RESET sequence is executed (see Fig.6).

FUNCTIONAL DESCRIPTION (continued)

IDLE mode (continued)

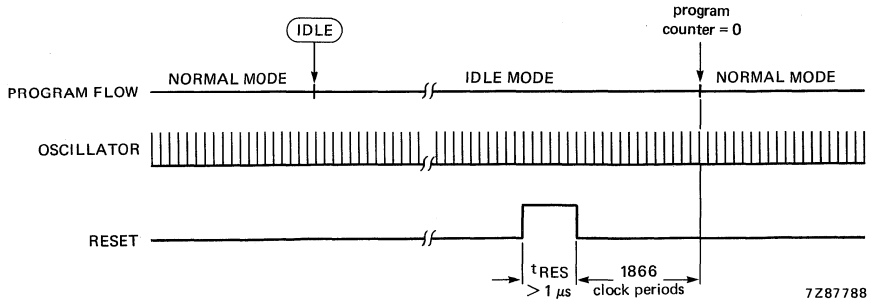


Fig.6 Exit from IDLE mode via a RESET.

An active signal coming from an enabled interrupt causes the execution of the normal interrupt routine since normal interrupt scanning is still being carried out. A HIGH-to-LOW transition on the external interrupt pin ($\overline{INT}$/T0) reactivates the microcontroller. A LOW level applied to $\overline{INT}$/T0 will reactivate the microcontroller only in the STOP mode. Thus, if $\overline{INT}$/T0 was LOW before the microcontroller entered the IDLE mode, it must go HIGH before the microcontroller can be reactivated (see Fig.7).

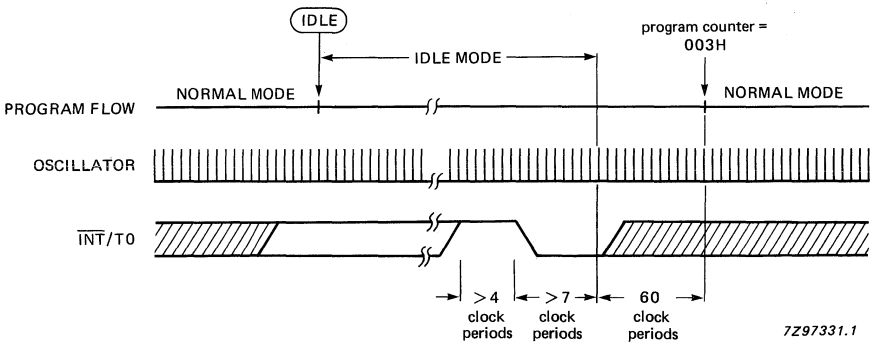


Fig.7 Exit from IDLE mode via an interrupt.

Wake-up from the IDLE mode is ensured when $\overline{INT}$/T0 is HIGH for at least 4 CP (clock periods) followed by a LOW for 7 CP. After the initial forced CALL 003 H operation (60 CP) the program continues with the external interrupt service routine. During IDLE mode operation, the address of the instruction immediately following that which caused the processor to enter the IDLE mode is present on the address bus.

STOP mode

The microcontroller enters the STOP mode via the STOP instruction (22H). The oscillator is switched off but internal status of the CPU, RAM contents and the state of I/O ports are unaffected. The microcontroller can be brought-out of the STOP mode by an active signal at the external interrupt input or by an external RESET signal. When one of these two signals is applied an internal delay of 1866 CP is provided to ensure that all internal clocks are operating correctly before restart (see Fig.8). Note: the start-up time of a crystal oscillator is measured in milliseconds, and the 1866 CP count begins after this start-up time.

If the microcontroller exits from the STOP mode by activating RESET, a normal RESET sequence is executed.

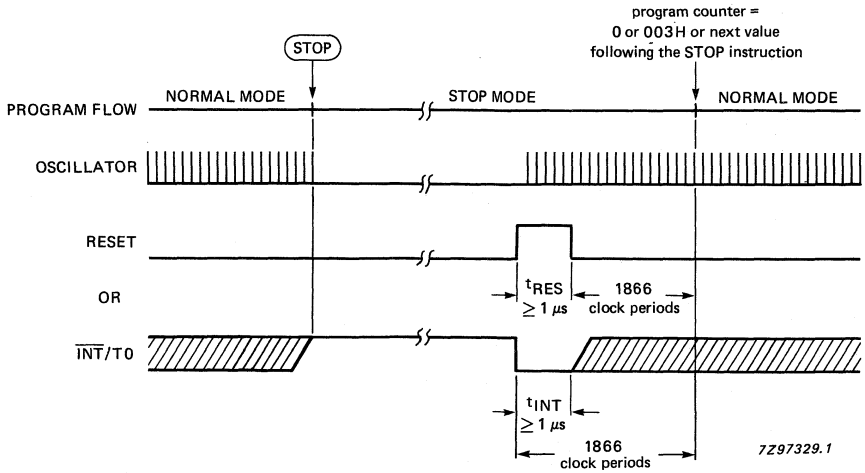


Fig.8 Entering and exiting the STOP mode.

DEVELOPMENT DATA

If the microcontroller exits the STOP mode by pulling the external interrupt input pin LOW, an interrupt sequence is executed only if the external interrupt is enabled. In this event the microcontroller resumes the normal program sequence after returning from the interrupt routine, as in the normal mode. If the interrupt is not enabled, it continues the normal program sequence, executing the instruction following the STOP instruction.

The microcontroller is restarted by a LOW level applied at the $\overline{INT}$/T0 pin.
Note: when leaving the STOP mode via an interrupt, a further instruction in the main program series is executed prior to entering the interrupt routine.

When the $\overline{INT}$/T0 level is active during the STOP instruction then no STOP is executed.

A LOW level on the external interrupt input of at least 1 µs will cause the microcontroller to exit the STOP mode. During the STOP mode, the address of the instruction immediately following the last STOP instruction is present on the address bus.

FUNCTIONAL DESCRIPTION (continued)

I/O facilities

The PCF84C230 has 34 I/O lines arranged as:

- Port 0 parallel port of 8 lines (P0.0 to P0.7)
- Port 1 parallel port of 4 lines (P1.0 to P1.3)
- LCD segment driver outputs 16 lines (S00 to S15)
- LCD backplane outputs 4 lines (BP0 to BP3)
- $\overline{INT}$/T0 external interrupt and test input. When used as a test input it can be directly tested by conditional branch instructions JT0 and JNT0
- T1 test input which can alter program sequences when tested by conditional jump instructions JT1 and JTN1. T1 also functions as an input to the 8-bit timer/event counter

Parallel ports

All parallel ports can be used as outputs or inputs, their structure is quasi-bidirectional. Output data written to a port is latched and remains unchanged until rewritten. Input data is not latched and so must be present until read by an input instruction.

Input lines are fully CMOS compatible, output lines can drive one TTL or CMOS load.

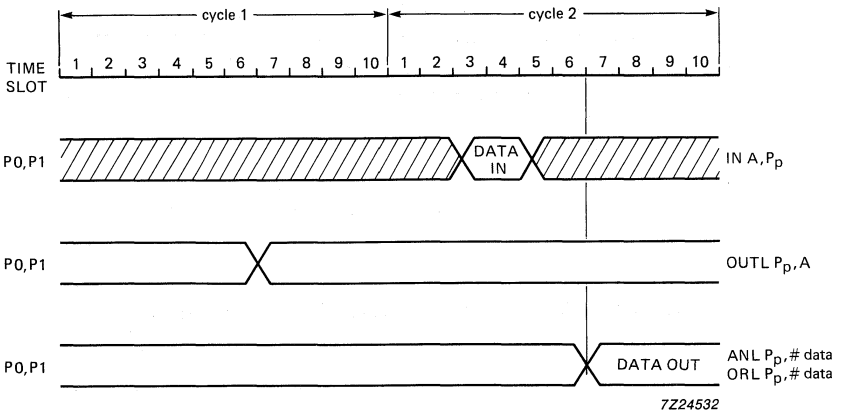


Fig.9 Timing diagram for all ports using IN, OUTL, ANL and ORL instructions.

Fig.10 shows the quasi-bidirectional I/O interface with push-pull output and switched pull-up current source.

Each line is pulled up to V_{DD} via a constant current source (TR4), which is enabled via TR3 whenever one of the two output latches contains a logic 1. This current source is sufficient for a TTL HIGH level, yet can be pulled LOW by an external CMOS device, thus allowing the same pin to be used for both input and output.

When a logic 1 is written to the line for the first time (MQ = 1, SQ = 0), TR2 is switched on for the duration of the internal write pulse (one oscillator period), to provide a fast transition from logic 0 to logic 1. Subsequent writing of a logic 1 to the port lines will not switch TR2 on. This prevents unnecessary current through external components connected to the port lines of the same port which might be in the input mode and also connected to ground.

When a logic 0 is written to the line, TR3 switches off the current source. Current sinking capability is provided by TR1, which is now switched on. When used as an input, a logic 1 must first be written to the line, otherwise TR1 will remain low impedance.

In different applications, this switched pull-up source may not be sufficient. Therefore the PCF84C230 offers the possibility to select individually the 12 parallel port pins by the following mask options:

Option 1 STANDARD PORT; quasi-bidirectional I/O with switched pull-up current source of 100 μA (typ.) and p-channel booster transistor TR2 (1.5 mA).
TR2 is only active during 1 clock cycle (see Fig.10).

Option 2 OPEN DRAIN; quasi-bidirectional I/O with only an n-channel open drain output. Application as an output requires connection of an external pull-up resistor (see Fig.11).

Option 3 PUSH-PULL OUTPUT; drive capability of the output is 1.5 mA (typ.) at V_{DD} = 3 V in both polarities. To avoid a large current flowing through the output transistors during the input mode, these push-pull pins must only be used as outputs (see Fig.12).

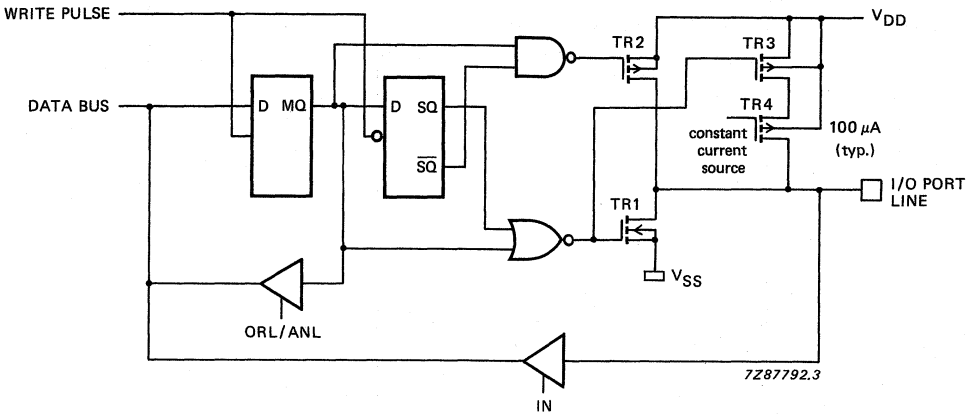


Fig.10 Standard output with switched pull-up current source.

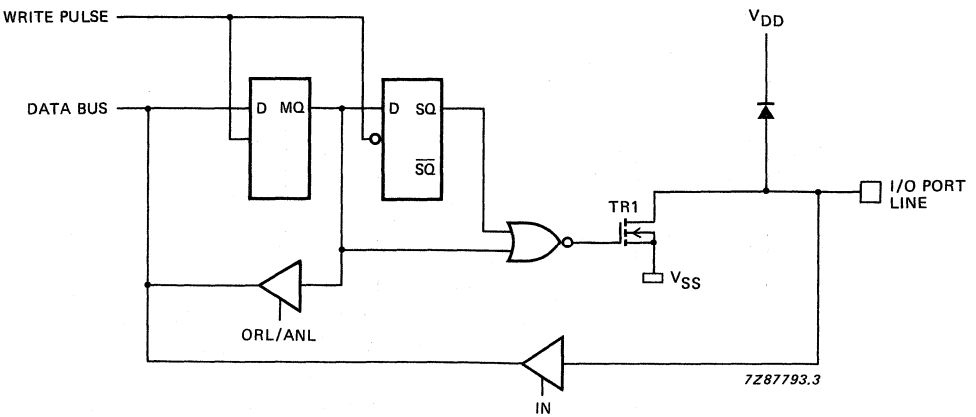


Fig.11 Open drain output.

FUNCTIONAL DESCRIPTION (continued)

I/O facilities (continued)

Parallel ports (continued)

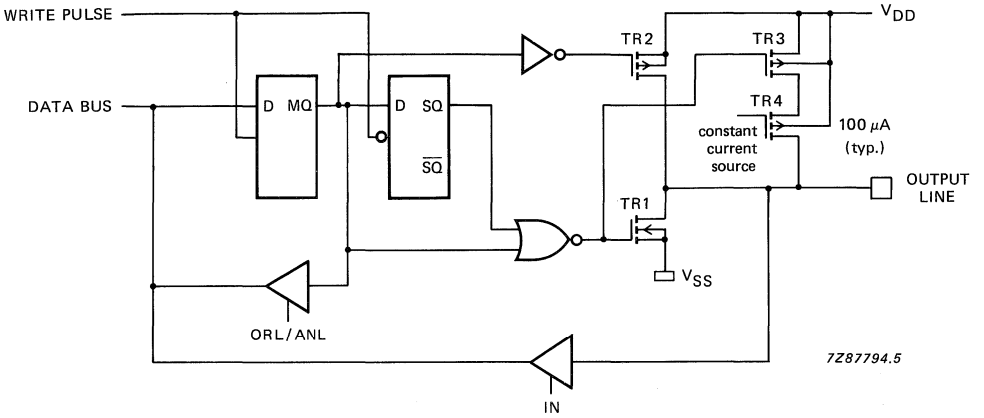


Fig.12 Push-pull output.

Interrupts

Upon entering an interrupt routine, the contents of the program counter and bits 4, 6 and 7 of the PSW are saved in the program counter stack. The contents of the accumulator must be saved by user software. Interrupt acknowledgement may be carried out in software via I/O ports. An interrupt routine can only be terminated by the RETR (return and restore) instruction. During an interrupt routine, further subroutine calls must be terminated using the RET instruction. Using the RETR instruction to terminate such a nested subroutine would terminate the interrupt routine prematurely.

External interrupt

When the external interrupt is enabled and no interrupt routine is in progress, a HIGH-to-LOW transition on the $\overline{INT}$/T0 pin sets the External Interrupt Flag (EIF) and invokes the external interrupt routine by forcing a CALL to location 3*. The program counter points to the external interrupt vector address (003H) between 2.6 and 3.6 machine cycles after the transition occurs. Interrupt latency will depend on the instruction that is being executed when the transition occurs. If an interrupt routine is already in progress, an external interrupt request is stored in the External Interrupt Flag (EIF). When the external interrupt is disabled the request is still latched into the digital filter. Execution of a DIS I instruction cancels a stored interrupt by clearing both the digital filter and the EIF.

Another external interrupt can be created by enabling the timer/event counter interrupt and loading FFH into the counter (one less than overflow). The STRT CNT instruction is then executed in user software, this enables the event counter mode and a LOW-to-HIGH transition on the T1 input will then initiate an interrupt subroutine and invoke a call to the timer/counter interrupt vector location 7.

Timer/counter interrupt

When no interrupt routine is in progress and the timer/counter is enabled, a timer/counter overflow sets the Timer Interrupt Flag (TIF). This initiates the timer interrupt routine by forcing a CALL to program location 7**. If an interrupt routine is in progress, the interrupt request is stored in the TIF only if the timer interrupt has been enabled. Execution of a DIS TCNTI instruction deletes a previously stored interrupt request. The Timer Flag (TF) is set every time the time/counter overflows and is not automatically reset after the timer routine is called. It can only be cleared by either the JTF or JNTF instruction or by a hardware RESET.

Interrupt priority

If simultaneous interrupts occur, their priority is as follows:

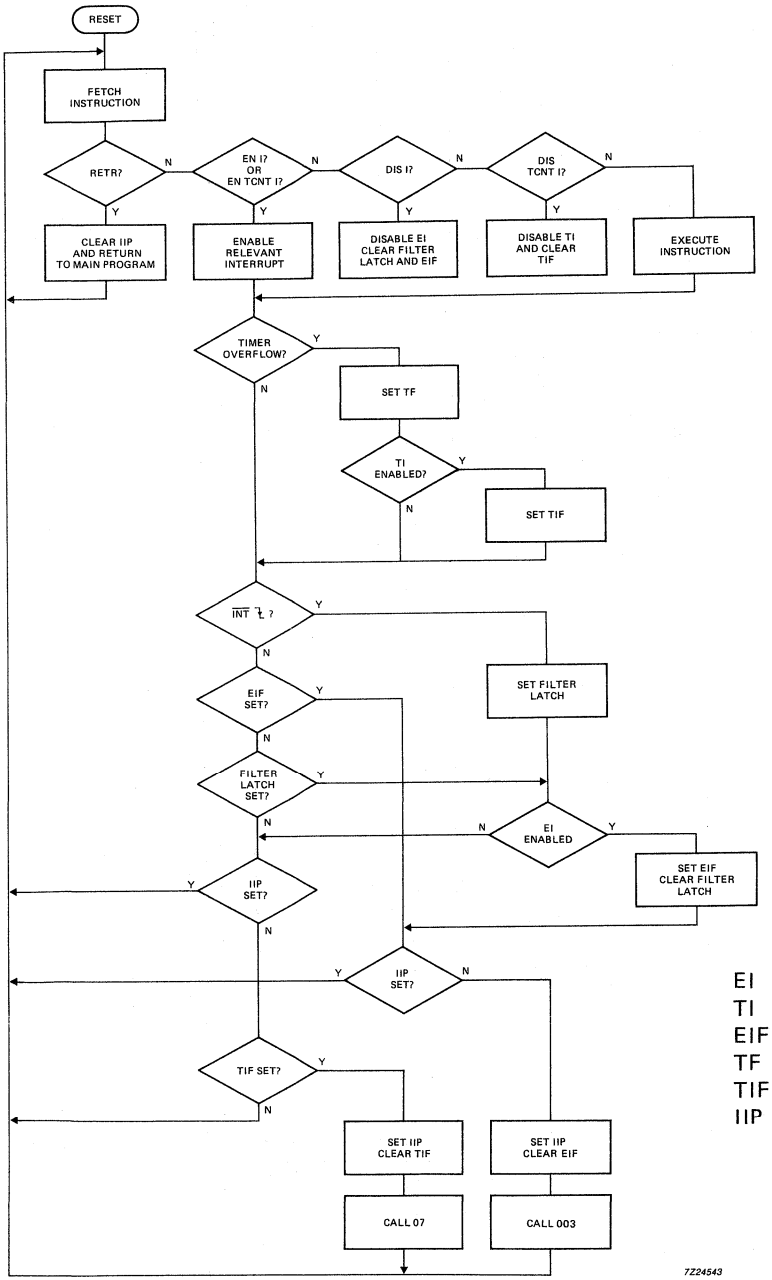
- External (highest)
- Timer/counter (lowest)

An interrupt routine can only be interrupted by a hardware RESET and cannot be interrupted by an other interrupt (which will be latched). When the interrupt routine is terminated by the RETR instruction, at least one instruction of the main program will be executed before another interrupt routine is entered.

* This CALL clears the EIF flag.
** This CALL clears the TIF flag.

FUNCTIONAL DESCRIPTION (continued)

Interrupts (continued)



EI External Interrupt
TI Timer/counter Interrupt
EIF External Interrupt Flag
TF Timer Flag
TIF Timer Interrupt Flag
IIP Interrupt In Progress flag

Fig.13 Flow chart illustrating the interrupt handling sequence.

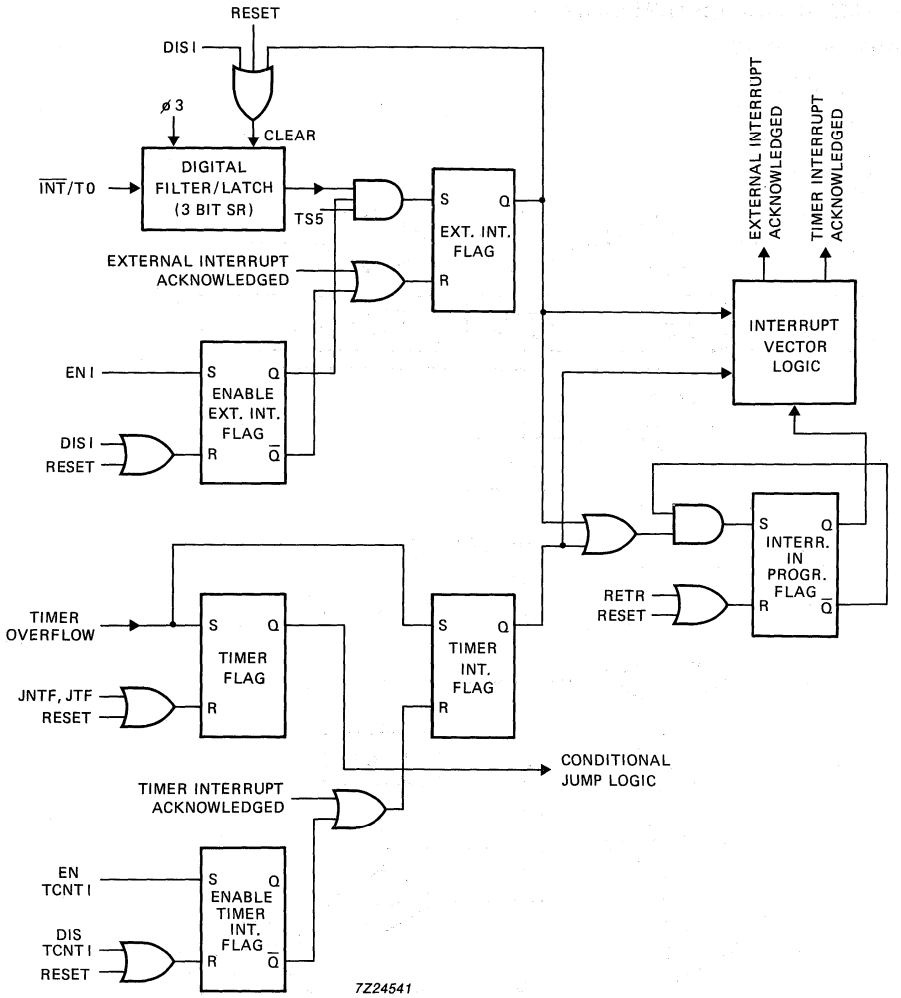


Fig.14 Interrupt logic.

Notes to Fig.14

1. $\overline{INT}$/T0 negative edge is always latched in the digital filter/latch.
2. Correct interrupt timing is ensured when $\overline{INT}$/T0 is HIGH for $>$ 4 CP followed by a LOW for $>$ 7 CP.
3. When the interrupt in progress flat is set, further external and timer interrupts are latched but ignored, until RETR is executed.
4. A DIS I instruction always clears a pending external interrupt.
5. For all flip-flops RESET overrules SET.

FUNCTIONAL DESCRIPTION (continued)

Oscillator (see Fig.15)

The oscillator can be inhibited by the STOP instruction under software control. It is also inhibited when a low-supply voltage condition is present to prevent discharge of a weak back-up battery. Provided the supply voltage is within the operating range the oscillator will be restarted after a STOP instruction by a LOW level at the $\overline{INT}$/T0 pin or a HIGH level at the RESET pin.

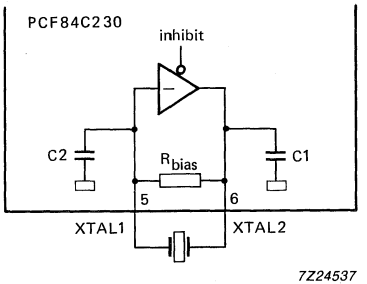


Fig.15 Oscillator with integrated elements.

The oscillator has an output drive capability for other CMOS devices via the XTAL 2 output. An external clock can be applied to the XTAL input. A machine cycle consists of 10 time slots, each time slot being 3 oscillator periods. A 10 MHz crystal provides a 3 μs machine cycle. The range of the clock frequency is from 100 kHz up to a maximum which is a function of the supply voltage.

Timer/event counter (see Fig.16)

An internal 8-bit up-counter is provided. It can count external events, modulo-32 machine cycles, or machine cycles directly. Table 1 gives the instructions that control the counter and the prescaler, and the functions performed.

When used as a timer, the input to the counter is either the overflow or input of a 5-bit prescaler. When used as an event counter, LOW-to-HIGH transitions on pin T1 are counted. The counter is incremented during a machine cycle only if the falling edge occurs during the first 7 time slots; otherwise is is incremented during the next cycle. The maximum rate at which the counter may be incremented is once every machine cycle. When the counter overflows, the Timer flag is set. The flag can be tested and reset using the JTF (jump if Timer flag = logic 1) or JNTF (jump if Timer flag = logic 0) instruction. Overflow also generates an interrupt request to the processor via setting of the Timer Interrupt Flag when the timer/event counter interrupt is enabled.

Table 1 Timer/event counter control

function	timer mode modulo-1, modulo-32*	counter mode
CLEAR	MOV T,A (A) = 0 or RESET	MOV T,A (A) = 0 or RESET
PRESET	MOV T,A	MOV T,A
START	STRT T	STRT CNT
STOP	STOP TCNT or RESET	STOP TCNT or RESET
TEST	JTF/JNTF	JTF/JNTF
READ**	MOV A,T	MOV A,T

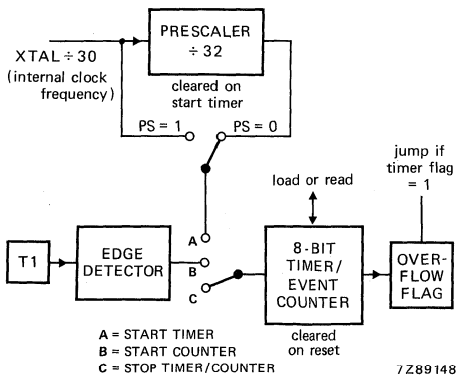


Fig.16 Timer/event counter.

* With prescaler select, PS = logic 0, the timer is incremented every 32 machine cycles; with PS = logic 1 the timer will be incremented every machine cycle (prescaler not used); the prescaler is cleared by the STRT T instruction and is not readable.

** READ does not disturb the counting process.

FUNCTIONAL DESCRIPTION (continued)

Program status word (see Fig.17)

The program status word (PSW) is an 8-bit word (1 byte) in the CPU which stores information about the current status of the microcontroller.
The PSW bits are:

- Bits 0 to 2 — stack pointer bits (SP_0, SP_1, SP_2)
- Bit 3 — prescaler select (PS);
  0 = modulo-32; 1 = modulo-1 (no prescaling)
- Bit 4 — working register bank select (RBS);
  0 = register bank 0; 1 = register bank 1
- Bit 5 — not used (1)
- Bit 6 — auxiliary carry (AC); half-carry bit generated by an ADD instruction and used by the decimal adjust instruction DA A
- Bit 7 — carry (CY); the carry flag indicates that previous operation has resulted in an overflow of the accumulator.

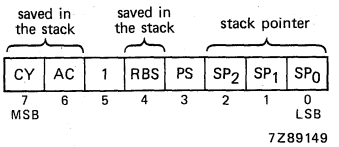


Fig.17 Program status word.

All bits can be read using the MOV A, PSW instruction. Bits 7 and 6 are set and cleared by CPU operation. Bit 4 can be changed by a SEL RB instruction, bit 3 by the MOV PSW, A instruction, and bits 0, 1 and 2 by the CALL, RET or RETR instructions in the event of an interrupt. Bits 7, 6 and 4 are stored in the program counter stack during subroutine and interrupt calls. These bits are restored in the PSW with a RETR (return and restore) instruction which must be used at the end of an interrupt and can be used at the end of a normal routine, which is not part of an interrupt subroutine. The RET instruction has no restore feature and must not be used at the end of an interrupt.

Program counter (see Fig.18)

An 11-bit program counter is used to address 2 K bytes of ROM. The arrangement of the bits is shown in Fig.18. Since the PCF84C230 can only address 2 K bytes of ROM, the maximum value that program counter can be loaded with is 7FFH. If a higher value is loaded the CPU ignores it. All 11 bits are saved in the stack during CALL and interrupt routines.

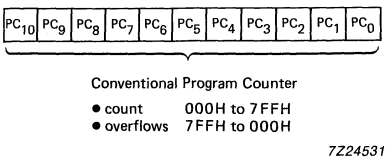


Fig.18 Program counter.

Central processing unit

The PCF84C230 has arithmetic, logical and branching capabilities. The DA A, SWAP A and XCHD instructions simplify BCD arithmetic and the handling of nibbles. The MOVP A, @A instruction permits efficient table look-up from the current ROM page.

Conditional branch logic

The conditional branch logic within the processor enables several conditions, internal and external to the processor, to be tested by the user's program. Table 2 lists the conditional jump instructions used to change the program sequence. The DJNZ instruction decrements a designated register or data memory location and branches if the contents are not zero. This instruction is useful for looping control. The JMPP@A instruction allows multiway branches to destinations determined by the contents of the accumulator.

Table 2 Conditional branches

test	jump condition	jump instruction
accumulator	all bits zero	JZ
	any bit non-zero	JNZ
accumulator bit test	1	JB0 to JB7
carry flag	1	JC
	0	JNC
timer overflow flag	1	JTF
	0	JNTF
test input T0	1	JT0
	0	JNT0
test input T1	1	JT1
	0	JNT1
register	non-zero	DJNZ

DEVELOPMENT DATA

Test input T1 (pin 28)

The T1 input line can be used as:

- A test input for branch instructions JT1 and JNT1
- An external input to the event counter

When used as a test input:

- JT1 instruction tests for logic 1 level
- JNT1 instruction tests for logic 0 level

When used as an input to the event counter, T1 must be LOW for $>$ 4 CP, followed by a HIGH for $>$ 4 CP. A transition can be recognized every 30 oscillator clock periods (1 machine cycle).

There is no internal pull-up or pull-down resistor connected to the T1 input. If required it must be externally connected to a resistor ($R = \leqslant 100\ k\Omega$). When T1 is not used pin 28 must be connected V_{DD} or V_{SS}.

FUNCTIONAL DESCRIPTION (continued)

Reset (pin 7)

A positive-going signal on the RESET input/output:

- Sets the program counter to zero
- Selects location 0 of memory bank 0 and register bank 0
- Sets the stack pointer to zero (000); pointing to RAM address 8
- Disables the interrupts (external and timer interrupts are all disabled)
- Stops the timer/event counter, then sets it to zero
- Sets the timer prescaler to modulo-32
- Resets the timer flag
- Sets all ports in accordance with reset states
- Sets the LCD segments and backplane outputs to V_{DD}
- Sets LCD drive mode 1:4 with 1/3 bias
- Cancels IDLE and STOP mode

After the voltage is applied to RESET, an internal delay of 1866 clock pulses is introduced before the microcontroller commences operation.

Power-on reset and low voltage detection (see Fig.19)

In many applications, correct operation of the PCF84C230 during moments of slowly changing supply voltages and low voltage conditions is essential. This is achieved by the addition of an internal power-on reset and low voltage detection circuit.

To allow an external reset signal to be fed into the PCF84C230, the RESET pin (pin 7) has been configured as an input/output. While a reset condition exists in the detection circuit, the RESET pin is pulled HIGH by transistor TR1 controlled by the reset circuit. When the reset condition is not present, a pull-down current source (TR2) will be activated. TR2 forces the RESET pin LOW, thus removing the reset signal from the microcontroller. Since the level at the RESET pin is detected by the microcontroller, the reset time constant can be stretched by connecting an external capacitor between V_{DD} and the RESET pin (see Fig.21).

The signal at the RESET pin can also be used as an output to reset other devices in the system. The internal reset circuit monitors the PCF84C230 supply voltage. If the supply voltage drops below the switching level (typically 1.3 V), a reset (HIGH) is applied to the RESET pin. This reset signal is removed (RESET pin goes LOW), after a fixed delay (t_d), when the supply voltage rises above the switching level again. The delay ensures a complete reset even when the supply voltage rises quickly above switching level after initial switch-on. During a low-voltage condition, the oscillator is inhibited to prevent complete discharge of a weak battery. The timing of the power-on reset and low voltage detection circuit is shown in Fig.25.

The internal power-on reset circuit monitors the PCF84C230 supply voltage V_{DD}. For as long as the supply voltage remains below the internal reference voltage level V_{ref}, the oscillator is inhibited and RESET has an undefined level. When V_{DD} rises above the internal reference level, the oscillator is released and RESET is pulled high to V_{DD} by TR1 for a period typically 50 μs. Because of the narrow bandwidth of the crystal, the start-up time of the oscillator is typically 10 ms.

DEVELOPMENT DATA

Power-on reset

Three modes of power-on reset are possible:

1. If V_{DD} can be switched with a fast rise time i.e. V_{DD} reaches its minimum operating value (corresponding to the selected oscillator frequency) before the RESET signal (t_d) has finished, then no extra components are required (see Fig.19 and 20). Note that the first instruction is executed after the oscillator start-up time plus 1866 clock periods have elapsed.
2. If V_{DD} has a slow rise time then the RESET signal should be stretched by an external RC circuit (see Fig.21 and 22). In the event of a short drop in the supply voltage, the diode path rapidly discharges the capacitor to ensure a reliable power-on reset. To ensure a correct reset, the RESET signal should reach at least 70% of the final value of V_{DD}. Given that the RESET voltage and V_{DD} rise exponentially, the above requirement is satisfied when the time constant τ of the RESET pulse is >8 times the time constant of V_{DD}. If V_{DD} rises linearly, then a RESET time constant >2 times the rise time of V_{DD} is required.

   When a reset is completed (RESET goes LOW) before the oscillator has started up, program execution begins after the oscillator start-up time plus 1866 clock periods have elapsed (see Fig.22). If the oscillator is started-up prior to the completion of RESET, then program execution begins 1866 clock periods after RESET goes LOW.
3. Figure 23 shows an external reset to the PCF84C230 during power-on. The external reset signal must remain HIGH until V_{DD} has reached its minimum operating value corresponding to the selected oscillator frequency. When a reset is completed (RESET goes LOW) before the oscillator has started up, program execution begins after the oscillator start-up time plus 1866 clock periods have elapsed (see Fig.24). If the oscillator is started-up prior to the completion of RESET, then program execution begins 1866 clock periods after RESET goes LOW.

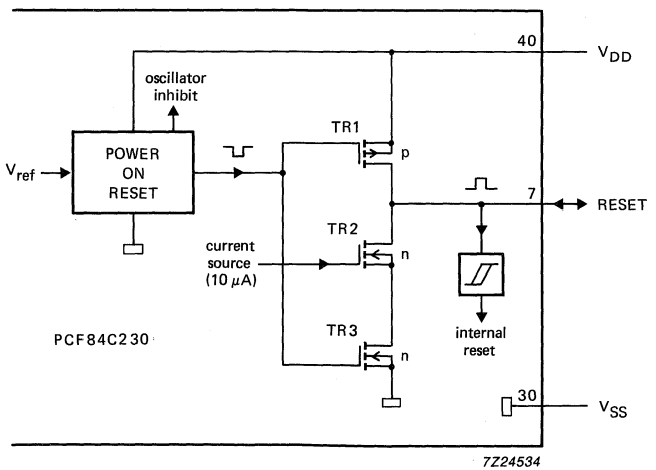


Fig.19 Power-on reset configuration.

FUNCTIONAL DESCRIPTION (continued)

Power-on reset (continued)

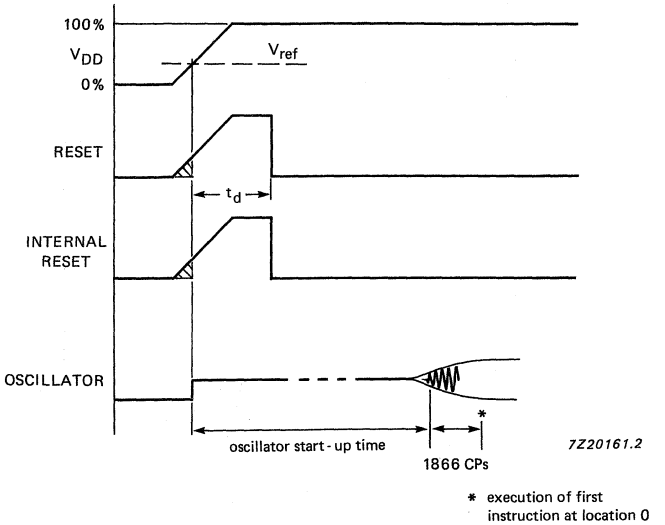


Fig.20 Timing of power-on reset with fast rise time.

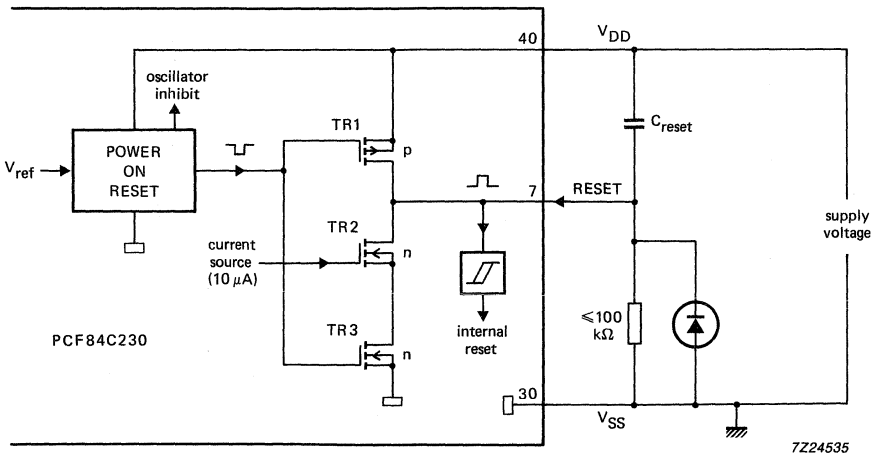


Fig.21 Stretched power-on reset with external components.

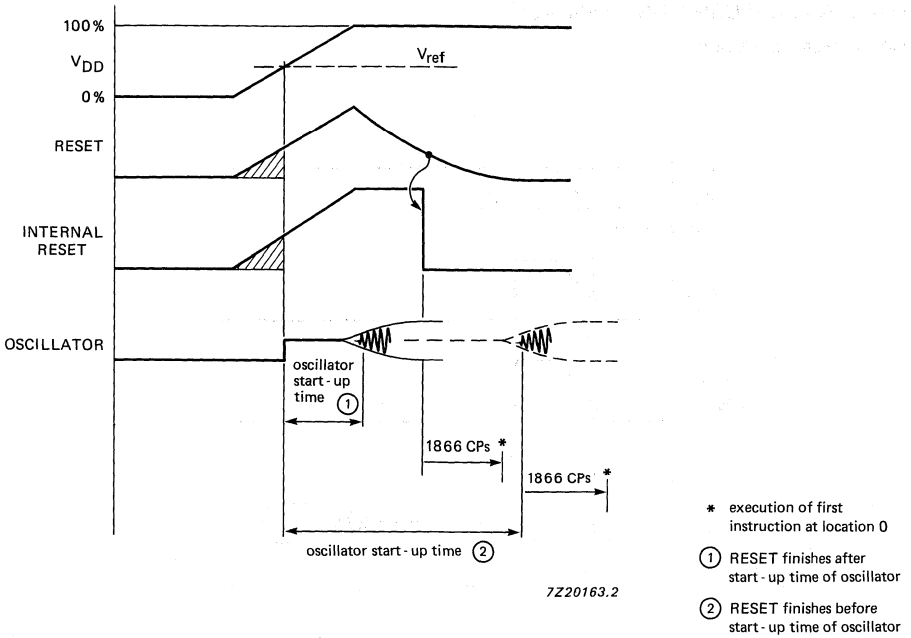


Fig.22 Timing of power-on reset with a slowly rising V_{DD} and a stretched RESET pulse.

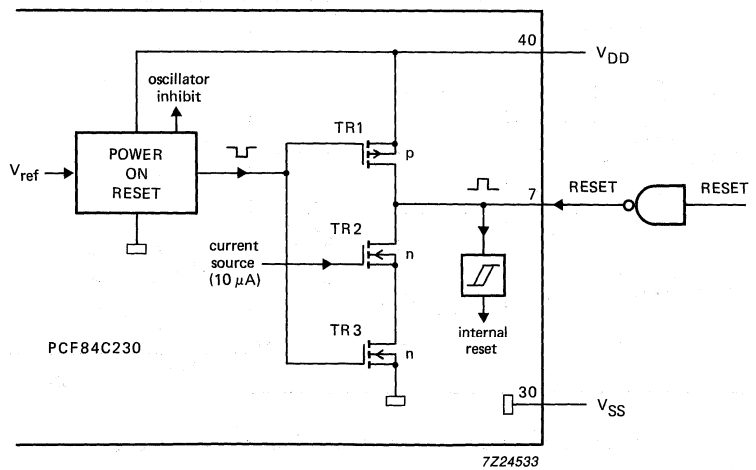


Fig.23 External power-on reset configuration.

FUNCTIONAL DESCRIPTION (continued)

Power-on reset (continued)

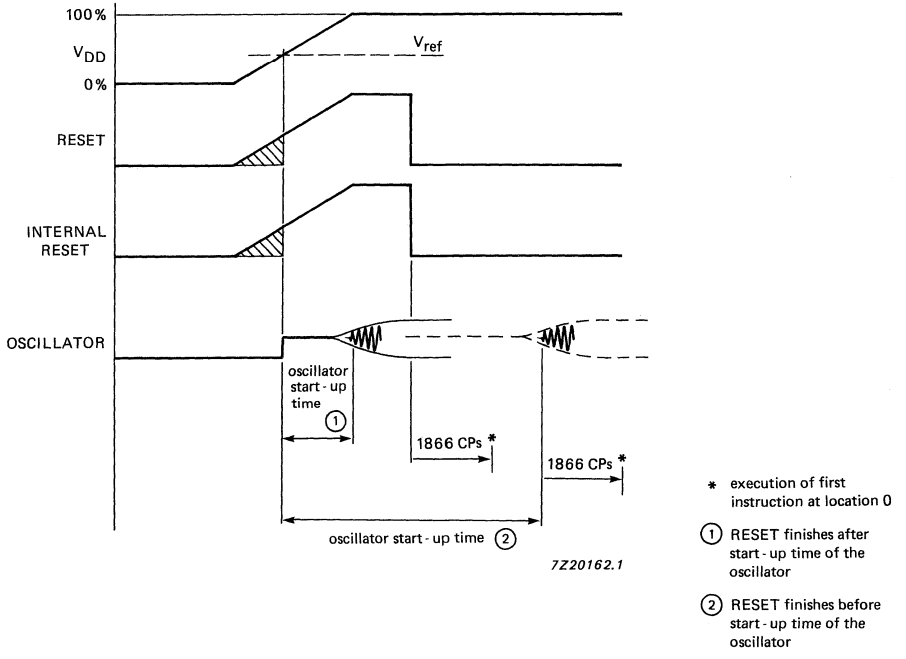
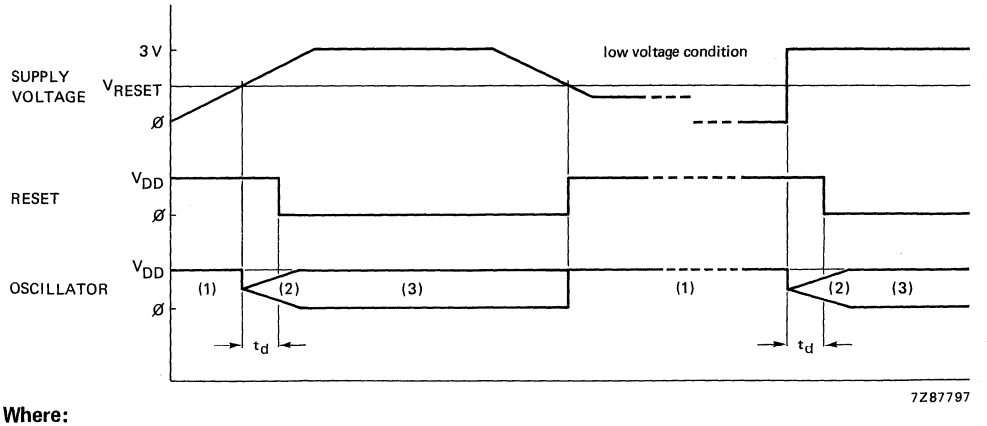


Fig.24 Timing of external power-on reset.



Where:

(1) oscillator inhibited
(2) oscillator started
(3) oscillator running, but may be stopped with a STOP condition

Fig.25 Timing of power-on reset and low voltage detection.

DEVELOPMENT DATA

Liquid crystal display driver

The PCF84C230 has a display driver which interfaces to almost any liquid crystal display (LCD) which has a low multiplex rate. The interface delivers drive signals for any static or multiplexed LCD panel that contains up to four backplanes and up to 16 segments.

The following features are incorporated:

- Selectable backplane drive configuration; static or 2/3/4 backplane multiplexing
- Selectable display bias configuration; 1/2 or 1/3 internal LCD bias generation
- 16 individual segment drivers can be used to provide:
  - up to eight 8-segment numeric characters
  - up to four 15 + 1 segment alphanumeric characters
  - graphics using up to 64 elements
- Eight 8-bit derivative registers for display data bits
- LCD and logic voltage supplies may be separated
- An LCD operating voltage range of between 2.5 V to 5.5 V
- A low-power zero-component oscillator keeps the LCD display running in the STOP mode

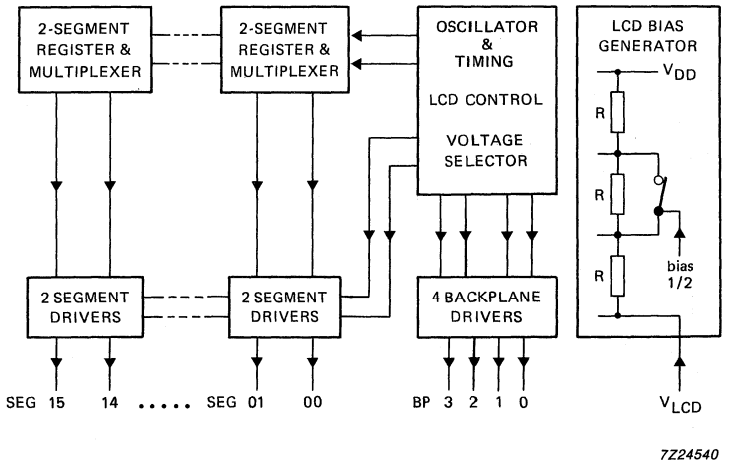


Fig.26 Block diagram of the LCD driver.

FUNCTIONAL DESCRIPTION (continued)

LCD driver (continued)

The LCD circuit can directly drive any static or multiplexed LCD containing up to four backplanes and up to 16 segments. The display configurations possible with the PCF84C230 depend upon the number of active backplane outputs required. A selection of display configurations is given in Table 3.

Table 3 Selection of display configurations

number of backplanes	number of segments	7-segment numeric	14-segment alphanumeric	dot matrix
4	64	9 digits plus 1 indicator symbol	4 characters plus 8 indicator symbols	64 dots (4 x 16)
3	48	6 digits plus 6 indicator symbols	3 characters plus 6 indicator symbols	48 dots (3 x 16)
2	32	4 digits plus 4 indicator symbols	2 characters plus 4 indicator symbols	32 dots (2 x 16)
1	16	2 digits plus 2 indicator symbols	1 character plus 2 indicator symbols	16 dots (1 x 16)

LCD bias generation

The LCD operating voltage (V_{op}) is the result of V_{DD} - V_{LCD}. V_{op} should be chosen so that the off voltage ($V_{off(RMS)}$) is just below the threshold voltage (V_{th}), typically when the LCD exhibits 10% contrast. The LCD voltage may be temperature compensated externally through the LCD supply. Fractional LCD biasing voltages are obtained from an internal voltage divider of three resistors connected between V_{DD} and V_{LCD}. The centre resistor may be switched out of circuit to provide a ½ bias voltage level for a 1:2 multiplex configuration.

LCD voltage selector

The LCD voltage selector coordinates the multiplexing of the LCD according to the selected drive configuration. The operation of the voltage selector is controlled by the MODE bits in the LCD control byte. The biasing configurations that apply to the preferred mode of operation, together with the biasing characteristics as functions of $V_{op} = V_{DD} - V_{LCD}$ and resulting discrimination ratios (D), are given in Table 4.

Table 4 LCD drive modes and characteristics

LCD drive mode	number of backplanes	LCD bias configuration	number of levels	$\frac{V_{off(rms)}}{V_{op}}$	$\frac{V_{on(rms)}}{V_{op}}$	$D = \frac{V_{on(rms)}}{V_{off(rms)}}$
static	1	static	2	0	1	∞
1:2	2	1/2	3	0.354	0.791	2.234
1:2	2	1/3	4	0.333	0.745	2.237
1:3	3	1/3	4	0.333	0.638	1.916
1:4	4	1/3	4	0.333	0.577	1.7321

Multiplex drive ratios of 1:3 and 1:4 with ½ bias are possible, but the discrimination and contrast ratios are reduced thus;

(1.732 for 1:3 or 1.528 for 1:4)

There is an advantage however, that these modes lead to a reduction in V_{op} as follows:

1:3 multiplex (½ bias). $V_{op} = 2.449\ V_{off(rms)}$

1:4 multiplex (½ bias). $V_{op} = 2.309\ V_{off(rms)}$

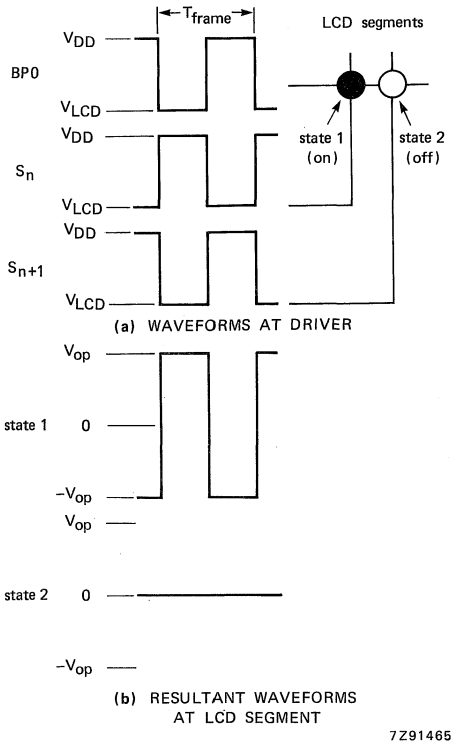
This compares with $V_{op} = 3\ V_{off(rms)}$ when 1/3 bias is used.

DEVELOPMENT DATA

FUNCTIONAL DESCRIPTION (continued)

LCD drive mode waveforms

The static LCD drive mode is used when a single backplane is provided in the LCD. Backplane and segment drive waveforms are shown in Fig.27.



At any instant (t):

$V_{state\ 1}(t) = V_{S_n}(t) - V_{BP0}(t)$

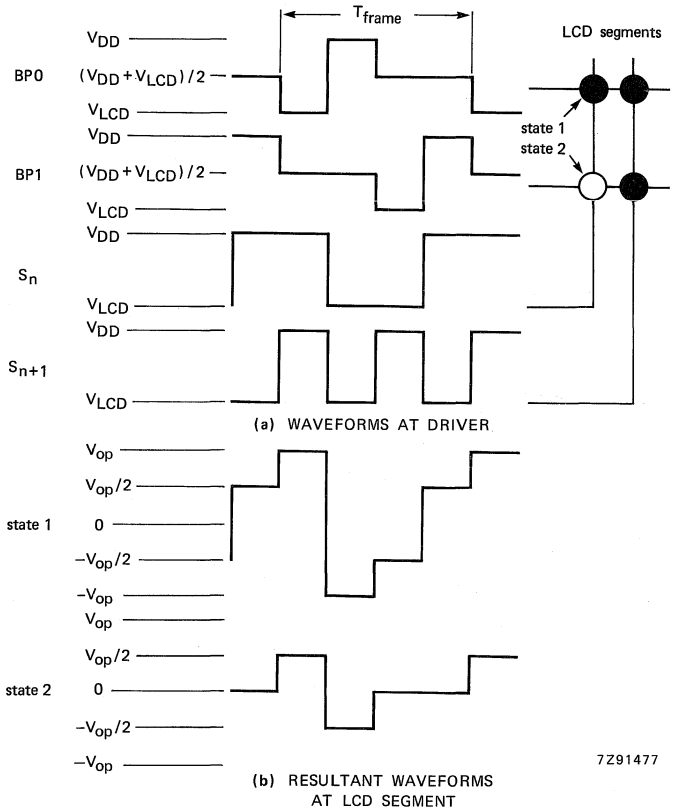
$V_{on(rms)} = V_{op}$

$V_{state\ 2}(t) = V_{S_{n+1}}(t) - V_{BP0}(t)$

$V_{off(rms)} = 0\ V$

Fig.27 Static drive mode waveforms; $V_{op} = V_{DD} - V_{LCD}$.

When two backplanes are provided in the LCD, the 1:2 multiplex drive mode applies. The PCF84C230 allows use of 1/2 or 1/3 bias in this mode as shown in Figs 28 and 29.



At any instant (t):

$V_{state\ 1}(t) = V_{S_n}(t) - V_{BP0}(t)$

$V_{on(rms)} = \frac{V_{op}}{4}\sqrt{10} = 0.791\ V_{op}$

$V_{state\ 2}(t) = V_{S_n}(t) - V_{BP1}(t)$

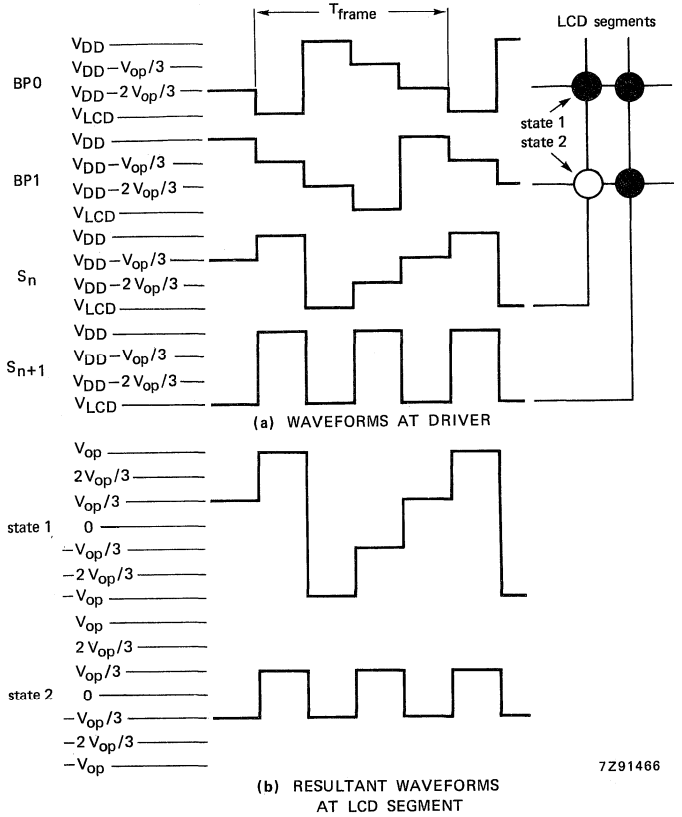
$V_{off(rms)} = \frac{V_{op}}{4}\sqrt{2} = 0.354\ V_{op}$

Fig.28 Waveforms for the 1:2 multiplex drive mode with 1/2 bias; $V_{op} = V_{DD} - V_{LCD}$.

DEVELOPMENT DATA

FUNCTIONAL DESCRIPTION (continued)

LCD drive mode waveforms (continued)



At any instant (t):

$V_{state\ 1}(t) = V_{S_n}(t) - V_{BP0}(t)$

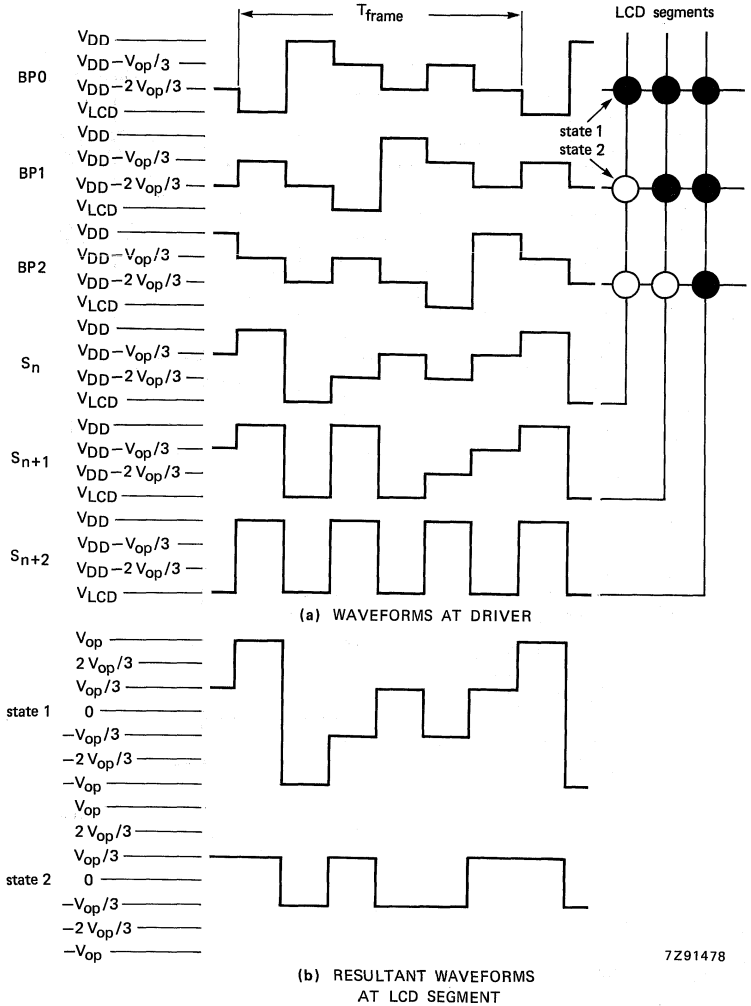
$V_{on(rms)} = \frac{V_{op}}{3}\sqrt{5} = 0.745\ V_{op}$

$V_{state\ 2}(t) = V_{S_n}(t) - V_{BP1}(t)$

$V_{off(rms)} = \frac{V_{op}}{3} = 0.333\ V_{op}$

Fig.29 Waveforms for the 1:2 multiplex drive mode with 1/3 bias; $V_{op} = V_{DD} - V_{LCD}$.

The backplane and segment drive waveforms for the 1:3 multiplex drive mode (three LCD backplanes) and for the 1:4 multiplex drive mode (four LCD backplanes) are shown in Figs 30 and 31 respectively.



At any instant (t):

$$V_{state\ 1}(t) = V_{S_n}(t) - V_{BP0}(t)$$

$$V_{on(rms)} = \frac{V_{op}}{9}\sqrt{33} = 0.638\ V_{op}$$

$$V_{state\ 2}(t) = V_{S_n}(t) - V_{BP1}(t)$$

$$V_{off(rms)} = \frac{V_{op}}{3} = 0.333\ V_{op}$$

Fig.30 Waveforms for the 1:3 multiplex drive mode; $V_{op} = V_{DD} - V_{LCD}$.

DEVELOPMENT DATA

FUNCTIONAL DESCRIPTION (continued)

LCD drive mode waveforms (continued)

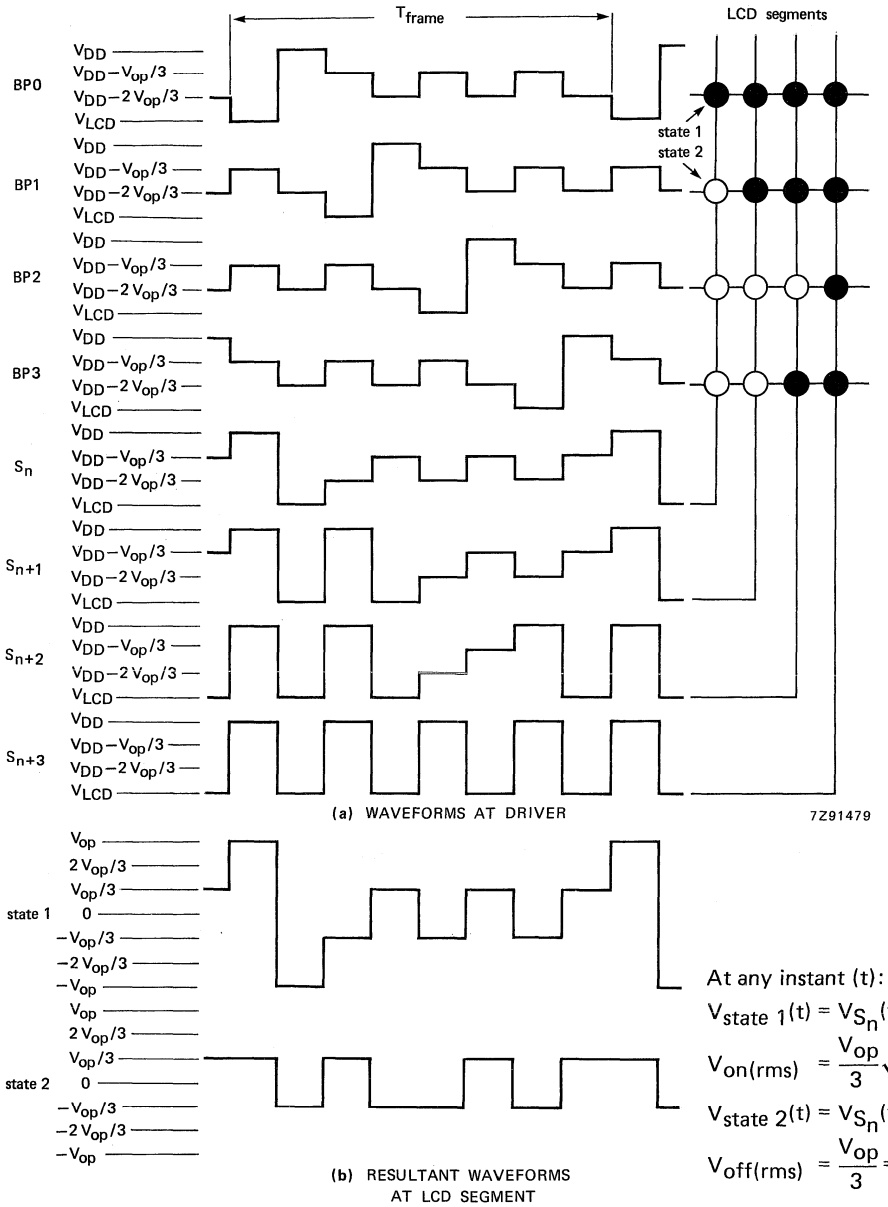


Fig.31 Waveforms for the 1:4 multiplex drive mode; $V_{op} = V_{DD} - V_{LCD}$.

LCD timing

The LCD clocking is performed by a separate, self-contained, on-chip oscillator with a nominal frequency of 30 kHz. The display may be kept active while in the STOP mode using a very low supply current.

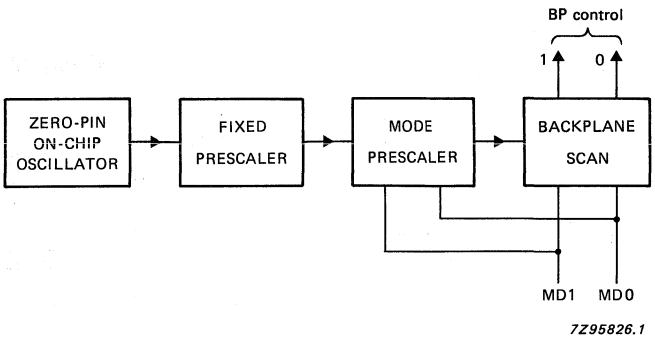


Fig.32 LCD timing control.

LCD segment driver outputs

The LCD drive section includes 16 segment outputs (S0 to S15) which should be connected directly to the LCD. The segment data bits (one nibble per segment) are multiplexed to the outputs in accordance with the backplane signals. If less than the 16 segment outputs are required then the unused driver outputs should be left open.

Backplane outputs

The LCD drive section includes 4 backplane outputs (BP0-BP3) which should be connected directly to the LCD. These backplane output signals are generated in accordance with the selected LCD drive mode. If less than four backplane outputs are required then the unused driver outputs should be left open-circuit.

In the 1:3 multiplex drive mode, BP3 carries the same signal as BP0, therefore these two outputs can be tied together to give enhanced drive capabilities. In the 1:2 multiplex drive mode, BP0 and BP3, BP1 and BP2 respectively carry the same signals and may also be paired to increase the drive capabilities. In the static drive mode the same signal is carried by all four backplane outputs and may be connected in parallel to give a very high drive capability.

FUNCTIONAL DESCRIPTION (continued)

LCD segment display registers

The LCD byte control register (LCDCON) is an 8-bit read/write register whose individual bits control the operation of the LCD display driver. LCDCON can be accessed through derivative address 16 (H'10').

LCDCON will be set to H'OC' during a reset or power-on reset. The function of each bit is given in Table 5.

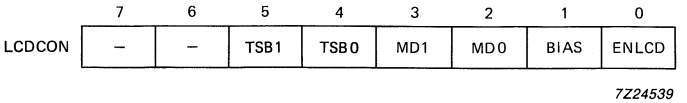


Table 5 LCDCON bit definition

bit	name	function		
0	ENLCD	0 = LCD driver disabled; set LCD segment and backplane outputs to V_{DD} 1 = LCD driver enabled		
1	BIAS	0 = bias voltage set to 1/3 bias 1 = bias voltage set to 1/2 bias		
2, 3	MD0, MD1	MD0	MD1	multiplex mode
		0 0 1 1	0 1 0 1	static 1:2 1:3 1:4
4, 5	TSB0, TSB1	used for testing; must be "00"		

The PCF84C230 has 8 additional 8-bit derivative (read/write) Segment Display Registers (SDR0 to SDR7) which store LCD segment data. They can be accessed by derivative addresses 17 H'11 to 24 H'18. A segment register bit which is set to logic 1 indicates the 'on' state of the corresponding LCD segment, similarly, a logic 0 indicates the 'off' state. There is a one-to-one relationship between the LCD segment register bits and the segment outputs. Every byte is divided into two nibbles. Each nibble corresponds to a segment driver; the first byte contains the bits earmarked for segment 1 and 2 etc. The first bit of a nibble corresponds to backplane 0, and the second to backplane 1 and so on. Fig.33 shows the display register bit map. Each bit is shown in the form Sxx.n, where (xx) is the segment output and (n) the backplane output.

LCDD : LCD display data

17 (H' 11')	R/W	S 0.0	S 0.1	S 0.2	S 0.3	S 1.0	S 1.1	S 1.2	S 1.3	SDR0
18 (H' 12')	R/W	S 2.0	S 2.1	S 2.2	S 2.3	S 3.0	S 3.1	S 3.2	S 3.3	SDR1
19 (H' 13')	R/W	S 4.0	S 4.1	S 4.2	S 4.3	S 5.0	S 5.1	S 5.2	S 5.3	SDR2
20 (H' 14')	R/W	S 6.0	S 6.1	S 6.2	S 6.3	S 7.0	S 7.1	S 7.2	S 7.3	SDR3
21 (H' 15')	R/W	S 8.0	S 8.1	S 8.2	S 8.3	S 9.0	S 9.1	S 9.2	S 9.3	SDR4
22 (H' 16')	R/W	S10.0	S10.1	S10.2	S10.3	S11.0	S11.1	S11.2	S11.3	SDR5
23 (H' 17')	R/W	S12.0	S12.1	S12.2	S12.3	S13.0	S13.1	S13.2	S13.3	SDR6
24 (H' 18')	R/W	S14.0	S14.1	S14.2	S14.3	S15.0	S15.1	S15.2	S15.3	SDR7

Fig.33 Display register bit map.

DEVELOPMENT DATA

In the static drive mode the eight display data bits are placed in bit 0 of the nibbles of four successive derivative display registers. In the 1:2 multiplex drive mode, these eight bits are placed in bits 0 and 1 of the nibbles of two successive registers. In the 1:3 multiplex drive mode the eight bits are placed in bits 0, 1 and 2 of three successive nibbles. In the 1:4 multiplex drive mode the eight bits are placed in bits 0, 1, 2 and 3 of the nibbles of the first display register. Fig.34 shows the relationship between LCD segment layout, drive mode and the LCD segment derivative register bit pattern.

FUNCTIONAL DESCRIPTION (continued)

LCD segment display registers (continued)

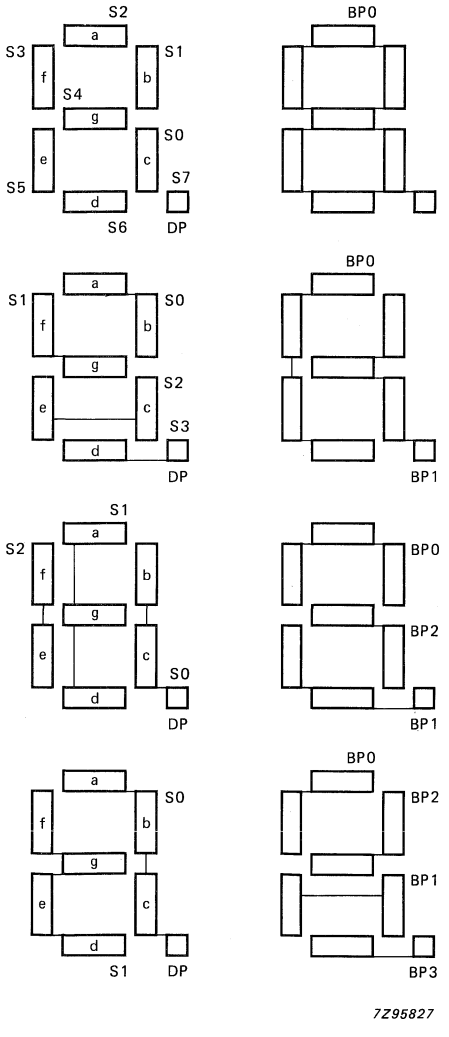


Fig.34 LCD segment layout and register bit pattern.

LCD derivative register bit map

byte

17	c	–	–	–	b	–	–	–
18	a	–	–	–	f	–	–	–
19	g	–	–	–	e	–	–	–
20	d	–	–	–	Dp	–	–	–

static mode

byte

17	a	b	–	–	f	g	–	–
18	e	c	–	–	d	Dp	–	–
19	–	–	–	–	–	–	–	–
20	–	–	–	–	–	–	–	–

1:2 multiplex mode

byte

17	b	Dp	c	–	a	d	g	–
18	f	e	–	–	–	–	–	–
19	–	–	–	–	–	–	–	–
20	–	–	–	–	–	–	–	–

1:3 multiplex mode

byte

17	a	c	b	Dp	f	e	g	d
18	–	–	–	–	–	–	–	–
19	–	–	–	–	–	–	–	–
20	–	–	–	–	–	–	–	–

1:4 multiplex mode

Table 6 Accessing each derivative register

register	function	R/W	instruction used to access*
LCDCON	LCD control register	R/W	MOV A, D16 (8C 10); read to accumulator MOV D16, A (8D 10); write from accumulator ANL D16, A (8E 10); AND with accumulator ORL D16, A (8F 10); OR with accumulator
SDRn	segment display register n	R/W	MOV A, Dx (8C 1y); read to accumulator MOV Dx, A (8D 1y); write from accumulator ANL Dx, A (8E 1y); AND with accumulator ORL Dx, A (8F 1y); OR with accumulator

Where

n represents the segment display register (SDR) number to be accessed; 0 to 7

x represents the derivative address (in decimal) of the particular SDR to be accessed; 17 to 24

y represents the corresponding derivative address (in hexadecimal) of the particular SDR to be accessed; 1 to 8

The segment display registers and the corresponding derivative addresses areas follows:

SDR0 <– – –> D17 (H'11')
SDR1 <– – –> D18 (H'12')
SDR2 <– – –> D19 (H'13')
SDR3 <– – –> D20 (H'14')
SDR4 <– – –> D21 (H'15')
SDR5 <– – –> D22 (H'16')
SDR6 <– – –> D23 (H'17')
SDR7 <– – –> D24 (H'18')

DEVELOPMENT DATA

* Hexadecimal code written in parenthesis.

INSTRUCTION SET

The PCF84C230 instruction set consists of over 80 one and two byte instructions. Program code efficiency is high because all RAM locations and all ROM locations on a 256 byte page require only a single byte address.

Table 9 gives the instruction set of the PCF84C230. Table 8 shows the instruction map and Table 7 details the symbols and definition descriptions that are used.

Table 7 Symbols and definitions used in Table 9

symbol	definition description
A	accumulator
addr	program memory address
Bb	bit designation (b = 0-7)
RBS	register bank select
C	carry bit (bit CY)
CNT	event counter
Dn	derivative register designation (n = 16-24)
data	8-bit number or expression
I	interrupt
MB	memory bank
MBFF	memory bank flip-flop
P	mnemonic for 'in-page' operation
PC	program counter
Pp	port designation (p = 0 or 1)
PSW	program status word
RB	register bank
Rr	register designation (r = 0-7)
SP	stack pointer
T	timer
TF	timer flag
T0, T1	test 0 and 1 inputs
#	immediate data prefix
@	indirect address prefix
(X)	contents of X
((X))	contents of location addressed by X
←	is replaced by
↔	is exchanged with

Table 8 PCF84C230 instruction map

first hexadecimal character of opcode ↓ / second hexadecimal character of opcode →	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
0	NOP	IDLE		ADD A, #data	JMP page 0	EN I	JNTF addr	DEC A	IN A, Pp							
1	INC @Rr		JB0	ADDC	CALL	DIS I	JTF	INC A	INC Rr							
	0	1	addr	A, #data	page 0		addr		0	1	2	3	4	5	6	7
2	XCH A, @Rr		STOP	MOV	JMP	EN	JNT0	CLR A	XCH A, Rr							
	0	1		A, #data	page 1	TCNTI	addr		0	1	2	3	4	5	6	7
3	XCHD A, @Rr		JB1		CALL	DIS	JT0	CPL A	OUTL Pp, A							
	0	1	addr		page 1	TCNTI	addr		0	1						
4	ORL A, @Rr		MOV	ORL	JMP	STRT	JNT1	SWAP	ORL A, Rr							
	0	1	A, T	A, #data	page 2	CNT	addr	A	0	1	2	3	4	5	6	7
5	ANL A, @Rr		JB2	ANL	CALL	STRT	JT1	DA A	ANL A, Rr							
	0	1	addr	A, #data	page 2	T	addr		0	1	2	3	4	5	6	7
6	ADD A, @Rr		MOV		JMP	STOP		RRC A	ADD A, Rr							
	0	1	T, A		page 3	TCNT			0	1	2	3	4	5	6	7
7	ADDC A, @Rr		JB3		CALL			RR A	ADDC A, Rr							
	0	1	addr		page 3				0	1	2	3	4	5	6	7
8				RET	JMP				ORL Pp, #data				MOV	MOV	ANL	ORL
					page 4				0	1			A,Dx	Dx,A	Dx,A	Dx,A
9			JB4	RETR	CALL		JNZ	CLR C	ANL Pp, #data							
			addr		page 4		addr		0	1						
A	MOV @Rr,A			MOVP	JMP			CPL C	MOV Rr, A							
	0	1		A, @A	page 5				0	1	2	3	4	5	6	7
B	MOV @Rr, #data		JB5	JMPP	CALL				MOV Rr, #data							
	0	1	addr	@A	page 5				0	1	2	3	4	5	6	7
C	DEC @Rr				JMP	SEL	JZ	MOV	DEC Rr							
	0	1			page 6	RB0	addr	A, PSW	0	1	2	3	4	5	6	7
D	XRL A, @Rr		JB6	XRL	CALL	SEL		MOV	XRL A, Rr							
	0	1	addr	A, #data	page 6	RB1		PSW, A	0	1	2	3	4	5	6	7
E	DJNZ @Rr, addr				JMP		JNC	RL A	DJNZ Rr, addr							
	0	1			page 7		addr		0	1	2	3	4	5	6	7
F	MOV A, @Rr		JB7		CALL		JC	RLC A	MOV A, Rr							
	0	1	addr		page 7		addr		0	1	2	3	4	5	6	7

INSTRUCTION SET (continued)

Table 9 Instruction set

	mnemonic	opcode (hex.)	bytes/ cycles	description	function		notes
ACCUMULATOR	ADD A, Rr	6*	1/1	Add register contents to A	(A)←(A) + (Rr)	r = 0–7	1
	ADD A, @Rr	60 61	1/1	Add RAM data, addressed by Rr, to A	(A)←(A) + ((R0)) (A)←(A) + ((R1))		1
	ADD A, #data	03 data	2/2	Add immediate data to A	(A)←(A) + data		1
	ADDC A, Rr	7*	1/1	Add carry and register contents to A	(A)←(A) + (Rr) + (C)	r = 0–7	1
	ADDC A, @Rr	70 71	1/1	Add carry and RAM data, addressed by Rr, to A	(A)←(A) + ((R0)) + (C) (A)←(A) + ((R1)) + (C)		1
	ADDC A, #data	13 data	2/2	Add carry and immediate data to A	(A)←(A) + data + (C)		1
	ANL A, Rr	5*	1/1	'AND' Rr with A	(A)←(A) AND (Rr)	r = 0–7	
	ANL A, @Rr	50 51	1/1	'AND' RAM data, addressed by Rr, with A	(A)←(A) AND ((R0)) (A)←(A) AND ((R1))		
	ANL A, #data	53 data	2/2	'AND' immediate data with A	(A)←(A) AND data		
	ORL A, Rr	4*	1/1	'OR' Rr with A	(A)←(A) OR (Rr)	r = 0–7	
	ORL A, @Rr	40 41	1/1	'OR' RAM data, addressed by Rr, with A	(A)←(A) OR ((R0)) (A)←(A) OR ((R1))		
	ORL A, #data	43 data	2/2	'OR' immediate data with A	(A)←(A) OR data		
	XRL A, Rr	D*	1/1	'XOR' Rr with A	(A)←(A) XOR (Rr)	r = 0–7	
	XRL A, @Rr	D0 D1	1/1	'XOR' RAM, addressed by Rr, with A	(A)←(A) XOR ((R0)) (A)←(A) XOR ((R1))		
	XRL A, #data	D3 data	2/2	'XOR' immediate data with A	(A)←(A) XOR data		
	INC A	17	1/1	increment A by 1	(A)←(A) + 1		
	DEC A	07	1/1	decrement A by 1	(A)←(A) − 1		
	CLR A	27	1/1	clear A to zero	(A)←0		
	CPL A	37	1/1	one's complement A	(A)←NOT(A)		
	RL A	E7	1/1	rotate A left	$(A_{n+1})\leftarrow(A_n)$ $(A_0)\leftarrow(A_7)$	n = 0–6	

DEVELOPMENT DATA

ACCUMULATOR	RLC A	F7	1/1	rotate left through carry	$(A_{n+1}) \leftarrow A_n$ $(A_0) \leftarrow (C), (C) \leftarrow (A_7)$	n = 0–6	2
	RR A	77	1/1	rotate A right	$(A_n) \leftarrow (A_{n+1})$ $(A_7) \leftarrow (A_0)$	n = 0–6	
	RRC A	67	1/1	rotate A right through carry	$(A_n) \leftarrow (A_{n+1})$ $(A_7) \leftarrow (C), (C) \leftarrow (A_0)$	n = 0–6	2
	DA A	57	1/1	decimal adjust A			2
	SWAP A	47	1/1	swap nibbles of A	$(A_{4-7}) \leftrightarrow (A_{0-3})$		
DATA MOVES	MOV A, Rr	F*	1/1	move register contents to A	(A)←(Rr)	r = 0–7	
	MOV A, @Rr	F0 F1	1/1	move RAM data, addressed by Rr, to A	(A)←((R0)) (A)←((R1))		
	MOV A, #data	23 data	2/2	move immediate data to A	(A)←data		
	MOV Rr, A	A*	1/1	move accumulator contents to register	(Rr)←(A)	r = 0–7	
	MOV @Rr, A	A0 A1	1/1	move accumulator contents to RAM location addressed by Rr	((R0))←(A) ((R1))←(A)		
	MOV Rr, #data	B* data	2/2	move immediate data to Rr	(Rr)←data		
	MOV @Rr, #data	B0 data B1 data	2/2	move immediate data to RAM location addressed by Rr	((R0))←data ((R1))←data		
	XCH A, Rr	2*	1/1	exchange accumulator contents with Rr	(A)↔(Rr)	r = 0–7	
	XCH A, @Rr	20 21	1/1	exchange accumulator contents with RAM data addressed by Rr	(A)↔((R0)) (A)↔((R1))		
	XCHD A, @Rr	30 31	1/1	exchange lower nibbles of A and RAM data addressed by Rr	$(A_{0-3}) \leftrightarrow ((R0_{0-3}))$ $(A_{0-3}) \leftrightarrow ((R1_{0-3}))$		
	MOV A, PSW	C7	1/1	move PSW contents to accumulator	(A)←(PSW)		
	MOV PSW, A	D7	1/1	move accumulator bit 3 to PSW_3	$(PSW_3) \leftarrow (A_3)$		3
	MOVP A, @A	A3	1/2	move indirectly addressed data in current page to A	$(PC_{0-7}) \leftarrow (A), (A) \leftarrow ((PC))$		
FLAGS	CLR C	97	1/1	clear carry bit	(C)←0		2
	CPL C	A7	1/1	complement carry bit	(C)←NOT(C)		2

INSTRUCTION SET (continued)

	mnemonic	opcode (hex.)	bytes cycles	description	function	notes
BRANCH	INC Rr	1*	1/1	increment register by 1	(Rr)←(Rr) + 1 r = 0–7	
	INC @Rr	10 11	1/1	increment RAM data, addressed by Rr, by 1	((R0))←((R0)) + 1 ((R1))←((R1)) + 1	
	DEC Re	C*	1/1	decrement register by 1	(Rr)←(Rr) − 1 r = 0–7	
	DEC @Rr	C0 C1	1/1	decrement RAM data, addressed by Rr, by 1	((R0))←((R0)) − 1 ((R1))←((R1)) − 1	
REGISTER	JMP addr	● 4 addr	2/2	unconditional jump within a 2 K bank	(PC_{8-10})←$addr_{8-10}$ (PC_{0-7})←$addr_{0-7}$ (PC_{11-12})←MBFF 0–1	
	JMPP @A	B3	1/2	indirect jump within a page	(PC_{0-7})←((A))	
	DJNZ Rr, addr	E* addr	2/2	decrement Rr by 1 and jump if not zero to addr	(Rr)←(Rr) − 1 r = 0–7 if (Rr) not zero (PC_{0-7})←addr	
	DJNZ @Rr, addr	E0 addr	2/2	decrement RAM data, addressed by Rr by 1 and jump if not zero to addr	((R0))←((R0)) − 1 if ((R0)) not zero (PC_{0-7})←addr	
		E1 addr			((R1))←((R1)) − 1 if ((R1)) not zero (PC_{0-7})←addr	
	JBb addr	▲ 2 addr	2/2	jump to addr if Acc. bit b = 1	if b = 1 : (PC_{0-7})←addr b = 0–7	
	JC addr	F6 addr	2/2	jump to addr if C = 1	if C = 1 : (PC_{0-7})←addr	
	JNC addr	E6 addr	2/2	jump to addr if C = 0	if C = 0 : (PC_{0-7})←addr	
	JZ addr	C6 addr	2/2	jump to addr if A = 0	if A = 0 : (PC_{0-7})←addr	
	JNZ addr	96 addr	2/2	jump to addr if A is NOT zero	if A≠0 : (PC_{0-7})←addr	
	JT0 addr	36 addr	2/2	jump to addr if T0 = 1	if T0 = 1: (PC_{0-7})←addr	
	JNT0 addr	26 addr	2/2	jump to addr if T0 = 0	if T0 = 0: (PC_{0-7})←addr	
	JT1 addr	56 addr	2/2	jump to addr if T1 = 1	if T1 = 1: (PC_{0-7})←addr	
	JNT1 addr	46 addr	2/2	jump to addr if T1 = 0	if T1 = 0: (PC_{0-7})←addr	
	JTF addr	16 addr	2/2	jump to addr if Timer Flag = 1	if TF = 1: (PC_{0-7})←addr	4
	JNTF addr	06 addr	2/2	jump to addr if Timer Flag = 0	if TF = 0: (PC_{0-7})←addr	

TIMER/EVENT COUNTER	MOV A, T	42	1/1	move timer/event counter contents to accumulator	(A)←(T)	
	MOV T, A	62	1/1	move accumulator contents to timer/event counter	(T)←(A)	
	STRT CNT	45	1/1	start event counter		
	STRT T	55	1/1	start timer		
	STOP TCNT	65	1/1	stop timer/event counter		
	EN TCNTI	25	1/1	enable timer/event counter interrupt		
	DIS TCNTI	35	1/1	disable timer/event counter interrupt		
CONTROL	EN I	05	1/1	enable external interrupt		
	DIS I	15	1/1	disable external interrupt		
	SEL RB0	C5	1/1	select register bank 0	(RBS)←0	5
	SEL RB1	D5	1/1	select register bank 1	(RBS)←1	5
	STOP	22	1/1	enter STOP mode		
	IDLE	01	1/1	enter IDLE mode		
SUBROUTINE	CALL addr	▲ 4 addr	2/2	jump to subroutine	((SP))←(PC), ($PSW_{4, 6, 7}$) (SP)←(SP) + 1 (PC_{8-10})←$addr_{8-10}$ (PC_{0-8})←$addr_{0-7}$ (PC_{11-12})←MBFF 0–1	6
	RET	83	1/2	return from subroutine	(SP)←(SP) – 1 (PC)←((SP))	6
	RETR	93	1/2	return from interrupt and restore bits 4, 6, 7 of PSW	(SP)←(SP) – 1 ($PSW_{4, 6, 7}$) + (PC)←((SP))	6

INSTRUCTION SET (continued)

	mnemonic	opcode (hex.)	bytes/ cycles	description	function	notes
PARALLEL INPUT/OUTPUT	IN A, Pp	08 09	1/2	input port p data ot accumulator	(A)←(P0) (A)←(P1)	7
	OUTL Pp, A	38 39	1/2	output accumulator data to port p	(P0)←(A) (P1)←(A)	
	ANL Pp, #data	98 data 99 data	2/2	AND port p data with immediate data	(P0)←(P0) AND data (P1)←(P1) AND data	
	ORL Pp, #data	88 data 89 data	2/2	OR port p data with immediate data	(P0)←(P0) OR data	
DERIVATIVE INPUT/OUTPUT	MOV A, Dn	8C direct	2/2	move derivative register contents to accumulator	(A)←(Dn)	8
	MOV Dn, A	8D direct	2/2	move contents of accumulator to derivative register	(Dn)←(A)	
	ANL Dn, A	8E direct	2/2	AND contents of accumulator with derivative register	(Dn)←(Dn) AND (A)	
	ORL Dn, A	8F direct	2/2	OR contents of accumulator with derivative register	(Dn)←(Dn) OR (A)	
	NOP	00	1/1	no operation		

Notes to Table 10

1. PSW CY, AC affected
2. PSW CY affected
3. PSW PS affected
4. Execution of JTF and JNTF instructions resets the Timer Flag (TF).

* : 8,9,A,B,C,D,E,F
● : 0,2,4,6,8,A,C,E
▲ : 0,3,5,7,9,B,D,F

5. PSW RBS affected
6. PSW SP_0, SP_1, SP_2 affected
7. For Port 1 it is the lower nibble of the accumulator that is moved/AND/ORed with P1.
8. n is 16, 17, 18, 19, 20, 21, 22, 23 or 24 depending on the derivative register to be accessed.

RATINGS

Limiting values in accordance with the Absolute Maximum System (IEC 134)

parameter	symbol	min.	max.	unit
Supply voltage (pin 40)	V_{DD}	−0.8	+ 8.0	V
All input voltages	V_I	−0.8	V_{DD} + 0.8	V
DC current into any input or output	$\pm I_I, \pm I_O$	–	10	mA
Total power dissipation	P_{tot}	–	500	mW
Power dissipation per output	P_O	–	50	mW
Storage temperature range	T_{stg}	−65	+ 150	°C
Operating ambient temperature range	T_{amb}	−40	+ 85	°C
Operating junction temperature	T_j	–	+ 125	°C

HANDLING

Inputs and outputs are protected against electrostatic discharge in normal handling. However, to be totally safe, it is desirable to take normal precautions appropriate to handling MOS devices (see 'Handling MOS devices').

DEVELOPMENT DATA

LCD DRIVER CHARACTERISTICS

V_{DD} = 2.5 to 5.5 V; V_{SS} = 0 V; T_{amb} = −40 to + 85 °C. All voltages with respect to V_{SS} unless otherwise specified

parameter	conditions	symbol	min.	typ.	max.	unit
LCD supply voltage	note 1	V_{LCD}	V_{SS}	–	$V_{DD}-2$	V
DC voltage component (BP0 to BP3)	V_{DD} = 5 V	$+ V_{BP}$	0	–	20	mV
DC voltage component (S00 to S15)	V_{DD} = 5 V	$\pm V_S$	–	–	20	mV
Output impedance (BP0 to BP3)	$V_{LCD} = V_{DD}-5$ V; note 2	R_{BP}	–	2.5	5	kΩ
Output impedance (S00 to S15)	$V_{LCD} = V_{DD}-5$ V; note 2	R_S	–	4.5	7	kΩ
Driver delays with test loads	$V_{LCD} = V_{DD}-5$ V	t_{PLCD}	–	–	30	μs
LCD frame scan frequency		f_{LCD}	–	60	–	Hz

Notes to the LCD driver characteristics

1. $V_{LCD} < V_{DD}-3$ V for 1/3 bias.
2. Outputs measured one at a time in DC mode; V_{DD} = 5.5 V.

DC CHARACTERISTICS

V_{DD} = 2.5 to 5.5 V; V_{SS} = 0 V; T_{amb} = −40 to + 85 °C; all voltages with respect to V_{SS}; f_{XTAL} = 10 MHz; R_S = 50 Ω unless otherwise specified

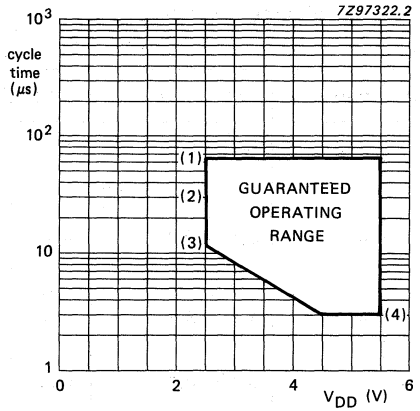
parameter	conditions	symbol	min.	typ.	max.	unit
Supply voltage		V_{DD}	2.5	–	5.5	V
Supply current	note 1					
operating	V_{DD} = 3 V					
	f_{XTAL} = 3.58 MHz	I_{DD}	–	0.4	0.8	mA
	V_{DD} = 5.0 V					
	f_{XTAL} = 6 MHz	I_{DD}	–	1.0	2.0	mA
	f_{XTAL} = 10 MHz	I_{DD}	–	1.6	3.2	mA
IDLE mode	V_{DD} = 3 V					
	f_{XTAL} = 3.58 MHz	I_{DD}	–	0.15	0.4	mA
	V_{DD} = 5.0 V					
	f_{XTAL} = 6 MHz	I_{DD}	–	0.5	1.0	mA
	f_{XTAL} = 10 MHz	I_{DD}	–	0.8	1.6	mA
STOP mode						
	V_{DD} = 2.5 V; T_{amb} = 25 °C	I_{DD}	–	15	30	μA
	V_{DD} = 2.5 V; T_{amb} = 85 °C	I_{DD}	–	12	24	μA
Inputs						
Input voltage LOW		V_{IL}	0	–	0.3 V_{DD}	V
Input voltage HIGH		V_{IH}	0.7 V_{DD}	–	V_{DD}	V
Input leakage current	$V_{SS} < V_I < V_{DD}$	$\pm I_{IL}$	–	–	1	μA
Outputs						
Output voltage LOW	V_I = V_{SS} or V_{DD}; $\lvert I_O \rvert < 1$ μA	V_{OL}	–	–	0.05	V
Output sink current LOW	V_{DD} = 5 V ± 10%; V_O = 0.4 V	I_{OL}	1.6	3.0	–	mA
Pull-up output source current HIGH	V_{DD} = 5 V ± 10%;					
	V_O = 0.7 V_{DD}	$-I_{OH}$	40	–	–	μA
	V_O = 0.4 V	$-I_{OH}$	–	–	400	μA
Push-pull output source current HIGH	V_{DD} = 5 V ± 10%;					
	V_O = V_{DD}−0.4 V	$-I_{OH}$	1.6	3.0	–	mA

Notes to the DC characteristics

1. V_{IL} = V_{SS}; V_{IH} = V_{DD}; all outputs unloaded; all open drain outputs connected to V_{SS}.

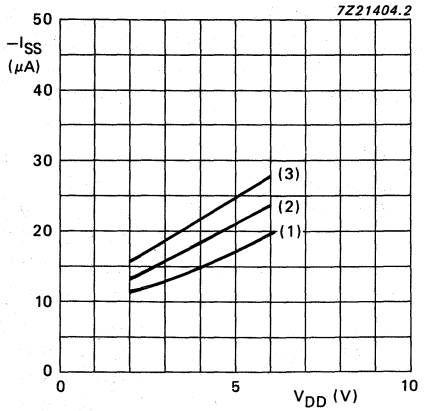
DEVELOPMENT DATA

AC CHARACTERISTICS



(1) clock frequency = 450 kHz
(2) clock frequency = 1 MHz
(3) clock frequency = 3 MHz
(4) clock frequency = 10 MHz

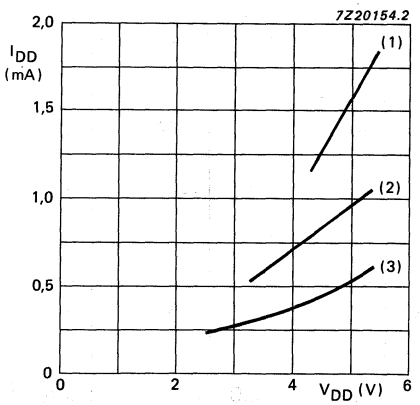
Fig.35 Maximum clock frequency (f_{XTAL}) as a function of the supply voltage (V_{DD}).



$f_{CLK} \triangleq 30$ kHz

(1) $T_{amb} = +85$ °C
(2) $T_{amb} = +25$ °C
(3) $T_{amb} = -40$ °C

Fig.36 Typical supply current ($-I_{SS}$) in STOP mode as a function of the supply voltage (V_{DD}).



(1) clock frequency = 10 MHz
(2) clock frequency = 6 MHz
(3) clock frequency = 3.58 MHz

Fig.37 Maximum supply current (I_{DD}) in IDLE mode as a function of the supply voltage (V_{DD}).

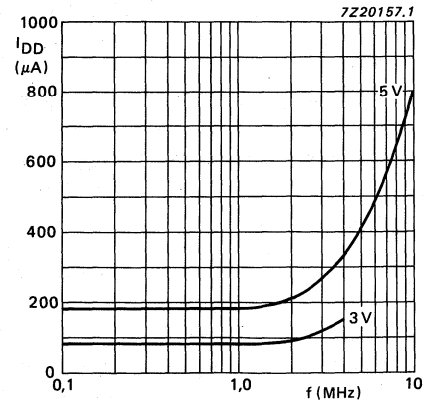
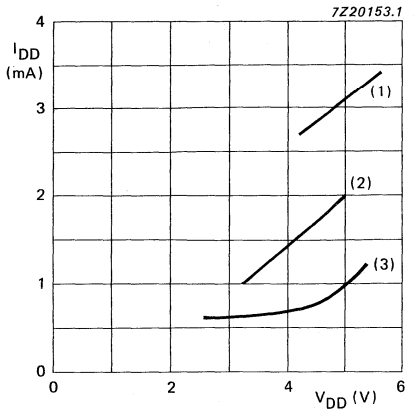


Fig.38 Typical supply current during IDLE mode as a function of frequency at $V_{DD} = 3$ V and $V_{DD} = 5$ V.

AC CHARACTERISTICS (continued)



(1) clock frequency = 10 MHz
(2) clock frequency = 6 MHz
(3) clock frequency = 3.58 MHz

Fig.39 Maximum supply current (I_{DD}) in operating mode as a function of the supply voltage.

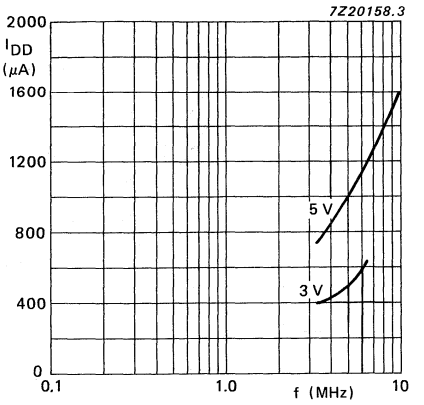


Fig.40 Typical supply current during operating mode as a function of frequency at V_{DD} = 3 V and V_{DD} = 5 V.

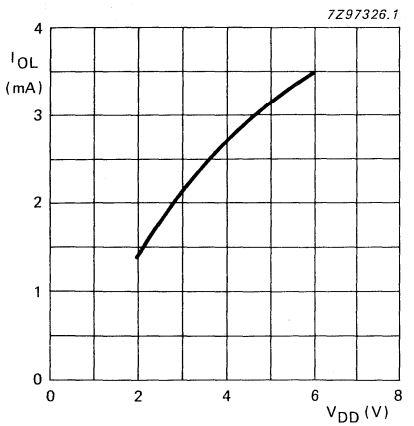
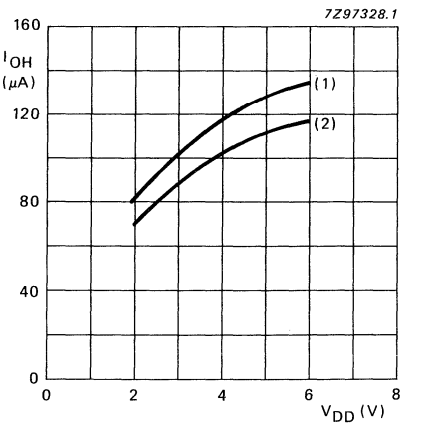
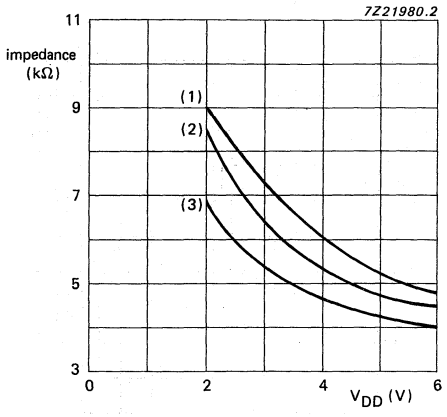


Fig.41 Typical output sink current (I_{OL}), outputs P0.0 to P0.7 and P1.0 to P1.3; as a function of the supply voltage (V_{DD}); V_O = 0.4 V.



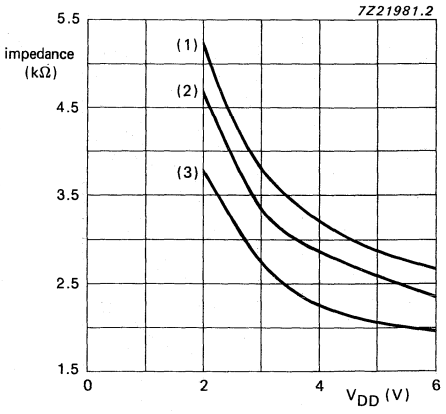
(1) $V_O = V_{SS}$
(2) $V_O = 0.7\ V_{DD}$

Fig.42 Typical output source current ($-I_{OH}$) as a function of the supply voltage (V_{DD}).



(1) + 90 °C
(2) + 25 °C
(3) −50 °C

Fig.43 Segment output impedance.



(1) + 90 °C
(2) + 25 °C
(3) −50 °C

Fig.44 Backplane output impedance.

DEVELOPMENT DATA

Philips Components

Data sheet	
status	Preliminary specification
date of issue	November 1990

PCF84C270/271/470

Single-chip 8-bit microcontrollers

INTRODUCTION

The PCF84C270, PCF84C271 and PCF84C470 CMOS microcontrollers are members of the PCF84CXXX microcontroller family. This data sheet should be read in conjunction with the PCF84CXXX Family Specification which describes the PCF84CXXX microcontroller family core.

FEATURES

- PCF84CXXX family core (see the PCF84CXXX Family Specification):
  - 2 K bytes ROM (PCF84C270 and PCF84C271); 4 K bytes (PCF84C470)
  - 128 bytes RAM
  - 2 single level vectored interrupts (external and timer/counter)
  - 8-bit quasi-bidirectional I/O port: 6 I/O lines have LED drive capabilities and 2 are open drain with high current drive capabilities
- 16 keyboard scanning output lines
- 8 sense amplifier input lines; the PCF84C470 has 8 sense amplifiers with hysteresis inputs
- Accurate sense amplifier reference voltage (externally adjustable)
- Clock frequency range: 450 kHz to 10 MHz
- Single supply voltage (2.5 V to 5.5 V)
- Operating temperature range: –40 to 85 °C

ORDERING AND PACKAGE INFORMATION

EXTENDED TYPENUMBER	PACKAGE				ORDER CODE
	PINS	PIN POSITION	MATERIAL	CODE	
PCF84C270T	40	VSO	plastic	SOT158	
PCF84C270P	40	DIL	plastic	SOT129	
PCF84C271T	40	VSO	plastic	SOT158	
PCF84C271P	40	DIL	plastic	SOT129	
PCF84C470P	40	DIL	plastic	SOT129	

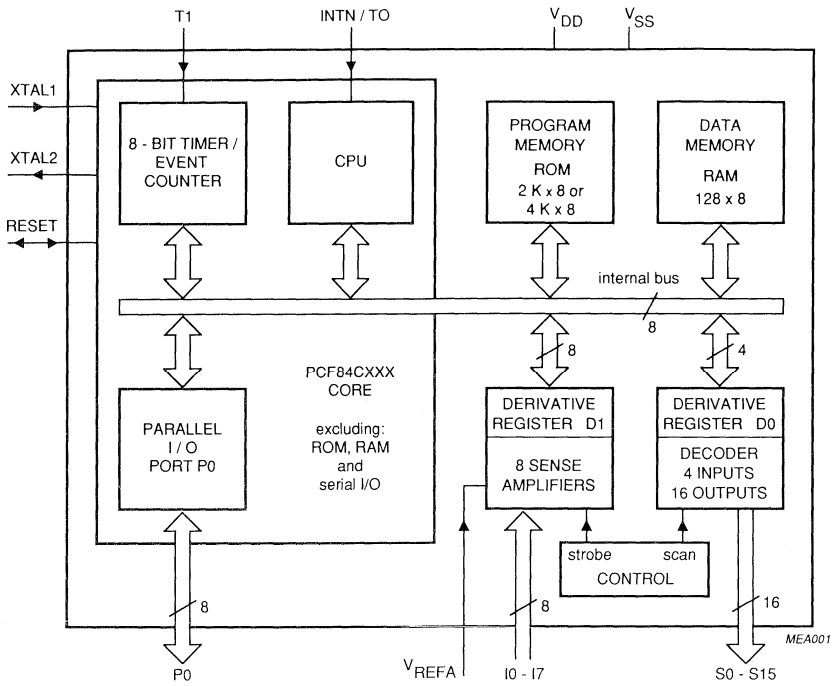


Fig. 1 Block diagram.

GENERAL DESCRIPTION

The PCF84C270, PCF84C271 and PCF84C470 CMOS microcontrollers have been specifically designed for use as keyboard controllers and may be used for efficient scanning, decoding and control of matrix type keyboards with up to 128 keys. The generic term PCF84C270 is used to refer to all three devices except where differences exist.

Key closures are detected by on-chip sense amplifiers. Two additional derivative registers form the interface between the CPU and the on-chip circuitry. One controls up to 16 scan output lines via a 4 to 16 decoder, and the other is used to read and latch the sense amplifier output pulses. This arrangement can accommodate keyboard matrices with up to 128 keys.

DEVICE	ROM	RAM	SCAN OUTPUTS	APPLICATION
PCF84C270	2 K bytes	128 bytes	push-pull	capacitive keyboards
PCF84C271	2 K bytes	128 bytes	open source	mechanical keyboards
PCF84C470	4 K bytes	128 bytes	pull-down resistor	capacitive and mechanical keyboards

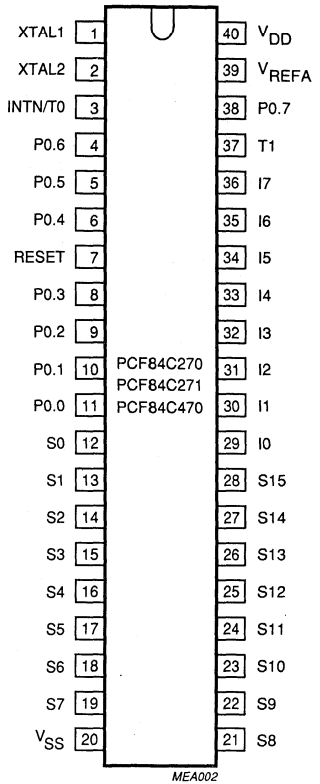


Fig. 2 Pin configuration.

PINNING

SYMBOL	PIN	TYPE	DESCRIPTION
XTAL1	1	I	Oscillator input: input from a crystal which determines the internal oscillator frequency, or input from an external clock generator.
XTAL2	2	O	Oscillator output: output of the inverting amplifier.
INTN/T0	3	I	Interrupt/Test 0: external interrupt input (negative edge triggered)/ test input pin.
P0.0 - P0.7	11, 10, 9, 8, 6, 5, 4, 38	I/O	Port 0: 8-bit quasi-bidirectional I/O port.
RESET	7	I/O	Reset input: active HIGH input used to initialize the microcontroller; also output of the power-on-reset circuit.
S0 - S15	12 - 19, 21 - 28	O	Scanning outputs: active HIGH scanning outputs. Only one of these outputs is active at any time. They are driven by the on-chip 4 to 16 decoder. An active HIGH pulse will appear on a selected line addressed by the four LSBs of derivative register 0.
V_{SS}	20	I	Ground.
I0 - I7	29 - 36	I	Sense inputs: input lines of the sense amplifiers. Signals appearing on each of these lines are input to on-chip comparators. The amplitude of the input signal is compared with an internal reference voltage to determine if a keypad has been pressed.
T1	37	I	Test 1: test input pin. T1 may also be used as an input to the 8-bit timer/event counter.
V_{REFA}	39	I	Reference voltage: reference voltage for the sense amplifier comparator. V_{REFA} is a stable, mask-programmable, internal reference voltage which is relatively insensitive to temperature variations. An external pull-up or pull-down resistor may be connected to this pin to adjust the reference voltage level.
V_{DD}	40	I	Power supply.

FUNCTIONAL DESCRIPTION

Parallel I/O port (P0)

The PCF84C270 has one 8-bit quasi-bidirectional parallel I/O port. When configured as outputs, P0.0 - P0.5 have LED drive capability. Mask options enable two further configurations of P0.0 - P0.5 (see the PCF84CXXX Family Specification). P0.6 and P0.7 are open drain and have high current sink capability.

Keyboard scanning and decoding

To implement keyboard scanning and decoding, the PCF84C270 incorporates two derivative registers: D0 and D1 for scanning and sensing respectively. D0 is used to select one of the 16 scan output lines, and the 8 sense input lines can be read via D1.

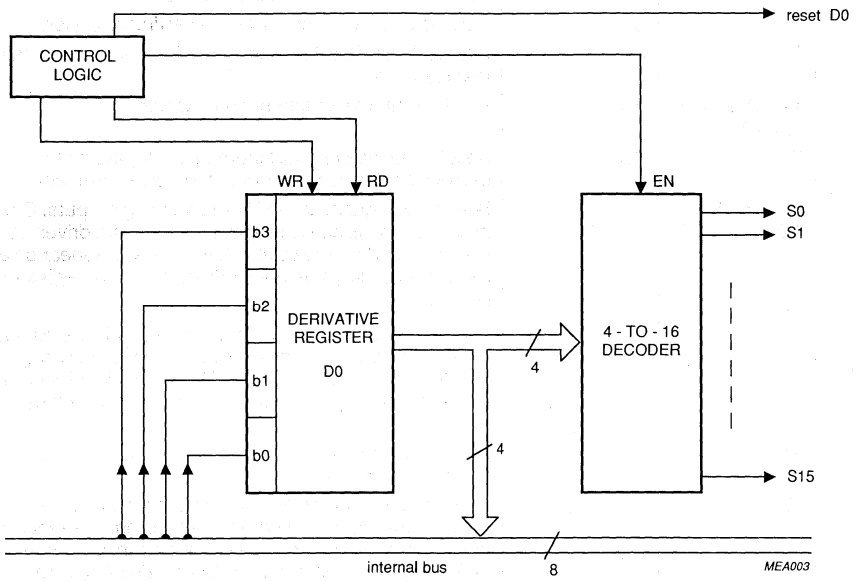


Fig.3 Scan outputs.

Scanning output lines

The scan circuitry is controlled by the 4 LSBs of derivative register D0 (see Fig.3). The four MSBs of D0 are not implemented and are always zero if read. If D0 is written, the 4 MSBs are 'don't care'. The 4 LSBs of D0 are coupled to an on-chip 4 to 16 decoder which drives the 16 scan lines. An active HIGH pulse will be generated on a selected decoder output (S0 - S15) when D0 is loaded with the corresponding 4-bit address (00H - 0FH). Writing 00H to D0 activates scan line S0, writing 01H to D0 activates scan line S1, etc.

When D0 is written to, derivative register D1 is first cleared and a scan pulse will be generated on the scanning output line corresponding to the new contents of D0. Approximately 100 ns after the scan pulse has been output, the sense amplifier strobe is enabled and the data pulse from the sense amplifier is latched into derivative register D1. When the scan outputs are inactive:

- PCF84C270 scan lines are pulled LOW by an N-channel transistor
- PCF84C271 scan lines are high impedance
- PCF84C470 scan lines are pulled LOW by an internal 80 kΩ (typical) pull-down resistor.

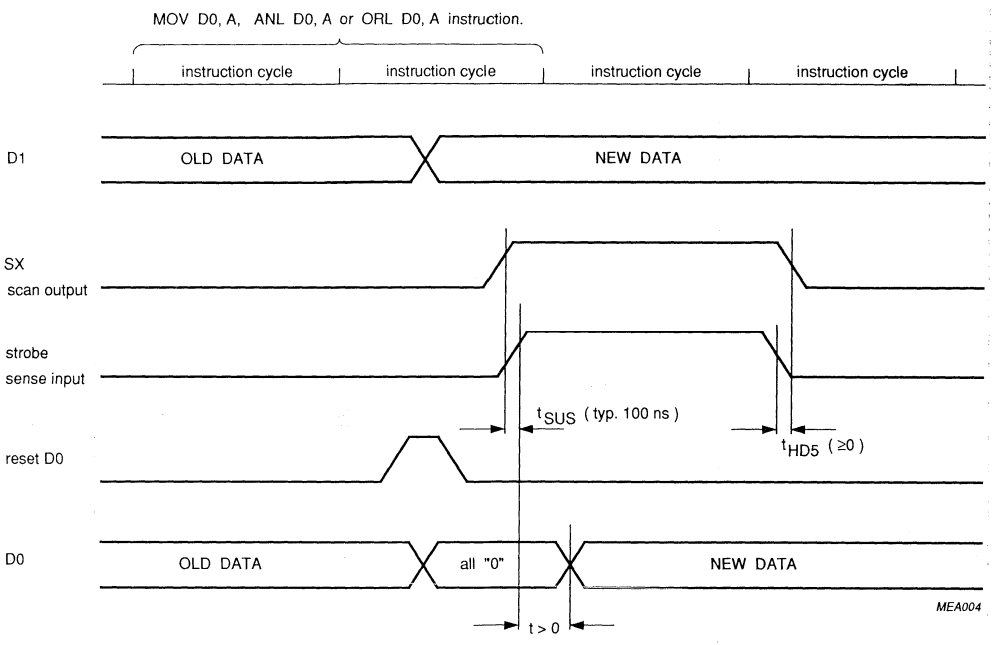


Fig.4 Scan and strobe signal timing.

Sense Input Lines

The PCF84C270 has 8 sense input lines (D0 - D7). The sense input lines and amplifier configuration are shown in Fig.5. Each of the 8 sense lines are input to sense amplifiers which compare the signal appearing on the sense input line to a reference voltage V_{REFA}. Input voltage levels greater than V_{REFA} will produce a logic 1 at the output of the comparator; input voltage levels lower than V_{REFA} will produce a logic 0 at the output of the comparator. During the strobe pulse (see Fig.4), the comparator outputs are latched into derivative register D1. D1 is a read-only register.

Reference voltage

The reference voltage (V_{REFA}) is mask programmable. $V_{REFA} = kV_{DD}$. For the PCF84C270 and PCF84C271, k = 0.067, 0.087, 0.111, 0.134 or 0.222. For the PCF84C271 k may also be set to 0.5. For the PCF84C470, k = 0.020, 0.040, 0.060, ... 0.960 or 0.980. The internal reference voltage is made available externally at the V_{REFA} pin. V_{REFA} may be adjusted by connecting a high value pull-up or pull-down resistor to this pin. To reduce power consumption, transistor TR1 (see Fig.5) is turned off during the Idle and Stop modes.

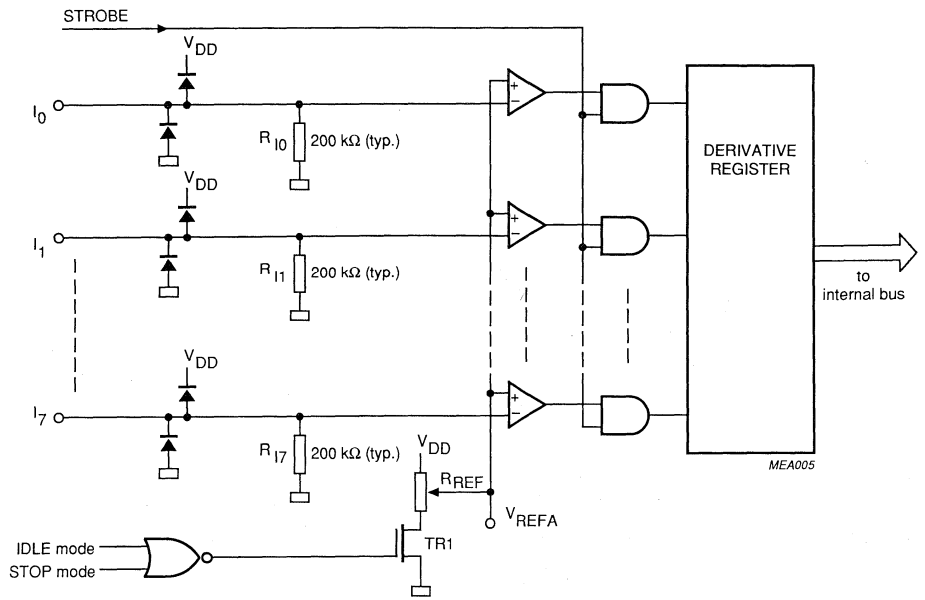


Fig. 5 Sense input lines and amplifier configuration.

Oscillator

External capacitors are not normally required since the on-chip circuitry contains capacitors. If external capacitors are used at XTAL1 and XTAL2, they must be of equal value. The crystal's series resistance and total load capacitance (C0 + wiring + external capacitors) must lie below the curve for the crystal frequency shown in Fig.6.

Reset

See the PCF84CXXX Family Specification. A positive going signal on the reset pin also:

- Clears the latches of derivative register D0 (D0 = 00H)
- Clears the latches of derivative register D1 (D1 = 00H).

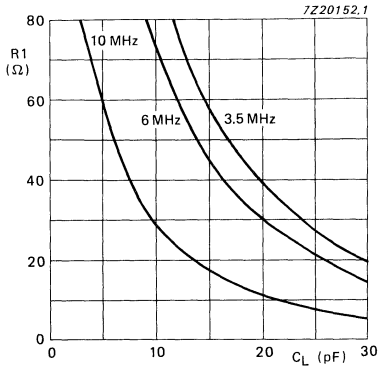


Fig.6 Crystal circuit series resistance (R_L) as a function of load capacitance (C_L).

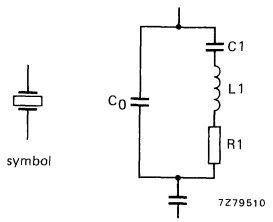


Fig.7 Crystal unit equivalent circuit.

INSTRUCTION SET

The PCF84CXXX Family Specification describes the complete PCF84CXXX instruction set. The PCF84C270 instruction set differs as follows:

EN SI and DIS SI	not available; the PCF84C270 does not have a serial interrupt.
SEL MB1, SEL MB2 and SEL MB3	not available to the PCF84C270 and PCf84C271; these microcontrollers have only 2 K bytes of on-chip ROM.
SEL MB2 and SEL MB3	not available to the PCF84C470; the PCf84C470 has only 4 K bytes of on-chip ROM.

HANDLING

Inputs and outputs are protected against electrostatic discharge in normal handling. However, it is good practice to take normal precautions appropriate to handling MOS devices (see 'Handling MOS devices').

RATINGS

Limiting values in accordance with the Absolute Maximum system (IEC 134).

SYMBOL	PARAMETER	MIN.	MAX.	UNIT
V_{DD}	Supply voltage range	−0.8	8.0	V
V_I	All input voltages	−0.5	V_{DD} + 0.5	V
$\pm I_I$, $\pm I_O$	DC current into any input or output (except P0.6, P0.7 sink current and S0 to S15 source current)	-	10	mA
$-I_{OH}$	S0 to S15 source current	-	16	mA
I_{OL}	P0.6, P0.7 sink current	-	16	mA
P_{tot}	Total power dissipation	-	125	mW
T_{stg}	Storage temperature	−65	150	°C
T_{amb}	Operating ambient temperature range	−40	85	°C
T_j	Operating junction temperature	-	90	°C

DC CHARACTERISTICS

V_{DD} = 2.5 to 5.5 V; V_{SS} = 0 V; T_{amb} = -40 to 85 °C; all voltages with respect to V_{SS} unless otherwise specified.

SYMBOL	PARAMETER	CONDITIONS	MIN.	TYP.	MAX.	UNIT
Supply						
V_{DD}	Operating supply voltage (see Fig.8)		2.5	-	5.5	V
I_{DD}	Operating supply current (see Flg.9)	V_{DD} = 5 V; f_{XTAL} = 10 MHz; note 1	-	1.6	3.2	mA
		V_{DD} = 5 V; f_{XTAL} = 6 MHz; note 1	-	1.0	2.0	mA
		V_{DD} = 3 V; f_{XTAL} = 3.58 MHz; note 1	-	0.4	0.6	mA
I_{DD}	IDLE mode supply current (see Fig.10)	V_{DD} = 5 V; f_{XTAL} = 10 MHz; note 1	-	0.8	1.6	mA
		V_{DD} = 5 V; f_{XTAL} = 6 MHz; note 1	-	0.5	1.0	mA
		V_{DD} = 3 V; f_{XTAL} = 3.58 MHz; note 1	-	0.2	0.4	mA
I_{DD}	STOP mode supply current	V_{DD} = 2.5 V; T_{amb} = 25 °C; note 1 and note 2	-	18	40	µA
		V_{DD} = 2.5 V; T_{amb} = 85 °C; note 1 and note 2	-	-	80	µA
Inputs						
V_{RES}	Power-on reset switching level	-	-	1.5	-	V
V_{IL}	Input LOW voltage (INTN/ T0, T1, XTAL1, RESET)	-	0	-	0.3V_{DD}	V
V_{IH}	Input HIGH voltage (INTN/ T0, T1, XTAL1, RESET)	-	0.7V_{DD}	-	V_{DD}	V
V_{TH}	Sense amplifier input threshold voltage	note 3	V_{REFA} - 50	-	V_{REFA} + 50	mV
V_{REFA}	Sense amplifier reference voltage	note 3	0.9kV_{DD}	-	1.1kV_{DD}	V
R_{IN}	Sense amplifier input impedance	note 3	-	200	-	kΩ
V_{IL}	Input LOW voltage (Port 0)	-	0	-	0.3V_{DD}	V
V_{IH}	Input HIGH voltage (Port 0)	-	0.7V_{DD}	-	V_{DD}	V
$\pm I_{IL}$	Port 0 input leakage current	$V_{SS} < V_{IN} < V_{DD}$	-	-	1	µA
Outputs						
I_{OL}	RESET output sink current		-	7	-	µA
V_{OL}	P0.0 - P0.5 output LOW voltage	V_{DD} = 5 V; I_{OL} = 1.6 mA	-	-	0.4	V
V_{OL}	P0.0 - P0.5 output LOW voltage	V_{DD} = 5 V; I_{OL} = 10 mA	-	-	1.2	V
V_{OL}	P0.6, P0.7 output LOW voltage	V_{DD} = 5 V; I_{OL} = 18 mA	-	-	1.2	V

SYMBOL	PARAMETER	CONDITIONS	MIN.	TYP.	MAX.	UNIT
V_{OH}	P0.0 - P0.5 output HIGH voltage (pull-up)	V_{DD} = 5 V; I_{OH} = 40 µA	0.7V_{DD}	-	-	V
V_{OH}	P0.0 - P0.5 output HIGH voltage (push-pull)	V_{DD} = 5 V; I_{OH} = 1.6 mA	V_{DD} – 0.4	-	-	V
I_{OH}	P0.0 to P0.5 pull-up source current	V_{IN} = V_{SS}	-	-	400	µA
V_{OL}	Scan output LOW voltage (PCF84C270)	V_{DD} = 5 V; I_{OL} = 0.5 mA; note 4	-	-	0.4	V
V_{OL}	Scan output LOW voltage (PCF84C470)	V_{DD} = 5 V; I_{OL} = 10 µA; note 4	-	-	0.4V_{DD}	V
V_{OH}	Scan output HIGH voltage	V_{DD} = 5 V; I_{OH} = 1.6 mA	V_{DD} – 0.4	-	-	V
I_{OL}	Scan output pull-down sink current (PCF84C470)	Output forced to V_{DD}	-	-	150	µA
$\pm I_{SL}$	Scan output leakage current	Scan output not selected	-	-	1	µA

Notes

1. V_{IL} = V_{SS}; V_{IH} = V_{DD}; all outputs and all sense input lines unloaded; all open drain ports connected to V_{SS}.
2. Crystal is connected between XTAL1 and XTAL2; T1 = V_{SS}; INTN = V_{DD}.
3. k is mask programmable.
4. PCF84C271, active HIGH open source outputs.

AC CHARACTERISTICS

V_{DD} = 2.5 to 5.5 V; V_{SS} = 0 V; T_{amb} = -40 to 85 °C. All voltages are with respect to V_{SS} unless otherwise specified.

SYMBOL	PARAMETER	CONDITIONS	MIN.	TYP.	MAX.	UNIT
f_{XTAL}	Oscillator frequency		0.45	-	10	MHz
t_{CY}	Cycle time (= 30 CP)	note 1	3	-	67	µs
t_R	Rise time all active pull-up outputs	V_{DD} = 5 V; C_L = 50 pF.	-	30	-	ns
t_F	Fall time all active pull-down outputs	V_{DD} = 5 V; C_L = 50 pF.	-	30	-	ns
t_{SCAN}	Length of scan pulse		-	39	-	CP
t_{SUS}	Setup time from activation of scan pulse until strobe pulse active (see Fig.4)			100	-	ns
t_{HDS}	Hold time from strobe pulse inactive until scan pulse inactive (see Fig.4)		-	0	-	s

Note

1. 1 clock pulse (CP) = 1/f_{XTAL}

DEVELOPMENT SUPPORT

The PCF84C270 is supported by the tools shown in Table 1.

Table 1 PCF8C270 development tools

TOOLS	TYPE NUMBER
Hardware	
SDS for 84C family without trace	OM1087
SDS for 84C family with trace	OM1087A
Probe for PCF84C270, PCF84C470	OM1077
Probe for PCF84C271	OM1078
Software	
MAC84 X-assembler 8400/3300	OM1088
8400/C00 Macro Assembler*	
SDS Window debugger	OM1089

* This assembler is delivered by 2500AD Software Inc.

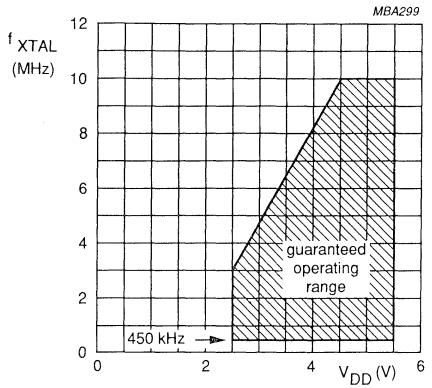
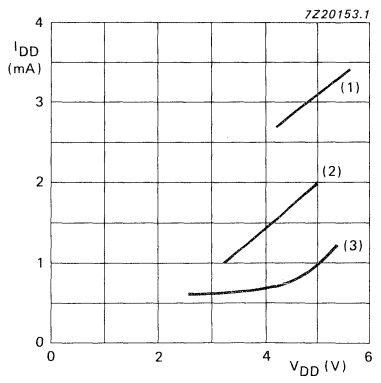
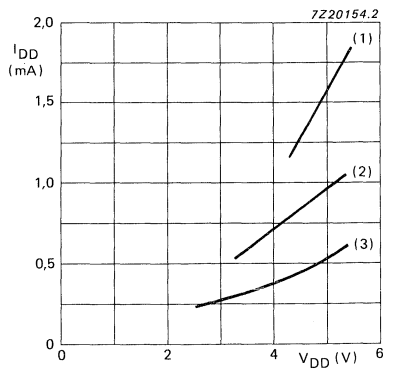


Fig.8 Maximum clock frequency (f_{XTAL}) as a function of the supply voltage (V_{DD}).



(1) f_{XTAL} = 10.0 MHz
(2) f_{XTAL} = 6.0 MHz
(3) f_{XTAL} = 3.58 MHz

Fig.9 Maximum operating supply current (I_{DD}) as a function of the supply voltage (V_{DD}).



(1) f_{XTAL} = 10.0 MHz
(2) f_{XTAL} = 6.0 MHz
(3) f_{XTAL} = 3.58 MHz

Fig.10 Maximum supply current (I_{DD}) in the Idle mode as a function of the supply voltage (V_{DD}).

SINGLE-CHIP 8-BIT MICROCONTROLLER WITH LCD DRIVER

DESCRIPTION

The PCF84C430 microcontroller is a derivative of the 84CXXX family of microcontrollers and is manufactured in CMOS technology. For detailed information see the 84CXXX family specification.

The PCF84C430 contains a PCF84CXX core CPU and is completely software compatible. In addition, the PCF84C430 contains an LCD driver supporting four back planes and a maximum driving capacity of up to 96 segments.

The PCF84C430 has 16 quasi-bidirectional I/O port lines, plus a derivative 8-bit port, a serial I/O interface, a single-level vectored interrupt circuit, an 8-bit timer/event counter and on-board clock oscillator and clock circuits.

Features

- 8-bit CPU, ROM, RAM, I/O in a single 64-lead QFP package
- 4 K ROM bytes
- 128 RAM bytes
- On-chip LCD driver with 24 outputs (max. 96 segments)
- LCD multiplexing rates at 1:1 (static), 1:2, 1:3 and 1:4
- Low-power oscillator for LCD driver during STOP mode
- 25 quasi-bidirectional I/O port lines are configured as two 8-bit ports, a 1-bit port (shared with SDA) and an 8-bit derivative port
- Two test inputs: one of which is also the external interrupt input
- Single-level vectored interrupts: external, timer/event counter, serial I/O
- I^2C-bus hardware interface for serial data transfer on two separate lines
- 8-bit programmable timer/event counter
- Clock frequency 100 kHz to 10 MHz
- Over 80 instructions (based on MAB8048) all of 1 or 2 cycles
- Single supply voltage from 2,5 V to 5,5 V ($V_{SS} \leqslant V_{LCD} < V_{DD}$)
- STOP and IDLE mode
- Power-on-reset circuit
- Operating temperature range: –40 to + 85 °C

PACKAGE OUTLINE

PCF84C430H: 64-lead quad flat-pack; plastic (SOT208).

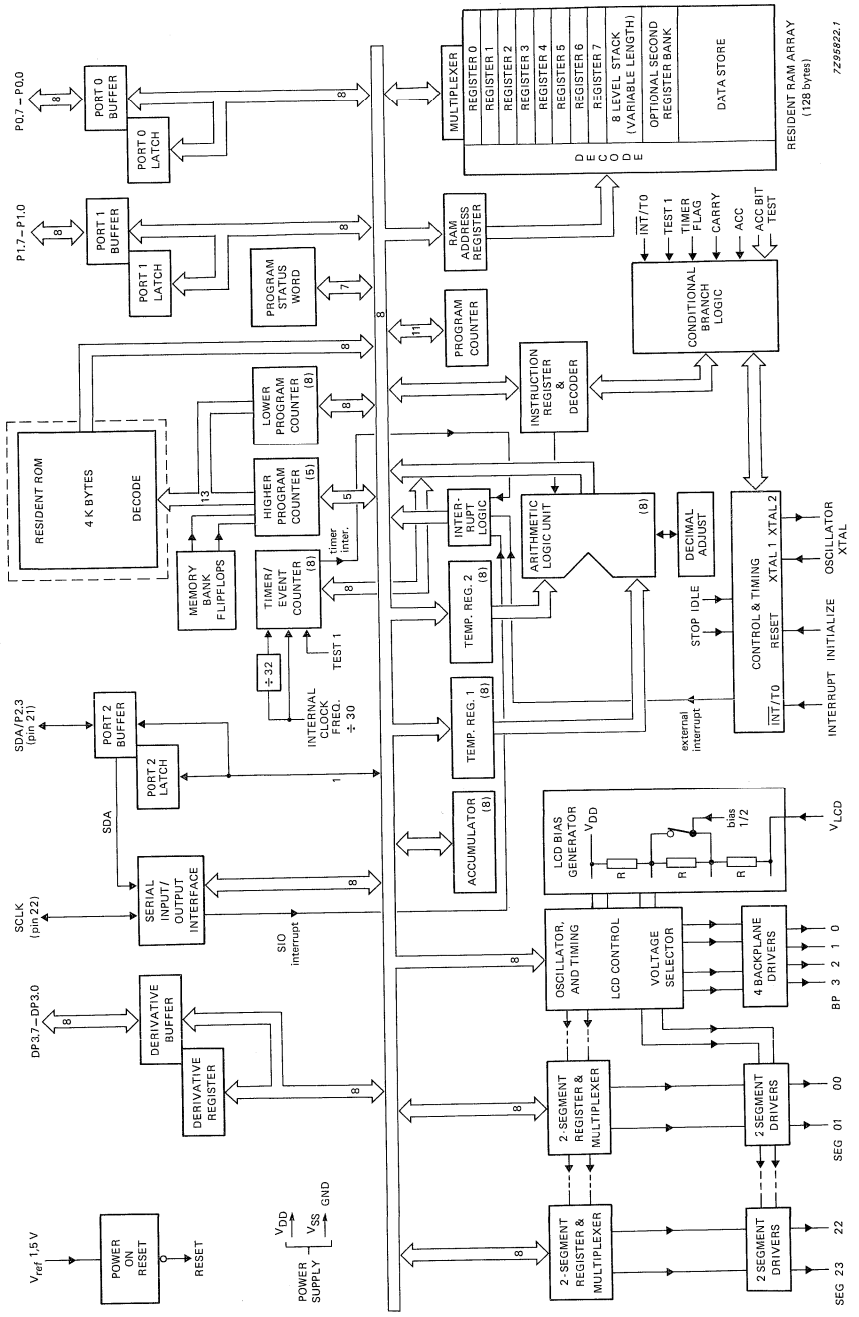


Fig. 1 Block diagram; PCF84C430.

DEVELOPMENT DATA

PINNING

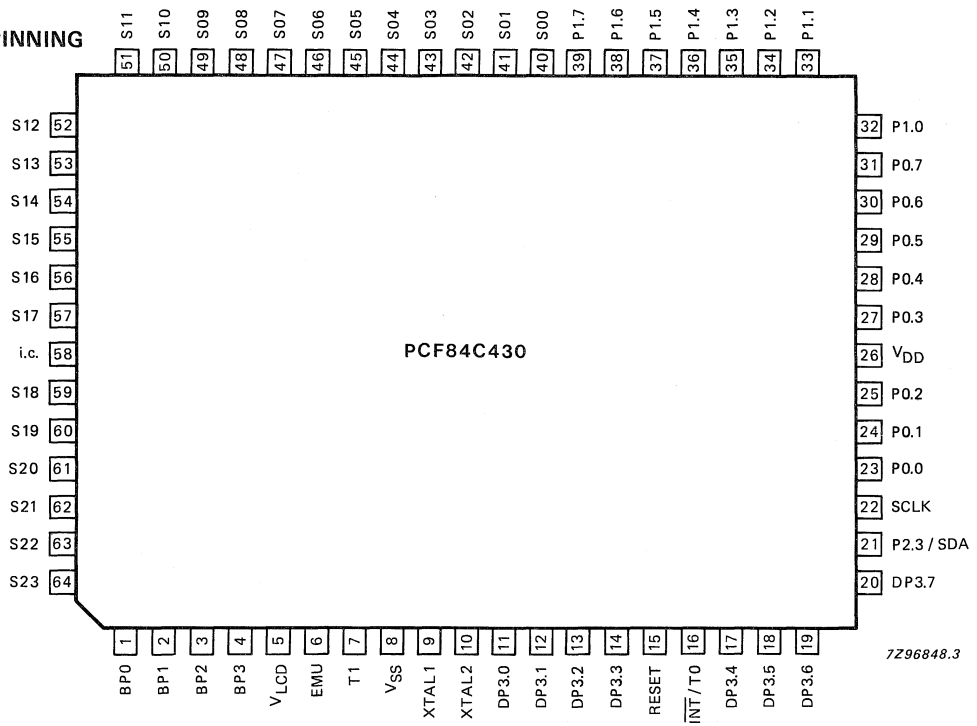


Fig. 2 Pin designation of the PCF84C430.

PIN DESIGNATION

pin	symbol	type	function
1-4	BP0-BP3	O	**LCD:** backplane outputs.
5	V_{LCD}		**LCD:** supply ground level.
6	EMU	I	**EMU pin:** to be grounded for normal operation.
7	T1	I	**Test 1:** test input pin, directly tested by conditional branch instructions JT1 and JNT1. T1 also functions as an input to the 8-bit timer/event counter, using the STRT CNT instruction.
8	V_{SS}		**Ground:** circuit earth potential.
9	XTAL1	I	**Oscillator input:** crystal which determines the internal oscillator frequency or the external clock generator.
10	XTAL2	I/O	Connection to other side of oscillator.
11-14	DP3.0-DP3.3	I/O	**Derivative port 3:** 8-bit parallel port LSBs.
15	RESET	I/O	**Reset input:** used to initialize the processor (active HIGH), or output of power-on-reset circuit.
16	$\overline{INT}$/T0	I	**Interrupt/Test 0:** external interrupt input (sensitive to negative going edge)/test input; when used as a test input directly tested by conditional branch instructions JT0 and JNT0.
17-20	DP3.4-DP3.7	I/O	**Derivative port 3:** 8-bit parallel port MSBs.
21	SDA/P23	I/O	**Serial data:** input/output in serial I/O mode. If not selected as serial data pin, P23 functions as a quasi-bidirectional I/O line.
22	SCLK	I/O	**Serial clock:** bidirectional clock for serial I/O.
23-25 27-31	P0.0-P0.7	I/O	**Port 0:** 8-bit quasi-bidirectional I/O port.
26	V_{DD}		**Power supply:** 2,5 V to 5,5 V.
32-39	P1.0-P1.7	I/O	**Port 1:** 8-bit quasi-bidirectional I/O port.
40-57 59-64	S00-S23	O	**LCD** segment driver outputs.

FUNCTIONAL DESCRIPTION

Program memory

The program memory consists of 4 K bytes, in a read-only memory (ROM).
Each location is directly addressable by the program counter. The memory is mask-programmed at the factory. Figure 3 shows the program memory map.

Four program memory locations are of special importance:

- Location 0; contains the first instruction to be executed after the processor is initialized (RESET)
- Location 3; contains the first byte of an external interrupt service subroutine
- Location 5; contains the first byte of a serial I/O and interrupt service subroutine
- Location 7; contains the first byte of a timer/event counter interrupt service subroutine

Program memory is arranged in banks of 2 K bytes, which are selected by SEL MB instructions. The program memory is further divided into location 'pages', each of 256 bytes. This latter division applies only for conditional branches. Memory bank boundaries can be crossed only by using the unconditional branch instructions after the appropriate memory bank has been selected. A CALL instruction can transfer control to a subroutine on any 'page'; RET and RETR instructions can transfer control from a subroutine back to the main program.

Data memory

Data memory consists of 128 bytes, random-access data memory (RAM).
All locations are indirectly addressable using RAM pointer registers; up to 16 designated locations are directly addressable. Memory also includes an 8-level program counter stack addressed by a 3-bit stack pointer. Figure 4 shows the data memory map.

Working registers

Locations 0 to 7 are designated as working registers, directly addressable by the direct register instructions. Ease of addressing, and a minimum requirement of instruction bytes to manipulate their contents, makes these locations suitable for storing frequently addressed intermediate results. This bank of registers can be selected by the SEL RB0 instruction.

Executing the select register bank instruction SEL RB1, designates locations 24 to 31 as working registers, instead of locations 0 to 7, and these are then directly addressable. This second bank of working registers may be used as an extension of the first or reserved for use during interrupt service subroutines saving the first bank for use in the main program. If the second bank is not used, locations 24 to 31 may serve as general purpose RAM.

The first locations of each bank contain the RAM pointer registers R0, R1, R0' and R1', which indirectly address all RAM locations.

All RAM locations make efficient program loop counters when used with the decrement register and test instruction DJNZ.

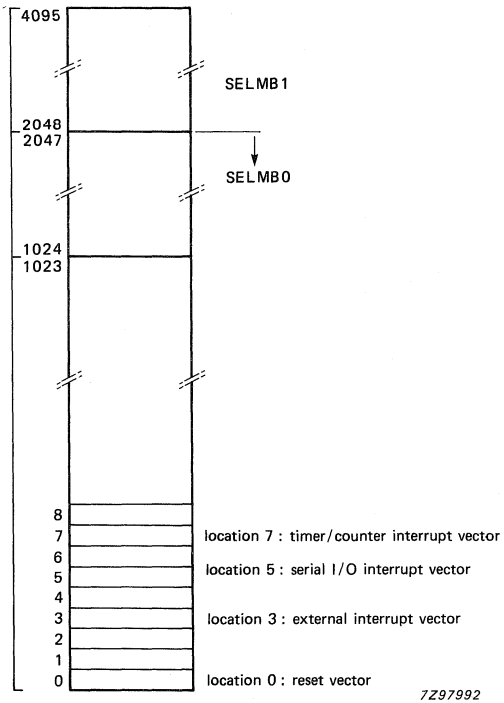


Fig. 3 Program memory map.

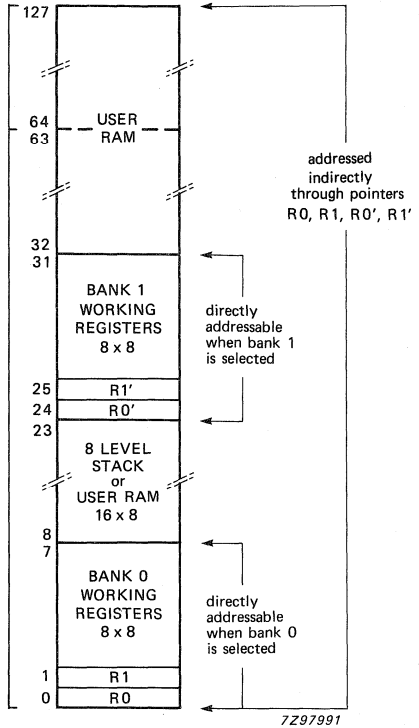


Fig. 4 Data memory map.

Program counter stack

Locations 8 to 23 may be designated as an 8-level program counter stack (2 locations per level), or as general purpose RAM. The program counter stack (Fig. 5) enables the processor to keep track of the return addresses and status generated by interrupts or CALL instructions by storing the contents of the program counter prior to servicing the subroutine. A 3-bit stack pointer determines which of the eight register pairs of the program counter stack will be loaded with the next generated return address.

The stack pointer, when initialized to 000 by RESET, points to RAM locations 8 and 9. On the first subroutine CALL or interrupt, the contents of the program counter and bits 4, 6 and 7 of the program status word (PSW) are transferred to locations 8 and 9. The stack pointer increments by one and points to locations 10 and 11 ready for another CALL. Because an address may be up to 13 bits long, two bytes must be used to store each address.

At the end of a subroutine, which is signalled by a return instruction (RET or RETR), the stack pointer decrements by one and the contents of the register pair on top of the stack are transferred to the program counter. The saved PSW bits are transferred to the PSW only by the RETR instruction.

If not all 8 levels of subroutine and interrupt nesting are used, the unused portion of the stack may be used as any other indirectly addressable RAM locations.

FUNCTIONAL DESCRIPTION (continued)

Program counter stack (continued)

Nesting of subroutines within subroutines can continue up to 8 times without overflowing the stack. If overflow does occur the deepest address stored (locations 8 and 9) will be overwritten and lost since the stack pointer overflows from 111 to 000. It also underflows from 000 to 111.

The value of the saved contents of the program counter is different for an interrupt CALL compared to a normal CALL to subroutine. With an interrupt CALL, the program counter return address is saved; with a subroutine CALL, the saved program counter value is one less than the program counter return address.

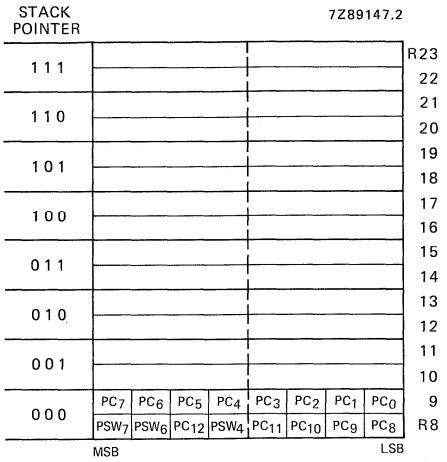


Fig. 5 Program counter stack.

IDLE and STOP modes

Idle mode

When the microcontroller enters the IDLE mode via the IDLE instruction (01 H) the oscillator, timer/counter and serial I/O are kept running. The microcontroller exits from the IDLE mode by one of three interrupts if they are enabled, or by activating a RESET. If the interrupt is not enabled the processor will remain in the IDLE mode. An active signal on the RESET pin restarts the microcontroller and a normal RESET sequence is executed (see Fig. 6).

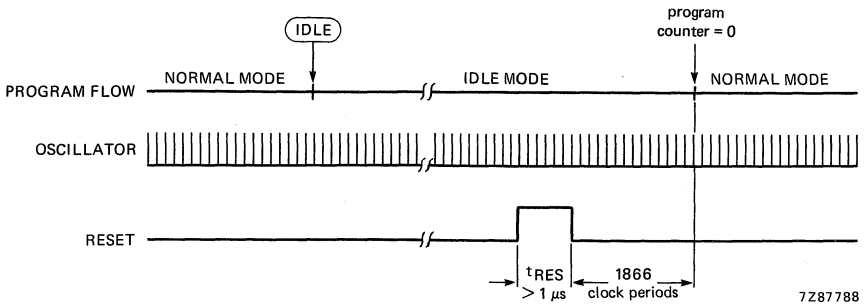


Fig. 6 Exit from IDLE mode via a RESET.

An active signal coming from an enabled interrupt causes the execution of the normal interrupt routine since normal interrupt scanning is still being carried out. A HIGH-to-LOW transition on the external interrupt pin ($\overline{INT}$/T0) reactivates the microcontroller. A LOW level applied to $\overline{INT}$/T0 will reactivate the microcontroller only in the STOP mode. Thus, if $\overline{INT}$/T0 was LOW before the microcontroller entered the IDLE mode, it must go HIGH before the microcontroller can be reactivated (see Fig. 7.).

DEVELOPMENT DATA

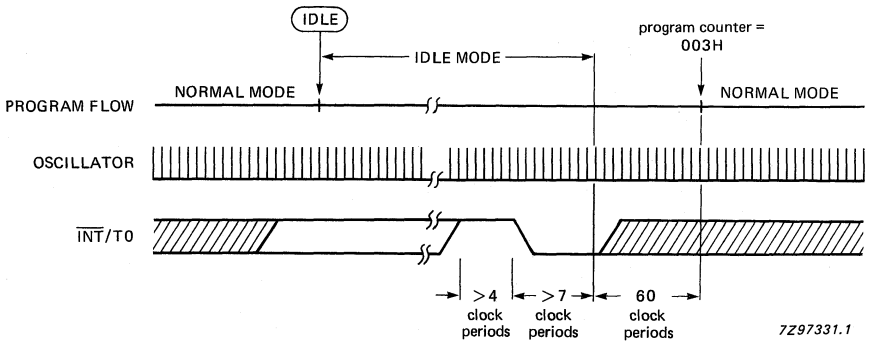


Fig. 7 Exit from IDLE mode via an interrupt.

Wake-up from the IDLE mode is ensured when $\overline{INT}$/T0 is HIGH for at least 4 CP (clock periods) followed by a LOW for 7 CP. After the initial forced CALL 003 H operation (60 CP) the program continues with the external interrupt service routine. During IDLE mode operation, the address of the instruction immediately following that which caused the processor to enter the IDLE mode is present on the address bus.

STOP mode

The microcontroller enters the STOP mode via the STOP instruction (22H). The oscillator is switched off but internal status of the CPU, RAM contents and the state of I/O ports are unaffected. The microcontroller can be brought-out of the STOP mode by an active signal at the external interrupt input or by an external RESET signal. When one of these two signals is applied an internal delay of 1866 CP is provided to ensure that all internal clocks are operating correctly before restart (see Fig. 8). Note: the start-up time of a crystal oscillator is measured in milliseconds, and the 1866 CP count begins after this start-up time.

If the microcontroller exits from the STOP mode by activating RESET, a normal RESET sequence is executed.

FUNCTIONAL DESCRIPTION (continued)

STOP mode (continued)

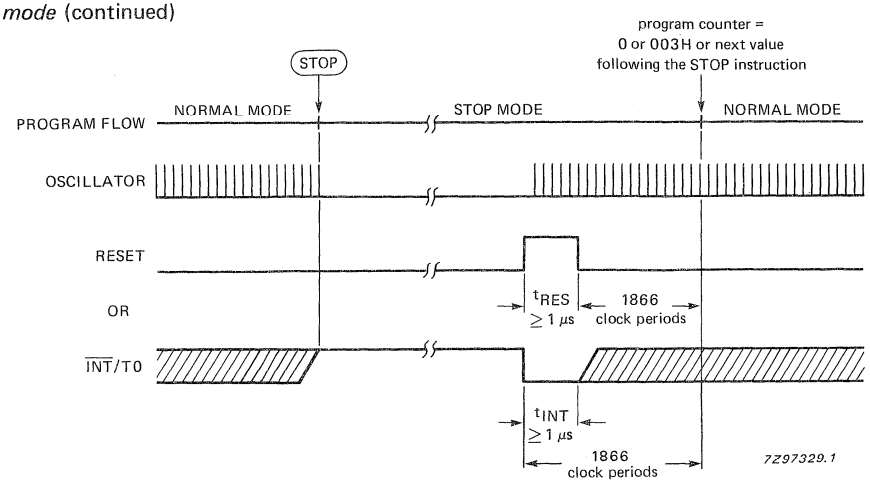


Fig. 8 Entering and exiting the STOP mode.

If the microcontroller exits the STOP mode by pulling the external interrupt input pin LOW, an interrupt sequence is executed only if the external interrupt is enabled. In this event the microcontroller resumes the normal program sequence after returning from the interrupt routine, as in the normal mode. If the interrupt is not enabled, it continues the normal program sequence, executing the instruction following the STOP instruction.

The microcontroller is restarted by a LOW level applied at the $\overline{INT}$/T0 pin.
Note: when leaving the STOP mode via an interrupt, a further instruction in the main program series is executed prior to entering the interrupt routine.

When the $\overline{INT}$/T0 level is active during the STOP instruction then no STOP is executed.

A LOW level on the external interrupt input of at least 1 μs will cause the microcontroller to exit the STOP mode. During the STOP mode, the address of the instruction immediately following the last STOP instruction is present on the address bus.

I/O facilities

The PCF84C430 has 25 quasi-bidirectional I/O lines arranged as:

- Port 0 — parallel port of 8 lines (P0.0 to P0.7)
- Port 1 — parallel port of 8 lines (P1.0 to P1.7)
- Port 2 — I/O port of 1 line (SDA/P2.3) active when not selected as serial I/O
- Port 3 — a derivative 8-bit parallel port (DP3.0 to DP3.7)

In addition 4 specialized I/O lines are comprised:

- SCLK — serial I/O clock line
- SDA/P2.3 — serial I/O data line
- $\overline{INT}$/T0 — external interrupt and test input. When used as a test input it can be directly tested by conditional branch instructions JT0 and JNT0
- T1 — test input which can alter program sequences when tested by conditional jump instructions JT1 and JNT1. T1 also functions as an input to the 8-bit timer/event counter.

Parallel ports

All parallel ports can be used as outputs or inputs, their structure is quasi-bidirectional.
Output data written to a port is latched and remains unchanged until rewritten.
Input data is not latched and so must be present until read by an input instruction.

Input lines are fully CMOS compatible, output lines can drive one TTL or CMOS load.

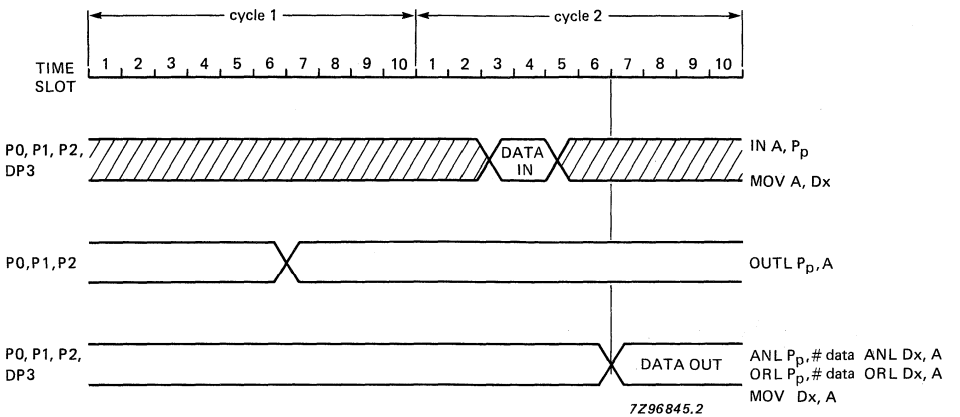


Fig. 9 Shows the timing diagram for all ports using IN, OUTL, ANL and ORL instructions. For the OUTL instruction data changes on time slot 7 of cycle 1. For the MOV, ANL and ORL instructions, the ports change on time slot 7 of cycle 2.

Fig. 10 shows the quasi-bidirectional I/O interface with push-pull output and switched pull-up current source.
Each line is pulled up to V_{DD} via a constant current source (TR4), which is enabled via TR3 whenever one of the two output latches contains a logic 1. This current source is sufficient for a TTL HIGH level, yet can be pulled LOW by an external CMOS device, thus allowing the same pin to be used for both input and output.

When a logic 1 is written to the line for the first time (MQ = 1, SQ = 0), TR2 is switched on for the duration of the internal write pulse (one oscillator period), to provide a fast transition from logic 0 to logic 1. Subsequent writing of a logic 1 to the port lines will not switch TR2 on. This prevents unnecessary current through external components connected to the port lines of the same port which might be in the input mode and also connected to ground.

When a logic 0 is written to the line, TR3 switches off the current source. Current sinking capability is provided by TR1, which is now switched on. When used as an input, a logic 1 must first be written to the line, otherwise TR1 will remain low impedance.

FUNCTIONAL DESCRIPTION (continued)

Parallel ports (continued)

The PCF84C430 family offers the possibility to select individually 24 of the 25 parallel port pins (not P2.3)* from the following mask options:

Option 1 – STANDARD PORT; quasi-bidirectional I/O with switched pull-up current source of 100 μA (typ.) and P-channel booster transistor TR2.
TR2 is only active during 1 clock cycle (Fig. 10).

Option 2 – OPEN DRAIN; quasi-bidirectional I/O with only an N-channel open drain output. Application as an output requires connection of an external pull-up resistor (Fig. 11). When an open drain port is unused it must be connected to V_{SS}.

Option 3 – PUSH-PULL OUTPUT; drive capability of the output is 1,6 mA (min.) at V_{DD} = 5 V in both polarities. To avoid a large current flowing through the output transistors during the input mode, these push-pull pins must only be used as outputs (Fig. 12).
Note: the port latch may be read back using the IN A,pp instruction.

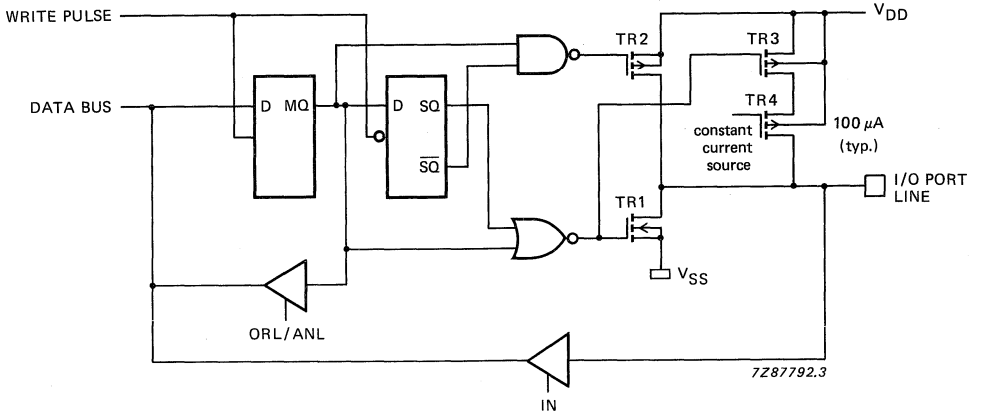


Fig. 10 Standard output with switch pull-up current source.

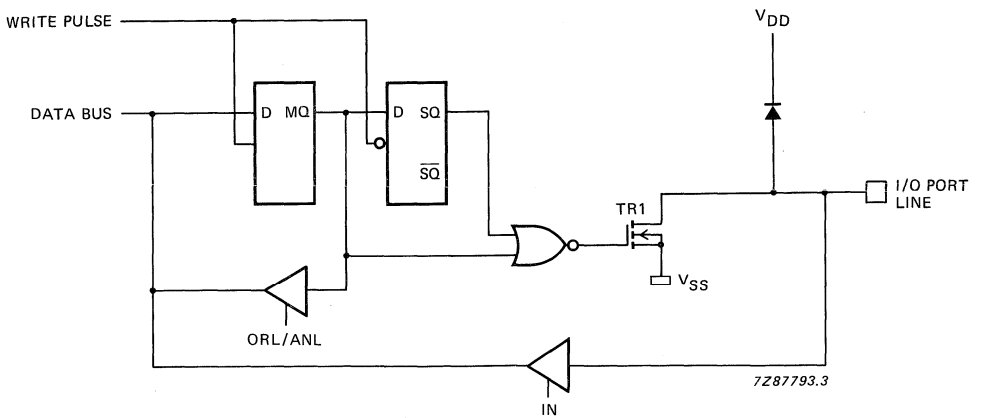


Fig. 11 Open drain output.

* P2.3 is always open drain.

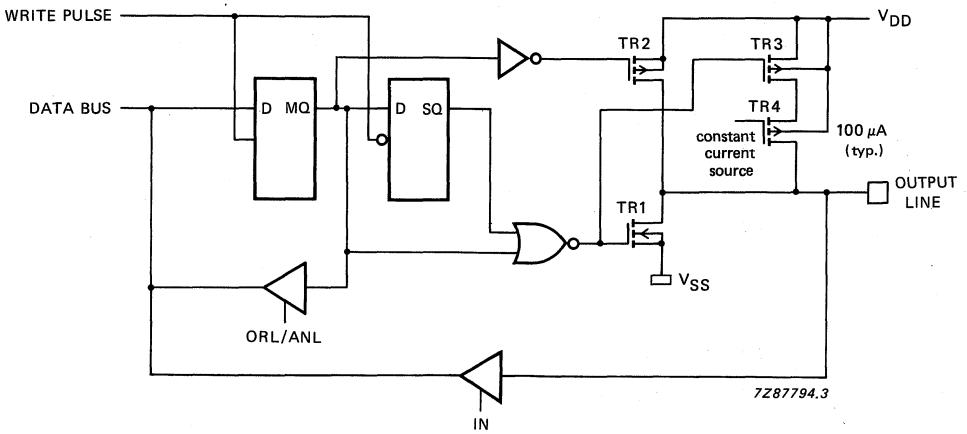


Fig. 12 Push-pull output.

Derivative I/O addresses

For extension of the I/O facilities, instructions MOV A,Dx, MOV Dx,A, ANL Dx,A and ORL Dx,A have been added to the instruction set. The derivative register address is the second byte of the instruction. The address map for the derivative registers is given in Table 1.

Table 1 address map of PCF84C430 derivative registers

Dx	R/W	bits	description
4	R/W	8	PR3; derivative I/O port 3 latch
5	R	8	PR3; derivative I/O port 3 input pins
16	R/W	8	LCDC; LCD control register
17-28	R/W	8	LCDB1-LCDB12; 12 consecutive segment data bytes

Serial I/O (SIO)

The PCF84C430 has an on-chip serial I/O interface that supports I^2C-bus communication. Whereas a normal microcontroller must regularly monitor the serial data bus for the presence of data, the serial I/O interface detects, receives and converts the serial data stream into parallel format without interrupting the execution of the current program. An interrupt is sent to the PCF84C430 only when a complete byte is received. It then reads the data byte in one instruction.

During transmission, the serial I/O interface performs parallel to serial conversion and subsequent serial output of the data. The microcontroller is only interrupted in the execution of its programmed tasks when a complete byte has been transmitted.

FUNCTIONAL DESCRIPTION (continued)

Serial I/O (SIO) (continued)

The design of the PCF84C430 serial I/O system allows any number of devices from PCF85XX family (clips) to be connected via the two-line serial bus. The ability of any devices to communicate, without interrupting the operation of any other devices on the bus, is an outstanding attribute of the system. This is achieved by allocating a specific 7-bit address to each device and providing a system whereby a device reacts only to a message prefixed with its own address or the 'general CALL' address. Address recognition is performed by the interface hardware so that operation of the microcontroller need only be interrupted when a valid address has been received. This saves significant processing time and memory space compared with a conventional microcontroller employing a software serial interface. When the addressing facility is not required, for instance in a system with only two microcontrollers, direct data transfer without addressing can be performed. In multi-master systems, an automatically, invoked arbitration procedure prevents two or more devices from continuing simultaneous transmission.

In NORMAL (running) and IDLE mode, the serial I/O logic remains active; its internal system clock will be switched off when there is no activity on the serial bus.

After execution of the STOP instruction, the oscillator of the PCF84C430 is switched off. This means that the serial I/O logic will remain in the state it was at the occurance of the STOP instruction. To avoid "bus block" problems and to assure correct start-up of the bus after exit from the STOP mode, the user should disable the serial logic (ESO = 0) prior to the execution of the STOP instruction. This must be carried out only when the PCF84C430 has finished a serial data transfer.

After a negative-going RESET signal, the first 30 clock pulses of the 1866-pulse initialization phase set P2.3/SDA and SCLK to HIGH. When P2.3/SDA or SCLK are not used they must be connected to V_{SS}.

Serial I/O interface

Figure 13 shows the serial I/O interface. The clock line of the serial bus has exclusive use of pin 22 (SCLK) while the data line shares pin 21 (SERIAL DATA) with the I/O line P2.3 of port 2. When the serial I/O is enabled, P2.3 is disabled as a parallel port line; (P2.3 and SCLK only open drain).

The microcontroller and interface communicate via the internal microcontroller bus and the Serial Interrupt Request line. Data and information controlling the operation of the interface are stored in four registers:

- Data shift register (S0)
- Serial I/O interface status word (S1)
- Serial clock control word (S2)
- Address register (S0')

Data shift register (S0)

Register S0 converts serial data to parallel format and vice versa. A pending interrupt is generated only after a complete byte has been transmitted, or after a complete data byte, specific address or 'general CALL' address has been received. The most significant bit is transmitted first.

Serial I/O interface status word (S1)

Register S1 provides information concerning the state of the interface and stores information from the microcontroller. Bits 0 to 3 are duplicated: control bits in these positions can only be written by the microcontroller, while status bits can only be read.

MST and TRX (see Table 2)

These bits determine the operating mode of the serial I/O interface.

Table 2 Operating modes of the serial I/O interface

MST	TRX	operating mode
0	0	slave receiver
1	0	master receiver
0	1	slave transmitter
1	1	master transmitter

BB: Bus Busy.

This is the flag which indicates the status of the bus.

PIN: Pending Interrupt Not

PIN = '0' indicates the presence of a pending interrupt, which will cause a Serial Interrupt Request when the serial interrupt mechanism is enabled.

ESO: Enable Serial Output

The ESO flag enables/disables the serial I/O interface: ESO = 1 enables, ESO = 0 disables. ESO can only be written by software.

BC0, BC1 and BC2

Bits BC0, BC1 and BC2 indicate the number of bits received or transmitted in a data stream. These bits can only be written by software.

AL: Arbitration Lost

The arbitration lost flag is set by hardware when the serial I/O interface, as master transmitter, loses a bus arbitration procedure.

AAS: Addressed As Slave

This flag is set by hardware when the interface detects either its own specific address or the 'general CALL' address as the first byte of a transfer and the interface has been programmed to operate in the address recognition mode.

AD0: Address Zero

This flag is set by hardware after detection of the 'general CALL' address when the interface is operating in the address recognition mode.

LRB: Last Received Bit

This contains either the last data bit received or, for a transmitting device in the acknowledgement mode, the acknowledgement signal from the receiving device.

Bits AL, AAS, AD0 and LRB can only be read by software.

FUNCTIONAL DESCRIPTION (continued)

Serial I/O interface (continued)

Serial clock control word (S2)

Bits 0 to 4 of the clock control register S2 are used to set the frequency of the serial clock signal. When a 6 MHz crystal is used, the frequency of the serial clock can be varied between 154 kHz and 1 kHz (see Table 3).
An asymmetrical clock with a HIGH-to-LOW ratio of 3:1 can be generated using bit 5. The asymmetrical clock allows a microcontroller more time per clock period for sampling the data line, making the timing of this action less critical. Bit 6 can be used to activate the acknowledge mode of the serial I/O.
S2 is a write only register.

Address register

The address register contains the 7-bit address back-up latches and the bit (ALS) used to enable/disable the address recognition mode. The address register can be written using the MOV S0, A and MOV S0, # data instructions, but only when ESO = 0 .

Serial I/O interrupt logic

An EN SI instruction enables and a DIS SI instruction disables the interrupt logic. When the logic is enabled, a pending interrupt results in a serial I/O interrupt to the processor, causing a CALL to location 5 in the ROM. When disabled, the presence of an interrupt is still indicated by PIN in S1, allowing the interrupt to be serviced. However, vectored interrupt will not occur.

DEVELOPMENT DATA

Table 3 SIO clock pulse frequency control when using a 6 MHz and a 10 MHz crystal

hexadecimal S20-S24 code	divisor	f_{XTAL} (6 MHz) f_{SCLK} (kHz)▲	f_{XTAL} (10 MHz) f_{SCLK} (kHz)▲
0	not allowed		
1	39	*154	*256
2	45	*133	*222
3	51	*118	*196
4	63	95	*159
5	75	80	*133
6	87	69	*115
7	99	61	*101
8	123	49	81
9	147	41	68
A	171	35	58
B	195	31	51
C	243	25	41
D	291	21	34
E	339	18	29
F	387	16	26
10	483	12	21
11	579	10	17
12	675	8,9	15
13	771	7,8	13,4
14	963	6,2	10,4
15	1155	5,2	8,7
16	1347	4,5	7,4
17	1539	3,9	6,5
18	1923	3,1	5,2
19	2307	2,6	4,3
1A	2691	2,2	3,7
1B	3075	2,0	3,3
1C	3843	1,6	2,6
1D	4611	1,3	2,2
1E	5379	1,1	1,9
1F	6147	1,0	1,6

* Not permitted for I^2C operation.

▲ The maximum clock frequency in the I^2C systems is 100 kHz.

FUNCTIONAL DESCRIPTION (continued)

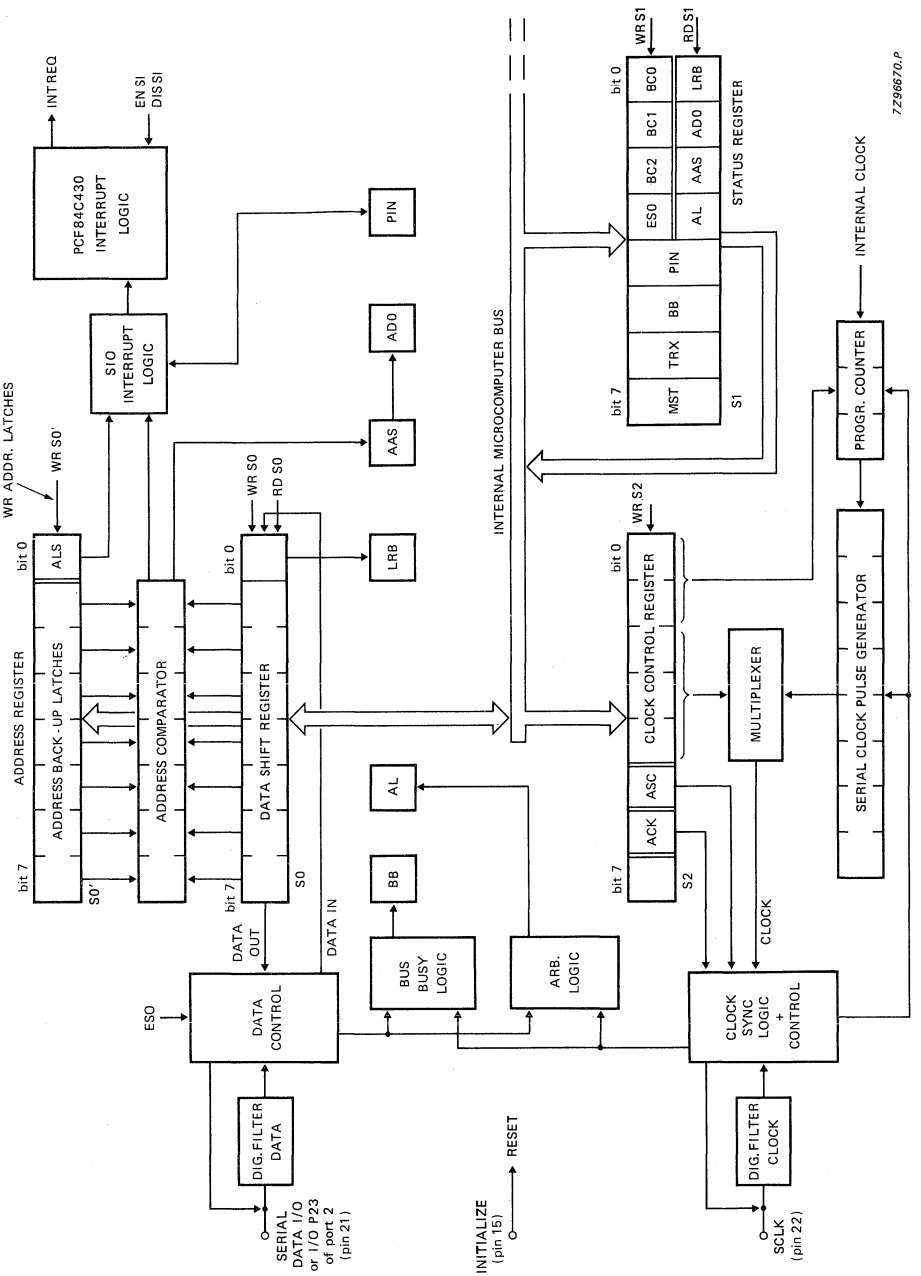


Fig. 13 Serial I/O interface.

Interrupts

Upon entering an interrupt routine, the contents of the program counter and bits 4, 6 and 7 of the PSW are saved in the program counter stack. The contents of the accumulator must be saved by user software. Interrupt acknowledgement may be carried out by software via I/O ports. All interrupt routines must reside in memory bank 0; the SEL MB instructions may not be used within an interrupt routine. An interrupt routine can only be terminated by the RETR (return and restore) instruction. During an interrupt routine, further subroutine calls must be terminated using the RET instruction. Using the RETR instruction to terminate such a nested subroutine would terminate the interrupt routine prematurely.

External interrupt

When the external interrupt is enabled and no interrupt routine is in progress, a HIGH-to-LOW transition on the $\overline{\text{INT}}$/T0 pin sets the External Interrupt Flag (EIF) and invokes the external interrupt routine by forcing a CALL to location 3*. The program counter points to the external interrupt vector address (003 H) between 2,6 and 3,6 machine cycles after the transition occurs. Interrupt latency will depend upon the instruction that is being executed when the transition occurs. If an interrupt routine is already in progress, an external interrupt request is stored in the External Interrupt flag (EIF). When the external interrupt is disabled the request is still latched into the digital filter. Execution of a DIS I instruction cancels a stored interrupt request by clearing both the digital filter and the (EIF).

Another external interrupt can be created by enabling the timer/event counter interrupt and loading FFH into the counter (one less than overflow). The STRT CNT instruction is then executed in user software, this enables the event counter mode and a LOW-to-HIGH transition on the T1 input will then initiate an interrupt subroutine and invoke a call to the timer/counter interrupt vector location 7.

SIO Interrupt

An interrupt request from the SIO hardware will set the PIN flag to its active LOW state. This action is fully independant of the Enable SIO interrupt flag. When the SIO interrupt is enabled and no interrupt routine is in progress, the PIN flag at active LOW will invoke the SIO interrupt routine by forcing a CALL to program memory location 5. After the SIO interrupt is initiated, the PIN flag is **not** automatically set back to a HIGH state, this must be done as part of the interrupt subroutine.

Timer/counter Interrupt

When no interrupt routine is in progress and the timer/counter is enabled, a timer/counter overflow sets the Timer interrupt flag (TIF). This initiates the timer interrupt routine by forcing a CALL to program memory location 7**. If an interrupt routine is in progress, the interrupt request is stored in the (TIF) only if the timer interrupt has been enabled. Execution of a DIS TCNTI instruction deletes a previously stored interrupt request. The timer flag (TF) is set every time the timer/counter overflows and is not automatically reset after the timer counter routine is called. It can only be cleared by either the JTF or JNTF or by a hardware RESET.

Interrupt Priority

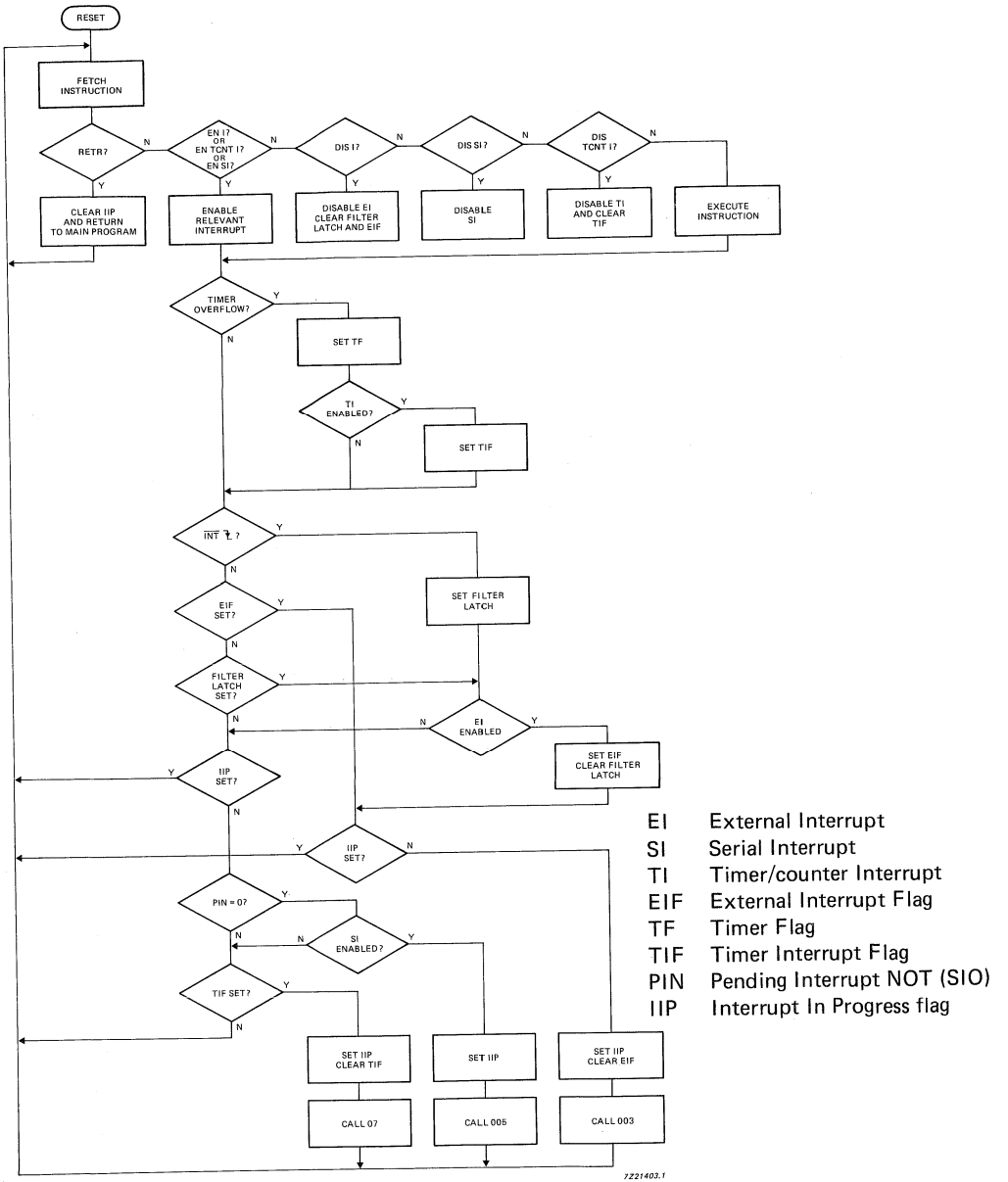
If simultaneous interrupts occur, their priority is as follows:

- External (highest)
- SIO
- Timer/Counter (lowest)

An interrupt routine can only be interrupted by a hardware RESET and cannot be interrupted by other interrupts (which will be latched). When the interrupt routine is terminated by the RETR instruction, at least one instruction of the main program will be executed before another interrupt routine is entered.

* This CALL clears the EIF flag.
** This CALL clears the TIF flag.

FUNCTIONAL DESCRIPTION (continued)



EI	External Interrupt
SI	Serial Interrupt
TI	Timer/counter Interrupt
EIF	External Interrupt Flag
TF	Timer Flag
TIF	Timer Interrupt Flag
PIN	Pending Interrupt NOT (SIO)
IIP	Interrupt In Progress flag

Fig. 14(a) Flow chart illustrating the interrupt handling sequence.

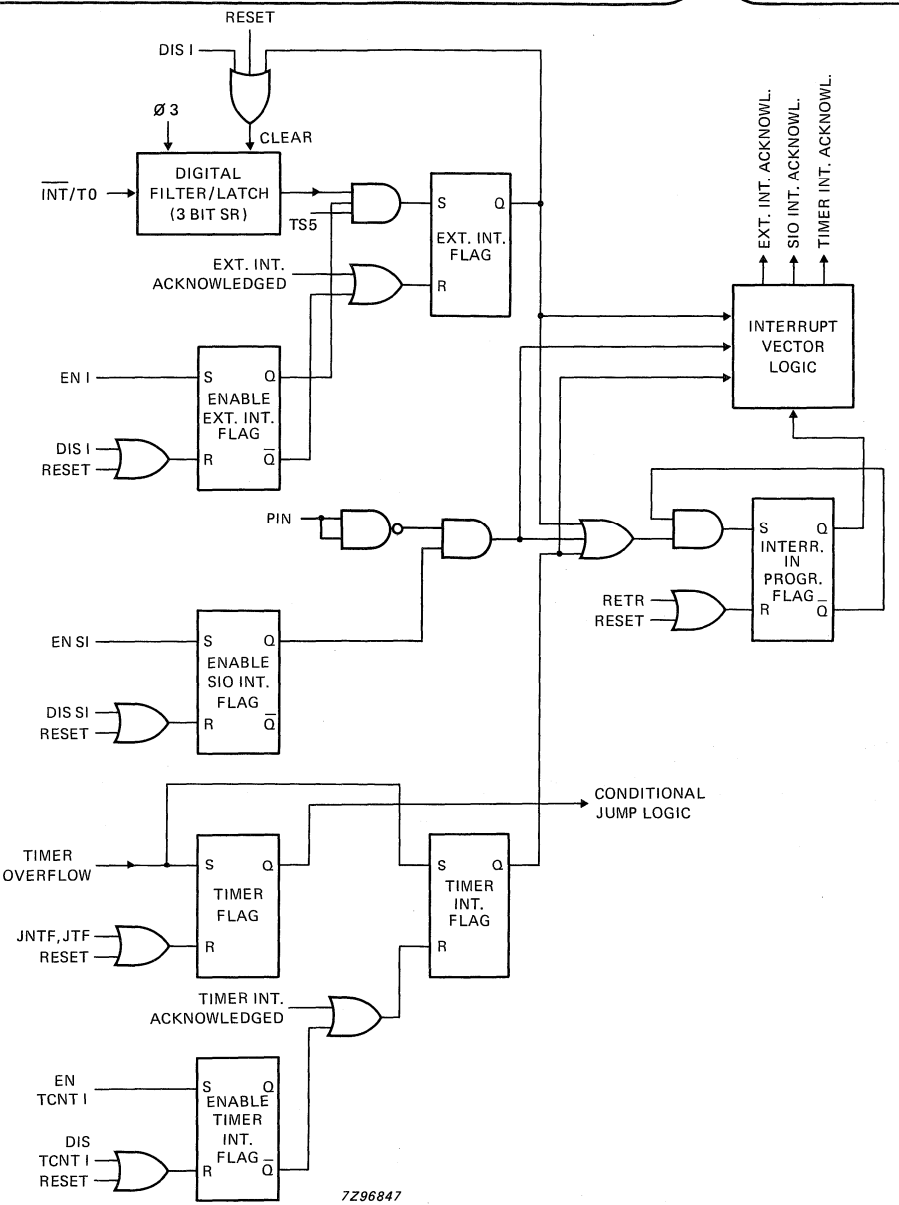


Fig. 14(b) Simplified schematic of interrupt logic, to be used in conjunction with the functional description.

Note to figure

1. $\overline{INT}$/T0 negative edge is always latched in the digital filter/latch.
2. Correct interrupt timing is ensured when $\overline{INT}$/T0 is HIGH for $>$ 4 CP followed by a LOW for $>$ 7 CP.
3. When the interrupt in progress flat is set, further external and timer interrupts are latched but ignored, until RETR is executed.
4. A DIS I instruction always clears a pending external interrupt.
5. For all flip-flops, RESET overrules SET.

Oscillator (see Fig. 15)

The oscillator can be inhibited by the STOP instruction under software control. It is also inhibited when a low-supply voltage condition is present to prevent discharge of a weak back-up battery. Provided the supply voltage is within the operating range the oscillator will be restarted after a STOP instruction by a LOW level at the $\overline{INT}$/T0 pin or a HIGH level at the RESET pin.

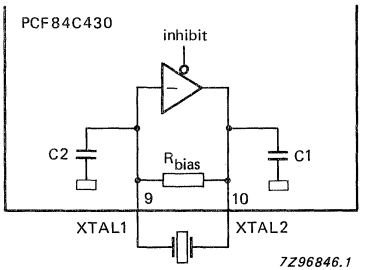


Fig. 15 Oscillator with integrated elements.

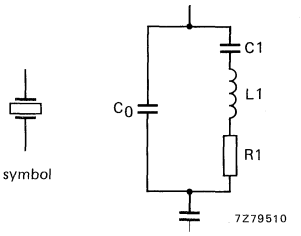


Fig. 16 Crystal unit equivalent circuit.

The values of crystal series resistance R1 and the crystal's total load capacitance C_L (C_o + wiring + external capacitors) must not be above the curve (Fig. 17) for the corresponding frequency.
Note: if external capacitors are connected to XTAL 1 and XTAL 2 they must be of equal value.

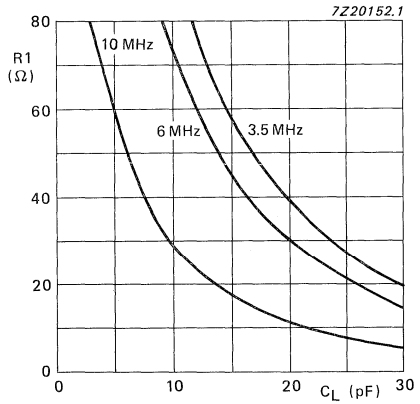


Fig. 17 Crystal circuit series resistance (R1) as a function of load capacitance (C_L).

The oscillator has the output drive capability via pin 10 (XTAL 2). An external clock can be applied to pin 9 (XTAL 1). A machine cycle consists of 10 time slots, each time slot being 3 oscillator periods.

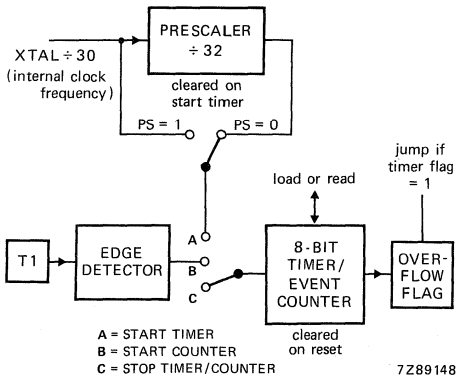
Timer/event counter (see Fig. 18)

An internal 8-bit up-counter is provided. It can count external events, modulo-32 machine cycles, or machine cycles directly. Table 4 gives the instructions that control the counter and the prescaler, and the functions performed.

When used as a timer, the input to the counter is either the overflow or input of a 5-bit prescaler. When used as an event counter, LOW-to-HIGH transitions on pin T1 are counted. The counter is incremented during a machine cycle only if the falling edge occurs during the first 7 time slots; otherwise it is incremented during the next cycle. The maximum rate at which the counter may be incremented is once every machine cycle. When the counter overflows, the Timer flag is set. The flag can be tested and reset using the JTF (jump if Timer flag = logic 1) or JNTF (jump if Timer flag = logic 0) instruction. Overflow also generates an interrupt request to the processor via setting of the Timer Interrupt Flag when the timer/event counter interrupt is enabled.

Table 4 Timer/event counter control

function	timer mode modulo-1, modulo-32*	counter mode
CLEAR	MOV T,A (A) = 0 or RESET	MOV T,A (A) = 0 or RESET
PRESET	MOV T,A	MOV T,A
START	STRT T	STRT CNT
STOP	STOP TCNT or RESET	STOP TCNT or RESET
TEST	JTF/JNTF	JTF/JNTF
READ**	MOV A,T	MOV A,T

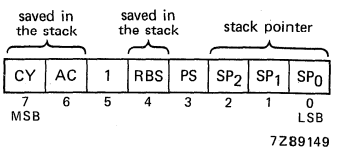


7Z89148 Fig. 18 Timer/event counter.

Program status word (see Fig. 19)

The program status word (PSW) is an 8-bit word (1 byte) in the CPU which stores information about the current status of the microcontroller.
The PSW bits are:

- Bits 0 to 2 — stack pointer bits (SP_0, SP_1, SP_2)
- Bit 3 — prescaler select (PS);
  0 = modulo-32; 1 = modulo-1 (no prescaling)
- Bit 4 — working register bank select (RBS);
  0 = register bank 0; 1 = register bank 1
- Bit 5 — not used (1)
- Bit 6 — auxiliary carry (AC); half-carry bit generated by an ADD instruction and used by the decimal adjust instruction DA A
- Bit 7 — carry (CY); the carry flag indicates that previous operation has resulted in an overflow of the accumulator.



7Z89149 Fig. 19 Program status word.

* With prescaler select, PS = logic 0, the timer is incremented every 32 machine sycles; with PS = logic 1 the timer will be incremented every machine cycle (prescaler not used); the prescaler is cleared by the STRT T instruction and is not readable.

** READ does not disturb the counting process.

DEVELOPMENT DATA

FUNCTIONAL DESCRIPTION (continued)

Program status word (continued)

All bits can be read using the MOV A, PSW instruction. Bits 7 and 6 are set and cleared by CPU operation. Bit 4 can be changed by a SEL RB instruction, bit 3 by the MOV PSW, A instruction, and bits 0, 1 and 2 by the CALL, RET or RETR instructions in the event of an interrupt. Bits 7, 6 and 4 are stored in the program counter stack during subroutine and interrupt calls. These bits are restored in the PSW with a RETR (return and restore) instruction which must be used at the end of an interrupt and can be used at the end of a normal routine, which is not part of an interrupt subroutine. The RET instruction has no restore feature and must not be used at the end of an interrupt.

Program counter (see Fig. 20)

The 12-bit program counter is able to address 4 K bytes of ROM. The arrangement of the bits is shown in Fig. 20. During an interrupt subroutine PC_{11} is forced to logic 0. All 12 bits are saved in the stack during CALL and interrupt routines.

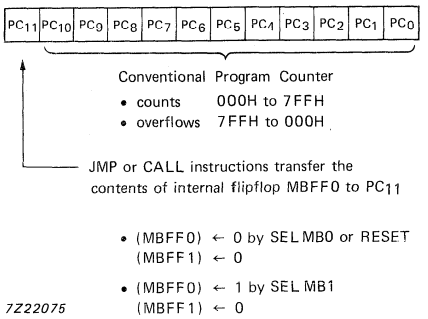


Fig. 20 Program counter.

Central processing unit

The PCF84C430 has arithmetic, logical and branching capabilities. The DA A, SWAP A and XCHD instructions simplify BCD arithmetic and the handling of nibbles. The MOVP A,@A instruction permits efficient table look-up from the current ROM page.

Conditional branch logic

The conditional branch logic within the processor enables several conditions, internal and external to the processor, to be tested by the user's program. Table 5 lists the conditional jump instructions used to change the program sequence. The DJNZ instruction decrements a designated register or data memory location and branches if the contents are not zero. This instruction is useful for looping control. The JMPP@A instruction allows multiway branches to destinations determined by the contents of the accumulator.

Table 5 Conditional branches

test	jump condition	jump instruction
accumulator	all bits zero	JZ
	any bit non-zero	JNZ
accumulator bit test	1	JB0 to JB7
carry flag	1	JC
	0	JNC
timer overflow flag	1	JTF
	0	JNTF
test input T0	1	JT0
	0	JNT0
test input T1	1	JT1
	0	JNT1
register	non-zero	DJNZ

Test input T1 (pin 7)

The T1 input line can be used as:

- A test input for branch instructions JT1 and JNT1
- An external input to the event counter

When used as a test input:

- JT1 instruction tests for logic 1 level
- JNT1 instruction tests for logic 0 level

When used as an input to the event counter, T1 must be LOW for > 4 CP, followed by a HIGH for > 4 CP. A transition can be recognized every 30 oscillator clock periods (1 machine cycle).

There is no internal pull-up or pull-down resistor connected to the T1 input. If required it must be externally connected to a resistor ($R = \leqslant 100\ k\Omega$). When T1 is not used pin 7 must be connected V_{DD} or V_{SS}.

Reset (pin 15)

A positive-going signal on the RESET input/output

- Sets the program counter to zero
- Selects location 0 of memory bank 0 and register bank 0
- Sets the stack pointer to zero (000); pointing to RAM address 8
- Disables the interrupts (external, timer and serial I/O)
- Stops the timer/event counter, then sets it to zero
- Sets the timer prescaler to divide by 32
- Resets the timer flag
- Sets all ports except P2.3 to input mode (see serial I/O section)
- Sets the serial I/O to slave receiver mode and disables the serial I/O
- Cancels IDLE and STOP mode
- Re-defines the LCD control byte

A negative-going signal on the RESET input/output:

- Sets P2.3/SDA and SCLK to HIGH after a maximum of 30 clock pulses
- Sets the serial I/O to slave receiver mode and disables the serial I/O after a maximum of 30 clock pulses
- Starts program execution after 1866 clock pulses

DEVELOPMENT DATA

FUNCTIONAL DESCRIPTION (continued)

Power-on reset

The internal power-on reset circuit monitors the PCF84C430 supply voltage V_{DD}. For as long as the supply voltage remains below the internal reference level V_{ref} (typically 1,5 V) the oscillator is inhibited and RESET (pin 15) has an undefined level. When V_{DD} rises above the internal reference level, the oscillator is released and RESET is pulled high to V_{DD} by TR1 for a period t_D (typically 50 μs).

N.B. Because of the narrow bandwidth of the crystal, the start-up time of the oscillator is typically 10 ms.

Three modes of power-on reset are possible:

1. If V_{DD} can be switched with a fast rise time i.e. V_{DD} reaches its minimum operating value (corresponding to the selected oscillator frequency) before the RESET signal (t_D) has finished, then no extra components are required (see Fig. 21 and 22). Note that the first instruction is executed after the oscillator start-up time plus 1866 clock periods have elapsed.
2. If V_{DD} has a slow rise time then the RESET signal should be stretched by an external RC circuit (see Fig. 23 and 24). In the event of a short drop in the supply voltage, the diode path rapidly discharges the capacitor to ensure a reliable power-on reset. To ensure a correct reset, the RESET signal should reach at least 70% of the final value of V_{DD}. Given that the RESET voltage and V_{DD} rise exponentially, the above requirement is satisfied when the time constant τ of the RESET pulse is >8 times the time constant of V_{DD}. If V_{DD} rises linearly, then a RESET time constant >2 times the rise time of V_{DD} is required.

   When a reset is completed (RESET goes LOW) before the oscillator has started up, program execution begins after the oscillator start-up time plus 1866 clock periods have elapsed (see Fig. 24). If the oscillator is started-up prior to the completion of RESET, then program execution begins 1866 clock periods after RESET goes LOW.
3. Figure 25 shows an external reset to the PCF84C430 during power-on. The external reset signal must remain HIGH until V_{DD} has reached its minimum operating value corresponding to the selected oscillator frequency. When a reset is completed (RESET goes LOW) before the oscillator has started up, program execution begins after the oscillator start-up time plus 1866 clock periods have elapsed (see Fig. 26). If the oscillator is started-up prior to the completion of RESET, then program execution begins 1866 clock periods after RESET goes LOW.

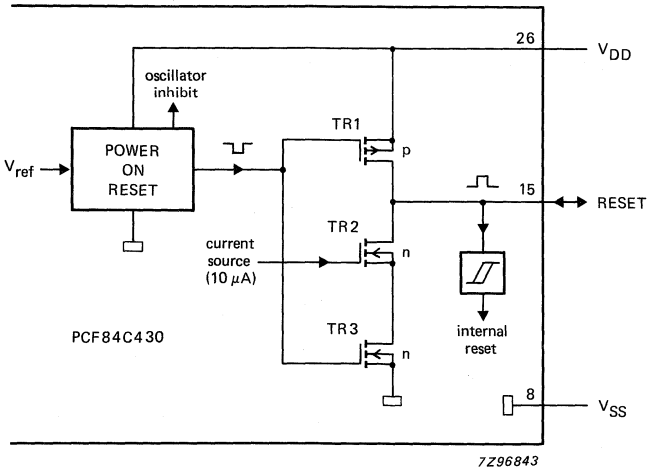


Fig. 21 Power-on-reset configuration.

DEVELOPMENT DATA

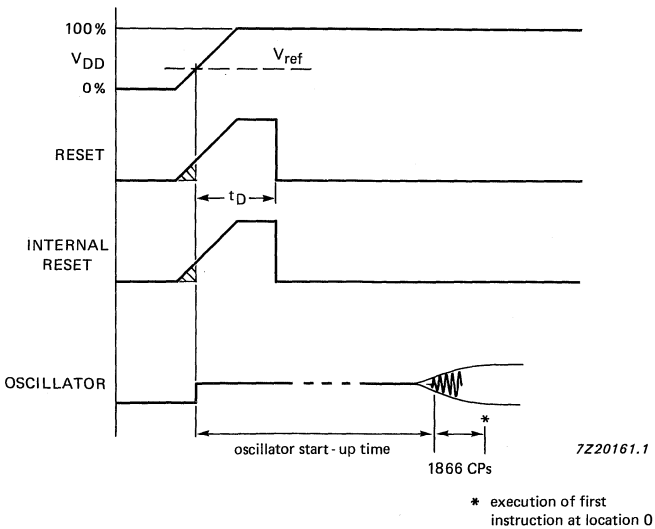


Fig. 22 Timing of power-on-reset with fast rise time.

FUNCTIONAL DESCRIPTION (continued)

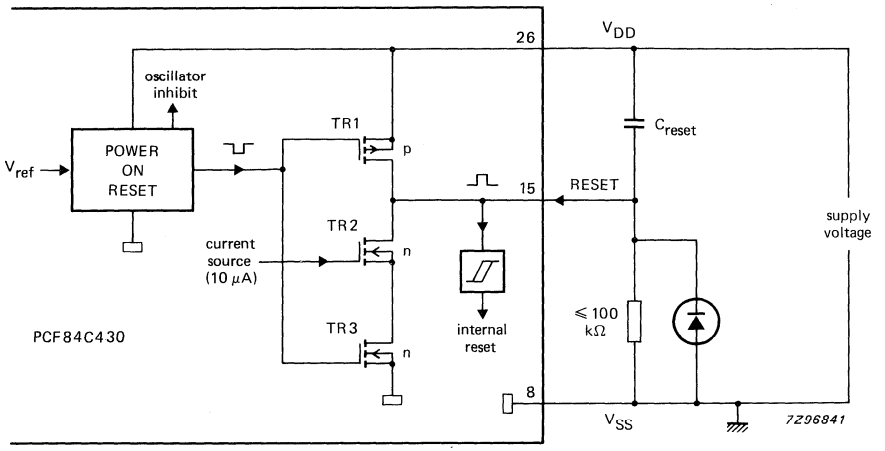


Fig. 23 Stretched power-on-reset with external components.

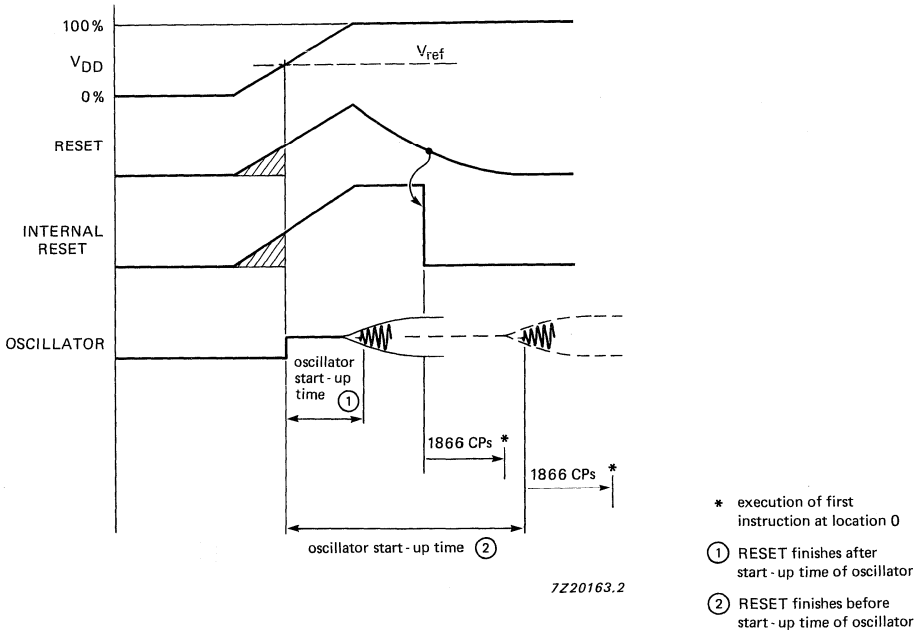


Fig. 24 Timing of power-on-reset with a slowly rising V_{DD} and a stretched RESET pulse.

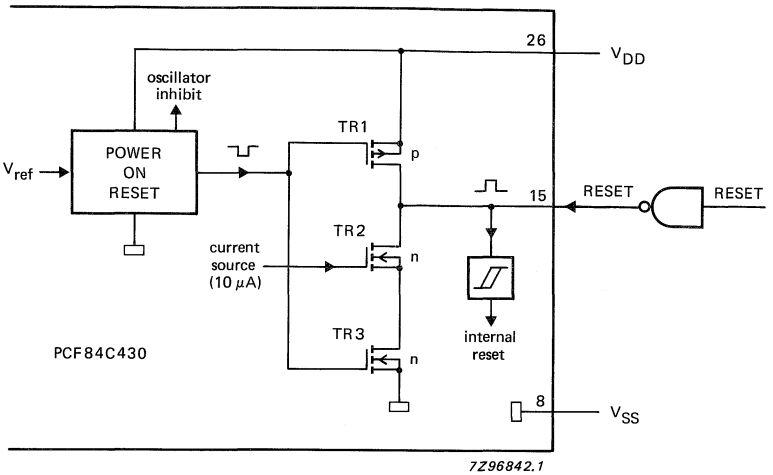


Fig. 25 External power-on-reset configuration.

DEVELOPMENT DATA

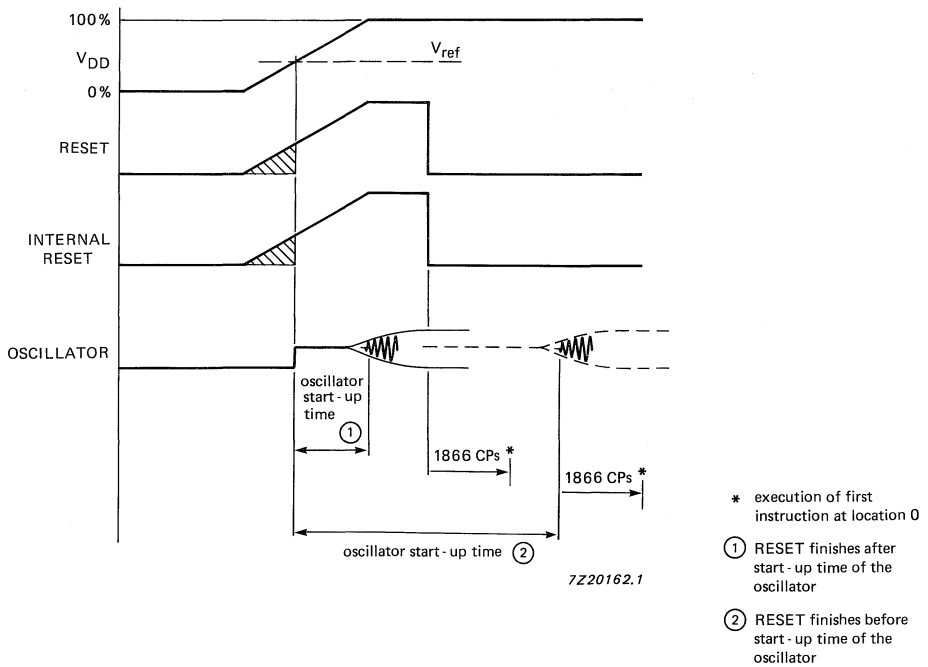


Fig. 26 Timing of external power-on-reset.

FUNCTIONAL DESCRIPTION (continued)

Liquid crystal display driver

The PCF84C430 has a display driver which interfaces to almost any liquid crystal display (LCD) which has a low multiplex rate. The interface delivers drive signals for any static or multiplexed LCD panel that contains up to four backplanes and up to 24 segments. Fig. 27 shows a block diagram of the LCD driver.

The following features are incorporated:

- Selectable backplane drive configuration; static or 2/3/4 backplane multiplexing
- Selectable display bias configuration; 1/2 or 1/3 internal LCD bias generation
- 24 individual segment drivers can be used to provide
  - up to twelve 8-segment numeric characters
  - up to six 15+1 segment alphanumeric characters
  - graphics using up to 96 elements
  - twelve 8-bit derivative registers for display data bits
- LCD and logic voltage supplies may be separated
- An LCD operating voltage range of between V_{SS} to V_{DD} −2,5 V ($V_{SS} \leqslant V_{LCD} < V_{DD}$)
- A low-power zero-component oscillator keeps the LCD display running in the STOP mode

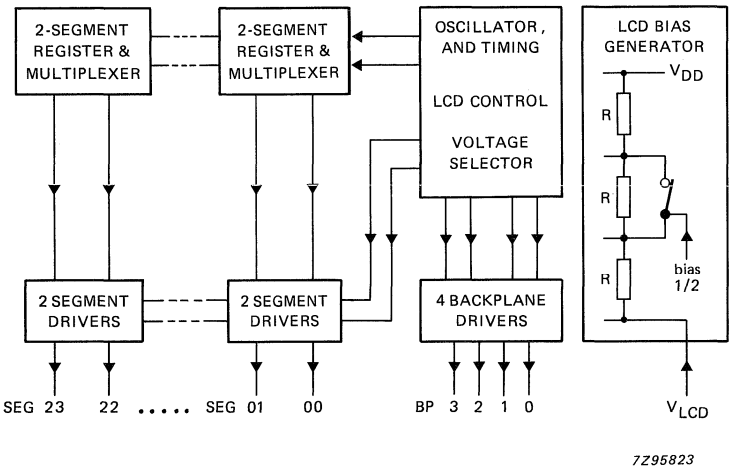


Fig. 27 Block diagram of the LCD driver.

The display configurations possible with the PCF84C430 depend upon the number of active backplane outputs required. A selection of display configurations is given in Table 6.

Table 6 Selection of display configurations

number of backplanes	number of segments	7-segment numeric	14-segment alphanumeric	dot matrix
4	96	12 digits + 12 indicator symbols	6 characters + 12 indicator symbols	96 dots (4 x 24)
3	72	9 digits + 9 indicator symbols	5 characters + 2 indicator symbols	72 dots (3 x 24)
2	48	6 digits + 6 indicator symbols	3 characters + 6 indicator symbols	48 dots (2 x 24)
1	24	3 digits + 3 indicator symbols	1 character + 10 indicator symbols	24 dots (1 x 24)

All of the display configurations given in Table 6 can be implemented in the typical system shown in Fig. 28. The appropriate biasing voltages for the multiplexed LCD wave forms are generated internally.

At power-on all the LCD driver control register bits are cleared. The LCD display is not affected by executing a STOP instruction.

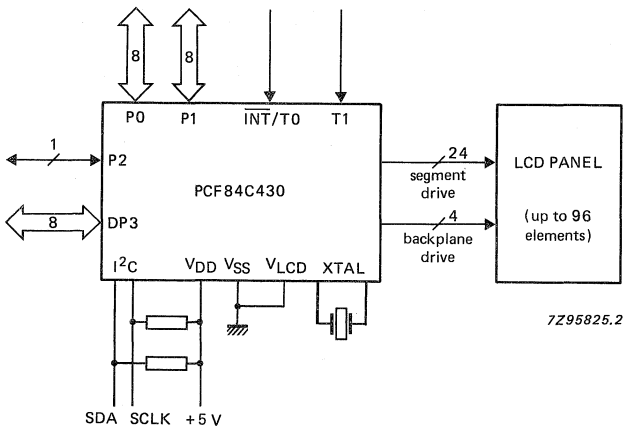


Fig. 28 Typical system configuration.

DEVELOPMENT DATA

FUNCTIONAL DESCRIPTION (continued)

LCD bias generation

The LCD operating voltage (V_{op}) is the result of V_{DD} - V_{LCD}. V_{op} should be chosen so that the off voltage ($V_{off(rms)}$) is just below the threshold voltage (V_{th}), typically when the LCD exhibits 10% contrast. The LCD voltage may be temperature compensated externally through the LCD supply. Fractional LCD biasing voltages are obtained from an internal voltage divider of three resistors connected between V_{DD} and V_{LCD}. The centre resistor may be switched out of circuit to provide a ½ bias voltage level for a 1:2 multiplex configuration.

LCD voltage selector

The LCD voltage selector coordinates the multiplexing of the LCD according to the selected drive configuration. The operation of the voltage selector is controlled by the MODE bits in the LCD control byte. The biasing configurations that apply to the preferred mode of operation, together with the biasing characteristics as functions of $V_{op} = V_{DD} - V_{LCD}$ and resulting discrimination ratios (D), are given in Table 7.

Table 7 LCD drive modes and characteristics

LCD drive mode	number of backplanes	LCD bias configuration	number of levels	$\frac{V_{off(rms)}}{V_{op}}$	$\frac{V_{on(rms)}}{V_{op}}$	$D = \frac{V_{on(rms)}}{V_{off(rms)}}$
static	1	static	2	0	1	∞
1:2	2	1/2	3	0,354	0,791	2,236
1:2	2	1/3	4	0,333	0,745	2,236
1:3	3	1/3	4	0,333	0,638	1,915
1:4	4	1/3	4	0,333	0,577	1,7321

Multiplex drive ratios of 1:3 and 1:4 with ½ bias are possible, but the discrimination and contrast ratios are reduced thus;

(1,732 for 1:3 or 1,528 for 1:4)

There is an advantage however, that these modes lead to a reduction in V_{op} as follows:

1:3 multiplex (½ bias). $V_{op} = 2{,}449\ V_{off(rms)}$

1:4 multiplex (½ bias). $V_{op} = 2{,}309\ V_{off(rms)}$

This compares with $V_{op} = 3\ V_{off(rms)}$ when 1/3 bias is used.

LCD driver (continued)

LCD drive mode waveforms

The static LCD drive mode is used when a single backplane is provided in the LCD. Backplane and segment drive waveforms are shown in Fig. 29.

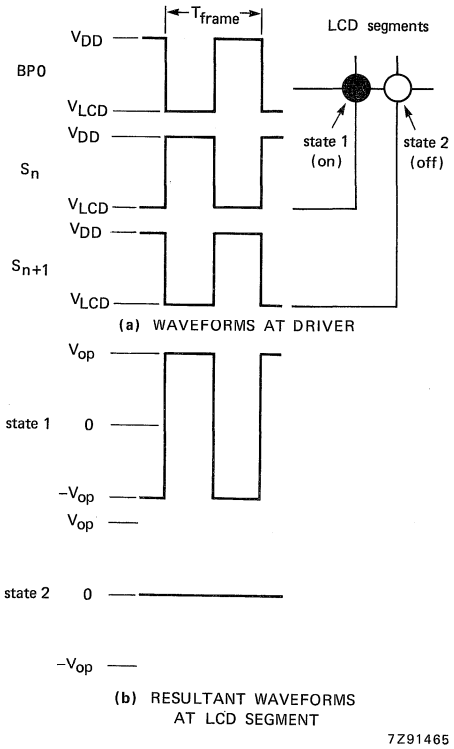


Fig. 29 Static drive mode waveforms.

DEVELOPMENT DATA

FUNCTIONAL DESCRIPTION (continued)

When two backplanes are provided in the LCD; the 1:2 multiplex drive mode applies. The PCF84C430 allows use of 1/2 or 1/3 bias in this mode as shown in Figs 30 and 31.

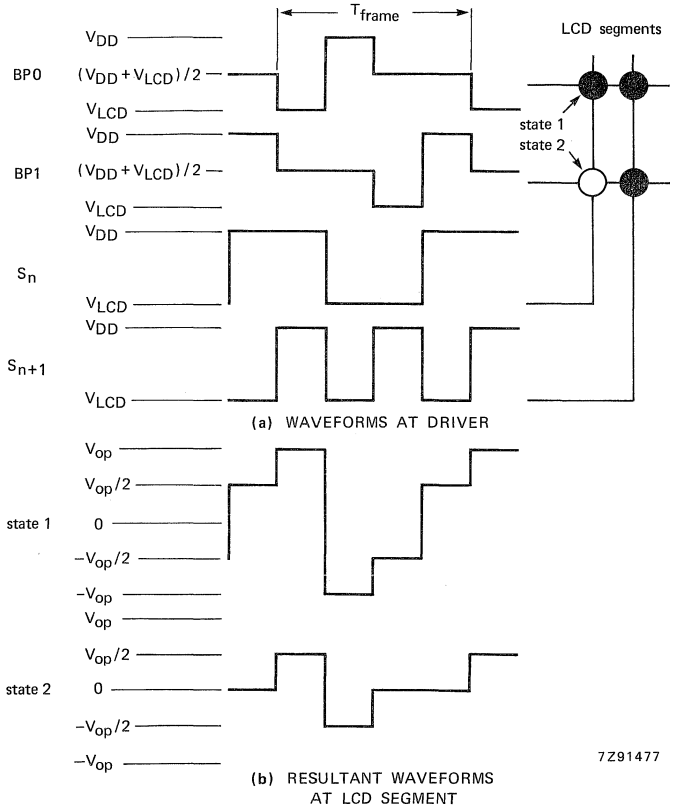


Fig. 30 Waveforms for the 1:2 multiplex drive mode with 1/2 bias.

LCD driver (continued)

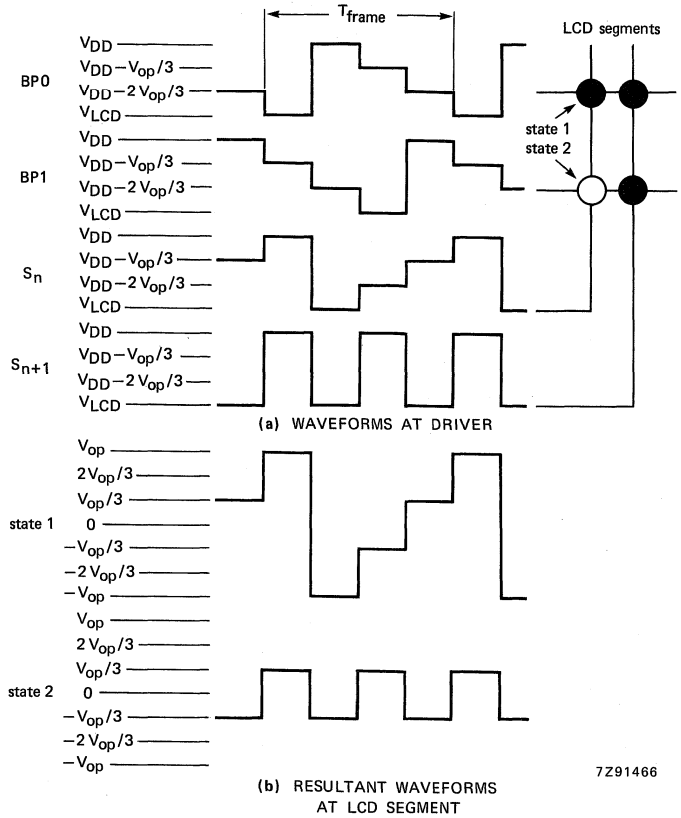


Fig. 31 Waveforms for the 1:2 multiplex drive mode with 1/3 bias.

DEVELOPMENT DATA

FUNCTIONAL DESCRIPTION (continued)

The backplane and segment drive waveforms for the 1:3 multiplex drive mode (three LCD backplanes) and for the 1:4 multiplex drive mode (four LCD backplanes) are shown in Figs 32 and 33 respectively.

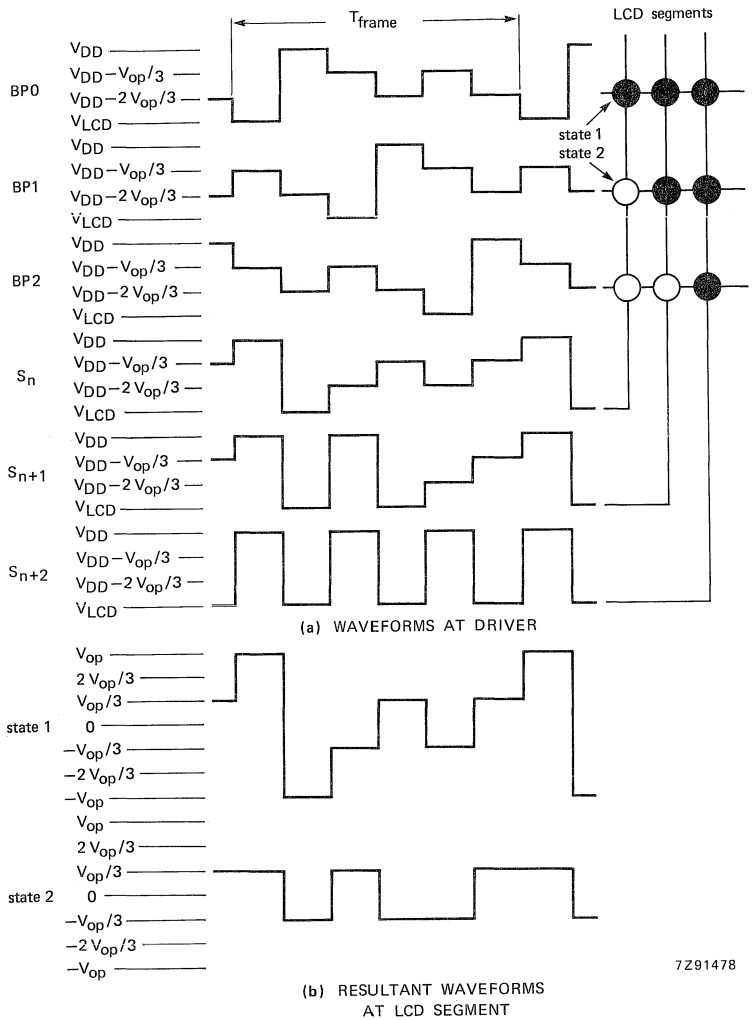


Fig. 32 Waveforms for the 1:3 multiplex drive mode.

LCD driver (continued)

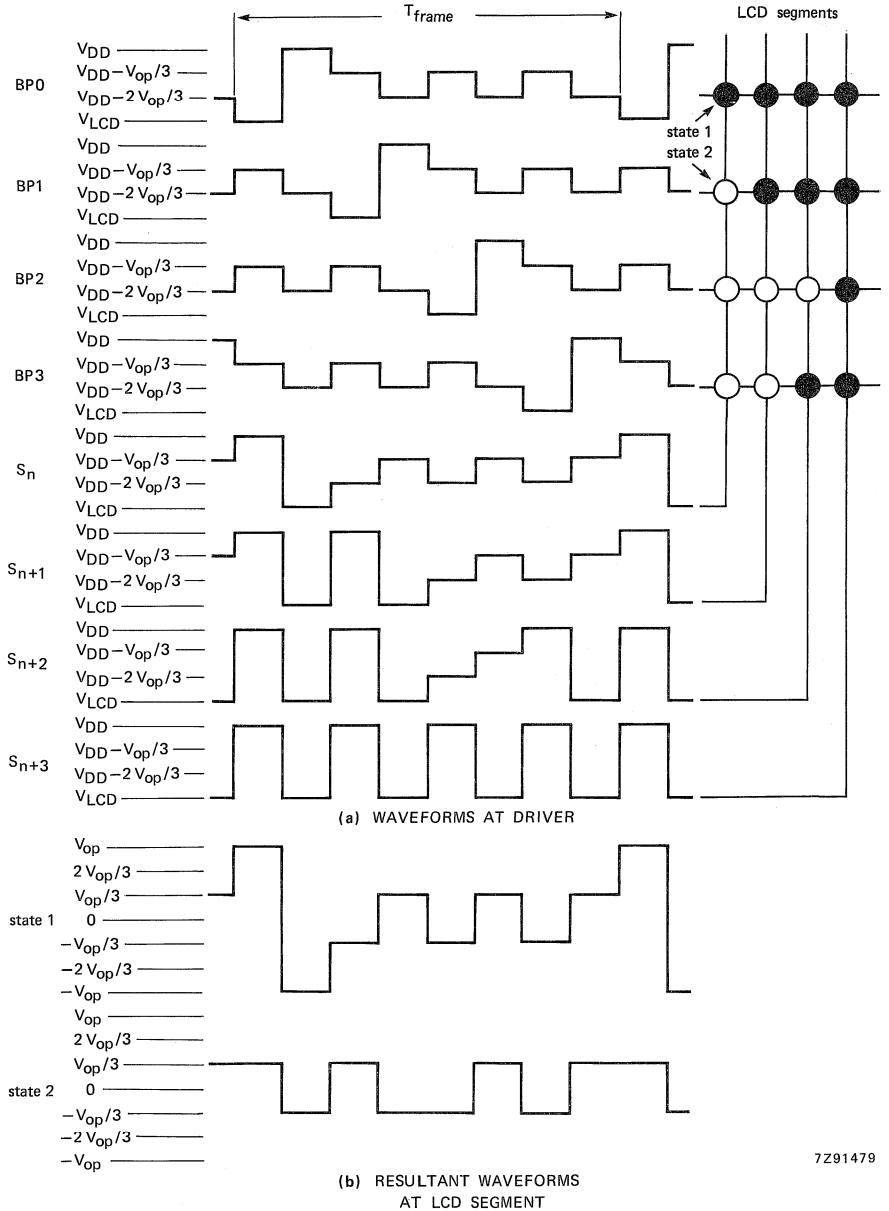


Fig. 33 Waveforms for the 1:4 multiplex drive mode.

FUNCTIONAL DESCRIPTION (continued)

LCD timing

The LCD clocking is performed by a separate, self-contained, on-chip oscillator with a nominal frequency of 30 kHz. The display may be kept active while in the STOP mode using a very low supply current. Fig. 34 shows the LCD timing control.

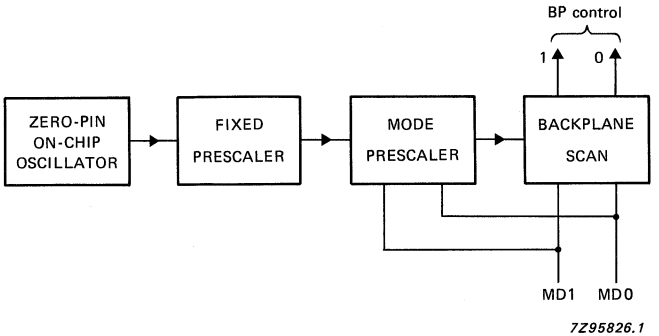


Fig. 34 LCD timing control.

LCD segment driver outputs

The LCD drive section includes 24 segment outputs (S0 to S23) which should be connected directly to the LCD. The segment data bits (one nibble per segment) are multiplexed to the outputs in accordance with the backplane signals. If less than the 24 segment outputs are required then the unused driver outputs should be left open.

Backplane outputs

The LCD drive section includes 4 backplane outputs (BP0-BP3) which should be connected directly to the LCD. These backplane output signals are generated in accordance with the selected LCD drive mode. If less than four backplane outputs are required then the unused driver outputs should be left open-circuit.

In the 1:3 multiplex drive mode, BP3 carries the same signal as BP0, therefore these two outputs can be tied together to give enhanced drive capabilities. In the 1:2 multiplex drive mode, BP0 and BP3, BP1 and BP2 respectively carry the same signals and may also be paired to increase the drive capabilities. In the static drive mode the same signal is carried by all four backplane outputs and may be connected in parallel to give a very high drive capability.

LCD driver (continued)

LCD segment display registers

The 12 segment display registers are 8-bit derivative (read/write) registers which store LCD segment data. A segment register bit which is set to a logic 1 indicates the 'on' state of the corresponding LCD segment, similarly, a logic 0 indicates the 'off' state. There is a one-to-one relationship between the LCD segment register bits and the segment outputs. Every byte is divided into two nibbles. Each nibble corresponds to a segment driver; the first byte contains the bits earmarked for segment 1 and 2 etc. The first bit of a nibble will correspond to backplane 0, and the second to backplane 1 and so on. Fig. 35 shows the display register bit map. Each bit is shown in the form Sxx.n, where (xx) is the segment output and (n) the backplane output.

LCDD : LCD display data

17	R/W	S 0.0	S 0.1	S 0.2	S 0.3	S 1.0	S 1.1	S 1.2	S 1.3
18	R/W	S 2.0	S 2.1	S 2.2	S 2.3	S 3.0	S 3.1	S 3.2	S 3.3
19	R/W	S 4.0	S 4.1	S 4.2	S 4.3	S 5.0	S 5.1	S 5.2	S 5.3
20	R/W	S 6.0	S 6.1	S 6.2	S 6.3	S 7.0	S 7.1	S 7.2	S 7.3
21	R/W	S 8.0	S 8.1	S 8.2	S 8.3	S 9.0	S 9.1	S 9.2	S 9.3
22	R/W	S10.0	S10.1	S10.2	S10.3	S11.0	S11.1	S11.2	S11.3
23	R/W	S12.0	S12.1	S12.2	S12.3	S13.0	S13.1	S13.2	S13.3
24	R/W	S14.0	S14.1	S14.2	S14.3	S15.0	S15.1	S15.2	S15.3
25	R/W	S16.0	S16.1	S16.2	S16.3	S17.0	S17.1	S17.2	S17.3
26	R/W	S18.0	S18.1	S18.2	S18.3	S19.0	S19.1	S19.2	S19.3
27	R/W	S20.0	S20.1	S20.2	S20.3	S21.0	S21.1	S21.2	S21.3
28	R/W	S22.0	S22.1	S22.2	S22.3	S23.0	S23.1	S23.2	S23.3

Fig. 35 Display register bit map.

In the static drive mode the eight display data bits are placed in bit 0 of the nibbles of four successive derivative display registers. In the 1:2 multiplex drive mode, these eight bits are placed in bits 0 and 1 of the nibbles of two successive registers. In the 1:3 multplex drive mode the eight bits are placed in

DEVELOPMENT DATA

FUNCTIONAL DESCRIPTION (continued)

bits 0, 1 and 2 of three successive nibbles. In the 1:4 multiplex drive mode the eight bits are placed in bits 0, 1, 2 and 3 of the nibbles of the first display register. Fig. 36 shows the relationship between LCD segment layout, drive mode and the LCD segment derivative register bit pattern.

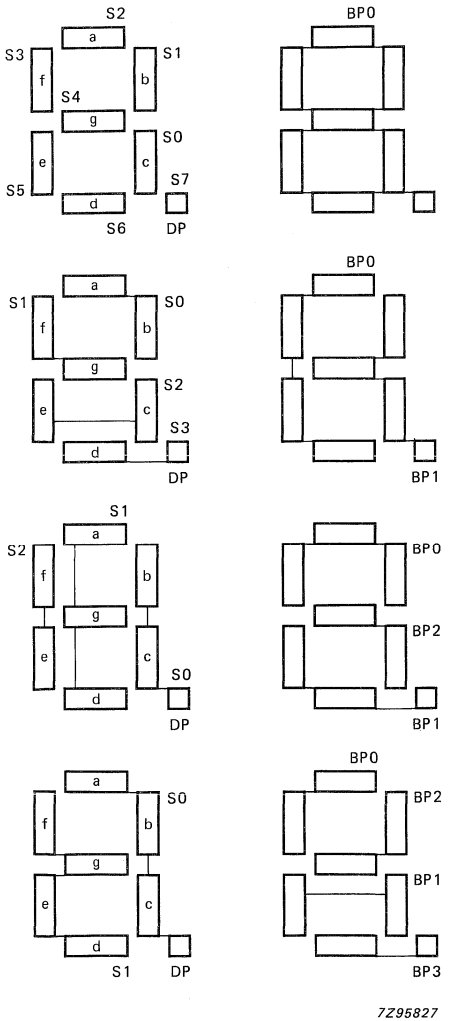


Fig. 36 LCD segment layout and register bit pattern.

LCD derivative register bit map

byte

17	c	–	–	–	b	–	–	–
18	a	–	–	–	f	–	–	–
19	g	–	–	–	e	–	–	–
20	d	–	–	–	Dp	–	–	–

static mode

byte

17	a	b	–	–	f	g	–	–
18	e	c	–	–	d	Dp	–	–
19	–	–	–	–	–	–	–	–
20	–	–	–	–	–	–	–	–

1:2 multiplex mode

byte

17	b	Dp	c	–	a	d	g	–
18	f	e	–	–	–	–	–	–
19	–	–	–	–	–	–	–	–
20	–	–	–	–	–	–	–	–

1:3 multiplex mode

byte

17	a	c	b	Dp	f	e	g	d
18	–	–	–	–	–	–	–	–
19	–	–	–	–	–	–	–	–
20	–	–	–	–	–	–	–	–

1:4 multiplex mode

LCD driver (continued)

LCD control register

The LCD operating mode is governed by four bits of the LCD control register (address 16 D). The LCD control register is an 8-bit derivative read/write register. The function of each bit is given in Table 8.

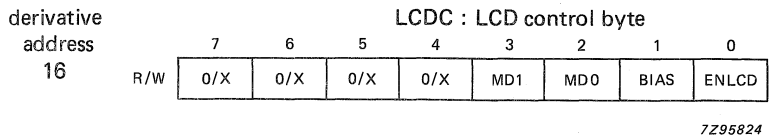


Table 8 Control register bit definition

bit	name	function		
0	ENLCD	0 = LCD display disabled 1 = LCD display enabled		
1	BIAS	0 = 1/3 V_{DD} 1 = 1/2 V_{DD}		
2, 3	MD0/MD1	MD1	MD0	multiplex mode
		0 0 1 1	0 1 0 1	static 1:2 1:3 1:4
4-7	unused			

ENLCD: When ENLCD is reset to 0, the LCD is disabled. Consequently all segment and backplane drivers are set to the V_{DD} level. When ENLCD is set to a 1, the LCD is enabled and character display is possible.

BIAS: The BIAS bit sets the LCD voltage bias generator to either 1/2 or 1/3 of V_{DD}, see Fig. 27.

MD0/MD1: Mode bits MD0 and MD1 determine the multiplex rate. Four multiplex rates are available; static, 1:2, 1:3 and 1:4.

FUNCTIONAL DESCRIPTION (continued)

LCD control byte after RESET

After an external or power-on reset the LCD control register is set with 0C H.

	7	6	5	4	3	2	1	0
	0/X	0/X	0/X	0/X	MD1	MD0	BIAS	ENLCD
reset value	0	0	0	0	1	1	0	0

7Z97990

- The LCD is disabled and segment and backplane drives are switched to the V_{DD} level.
- BIAS is set to generate 1/3 V_{DD}.
- Bits MD0 and MD1 reset the multiplex mode to the 1:4 mode.

INSTRUCTION SET

The PCF84C430 instruction set consists of over 80 one and two byte instructions. Program code efficiency is high because all RAM locations and all ROM locations on a 256 byte page require only a single byte address.

Table 11 gives the instruction set of the PCF84C430. Table 10 shows the instruction map and Table 9 details the symbols and definition descriptions that are used.

Table 9 Symbols and definitions used in Table 11

symbol	definition description
A	accumulator
addr	program memory address
Bb	bit designation (b = 0-7)
RBS	register bank select
C	carry bit (bit CY)
CNT	event counter
Dx	Derivative register designation (x = 4,5,16,17-28)
data	8-bit number or expression
I	interrupt
MB	memory bank
MBFF	memory bank flip-flop
P	mnemonic for 'in-page' operation
PC	program counter
Pp	port designation (p = 0, 1 or 2)
PSW	program status word
RB	register bank
Rr	register designation (r = 0-7)
Sn	serial I/O register
SP	stack pointer
T	timer
TF	timer flag
T0, T1	test 0 and 1 inputs
#	immediate data prefix
@	indirect address prefix
(X)	contents of X
((X))	contents of location addressed by X
←	is replaced by
↔	is exchanged with

DEVELOPMENT DATA

INSTRUCTION SET (continued)

Table 10 PCF84C430 instruction map

first hexadecimal character of opcode

second hexadecimal character of opcode

	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
0	NOP	IDLE		ADD A,#data	JMP page 0	EN I	JNTF addr	DEC A	IN A,Pp				MOV A,Sn			
									0	1	2		0	1		
1	INC @Rr		JBO addr	ADDC A,#data	CALL page 0	DIS I	JTF addr	INC A	INC Rr							
	0	1							0	1	2	3	4	5	6	7
2	XCH A,@Rr		STOP	MOV A,#data	JMP page 1	EN TCNTI	JNTO addr	CLR A	XCH A,Rr							
	0	1							0	1	2	3	4	5	6	7
3	XCHD A,@Rr		JB1 addr		CALL page 1	DIS TCNTI	JTO addr	CPL A	OUTL Pp,A				MOV Sn,A			
	0	1							0	1	2		0	1	2	
4	ORL A,@Rr		MOV A,T	ORL A,#data	JMP page 2	STRT CNT	JNT1 addr	SWAP A	ORL A,Rr							
	0	1							0	1	2	3	4	5	6	7
5	ANL A,@Rr		JB2 addr	ANL A,#data	CALL page 2	STRT T	JT1 addr	DA A	ANL A,Rr							
	0	1							0	1	2	3	4	5	6	7
6	ADD A,@Rr		MOV T,A		JMP page 3	STOP TCNT		RRC A	ADD A,Rr							
	0	1							0	1	2	3	4	5	6	7
7	ADDC A,@Rr		JB3 addr		CALL page 3			RR A	ADDC A,Rr							
	0	1							0	1	2	3	4	5	6	7
8				RET	JMP page 4	EN SI			ORL Pp,#data				MOV A,Dx	MOV Dx,A	ANL Dx,A	ORL Dx,A
									0	1	2					
9			JB4 addr	RETR	CALL page 4	DIS SI	JNZ addr	CLR C	ANL Pp,#data				MOV Sn,#data			
									0	1	2		0	1	2	
A	MOV @Rr,A			MOVP A,@A	JMP page 5			CPL C	MOV Rr,A							
	0	1							0	1	2	3	4	5	6	7
B	MOV @Rr,#data		JB5 addr	JMPP @A	CALL page 5				MOV Rr,#data							
	0	1							0	1	2	3	4	5	6	7
C	DEC @Rr				JMP page 6	SEL RB0	JZ addr	MOV A,PSW	DEC Rr							
	0	1							0	1	2	3	4	5	6	7
D	XRL A,@Rr		JB6 addr	XRL A,#data	CALL page 6	SEL RB1		MOV PSW,A	XRL A,Rr							
	0	1							0	1	2	3	4	5	6	7
E	DJNZ @Rr,addr				JMP page 7	SEL MB0	JNC addr	RL A	DJNZ Rr, addr							
	0	1							0	1	2	3	4	5	6	7
F	MOV A,@Rr		JB7 addr		CALL page 7	SEL MB1	JC addr	RLC A	MOV A,Rr							
	0	1							0	1	2	3	4	5	6	7

Table 11 Instruction set

	mnemonic	opcode (hex.)	bytes/ cycles	description	function		notes
ACCUMULATOR	ADD A, Rr	6*	1/1	Add register contents to A	(A)←(A) + (Rr)	r = 0–7	1
	ADD A, @Rr	60 61	1/1	Add RAM data, addressed by Rr, to A	(A)←(A) + ((R0)) (A)←(A) + ((R1))		1
	ADD A, #data	03 data	2/2	Add immediate data to A	(A)←(A) + data		1
	ADDC A, Rr	7*	1/1	Add carry and register contents to A	(A)←(A) + (Rr) + (C)	r = 0–7	1
	ADDC A, @Rr	70 71	1/1	Add carry and RAM data, addressed by Rr, to A	(A)←(A) + ((R0)) + (C) (A)←(A) + ((R1)) + (C)		1
	ADDC A, #data	13 data	2/2	Add carry and immediate data to A	(A)←(A) + data + (C)		1
	ANL A, Rr	5*	1/1	'AND' Rr with A	(A)←(A) AND (Rr)	r = 0–7	
	ANL A, @Rr	50 51	1/1	'AND' RAM data, addressed by Rr, with A	(A)←(A) AND ((R0)) (A)←(A) AND ((R1))		
	ANL A, #data	53 data	2/2	'AND' immediate data with A	(A)←(A) AND data		
	ORL A, Rr	4*	1/1	'OR' Rr with A	(A)←(A) OR (Rr)	r = 0–7	
	ORL A, @Rr	40 41	1/1	'OR' RAM data, addressed by Rr, with A	(A)←(A) OR ((R0)) (A)←(A) OR ((R1))		
	ORL A, #data	43 data	2/2	'OR' immediate data with A	(A)←(A) OR data		
	XRL A, Rr	D*	1/1	'XOR' Rr with A	(A)←(A) XOR (Rr)	r = 0–7	
	XRL A, @Rr	D0 D1	1/1	'XOR' RAM, addressed by Rr, with A	(A)←(A) XOR ((R0)) (A)←(A) XOR ((R1))		
	XRL A, #data	D3 data	2/2	'XOR' immediate data with A	(A)←(A) XOR data		
	INC A	17	1/1	increment A by 1	(A)←(A) + 1		
	DEC A	07	1/1	decrement A by 1	(A)←(A) −1		
	CLR A	27	1/1	clear A to zero	(A)←0		
	CPL A	37	1/1	one's complement A	(A)←NOT(A)		
	RL A	E7	1/1	rotate A left	$(A_{n+1})\leftarrow(A_n)$ $(A_0)\leftarrow(A_7)$	n = 0–6	

INSTRUCTION SET (continued)

ACCUMULATOR (cont.)	RLC A	F7	1/1	rotate A left through carry	$(A_{n+1})\leftarrow A_n$ $(A_0)\leftarrow(C)$, $(C)\leftarrow(A_7)$	n = 0–6	2
	RR A	77	1/1	rotate A right	$(A_n)\leftarrow(A_{n+1})$ $(A_7)\leftarrow(A_0)$	n = 0–6	
	RRC A	67	1/1	rotate A right through carry	$(A_n)\leftarrow(A_{n+1})$ $(A_7)\leftarrow(C)$, $(C)\leftarrow(A_0)$	n = 0–6	2
	DA A	57	1/1	decimal adjust A			2
	SWAP A	47	1/1	swap nibbles of A	$(A_{4-7})\leftrightarrow(A_{0-3})$		
DATA MOVES	MOV A, Rr	F*	1/1	move register contents to A	(A)←(Rr)	r = 0–7	
	MOV A, @Rr	F0 F1	1/1	move RAM data, addressed by Rr, to A	(A)←((R0)) (A)←((R1))		
	MOV A, #data	23 data	2/2	move immediate data to A	(A)←data		
	MOV Rr, A	A*	1/1	move accumulator contents to register	(Rr)←(A)	r = 0–7	
	MOV @Rr, A	A0 A1	1/1	move accumulator contents to RAM location addressed by Rr	((R0))←(A) ((R1))←(A)		
	MOV Rr, #data	B* data	2/2	move immediate data to Rr	(Rr)←data		
	MOV @Rr, #data	B0 data B1 data	2/2	move immediate data to RAM location addressed by Rr	((R0))←data ((R1))←data		
	XCH A, Rr	2*	1/1	exchange accumulator contents with Rr	(A)↔(Rr)	r = 0–7	
	XCH A, @Rr	20 21	1/1	exchange accumulator contents with RAM data addressed by Rr	(A)↔((R0)) (A)↔((R1))		
	XCHD A, @Rr	30 31	1/1	exchange lower nibbles of A and RAM data addressed by Rr	$(A_{0-3})\leftrightarrow((R0_{0-3}))$ $(A_{0-3})\leftrightarrow((R1_{0-3}))$		
	MOV A, PSW	C7	1/1	move PSW contents to accumulator	(A)←(PSW)		
	MOV PSW, A	D7	1/1	move accumulator bit 3 to PSW_3	$(PSW_3)\leftarrow(A_3)$		3
	MOVP A, @A	A3	1/2	move indirectly addressed data in current page to A	$(PC_{0-7})\leftarrow(A)$, $(A)\leftarrow((PC))$		
FLAGS	CLR C	97	1/1	clear carry bit	(C)←0		2
	CPL C	A7	1/1	complement carry bit	(C)←NOT(C)		2

DEVELOPMENT DATA

	mnemonic	opcode (hex.)	bytes/ cycles	description	function	notes
REGISTER	INC Rr	1*	1/1	increment register by 1	(Rr)←(Rr) + 1 r = 0–7	
	INC @Rr	10 11	1/1	increment RAM data, addressed by Rr, by 1	((R0))←((R0)) + 1 ((R1))←((R1)) + 1	
	DEC Rr	C*	1/1	decrement register by 1	(Rr)←(Rr) – 1 r = 0–7	
	DEC @Rr	C0 C1	1/1	decrement RAM data, addressed by Rr, by 1	((R0))←((R0)) – 1 ((R1))←((R1)) – 1	
BRANCH	JMP addr	● 4 addr	2/2	unconditional jump within a 2 K bank	(PC_{8-10})←$addr_{8-10}$ (PC_{0-7})←$addr_{0-7}$ (PC_{11-12})←MBFF 0–1	
	JMPP @A	B3	1/2	indirect jump within a page	(PC_{0-7})←((A))	
	DJNZ Rr, addr	E* addr	2/2	decrement Rr by 1 and jump if not zero to addr	(Rr)←(Rr) – 1 r = 0–7 if (Rr) not zero (PC_{0-7})←addr	
	DJNZ @Rr, addr	E0 addr	2/2	decrement RAM data, addressed by Rr by 1 and jump if not zero to addr	((R0))←((R0)) – 1 if ((R0)) not zero (PC_{0-7})←addr	
		E1 addr			((R1))←((R1)) – 1 if ((R1)) not zero (PC_{0-7})←addr	
	JBb addr	▲ 2 addr	2/2	jump to addr if Acc. bit b = 1	if b = 1 : (PC_{0-7})←addr b = 0–7	
	JC addr	F6 addr	2/2	jump to addr if C = 1	if C = 1 : (PC_{0-7})←addr	
	JNC addr	E6 addr	2/2	jump to addr if C = 0	if C = 0 : (PC_{0-7})←addr	
	JZ addr	C6 addr	2/2	jump to addr if A = 0	if A = 0 : (PC_{0-7})←addr	
	JNZ addr	96 addr	2/2	jump to addr if A is NOT zero	if A≠0 : (PC_{0-7})←addr	
	JT0 addr	36 addr	2/2	jump to addr if T0 = 1	if T0 = 1: (PC_{0-7})←addr	
	JNT0 addr	26 addr	2/2	jump to addr if T0 = 0	if T0 = 0: (PC_{0-7})←addr	
	JT1 addr	56 addr	2/2	jump to addr if T1 = 1	if T1 = 1: (PC_{0-7})←addr	
	JNT1 addr	46 addr	2/2	jump to addr if T1 = 0	if T1 = 0: (PC_{0-7})←addr	
	JTF addr	16 addr	2/2	jump to addr if Timer Flag = 1	if TF = 1: (PC_{0-7})←addr	4
	JNTF addr	06 addr	2/2	jump to addr if Timer Flag = 0	if TF = 0: (PC_{0-7})←addr	

INSTRUCTION SET (continued)

TIMER/EVENT COUNTER	MOV A, T	42	1/1	move timer/event counter contents to accumulator	(A)←(T)	
	MOV T, A	62	1/1	move accumulator contents to timer/event counter	(T)←(A)	
	STRT CNT	45	1/1	start event counter		
	STRT T	55	1/1	start timer		
	STOP TCNT	65	1/1	stop timer/event counter		
	EN TCNTI	25	1/1	enable timer/event counter interrupt		
	DIS TCNTI	35	1/1	disable timer/event counter interrupt		
CONTROL	EN I	05	1/1	enable external interrupt		
	DIS I	15	1/1	disable external interrupt		
	SEL RB0	C5	1/1	select register bank 0	(RBS)←0	5
	SEL RB1	D5	1/1	select register bank 1	(RBS)←1	5
	SEL MB0	E5	1/1	select program memory bank 0	(MBFF0)←0, (MBFF1)←0	10
	SEL MB1	F5	1/1	select program memory bank 1	(MBFF0)←1, (MBFF1)←0	10
	STOP	22	1/1	enter STOP mode		
	IDLE	01	1/1	enter IDLE mode		
SUBROUTINE	CALL addr	▲ 4 addr	2/2	jump to subroutine	$((SP))\leftarrow(PC), (PSW_{4,6,7})$ 6 $(SP)\leftarrow(SP)+1$ $(PC_{8-10})\leftarrow addr_{8-10}$ $(PC_{0-7})\leftarrow addr_{0-7}$ $(PC_{11-12})\leftarrow$MBFF 0–1	6
	RET	83	1/2	return from subroutine	$(SP)\leftarrow(SP)-1$ $(PC)\leftarrow((SP))$	6
	RETR	93	1/2	return from interrupt and restore bits 4, 6, 7 of PSW	$(SP)\leftarrow(SP)-1$ $(PSW_{4,6,7}) + (PC)\leftarrow((SP))$	6

DEVELOPMENT DATA

	mnemonic	opcode (hex.)	bytes/ cycles	description	function	notes
PARALLEL INPUT/OUTPUT	IN A, Pp	08 09 0A	1/2	input port p data to accumulator	(A)←(P0) (A)←(P1) (A)←(P2)	7
	OUTL Pp, A	38 39 3A	1/2	output accumulator data to port p	(P0)←(A) (P1)←(A) (P2)←(A)	
	ANL Pp, #data	98 data 99 data 9A data	2/2	AND port p data with immediate data	(P0)←(P0) AND data (P1)←(P1) AND data (P2)←(P2) AND data	
	ORL Pp, #data	88 data 89 data 8A data	2/2	OR port p data with immediate data	(P0)←(P0) OR data (P1)←(P1) OR data (P2)←(P2) OR data	
DERIVATIVE INPUT/OUTPUT	MOV A, Dx	8C Dx	2/2	Move derivative register/port contents addressed by Dx to accumulator	(A)←(Dx) x = 4,5,16,17-28	8
	MOV Dx, A	8D Dx	2/2	Move contents of accumulator to derivative register addressed by Dx	(Dx)←(A) x = 4,16,17-28	8
	ANL Dx, A	8E Dx	2/2	AND contents of accumulator with derivative register addressed by Dx	(Dx)←(Dx) AND (A) x = 4,16,17-28	8
	ORL Dx, A	8F Dx	2/2	OR contents of accumulator with derivative register addressed by Dx	(Dx)←(Dx) OR (A) x = 4,16,17-28	8

INSTRUCTION SET (continued)

	mnemonic	opcode (hex.)	bytes/ cycles	description	function	notes
SERIAL INPUT/OUTPUT	MOV A, S_n	0C 0D	1/2	move serial I/O register contents to accumulator	(A)←(S0) (A)←(S1)	9
	MOV S_n, A	3C 3D 3E	1/2	move accumulator contents to serial I/O register	(S0)←(A) (S1)←(A) (S2)←(A)	
	MOV S_n, #data	9C data 9D data 9E data	2/2	move immediate data to serial I/O register	(S0)←data (S1)←data (S2)←data	
	EN SI	85	1/1	enable serial I/O interrupt		
	DIS SI	95	1/1	disable serial I/O interrupt		
	NOP	00	1/1	no operation		

Notes to Table 8

1. PSW CY, AC affected
2. PSW CY affected
3 PSW PS affected
4. Execution of JTF and JNTF instructions resets the Timer Flag (TF).

* : 8,9,A,B,C,D,E,F
● : 0,2,4,6,8,A,C,E
▲ : 0,3,5,7,9,B,D,F

5. PSW RBS affected
6. PSW SP_0, SP_1, SP_2 affected
7. (A) = 0000 (P23) 111
8. Dx = 04, 05, 16, 17-28
9. (S1) has a different function in read and write operations, see serial I/O interface.
10. SEL MB0 and SEL MB1 must not be used within interrupt routines.

RATINGS

Limiting values in accordance with the Absolute Maximum System (IEC 134)

parameter	symbol	min.	max.	unit
Supply voltage (pin 26)	V_{DD}	−0,8	+ 8	V
All input voltages	V_I	−0,8	V_{DD} + 0,8	V
DC current into input or output	$\pm I_I \pm I_O$	–	10	mA
Power dissipation per output	P_O	–	50	mW
Storage temperature	T_{stg}	−65	+ 150	°C
Ambient operating temperature range (if P_{tot}max. = 100 mW)	T_{amb}	−40	+ 70	°C
Ambient operating temperature range (if P_{tot}max. = 30 mW)	T_{amb}	−40	+ 85	°C
Operating junction temperature	T_j	–	90	°C

HANDLING

Inputs and outputs are protected against electrostatic discharge in normal handling. However, to be totally safe, it is desirable to take normal precautions appropriate to handling MOS devices (see 'Handling MOS devices').

DEVELOPMENT DATA

LCD DRIVER CHARACTERISTICS

V_{DD} = 2,5 to 5,5 V; V_{SS} = 0 V; T_{amb} = −40 to + 85 °C. All voltages with respect to V_{SS} unless otherwise specified.

parameter	symbol	min.	typ.	max.	unit
LCD supply voltage (note 1)	V_{LCD}	V_{SS}	–	V_{DD}−2,5	V
DC voltage component (BP0 to BP3) at V_{DD} = 5 V	$+/-V_{BP}$	0	20	–	mV
DC voltage component (S0 to S23) at V_{DD} = 5 V	$+/-V_S$	–	20	–	mV
Output impedance (BP0 to BP3) at $V_{LCD} = V_{SS}$ (note 2)	R_{BP}	–	–	5	kΩ
Output impedance (S0 to S23) at $V_{LCD} = V_{SS}$ (note 2)	R_S	–	–	7,0	kΩ
Driver delays with test loads at $V_{LCD} = V_{SS}$, I_{OS} = 15 μA, I_{OB} = 25 μA	tp_{LCD}	–	–	30	μs
LCD scan frame frequency	f_{LCD}	–	60	–	Hz

Notes to the LCD driver characteristics

1. $V_{LCD} < V_{DD}$−3 V for 1/3 bias.
2. Outputs measured one at a time.

DC CHARACTERISTICS

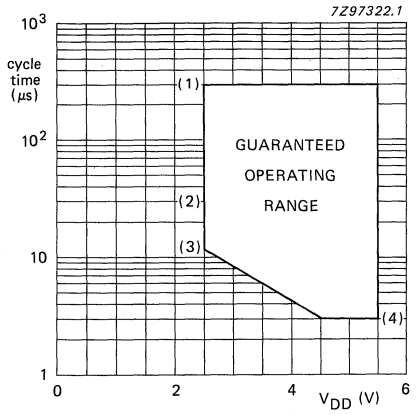
V_{DD} = 2,5 to 5,5 V; V_{SS} = 0 V; T_{amb} = −40 to + 85 °C; all voltages with respect to V_{SS}; unless otherwise specified.

parameter	symbol	min.	typ.	max.	unit
Supply voltage					
operating (see Fig. 37)	V_{DD}	2,5	–	5,5	V
Supply current (note 1)					
operating mode (see Figs 41 and 42)					
at V_{DD} = 5 V; f_{XTAL} = 10 MHz	I_{DD}	–	2,2	4,5	mA
at V_{DD} = 5 V; f_{XTAL} = 6 MHz	I_{DD}	–	1,3	2,6	mA
at V_{DD} = 3 V; f_{XTAL} = 3,58 MHz	I_{DD}	–	0,4	0,8	mA
IDLE mode (see Figs 39 and 40)					
at V_{DD} = 5 V; f_{XTAL} = 10 MHz	I_{DD}	–	0,8	1,6	mA
at V_{DD} = 5 V; f_{XTAL} = 6 MHz	I_{DD}	–	0,5	1	mA
at V_{DD} = 3 V; f_{XTAL} = 3,58 MHz	I_{DD}	–	0,15	0,4	mA
STOP mode (see Fig.38 and note 2)					
at V_{DD} = 2,5 V	I_{DD}	–	26	60	μA
at V_{DD} = 5 V	I_{DD}	–	30	80	μA
Inputs					
Input voltage LOW	V_{IL}	0	–	$0{,}3V_{DD}$	V
Input voltage HIGH	V_{IH}	$0{,}7V_{DD}$	–	V_{DD}	V
Input leakage current					
at $V_{SS} < V_I < V_{DD}$	$\pm I_{IL}$	–	–	1	μA
Outputs					
Output sink current LOW					
at V_{DD} = 5 V ± 10%; V_O = 0,4 V					
except P2.3/SDA, SCLK (see Fig.43)	I_{OL}	1,6	3	–	mA
P2.3/SDA, SCLK (see Fig.44)	I_{OL}	3	–	–	mA
Pull-up output source current HIGH (see Fig.45)					
at V_{DD} = 5 V ± 10%; $V_O = 0{,}7V_{DD}$	$-I_{OH}$	40	–	–	μA
at V_{DD} = 5 V ± 10%; $V_O = V_{SS}$	$-I_{OH}$	–	–	400	μA
Push-pull output source current HIGH					
at V_{DD} = 5 V ± 10%; $V_O = V_{DD}-0{,}4$ V	$-I_{OH}$	1,6	3	–	mA

Notes to the DC characteristics

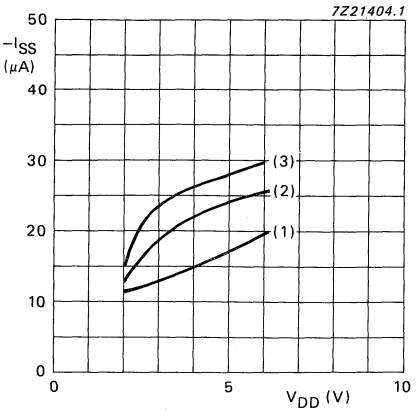
1. $V_{IL} = V_{SS}$, $V_{IH} = V_{DD}$; all outputs unloaded; all open drain outputs connected to V_{SS}.
2. Crystal connected between XTAL 1 and XTAL 2; SCLK and SDA pulled to V_{DD} via a 5,6 kΩ resistor; T1 at V_{SS}, $\overline{INT}$ at V_{DD}.

AC CHARACTERISTICS



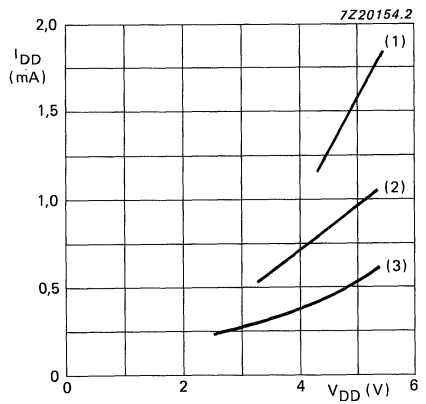
(1) clock frequency = 100 kHz
(2) clock frequency = 1 MHz
(3) clock frequency = 3 MHz
(4) clock frequency = 10 MHz

Fig. 37 Maximum clock frequency (f_{XTAL}) as a function of the supply voltage (V_{DD}).



$f_{CLK} \triangleq 30$ kHz (1) $T_{amb} = +85$ °C
(2) $T_{amb} = +25$ °C
(3) $T_{amb} = -40$ °C

Fig. 38 Typical supply current ($-I_{SS}$) in STOP mode as a function of the supply voltage (V_{DD}).



(1) clock frequency = 10 MHz
(2) clock frequency = 6 MHz
(3) clock frequency = 3,58 MHz

Fig. 39 Maximum supply current (I_{DD}) in IDLE mode as a function of the supply voltage (V_{DD}).

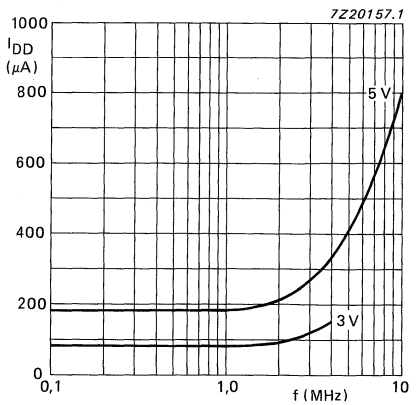
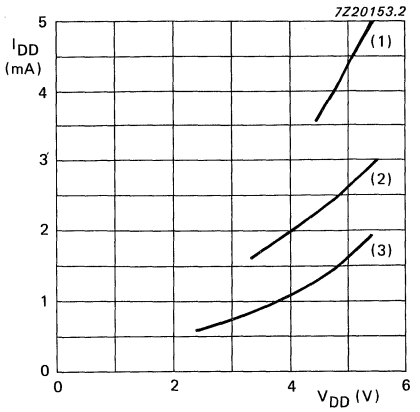


Fig. 40 Typical supply current during IDLE mode as a function of frequency at $V_{DD} = 3$ V and $V_{DD} = 5$ V.

DEVELOPMENT DATA

AC CHARACTERISTICS (continued)



(1) clock frequency = 10 MHz
(2) clock frequency = 6 MHz
(3) clock frequency = 3,58 MHz

Fig. 41 Maximum supply current (I_{DD}) in operating mode as a function of the supply voltage.

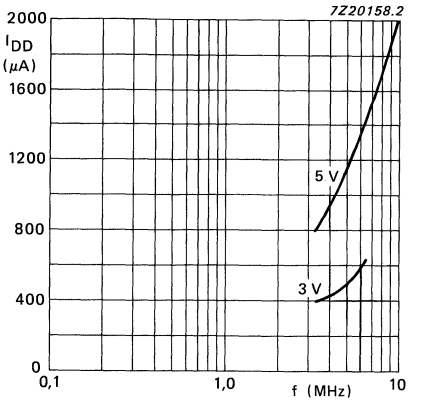


Fig. 42 Typical supply current during operating mode as a function of frequency at V_{DD} = 3 V and V_{DD} = 5 V.

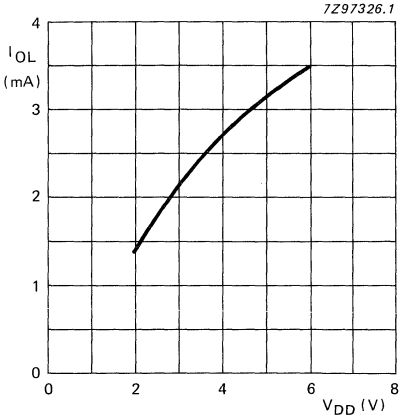


Fig. 43 Typical output sink current (I_{OL}), outputs P0.0 to P0.7, P1.0 to P1.7, DP3.0 to DP3.7, as a function of the supply voltage (V_{DD}); V_O = 0,4 V.

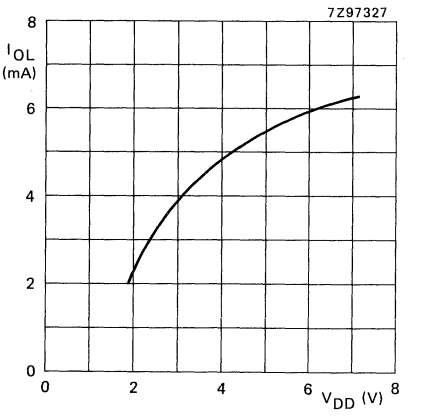
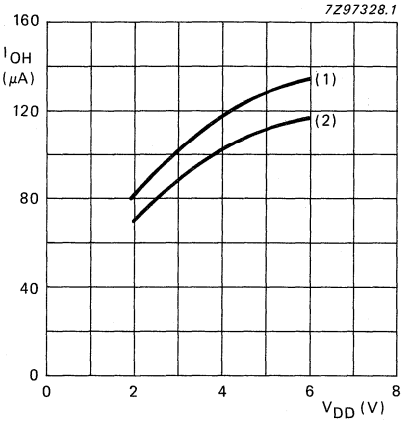
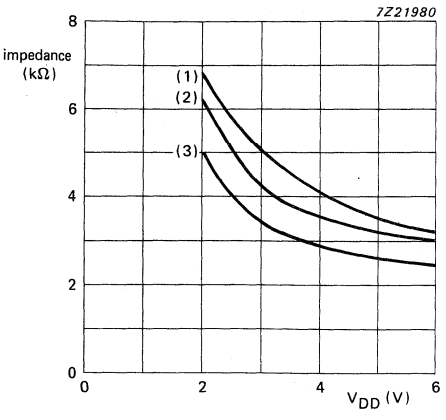


Fig. 44 Typical output sink current (I_{OL}), outputs P2.3/SDA and SCLK, as a function of the supply voltage (V_{DD}); V_O = 0,4 V.

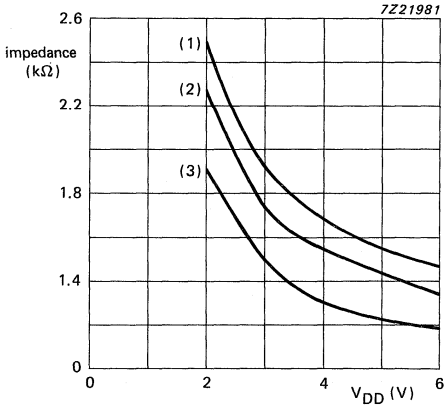


(1) $V_O = V_{SS}$
(2) $V_O = 0{,}7\ V_{DD}$
Fig. 45 Typical output source current ($-I_{OH}$) as a function of the supply voltage (V_{DD}).



(1) + 90 °C
(2) + 25 °C
(3) –50 °C
Fig. 46 Segment output impedance.

DEVELOPMENT DATA



(1) + 90 °C
(2) + 25 °C
(3) –50 °C
Fig. 47 Backplane output impedance.

Table 12 Input timing shown in Fig. 48.

symbol	timing
t_{BUF}	$\geqslant 14t_{XTAL}$
$t_{HD;STA}$	$\geqslant 14t_{XTAL}$
t_{HIGH}	$\geqslant 17t_{XTAL}$
t_{LOW}	$\geqslant 17t_{XTAL}$
$t_{SU;STO}$	$\geqslant 14t_{XTAL}$
$t_{HD;DAT}$	> 0
$t_{SU;DAT}$	$\geqslant 250$ ns
t_{RD}	$\leqslant 1\ \mu s$
t_{RC}	$\leqslant 1\ \mu s$
t_{FD}	$\leqslant 1\ \mu s$
t_{FC}	$\leqslant 0{,}3\ \mu s$

Notes to Table 12

t_{XTAL} = one period of the XTAL input frequency (f_{XTAL})
= 167 ns for f_{XTAL} = 6 MHz.
These figures apply to all modes.

AC CHARACTERISTICS (continued)

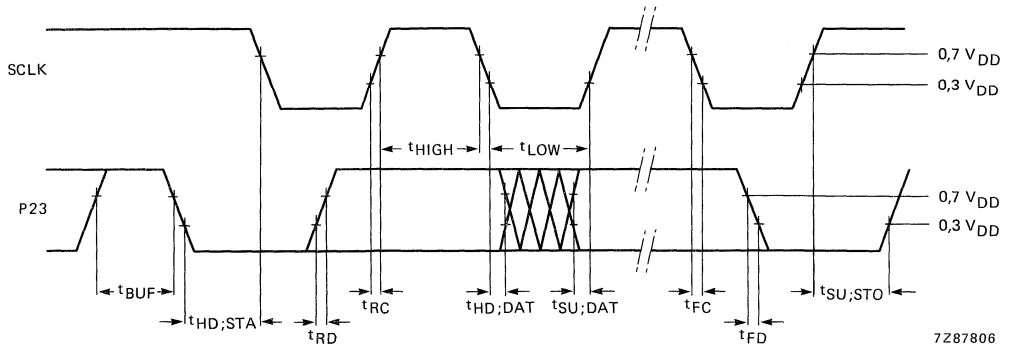


Fig. 48 PCF84C430 timing requirements for the P2.3 and SCLK *input* signals.

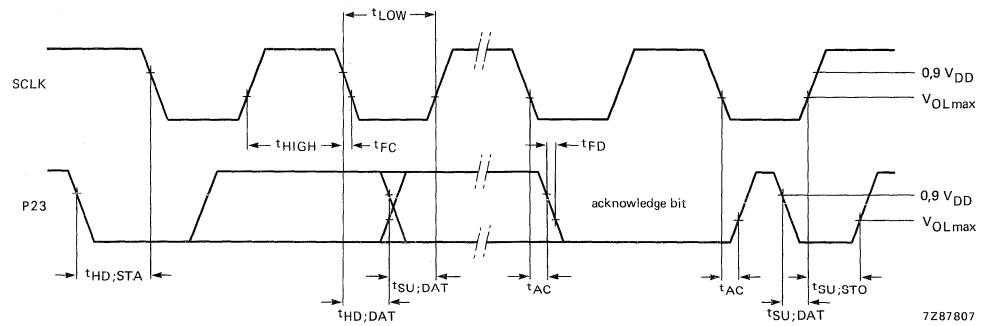


Fig. 49 PCF84C430 timing requirements for the P2.3 and SCLK *output* signals.

Purchase of Philips' I²C components conveys a license under the Philips' I²C patent to use the components in the I²C-system provided the system conforms to the I²C specifications defined by Philips.

Table 13 Output timing shown in Fig. 49

symbol	timing	
	normal mode (ASC in S2 = 0)	low-speed mode (ASC in S2 = 1
$t_{HD;STA}$	$\frac{1}{2}$ (DF + 9) t_{XTAL}	$\frac{3}{4}$ (DF + 9) t_{XTAL}
t_{HIGH}	$\frac{1}{2}$ (DF) t_{XTAL}	$\frac{3}{4}$ (DF) t_{XTAL}
t_{LOW}	$\frac{1}{2}$ (DF) t_{XTAL}	$\frac{1}{4}$ (DF) t_{XTAL}
$t_{SU;STO}$	$\frac{1}{2}$ (DF − 3) t_{XTAL}	$\frac{1}{4}$ (DF − 3) t_{XTAL}
$t_{HD;DAT}$ (slave transmitter)		
any DF	$\geqslant 9t_{XTAL}$ $\leqslant 12t_{XTAL}$	$\geqslant 9t_{XTAL}$ $\leqslant 12t_{XTAL}$
$t_{HD;DAT}$ (master transmitter)		
for DF $\leqslant$ 51	$\geqslant 9t_{XTAL}$ $\leqslant 12t_{XTAL}$	–
for DF $\leqslant$ 99	–	–
	–	$\geqslant 9t_{XTAL}$ $\leqslant 12t_{XTAL}$
$t_{SU;DAT}$ (master transmitter)		
for DF > 51	$\geqslant 15t_{XTAL}$ $\leqslant 24t_{XTAL}$	–
for DF > 99	–	–
	–	$\geqslant 15t_{XTAL}$ $\leqslant 24t_{XTAL}$
t_{AC}	$\geqslant 9t_{XTAL}$ $\leqslant 12t_{XTAL}$	$\geqslant 9t_{XTAL}$ $\leqslant 12t_{XTAL}$
$t_{FD;FC}$	$\leqslant$ 100 ns at C_b = 400 pF	$\leqslant$ 100 ns at C_b = 400 pF

Notes to Table 13

t_{XTAL} = one period of the XTAL input frequency (f_{XTAL})
= 167 ns for f_{XTAL} = 6 MHz.
DF = divisor (see Table 3 Serial I/O section).
C_b = the maximum bus capacitance for each line.

DEVELOPMENT DATA

SINGLE-CHIP 8-BIT MICROCONTROLLER WITH LCD DRIVERS, DERIVATIVE PORT, TIMER/CAPTURE AND TIMER/COUNTER

DESCRIPTION

The PCF84C633A is a microcontroller with 20 on-chip liquid crystal display (LCD) outputs. These can be configured for one to four backplanes and 19 to 16 segment lines, yielding a maximum of 64 display elements. In addition to the shared features of the PCF84CXX family of microcontrollers, the PCF84C633A includes a 16-bit timer with capture and compare registers, a 16-bit up/down counter/timer and two filtered control inputs. Together with additional derivative port lines, these powerful extensions make the device attractive for demanding real-time applications.

FEATURES

- 8-bit CPU, ROM, RAM, I/O in a single 56-lead package
- 6 K bytes ROM
- 256 bytes RAM
- Over 80 instructions (based on MAB 8048) all of 1 or 2 cycles
- 28 quasi-bidirectional I/O port lines
- 8-bit programmable timer/event counter
- 16-bit derivative timer with capture and compare registers (T2)
- 16-bit up/down counter/timer (T3)
- 2 filtered input lines coupled to T2 and T3
- 3 single-level vectored interrupts; external, 8-bit programmable timer/event counter, derivative (triggered by 4 events in T2 and T3)
- 2 test inputs of which one also serves as the external interrupt input
- 20 LCD output configurable for one to four backplanes and 19 to 16 segment lines
- Drive for up to 64 display elements
- Display memory bank switching in static and duplex drive modes
- 19 of the LCD outputs may serve as additional low-drive logic outputs with optional level-shift
- Stop and idle modes
- Logic supply V_{DD}; 2.5 V to 5.5 V
- Independent LCD supply V_{DDLC}; 2.5 V to 5.5 V
- Clock frequency: 1 MHz to 16 MHz
- Operating temperature range: −40 °C to 85 °C
- Manufactured in silicon gate CMOS process

IMPORTANT

This data sheet details the specific properties of the PCF84C633A. The shared characteristics of the 84CXXX family of microcontrollers are described in 84CXXX family specification, which should be read in conjunction with this publication.

PACKAGE OUTLINE

PCF84C633AT: 56-lead mini-pack; plastic (VSO56; SOT190)

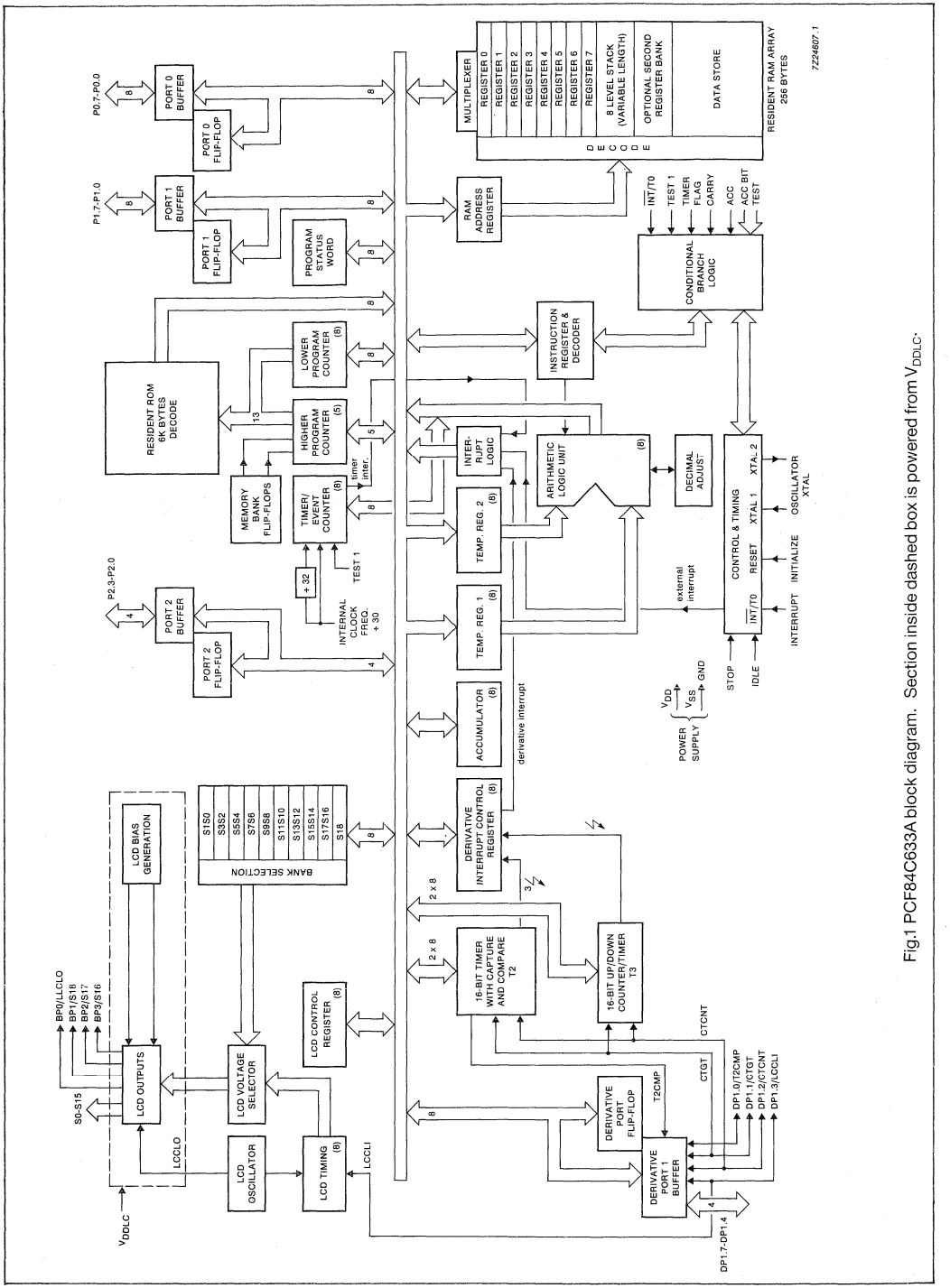


Fig.1 PCF84C633A block diagram. Section inside dashed box is powered from V_{DDLC}.

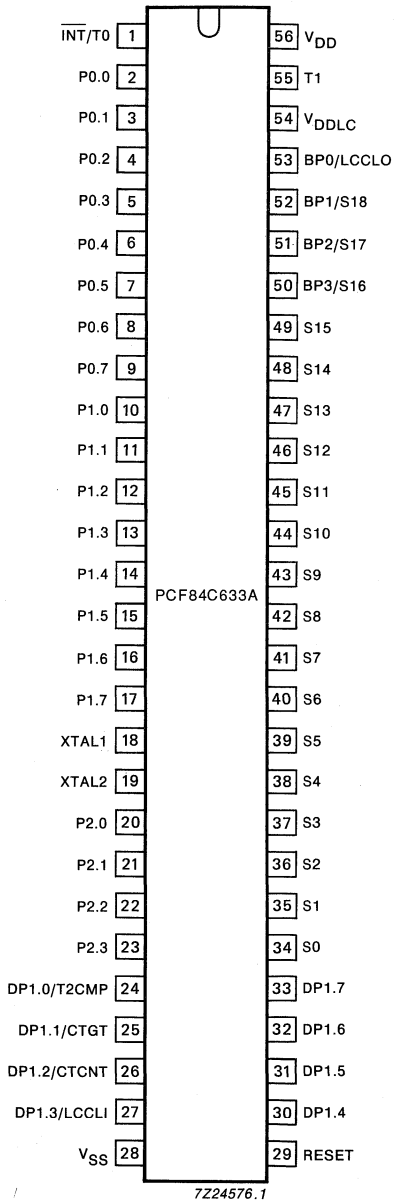


Fig.2 Pinning diagram.

Pinning

PIN	SYMBOL	TYPE	FUNCTION
1	INT/T0	I	Interrupt/Test 0
2-9	P0.0-P0.7	I/O	Port 0; 8-bit quasi-bidirectional I/O port
10-17	P1.0-P1.7	I/O	Port 1; 8-bit quasi-bidirectional I/O port
18	XTAL1	I	Crystal oscillator/external clock input
19	XTAL2	O	Crystal oscillator output
20-23	P2.0-P2.3	I/O	Port 2; 4-bit quasi-bidirectional I/O port
24	DP1.0/T2CMP	I/O	Derivative Port 1; quasi-bidirectional I/O line/compare output of T2
25	CP1.1/CTGT	I/O	Derivative Port 1; quasi-bidirectional I/O line/capture input (T2) and/or gate input (T3)
26	DP1.2/CTCNT	I/O	Derivative Port 1; quasi-bidirectional I/O line/capture input (T2) and/or count input (T3)
27	DP1.3/LCCLI	I/O	Derivative Port 1; quasi-bidirectional I/O line/LCD clock input
28	V_{SS}	P	Ground
29	RESET	I	Reset input
30-33	DP1.4-DP1.7	I/O	Derivative Port 1; quasi-bidirectional I/O port
34-49	S0-S15	O	LCD segment outputs
50-52	BP3/S16-BP1/S18	O	LCD backplane/segment outputs
53	BP0/LCCLO	O	LCD backplane/LCD oscillator output
54	V_{DDLC}	P	LCD supply voltage
55	T1	I	Test 1/count input of 8-bit timer/event counter
56	V_{DD}	P	Logic supply voltage

PARALLEL PORTS

All standard quasi-bidirectional I/O ports are available:

- Port 0 parallel port of 8 lines (P0.0 to P0.7)
- Port 1 parallel port of 8 lines (P1.0 to P1.7)
- Port 2 parallel port of 4 lines (P2.0 to P2.3)

Since no serial I/O interface is provided, P2.3 is a general purpose line without restrictions.
In addition to the standard ports, a derivative I/O port is available:

- DP1 derivative port of 8 lines (DP1.0/T2CMP, DP1.1/CTGT, DP1.2/CTCNT, DP1.3/LCCLI, DP1.4 to DP1.7)

Table 1 summarizes the derivative addresses of DP1.

Table 1 Derivative port address pair.

Dx ADDRESS (HEX)	TYPE	DESCRIPTION
02	R	DP1 derivative port lines
03	R/W	DP1 derivative port flip-flop

Four lines of derivative Port 1 are shared with signals used for timer T2 , up/down counter/timer T3 and the LCD driver section; DP1.0/T2CMP, DP1.1/CTGT, DP1.2/CTCNT and DP1.3/LCCLI. Before an alternative function can be used, the output driver FET of the corresponding port-line must be turned off by writing a '1' to the port latch. The port pin is then pulled high by the internal pull-up (standard I/O and push-pull I/O) or is floating (open drain I/O), but may be driven by an external signal. In their non-port functions, DP1.1/CTGT, DP1.2/CTCNT and DP1.3/LCCLI serve as inputs. Therefore, they may only be configured as standard outputs (option 1) or open drain outputs (option 2). All other standard or derivative port lines are eligible for all three mask options.

The alternative functions are controlled by the corresponding control bits. When an alternative function is used the corresponding I/O port is not disabled, so the state of the pin and latch may still be read using the MOV A,DX instruction. Note, if a MOV DX,A instruction sets the port latch to 'O', it can no longer be driven by an external signal anymore.

DP1.1/CTGT AND DP1.2/CTCNT INPUT FILTERS

The T2 capture transitions and the T3 gating and count signals are digitally filtered, to remove spikes shorter than one machine cycle. The signals which are presented on DP1.1/CTGT and DP1.2/CTCNT are sampled on the positive edge of the internal machine cycle signal ($f_{XTAL}/30$). A change on the input lines must be stable for two consecutive samples to be accepted. Therefore the filter delay ranges from one to two machine cycles (see Fig.3). The T2 capture and a T3 counting pulses are delayed by two to three machine cycles with respect to the corresponding input signal transition (see Fig.3). The internal signals GT, CT and CNT occur only if the corresponding derivative function is enabled.
The filter delays shown in Fig.3 are only valid if the corresponding function is enabled.

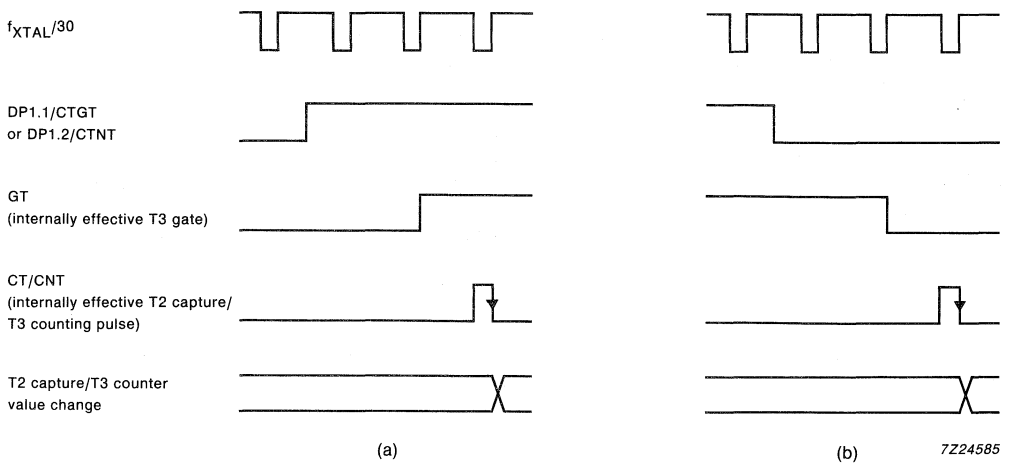


Fig.3 Filter delays at positive edge (a) or negative edge (b) on DP1.1/CTGT and DP1.2/CTCNT.

16-BIT TIMER T2

A derivative 16-bit timer (T2) with capture and compare registers is provided. Communication between the CPU and T2 is handled through derivative registers, making use of the derivative input/output instructions. 16-bit values are accessed by two consecutive byte accesses. Fig.4 gives the block diagram of the T2 section.
Typical applications of T2 are for measuring signal deviations (in conjunction with the capture register) or for generating pulse deviations (in conjunction with the compare register).

T2 Derivative Registers

Table 2 summarizes the derivative addresses, the register mnemonics and the access types for the T2 section.

Table 2 Derivative addresses for the T2 section.

Dx ADDRESS (HEX)	TYPE	MNEMONIC	DESCRIPTION
04	R	T2H	T2 timer register high byte
05	R	T2L	T2 timer register low byte
06	R	T2LB	T2 low byte register
07	R/W	T2CMH	T2 compare register high byte
08	R/W	T2CML	T2 compare register low byte
09	R	T2CTH	T2 capture register high byte
0A	R	T2CTL	T2 capture register low byte
0B	R/W	T2CON	T2 control register

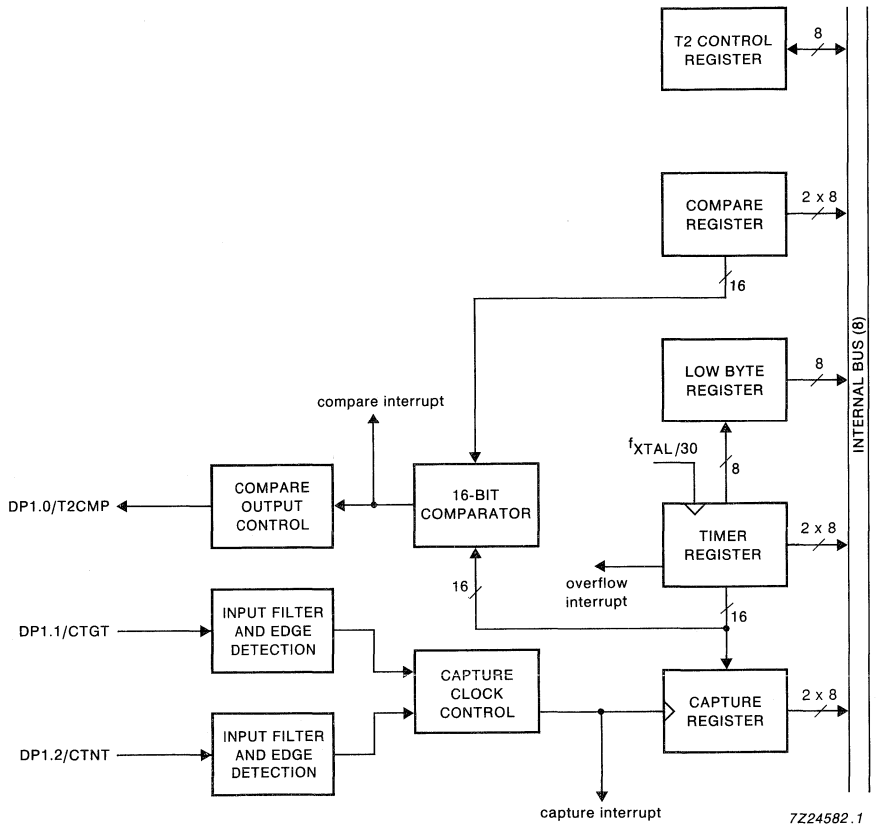


Fig.4 Block diagram of the derivative 16-bit timer T2.

Timer Register and Low Byte Register

The central unit of T2 is the 16-bit timer register. Its behaviour depends on the state of bits ET2 and RUN in the T2 control register (see Table 3). If ET2 = 0, the timer register is disabled and cleared. Otherwise the timer register is enabled and keeps its value (RUN = 0) or increments (RUN = 1) every machine cycle ($f_{XTAL}/30$).

When the timer register overflows from FFFFH to 0000H, the T2 overflow flag T20V is set in the derivative interrupt control register (see section on derivative interrupts). This event can be used to generate one of four different derivative interrupt request. The source of the interrupt is found by reading DIRCON (see section Derivative Interrupts).

The 16-bit timer register is subdivided into two 8-bit registers. T2H contains the high byte of the timer value, T2L contains the low byte. Both are read-only registers and can be read anytime without disturbing the timer function. Whenever T2H is read, T2L is simultaneously copied into the low byte register T2LB. This allows the full 16-bit timer value to be read consistently. The value in T2LB remains fixed until T2H is read again.

Capture Function

The 16-bit capture register latches the value of the 16-bit timer register when a predefined transition on DP1.1/CTGT or DP1.2/CTCNT occurs. Bits CT0 and CT1 in the T2 control register (see Table 3) are used to either disable the capture function or to select the positive, the negative or both edges of the input signal to latch the timer value into the capture register.

Bit CTS in the T2 control register selects DP1.1/CTGT (CTS = 1) or DP1.2/CTCNT (CTS = O) as the source for the capture clock. Signals on DP1.1/CTGT and DP1.2/CTCNT are filtered (see Fig.4). For a transition to be detected, the signal on DP1.1/CTGT and DP1.2/CTCNT must be stable for at least two machine cycles ($60/f_{XTAL}$) before and after the selected edge. If several consecutive capture conditions occur, the capture register always contains the most recent value.

When the capture trigger occurs, the capture interrupt flag CTI is set in the derivative interrupt control register (see section on derivative interrupts). This event can be used to generate a derivative interrupt request. The source of the interrupt is found by reading DIRCON (see section Derivative Interrupts).

The 16-bit register is subdivided into two 8-bit registers. T2CTH contains the high byte of the capture value, T2CTL contains the low byte. Both are read-only. When T2CTH is read, the capture function is inhibited until T2CTL is also read. This allows the full 16-bit capture value to be consistently read.

Compare Function

If (ET2 = 1).(RUN = 1) in the T2 control register, a match between the timer register and the 16-bit compare register will set the compare interrupt flag CMI in the derivative interrupt control register (see section on Derivative Interrupts). This event can be used to generate a derivative interrupt request. The source of the interrupt is found by reading DIRCON (see section Derivative Interrupts).

The 16-bit compare register is subdivided into two 8-bit registers. T2CMH contains the high byte of the compared value, T2CML contains the low byte. Both can be read or written. If the compare function is not used, T2CMH and T2CML may be used as storage locations. When T2CMH is written, the compare function is inhibited until T2CML is also written. This allows the full 16-bit compare value to be consistently defined.

If bit ECO = 1 in the T2 control register, a compare event can generate an output on DP1.0/T2CMP. Bit CM0 and CM1 in the T2 control register select between no change, output low, output high or output toggle. This may be useful to generate a pulse width or to signal a lapse of time.

T2 Control Register (T2CON)

The 8-bit T2 control register defines the behaviour of the T2 section. It can be read or written. Fig.5 gives the structure of the T2 control register, and Table 3 summarizes the significance of the individual control bits.

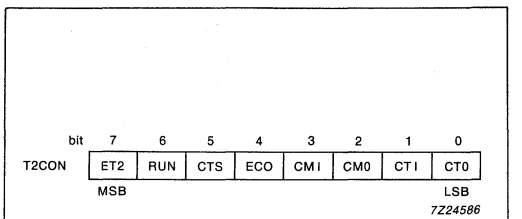


Fig.5 T2 control register (T2CON).

Table 3 Overview of T2 control register bits.

BIT	NAME	DESCRIPTION
ET2	Enable T2	ET2 = 0: timer register disabled and cleared ET2 = 1: timer register enabled (see RUN)
RUN	RUN	RUN = 0: timer register value held RUN = 1: timer register increments if ET2 = 1
CTS	Capture Source	CTS = 0: DP1.2/CTCNT capture signal source CTS = 1: DP1.1/CTGT capture signal source
ECO	Enable Compare Out	ECO = 0: DP1.0/T2CMP derivative port line ECO = 1: DP1.0/T2CMP compare output
CM1 CM0	Compare 1 Compare 0	**CM1** **CM0** 0 0 no change on DP1.0/T2CMP after compare match 0 1 low level on DP1.0/T2CMP after compare match if ECO = 1 1 0 high level on DP1.0/T2CMP after compare match if ECO = 1 1 1 toggles DP1.0/T2CMP after compare match if ECO = 1
CT1 CT0	Capture 1 Capture 0	**CT1** **CT0** 0 0 capture function disabled 0 1 capture on positive edge of capture signal source 1 0 capture on negative edge of capture signal source 1 1 capture on any edge of capture signal source

16-BIT UP/DOWN COUNTER/TIMER T3

A derivative 16-bit up/down counter/timer (T3) is provided. Communication between the CPU and T3 is handled through derivative registers, making use of the derivative input/output instructions. The 16-bit counter/timer value is accessed by two consecutive byte accesses. Fig.6 gives the block diagram of the T3 section.

Typical applications of T3 are for measuring signal durations (in timer mode in conjunction with the gating signal on DP1.1/CTGT), for interval generation (in timer mode) and for counting signal edges (in counter mode).

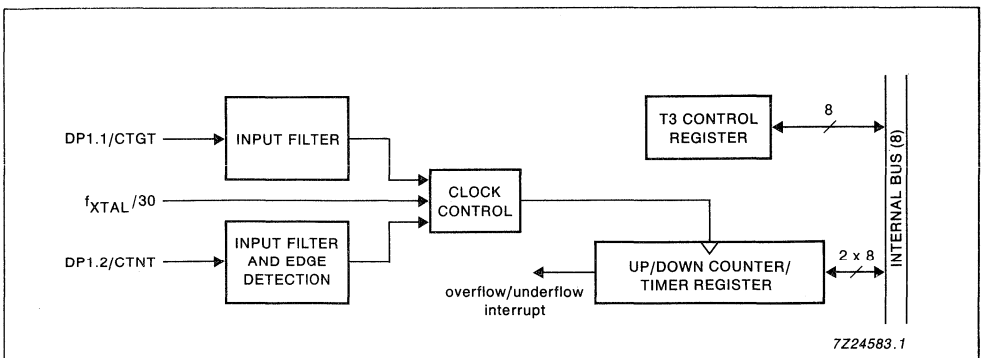


Fig.6 Block diagram of the derivative 16-bit up/down counter/timer T3.

T3 Derivative Registers

Table 4 summarizes the derivative addresses, the register mnemonics and the access types for the T3 section.

Table 4 Derivative addresses for the T3 section.

Dx ADDRESS (HEX)	TYPE	MNEMONIC	DESCRIPTION
0D	R/W	T3H	T3 counter/timer register high byte
0E	R/W	T3L	T3 counter/timer register low byte
0F	R/W	T3CON	T3 control register

Up/Down Counter/Timer Register

The central unit of T3 is the 16-bit up/down counter/timer register. Its behaviour is defined by the state of mode bits MD0, MD1 and MD2 in the T3 control register (see Tables 5 and 6 and also Fig.7).

In timer mode (MD2 = 0), the register increments (MD1 = 1) or decrements (MD0 = 1) every machine cycle ($f_{XTAL}/30$).

In counter mode(MD2 = 1), the register increments (MD1 = 1) with the positive edge on DP1.2/CTCNT. The register decrements (MD = 1) with the negative edge on DP1.2/CTCNT. For a transition to be detected, the clock on DP1.2/CTCNT must be stable for at least two machine cycles ($60/f_{XTAL}$) before and after the signal change.

In both timer and counter modes, counting can be inhibited if the gating signal on DP1.1/CTGT equals zero. This feature is turned on by bit ET3GT (enable T3 gate) in the T3 control register. Being digitally filtered, the state of DP1.1/CTGT must be stable for at least two machine cycles ($60/f_{XTAL}$) before it is recognized as a valid gating signal. It is possible to simultaneously enable counting up and counting down in count mode (MD1 = MD0 = 1). When this is done, the gating signal should be enabled by setting bit ET3GT (in DIRCON) to '1'. Then signal edges will only be counted while the gating signal (DP1.1/CTGT) is '1'.

In both timer and counter modes, an overflow (from FFFFH to 0000H) or an underflow (from 0000H to FFFFH) assert the overflow flag T30V in the derivative interrupt control register (see section on derivative interrupts). This event can be used to generate a derivative interrupt request. The source of the interrupt is found by reading DIRCON (see section Derivative Interrupts).

The 16-bit up/down counter/timer register is subdivided into two 8-bit registers. T3H contains the high byte of the counter/timer value, T3L contains the low byte. Both can be read or written. It is the responsibility of the user software to ensure that accesses to T3H and T3L do not violate data consistency. One way to guarantee this is by disabling the T3 section for the duration of the access. If the up/down counter/timer is not used, T3H and T3L may serve as storage locations.

Table 5 Truth table T3 Up/Down Timer.

T3 CLOCK INPUT		T3 CONTROL REGISTER				T3 GATE	TIMER MODE
$f_{XTAL}/30$	DP1.2/ CTCNT	MD2	MD1	MD0	ET3GT	DP1.1/ CTGT	
⎍	X	0	1	0	0	X	Timer Up
⎍	X	0	1	0	1	1	Timer Up
⎍	X	0	1	0	1	0	Inhibit Timer Up by DP1.1/CTGT
⎍	X	0	0	1	0	X	Timer Down
⎍	X	0	0	1	1	1	Timer Down
⎍	X	0	0	1	1	0	Inhibit Timer Down by DP1.1/CTGT
X	X	X	0	0	X	X	Inhibit Timer

'X' denotes don't care states

Table 6 Truth table Up/Down Counter.

T3 CLOCK INPUT		T3 CONTROL REGISTER				T3 GATE	COUNTER MODE
$f_{XTAL}/30$	DP1.2/ CTCNT	MD2	MD1	MD0	ET3GT	DP1.1/ CTGT	
X	↓	1	1	0	0	X	Count Up
X	↓	1	1	0	1	1	Count Up
X	↓	1	1	0	1	0	Inhibit Count Up by DP1.1/CTGT
X	↑	1	0	1	0	X	Count Down
X	↑	1	0	1	1	1	Count Down
X	↑	1	0	1	1	0	Inhibit Count Down by DP1.1/CTGT
X	↓	1	1	1	1	1	Count Up
X	↑	1	1	1	1	1	Count Down
X	X	1	1	1	1	0	Inhibit Count Down by DP1.1/CTGT
X	X	X	0	0	X	X	Inhibit Counter

'X' denotes don't care states

T3 Control Register (T3CON)

The 8-bit T3 control register defines the behaviour of the T3 section. It can be read or written. Fig.7 gives the structure of T3CON, and Table 7 summarizes the significance of the individual control bits.

Bits 5 to 7 are fixed at zero.

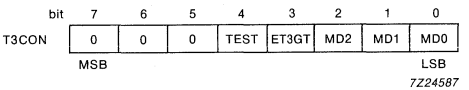


Fig.7 T3 control register (T3CON).

Table 7 Overview of T3 control register bits.

BIT	NAME	DESCRIPTION			
TEST	Test	TEST = 0: test mode disabled (state for user software) TEST = 1: test mode enabled (not allowed in user software)			
ET3GT	Enable T3 Gate	ET3GT = 0: state of DP1.1/CTGT irrelevant to T3 ET3GT = 1: counter/timer T3 inhibited/enabled if DP1.1/CTGT = 0/1			
MD2 MD1 MD0	Mode 2 Mode 1 Mode 0	**MD2**	**MD1**	**MD0**	
		X	0	0	counter/timer T3 inhibited
		0	0	1	decrementing timer mode
		0	1	0	incrementing timer mode
		0	1	1	not allowed
		1	0	1	decrementing counter mode
		1	1	0	incrementing counter mode
		1	1	1	decrementing and incrementing counter mode (use with ET3GT = 1)

DERIVATIVE INTERRUPTS

Since the PCF84C633A includes no serial I/O interface, the SIO/derivative interrupt (see 84CXX family data sheet) reduces to a derivative interrupt. Four derivative interrupt events are defined. These events are controlled by the derivative interrupt control register (DIRCON). Table 8 summarizes the significance of the individual control bits.

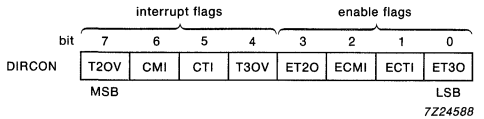


Fig.8 Derivative interrupt control register (DIRCON).

A derivative interrupt request is honoured if:-

- no interrupt routine proceeds
- no external interrupt request is pending
- the SIO/derivative interrupt is enabled
- the corresponding enable bit in the derivative interrupt control register is set

The derivative interrupt routing must include instructions that will remove the cause of the derivative interrupt by implicitly clearing the corresponding interrupt flag. Table 9 gives derivative address, the register mnemonics and the access type for the derivative interrupt control register.

DIRCON can be read or written. Read access is necessary to determine the cause of a derivative interrupt request. If the derivative interrupt is not used, the flags may be directly tested by the program. Write access is necessary to remove the cause of the derivative interrupt request. The flags may also be directly written to generate a software interrupt.

Table 8 Overview of derivative interrupt control register bits.

BIT	NAME	DESCRIPTION
T20V	T2 Overflow	Set: if T2 timer register overflows from FFFFH to 0000H (or by program) Reset: by program and by RESET
CMI	Compare Interrupt	Set: if T2 timer register equals T2 compare register while (ET2 = 1) and (RUN = 1) in the T2 control register (or by program) Reset: by program and by RESET
CTI	Capture Interrupt	Set: if capture trigger occurs (or by program) Reset: by program and by RESET
T30V	T3 Overflow	Set: if T3 up/down counter/timer register over flows from FFFFH to 0000H or underflows from 0000H to FFFFH (or by program) Reset: by program and by RESET
ET20	Enable T2 Overflow	ET20 = 0: T20V event cannot request interrupt ET20 = 1: T20V event requests interrupt
ECMI	Enable Compare Interrupt	ECMI = 0: CMI event cannot request interrupt ECMI = 1: CMI event requests interrupt
ECTI	Enable Capture Interrupt	ECTI = 0: CTI event cannot request interrupt ECTI = 1: CTI event requests interrupt
ET30	Enable T3 Overflow	ET30 = 0: T30V event cannot request interrupt ET30 = 1: T30V event requests interrupt

Table 9 Derivative interrupt control register address.

Dx ADDRESS (HEX)	TYPE	MNEMONIC	DESCRIPTION
0C	R/W	DIRCON	Derivative interrupt control register

LCD DRIVER SECTION

A versatile derivative LCD driver section is provided which interfaces the PCF84C633A to a wide variety of LCDs. The 20 LCD outputs can be configured as one backplane/19 segments (static drive), as two backplanes/18 segments (1:2 multiplex), as three backplanes/17 segments (1:3 multiplex) and as four backplanes/16 segments (1:4 multiplex).

It is also possible to configure the LCD section as up to 19 low-drive logic outputs including a level-shift to V_{DDLC}. These may be useful as an extension of port output when a large number of signal lines must be controlled, possibly referenced to another supply.

Table 10 summarizes the most typical applications of the LCD driver section. As indicated in the block diagram (Fig.9), communication between the CPU and the LCD driver section is handled through derivative registers, making use of the derivative input/output instructions.

Table 10 Typical applications of the LCD driver section.

ACTIVE BACKPLANES	NO. OF SEGMENTS	7-SEGMENT NUMERIC	14-SEGMENT ALPHANUMERIC	DOT MATRIX
4	16	8 digits + 8 indicator symbols	4 characters + 8 indicator symbols	64 dots (4 x 16)
3	17	6 digits + 9 indicator symbols	3 characters + 9 indicator symbols	51 dots (3 x 17)
2	18	4 digits + 8 indicator symbols	2 characters + 8 indicator symbols	36 dots (2 x 18)
1	19	2 digits + 5 indicator symbols	1 characters + 5 indicator symbols	19 dots (1 x 19)
–	19 low-drive logic outputs with level-shift (referenced to V_{SS} and V_{DDLC})			

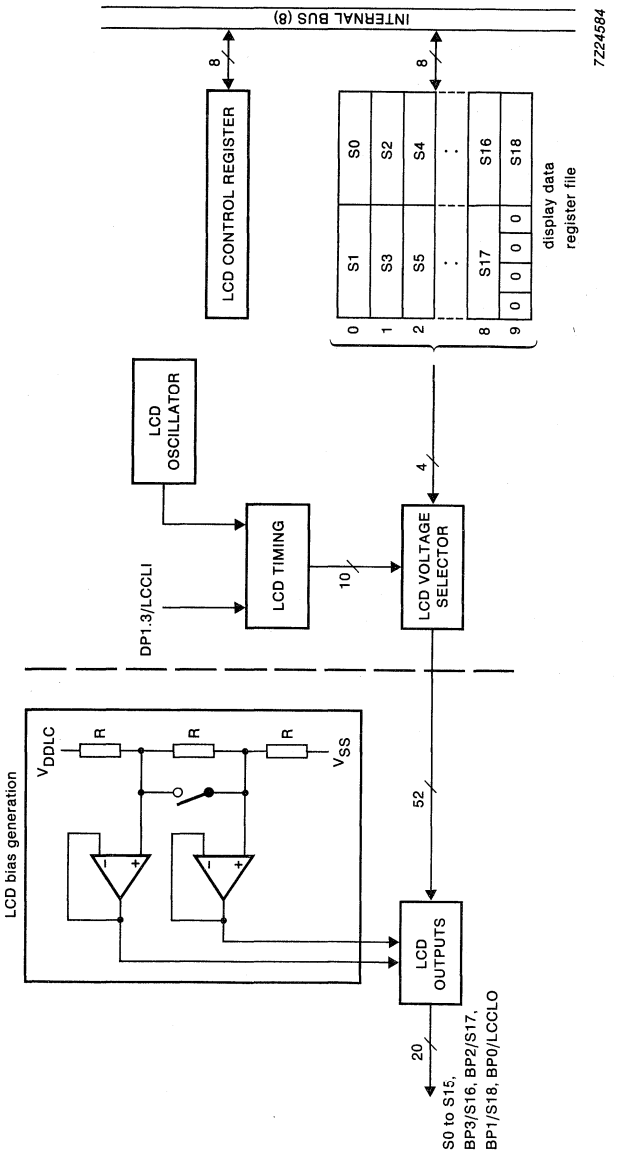


Fig.9 Block diagram of the LCD driver section.

LCD Derivative Registers

Table 11 summarizes the derivative addresses, the register mnemonics and the access types for the LCD driver section.

Table 11 Derivative addresses for the LCD driver section.

Dx ADDRESS (HEX)	TYPE	MNEMONIC	DESCRIPTION
20	R/W	S1S0	Display data nibbles for S0 and S1
21	R/W	S3S2	Display data nibbles for S2 and S3
22	R/W	S5S4	Display data nibbles for S4 and S5
23	R/W	S7S6	Display data nibbles for S6 and S7
24	R/W	S9S8	Display data nibbles for S8 and S9
25	R/W	S11S10	Display data nibbles for S10 and S11
26	R/W	S13S12	Display data nibbles for S12 and S13
27	R/W	S15S14	Display data nibbles for S14 and S15
28	R/W	S17S16	Display data nibbles for S16 and S17
29	R/W	S18	Display data nibble for S18
2A	R/W	LCDCON	LCD control register

Display Data Register File

The display data register file defines the LCD segment outputs. There is a one-to-one correspondence between segment outputs (S0 to BP1/S18) and nibbles in the display data register file (see Fig.10).

The four nibble bits A, B, C and D represent the four banks of data behind every LCD segment output. In 1:4 multiplex, banks A to D represent the states of the LCD elements multiplexed with backplanes BP0 to BP3/S16, respectively. Likewise, in 1:3 multiplex, banks A to C correspond to LCD elements multiplexed with BP0 to BP2/S17. In this case, bank D will not be used for LCD output.

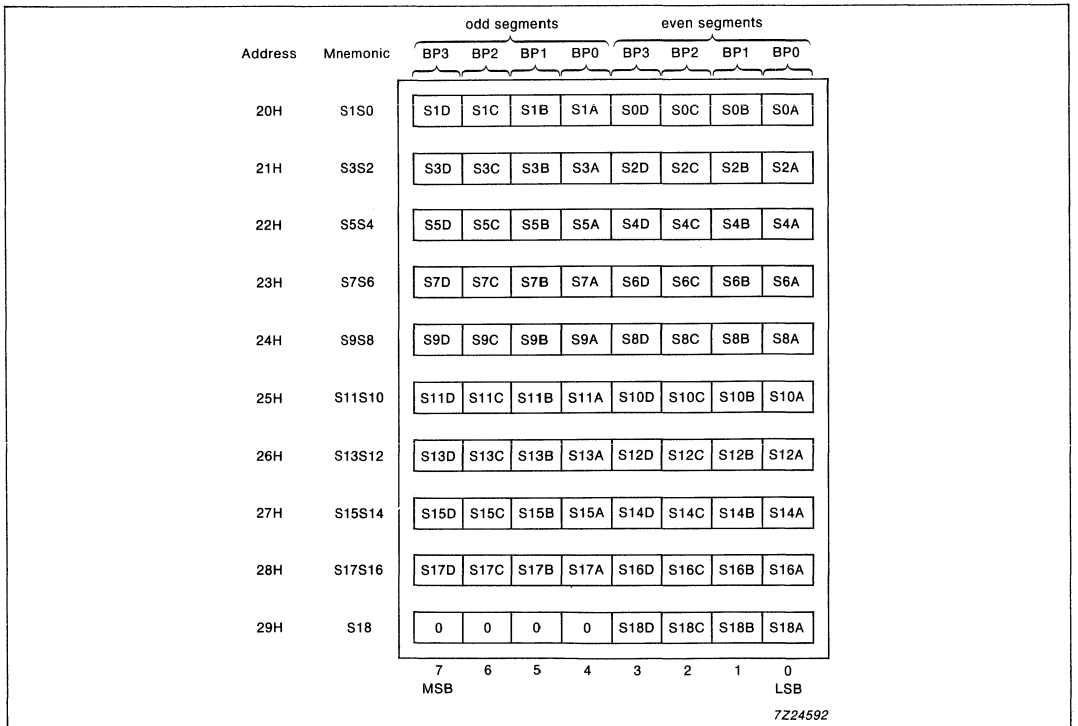


Fig.10 Display data register file.

In 1:2 multiplex, either bank pair A/B or bank pair C/D may be multiplexed with BP0 and BP1/S18. Likewise, in static drive and in low-drive logic mode, a choice of four banks exists. Bank selection allows preparation of additional information in alternative banks and to efficiently switch back and forth, making use of bits BK0 and BK1 in the LCD control register. This feature may be useful to efficiently blink groups of display elements.

All locations of the display data register file can be read or written. Note that bits 4 to 7 of the location at derivative address 29H are fixed at zero (see Fig.10). A logic one or zero in a bit of the display register file respectively translate into an on or off-waveform for the corresponding LCD segment output. In low-drive logic mode, a logic one or zero in a bit of the display data register file respectively translate into a HIGH or LOW level for the corresponding output.

Display data register file locations that are unused for output may serve as general purpose storage. In particular, this applies to nibbles corresponding to segment outputs shared with active backplane outputs, in bank D in 1:3 multiplex and to non-selected banks in static, duplex and low-drive logic modes.

LCD Control Register (LCDCON)

The LCD control register completely defines the behaviour of the LCD driver section. It can be read or written. Fig.11 gives the structure of LCDCON, and Table 12 summarizes the significance of the individual control bits.

Bit 7 fixed at zero

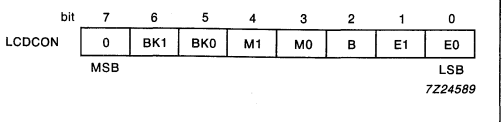


Fig.11 LCD control register (LCDCON).

Table 12 Overview of LCD control register bits.

BIT	NAME	DESCRIPTION							
BK1 BK0	Bank 1 Bank 0	**BK1**	**BK0**	**Bank**					
		0	0	A	selected in static drive and in low-drive mode				
		0	1	B	selected in static drive and in low-drive mode				
		1	0	C	selected in static drive and in low-drive mode				
		1	1	D	selected in static drive and in low-drive mode				
		X	0	pair A/B	selected in 1:2 multiplex mode				
		X	1	pair C/D	selected in 1:2 multiplex mode				
		X	X	triplet A/B/C	selected in 1:3 mode				
		X	X	quartet A/B/C/D	selected in 1:4 mode				
M1 M0	Multiplex 1 Multiplex 0	**M1**	**M0**						
		0	0		1:4 multiplex (4 BPs)				
		0	1		static drive (1 BP)				
		1	0		1:2 multiplex (2 BPs)				
		1	1		1:3 multiplex (3 BPs)				
		X	X		irrelevant in low-drive logic mode (E1 = 0, E0 = 1)				
B	Bias	B = 0		1/3 bias	in 1:2, 1:3 and 1:4 multiplex				
		B = 1		1/2 bias	in 1:2, 1:3 and 1:4 multiple				
		B = X		–	irrelevant in static drive and in low-drive logic mode (E1 = 0, E0 = 1)				
E1 E0	Enable 1 Enable 0	**E1**	**E0**	**Mode**	**S0 to BP1/S18**	**BP0/LCCLO**	**LCD oscillator**	**DP1.3/LCCLI**	
		0	0	disable (LCD blank)	at V_{SS}	at V_{SS}	stopped	derivative line	
		0	1	low-drive logic	logic outputs	LCD clock output	running	derivative port line	
		1	0	LCD drive 1	LCD outputs	BP0 output	stopped	LCD clock input	
		1	1	LCD drive 2	LCD outputs	BP0 output	running	derivative	

'X' denotes don't care states.

LCD Timing

The LCD timing organizes the internal data flow of the LCD driver section. This includes the transfer of data from the display data register file to the LCD outputs.
In LCD drive modes (see bits E0 and E1 in the LCD control register), the LCD frame corresponds to 24 periods of the LCD clock. At a nominal LCD frame frequency of 64 Hz, the LCD clock frequency must be about 1.5 kHz. The LCD clock must always be supplied otherwise, the LCD may be frozen in a DC state. As a clocking source, either an external clock on DP1.3/LCCLI (LCD drive mode 1, Table 12) or the internal LCD oscillator (LCD drive mode 2, Table 12) may be selected. If an external LCD clock is used, the port flip-flop corresponding to DP1.3/LCCLI should remain set to avoid conflict between input drive and port output.

LCD Oscillator

A low-power oscillator delivering a nominal frequency of 1.5 kHz is provided as a clocking source for the LCD timing. This clock can be made available on BP0/LCCLO in low-drive logic mode (see bit E0, E1 in the LCD control register).
The LCD register is stopped in disable and LCD drive 1 modes (see Table 12).

LCD Bias Generation

The full scale LCD voltage (V_{op}) is obtained from $V_{DDLC} - V_{SS}$. V_{op} may be temperature compensated externally through the V_{DDLC} supply. Fractional LCD biasing voltages are generated by an internal voltage divider of three series connected resistors between V_{DDLC} and V_{SS}. The center resistor can be switched out of circuit to provide a 1/2 bias voltage level (see bit B in the LCD control register).

LCD Outputs

The LCD driver section includes 20 outputs. They are level-shifted to $V_{DDLC} - V_{SS}$ levels.
In LCD drive modes, the number of segment varies backplane drivers depends on the type of multiplexer. Segment and backplane drivers should be connected directly to the LCD. Unused segment drivers should be left open.
In disable mode, all LCD outputs are forced to V_{SS}, independently of the display data register file. This can be used as an efficient means to blank a connected LCD. If the disable mode is alternated with an LCD drive mode, the whole display can be blinked. The disable mode may be held indefinitely since no DC appears across the LCD.
In low-drive logic mode, up to 19 binary output lines are available. If a level-shift is not required, V_{DDLC} can be tied to V_{DD}.

LCD Voltage Selector

The LCD voltage selector co-ordinates the multiplexing of the LCD according to the selected multiplex and bias configurations (see bits B, M0 and M1 in the LCD control register). The preferred multiplex and bias configurations, together with their characteristics as functions of $V_{op} = V_{DDLC} - V_{SS}$ and the resulting discrimination ratio (D), are given in Table 13.

Table 13 Preferred LCD multiplexing and bias configurations: summary of characteristics.

LCD MULTIPLEX CONFIGURATION	LCD BIAS CONFIGURATION	$V_{off(rms)}/V_{op}$	$V_{on(rms)}/V_{op}$	$D = V_{on(rms)}/V_{off(rms)}$
static (1 BP)	static (2 levels)	0	1	∞
1:2 MUX (2 BP)	1/2 (3 levels)	$\sqrt{2}/4 = 0.354$	$\sqrt{10}/4 = 0.791$	$\sqrt{5} = 2.236$
1:2 MUX (2 BP)	1/3 (4 levels)	$1/3 = 0.333$	$\sqrt{5}/3 = 0.745$	$\sqrt{5} = 2.236$
1:3 MUX (3 BP)	1/3 (4 levels)	$1/3 = 0.333$	$\sqrt{33}/9 = 0.638$	$\sqrt{33}/3 = 1.915$
1:4 MUX (4 BP)	1/3 (4 levels)	$1/3 = 0.333$	$\sqrt{3}/3 = 0.577$	$\sqrt{3} = 1.732$

A practical value for V_{op} is determined by equating $V_{off(rms)}$ with a defined LCD threshold voltage (V_{th}), typically where the LCD exhibits approximately 10% contrast. In the static drive mode a suitable choice is $V_{op} \geq 3\,V_{th}$.
Multiplex configurations of 1:3 and 1:4 with 1/2 bias are possible but the discrimination and hence the contrast ratios are smaller ($\sqrt{3} = 1.732$ for 1:3 multiplex or $\sqrt{21}/3 = 1.528$ for 1:4 multiplex). The advantage of these modes is a reduction of the LCD full scale voltage V_{op} as follows:

1:3 multiplex (1/2 bias) : $V_{op} = 6\,V_{off(rms)} = 2.449\,V_{off(rms)}$

1:4 multiplex (1/2 bias) : $V_{op} = \sqrt{43}/3\,V_{off(rms)} = 2.309\,V_{off(rms)}$

These compare with $V_{op} = 3\,V_{off(rms)}$ when 1/3 bias is used.

LCD Waveforms

The static LCD drive is used when a single backplane is provided in the LCD. Backplane and segment drive waveforms for this mode are shown in Fig.12.

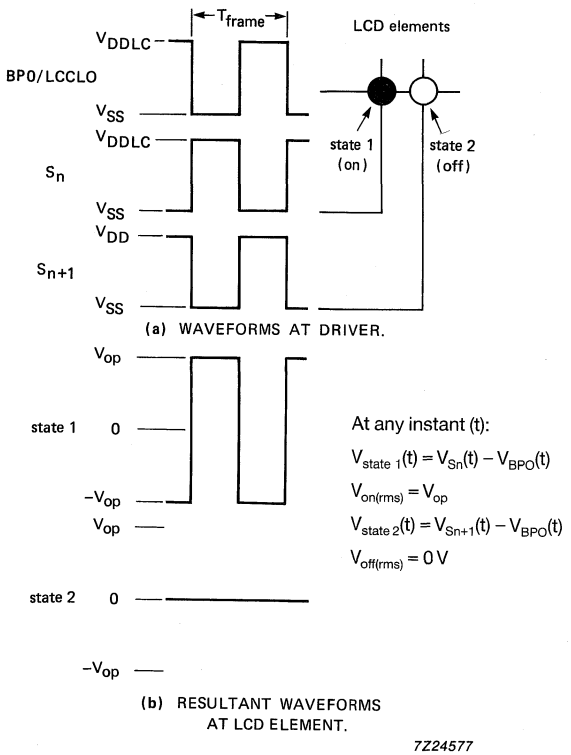


Fig.12 Static drive waveforms ($V_{op} = V_{DDLC} - V_{SS}$).

When two backplanes are provided in the LCD the 1:2 multiplex drive applies. The PCF84C633A allows use of 1/2 and 1/3 bias in this mode as shown in Figs.13 and 14.

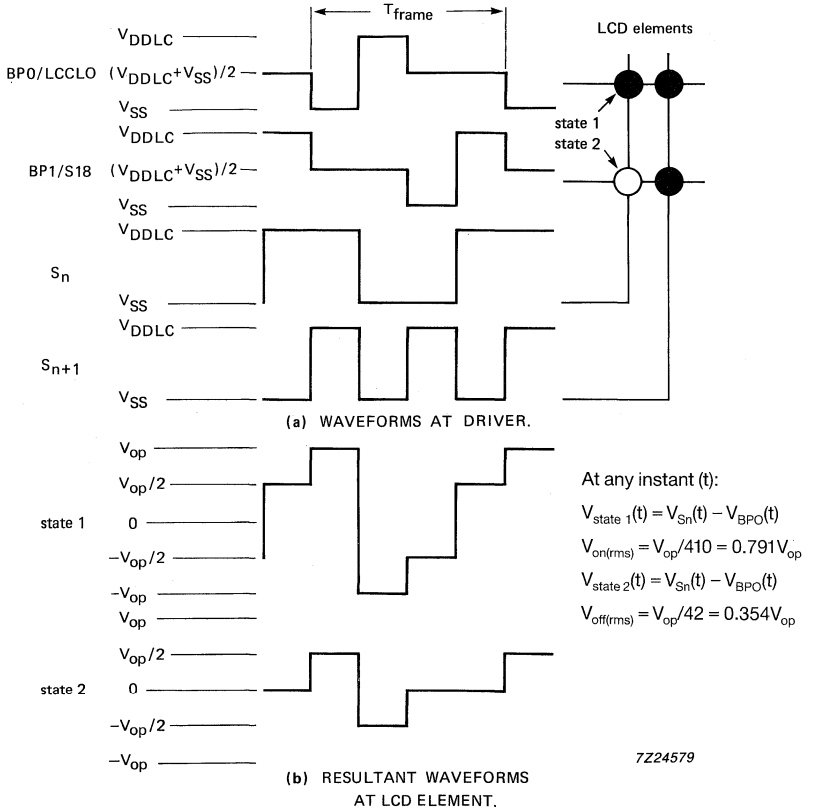


Fig.13 Waveforms for 1:2 multiplex drive with 1/2 bias ($V_{op} = V_{DDLC} - V_{SS}$).

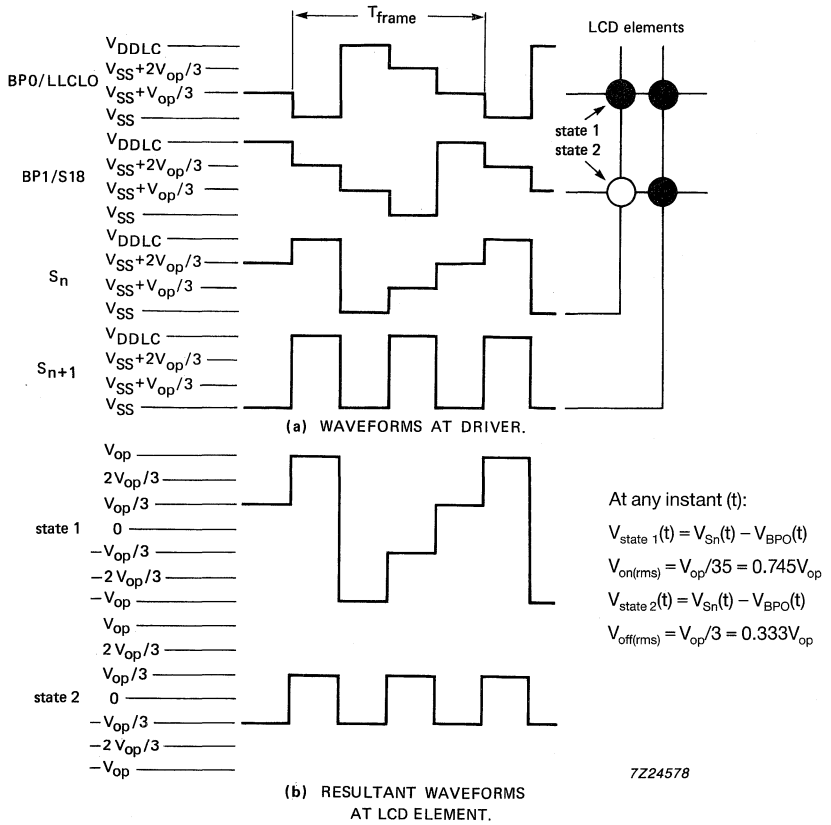


Fig.14 Waveforms for 1:2 multiplex drive with 1/3 bias.

The backplane and segment drive waveform for the 1:3 multiplex drive (three LCD backplanes) and for the 1:4 multiplex drive (four LCD backplanes) are shown in Figs.15 and 16 respectively.

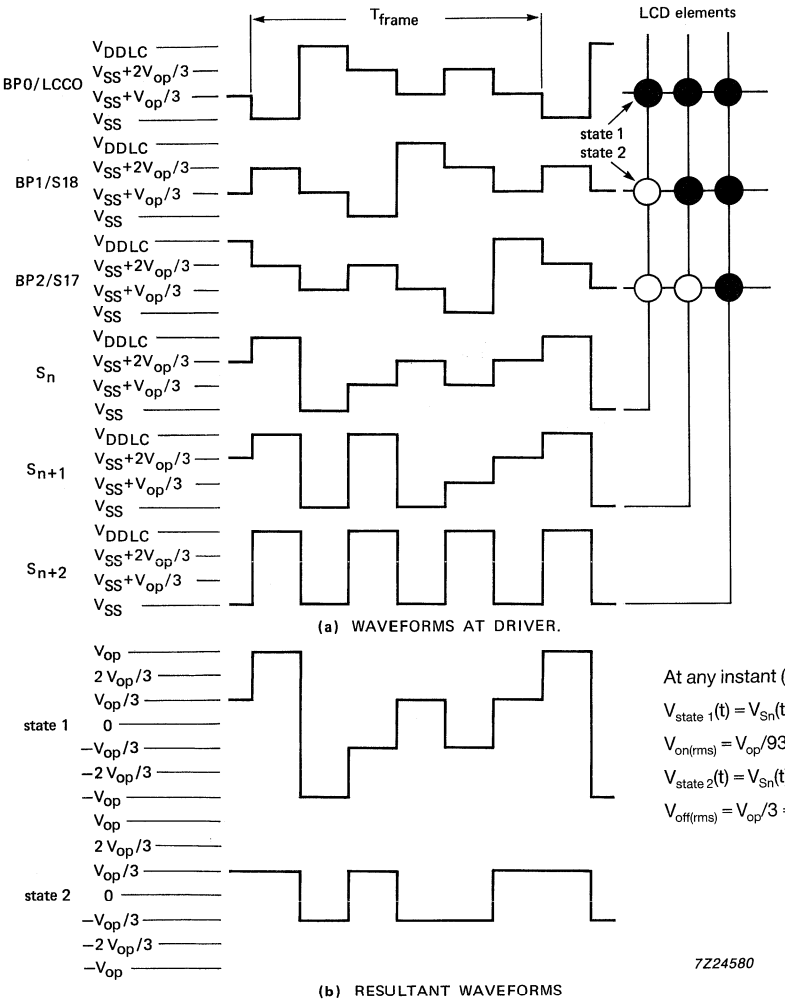


Fig.15 Waveforms for 1:3 multiplex drive ($V_{op} = V_{DDLC} - V_{SS}$).

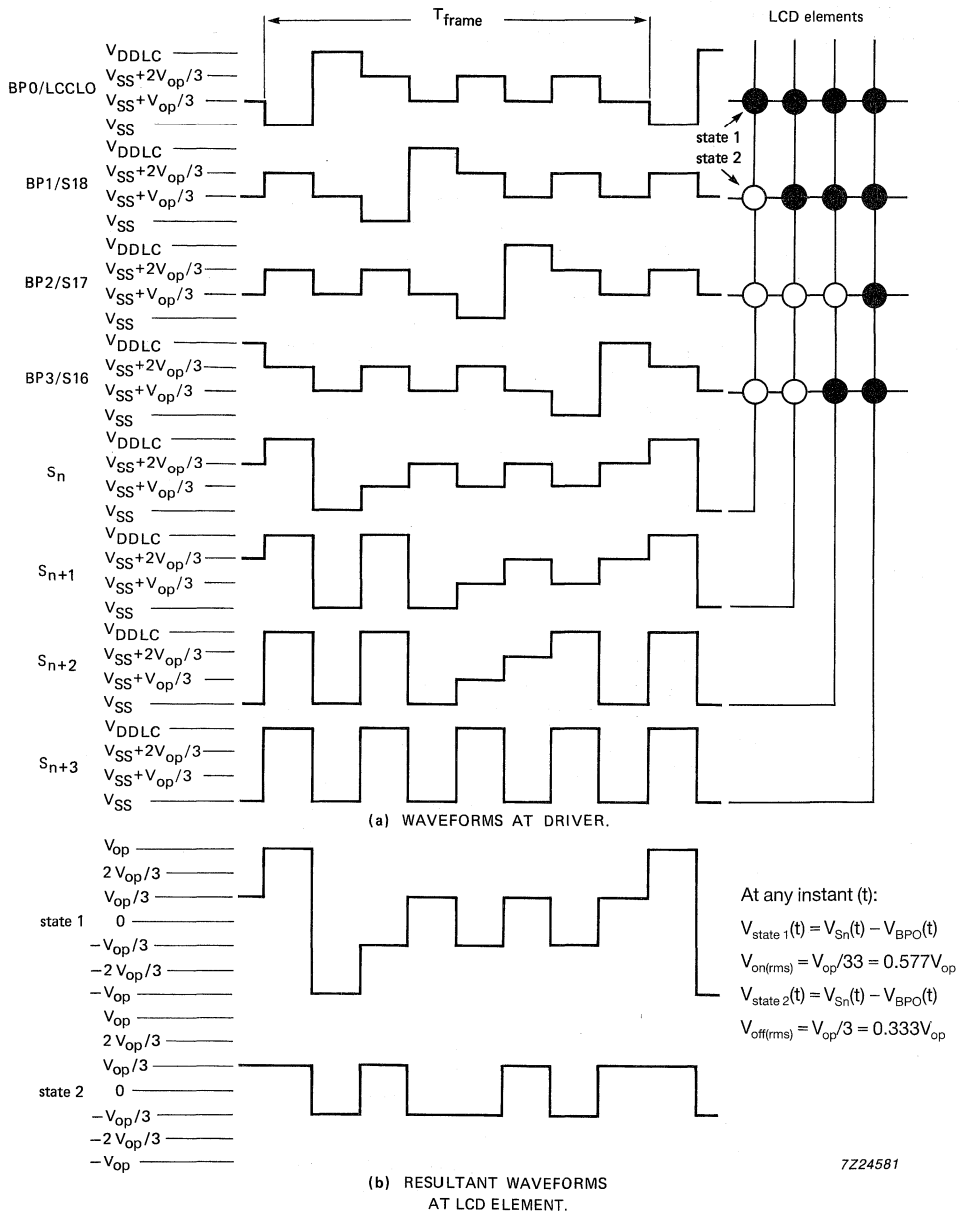
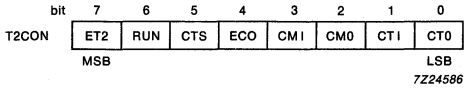
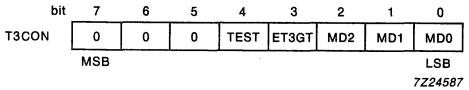


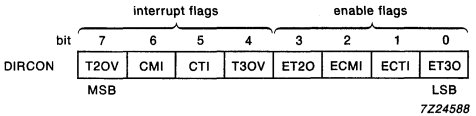
Fig.16 Waveforms for 1:4 multiplex drive ($V_{op} = V_{DDLC} - V_{SS}$).

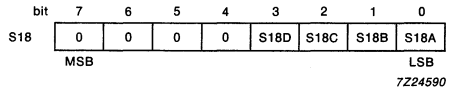
SUMMARY OF DERIVATIVE ADDRESSES AND CONTROL REGISTERS

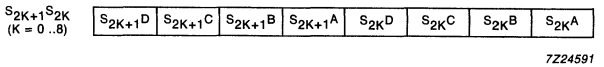
Dx ADDRESS (HEX)	TYPE	MNEMONIC	DESCRIPTION
00 to 01	–	–	not used
02	R	DP1I	DP1 derivative port lines
03	R/W	DP1FF	DP1 derivative port flip-flop
04	R	T2H	T2 timer register high byte
05	R	T2L	T2 timer register low byte
06	R	T2LB	T2 low byte register
07	R/W	T2CMH	T2 compare register high byte
08	R/W	T2CML	T2 compare register low byte
09	R	T2CTH	T2 capture register high byte
0A	R	T2CTL	T2 capture register low byte
0B	R/W	T2CON	T2 control register
0C	R/W	DIRCON	Derivative interrupt control register
0D	R/W	T3H	T3 counter/timer register high byte
0E	R/W	T3L	T3 counter/timer register low byte
0F	R/W	T3CON	T3 control register
10 to 1F	–	–	not used
20	R/W	S1S0	Display data nibbles for S0 and S1
21	R/W	S3S2	Display data nibbles for S2 and S3
22	R/W	S5S4	Display data nibbles for S4 and S5
23	R/W	S7S6	Display data nibbles for S6 and S7
24	R/W	S9S8	Display data nibbles for S8 and S9
25	R/W	S11S10	Display data nibbles for S10 and S11
26	R/W	S13S12	Display data nibbles for S12 and S13
27	R/W	S15S14	Display data nibbles for S14 and S15
28	R/W	S17S16	Display data nibbles for S16 and S17
29	R/W	S18	Display data nibble for S18
2A	R/W	LCDCON	LCD control register
2B to FF	–	–	not used


bit 7 6 5 4 3 2 1 0
T2CON
ET2 RUN CTS ECO CMI CM0 CTI CT0
MSB
LSB
7Z24586


bit 7 6 5 4 3 2 1 0
T3CON
0 0 0 TEST ET3GT MD2 MD1 MD0
MSB
LSB
7Z24587


interrupt flags
enable flags
bit 7 6 5 4 3 2 1 0
DIRCON
T2OV CMI CTI T3OV ET2O ECMI ECTI ET3O
MSB
LSB
7Z24588


bit 7 6 5 4 3 2 1 0
S18
0 0 0 0 S18D S18C S18B S18A
MSB
LSB
7Z24590


S2K+1S2K
(K = 0 ..8)
S2K+1D S2K+1C S2K+1B S2K+1A S2KD S2KC S2KB S2KA
7Z24591

TIMING

The PCF84C633 has been optimized for high speed. Allowed clock frequencies range from 1 MHz to a maximum which is a function of supply voltage. At $V_{DD} \geq 4.5$ V, a 16 MHz maximum clock frequency is guaranteed.

OSCILLATOR

The transconductance (gm) of the inverter stage can be mask-programmed, optimizing the oscillator for a specific frequency and resonator. Three standard transconductance options, referred to as 'LOW', 'MEDIUM' and 'HIGH' gm, can be specified by the user. Table 14 is intended as a guide for typical quartz and PXE resonators.

With C1 = C2 = 8 pF on-chip, external capacitors are not required for quartz oscillators. However, for adequate frequency stability, PXE resonators need external capacitors on the order of the static resonator capacitance C_O, such as external C1 = C2 = 30 to 100 pF.

RESET

The PCF84C633A does not contain a power-on reset circuit. This has the following consequences:

- RESET is an input pin only
- Passive reset always requires an external RC circuit
- When the supply voltage drops, the oscillator is not inhibited

In addition to the conditions given in the PCF84CXX family data sheet, the reset state is characterized as follows:

- all port flip-flops set to 1 (including P2.3)
- all T2 and T3 registers cleared
- derivative interrupt control register cleared
- LCD control register cleared

IDLE MODE

In addition to the oscillator and the 8-bit programmable timer/event counter, the T2, T3 and LCD driver sections remain operative.

STOP MODE

Since the crystal oscillator is switched off, T2, T3 and the digital filters of DP1.1/CTGT and DP1.2/CTCNT receive no clock. Therefore, the complete T2 and T3 sections are frozen. After exit from STOP mode by a LOW level on INT/T0, T2 and T3 proceed from the held state.

LCD waveforms continue during STOP mode because they are independent of the crystal oscillator. If it is permissible to blank the LCD during STOP mode, power dissipation can be further reduced by switching to the disable mode (see bits E0 and E1 in the LCD control register) before executing the STOP instruction.

In low-drive logic mode (Table 10), the levels on S0 to BP1/S18 are held while the LCD oscillator is available on BP0/LCCLO.

Table 14 Recommended transconductance options for popular quartz and PXE resonators.

OPTION	TYPICAL gm (mA/V) AT 5 V	f_{osc} (MHz) FOR QUARTZ CRYSTAL	f_{osc} (MHz) FOR PXE RESONANCE
LOW (gmL)	0.4	1 to 6	–
MEDIUM (gmM)	1.2	4 to 12	1 to 5
HIGH (gmH)	4	10 to 16	3 to 16

INSTRUCTION SET

Since no serial I/O interface is provided, the serial input/output instructions are not available except EN SI and DIS SI, which enable and disable respectively, the derivative interrupt.
ROM space being restricted to 6K bytes, the SEL MB3 instruction would define a non-existing program memory bank and should therefore be avoided.

RATINGS

Limiting values in accordance with the Absolute Maximum System (IEC 134).

PARAMETER	SYMBOL	MIN.	MAX.	UNIT
Supply voltages	V_{DD}, V_{DDLC}	−0.8	+ 7	V
All input voltages	V_I	−0.5	V_{DD} + 0.5	V
DC current into any input or output	I_I, I_O	–	±10	mA
Total power dissipation	P_{tot}	–	+ 125	mW
Storage temperature range	T_{stg}	−65	+ 150	°C
Operating ambient temperature range ($P_{tot} \leqslant 100$ mW)	T_{amb}	−40	+ 70	°C
Operating ambient temperature range ($P_{tot} \leqslant 30$ mW)	T_{amb}	−40	+ 85	°C
Operating junction temperature	T_j	–	+ 90	°C

HANDLING

Inputs and outputs are protected against electrostatic discharge in normal handling. However, it is good practice to take normal precautions appropriate to handling MOS devices (see 'Handling MOS devices' Data Handbook IC14).

DC CHARACTERISTICS

V_{DD} = 2.5 to 5.5 V; V_{DDLC} = 2.5 to 5.5 V; V_{SS} = 0 V; T_{amb} = −40 to + 85 °C; all voltages with respect to V_{SS}; unless otherwise specified. See Figs.17 to 31.

PARAMETER	CONDITIONS	SYMBOL	MIN.	TYP.	MAX.	UNIT
Logic supply voltage operating		V_{DD}	2.5	–	5.5	V
LCD supply voltage	$V_{DDLC} \geqslant 3V$ for 1/3 bias	V_{DDLC}	2.5	–	5.5	V
Logic supply current operating	(note 1)					
	V_{DD} = 5 V, f_{XTAL} = 16 MHz	I_{DD}	–	2.6	5.2	mA
	V_{DD} = 5 V, f_{XTAL} = 10 MHz	I_{DD}	–	1.6	3.2	mA
	V_{DD} = 3 V; f_{XTAL} = 3.58 MHz	I_{DD}	–	0.3	0.6	mA
IDLE mode	(note 1)					
	V_{DD} = 5 V; f_{XTAL} = 16 MHz	I_{DD}	–	1.3	2.6	mA
	V_{DD} = 5 V; f_{XTAL} = 10 MHz	I_{DD}	–	0.8	1.6	mA
	V_{DD} = 3 V; f_{XTAL} = 3.58 MHz	I_{DD}	–	0.15	0.4	mA
STOP mode	(notes 1 and 2)					
	LCD oscillator running, V_{DD} = 2.5 V	I_{DD}	–	10	30	µA
STOP mode	(notes 1, 2 and 3)					
	LCD oscillator stopped, V_{DD} = 2.5 V	I_{DD}	–	5	10	µA
LCD supply current	(note 4)	I_{DDLC}	–	10	20	µA
Inputs						
Input voltage LOW		V_{IL}	0	–	$0.3V_{DD}$	V
Input voltage HIGH		V_{IH}	$0.7V_{DD}$	–	V_{DD}	V
Input leakage current	$V_{SS} \leqslant V_I \leqslant V_{DD}$	$\pm I_L$	–	–	1	µA
Port Outputs						
Output sink current LOW	V_{DD} = 5 V; V_O = 0.4 V	I_{OL}	1.6	5.5	–	mA
Pull-up output source current HIGH	V_{DD} = 5 V; $V_O = 0.7V_{DD}$	$-I_{OH}$	40	–	–	µA
	V_{DD} = 5 V; $V_O = V_{SS}$	$-I_{OH}$	–	–	400	µA
Push-pull output source current HIGH	V_{DD} = 5 V; $V_O = V_{DD}$ − 0.4 V	$-I_{OH}$	1.6	4	–	mA

PARAMETER	CONDITIONS	SYMBOL	MIN.	TYP.	MAX.	UNIT
LCD Outputs						
Output sink current LOW	V_{DDLC} = 5 V; V_O = 0.4 V	I_{OL}	1	2.8	–	mA
Output source current HIGH	V_{DDLC} = 5 V; V_O = V_{DDLC} – 0.4 V	$-I_{OH}$	1	2	–	mA
Output impedance (BP3/S16 to BP0/LCCLO)	V_{DDLC} = 5 V and I_{BP} = 100 µA Outputs measured one at a time	R_{BP}	–	–	5	kΩ
Output impedance (S0 to S15)	V_{DDLC} = 5 V and I_S = 100 µA Outputs measured one at a time	R_S	–	–	7	kΩ
DC voltage component (BP3/S16 to BP0/LCCLO)	V_{DDLC} = 5 V; LCD modes	$\pm V_{BP}$	–	20	–	mV
DC voltage component (S0 to S15)	V_{DDLC} = 5 V; LCD modes	$\pm V_S$	–	20	–	mV
Oscillator transconductance						
option 'LOW' gm	V_{DD} = 5 V	gmL	0.3	0.4	–	mA/V
option 'MEDIUM' gm	V_{DD} = 5 V	gmM	0.9	1.2	–	mA/V
option 'HIGH' gm	V_{DD} = 5 V	gmH	3	4	–	mA/V
Oscillator feedback resistor		R_f	0.4	0.8	1.6	MΩ

Notes to characteristics

1. V_{IL} = 0, V_{IH} = V_{DD}; open drain outputs connected to V_{SS}; all other outputs open.
2. Crystal connected between XTAL1 and XTAL2; pin T1 at V_{SS}; pin INT/T0 at V_{DD}.
3. E0 = 0 in LCD control register; if E1 = 1, f_{LCCLI} = 1.5 kHz.
4. f_{LCCLI} = 1.5 kHz if measured in LCD drive 1 mode.

AC CHARACTERISTICS

V_{DD} = 2.5 to 5.5 V; V_{SS} = 0 V; T_{amb} = −40 to + 85 °C. All voltages with respect to V_{SS}; unless otherwise specified. See Fig.17.

PARAMETER	CONDITIONS	SYMBOL	MIN.	TYP.	MAX.	UNIT
Rise time all outputs	V_{DD} = 5 V, T_{amb} = 25 °C, C_L = 50 pF	t_R	–	30	–	ns
Fall time all outputs	V_{DD} = 5 V, T_{amb} = 25 °C, C_L = 50 pF	t_F	–	30	–	ns
Clock frequency		f_{XTAL}	1	–	16	MHz
LCD oscillator frequency	V_{DD} = 5 V	f_{LCCLO}	0.9	1.5	2.4	kHz

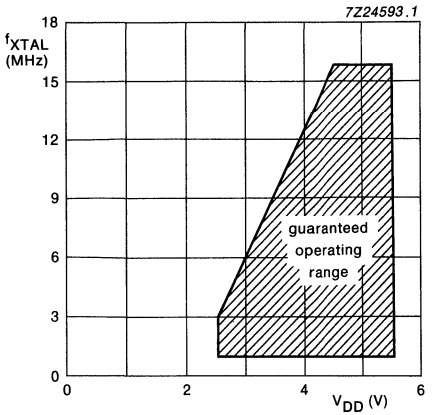


Fig.17 Maximum clock frequency (f_{XTAL}) as a function of logic supply voltage (V_{DD}).

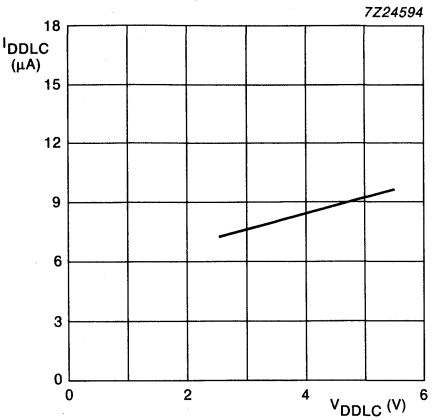
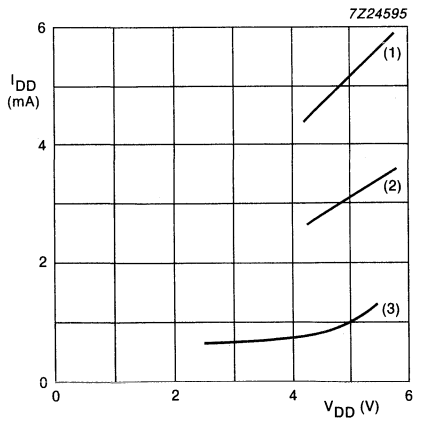


Fig.18 Typical supply current of the LCD section (I_{DDLC}) as a function of LCD supply voltage (V_{DDLC}).



(1) f_{xtal} = 16 MHz
(2) f_{xtal} = 10 MHz
(3) f_{xtal} = 3.58 MHz

Fig.19 Maximum supply current (I_{DD}) in operation as a function of logic supply voltage (V_{DD}).

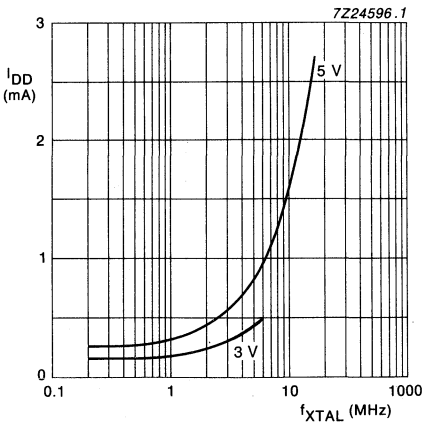
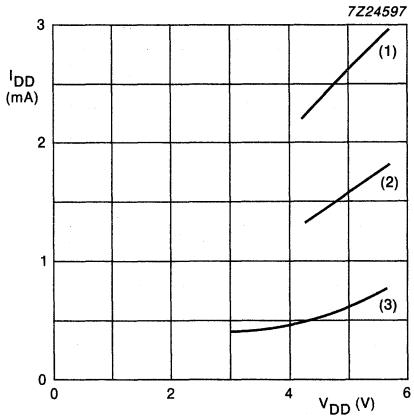


Fig.20 Typical supply current during operation as a function of frequency at V_{DD} = 3 V and V_{DD} = 5 V.



(1) f_{xtal} = 16 MHz
(2) f_{xtal} = 10 MHz
(3) f_{xtal} = 3.58 MHz

Fig.21 Maximum supply current (I_{DD}) in IDLE mode as a function of logic supply voltage (V_{DD}).

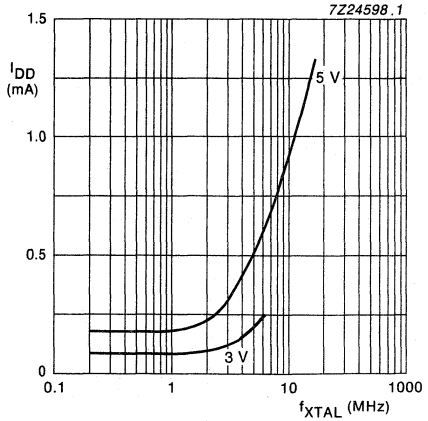


Fig.22 Typical supply current during IDLE mode as a function of frequency at V_{DD} = 3 V and V_{DD} = 5 V.

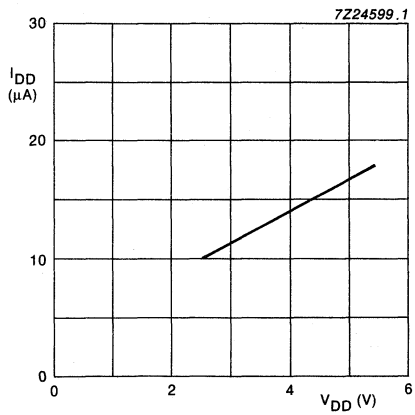


Fig.23 Typical supply current (I_{DD}) in STOP mode with LCD oscillator running as a function of logic supply voltage (V_{DD}).

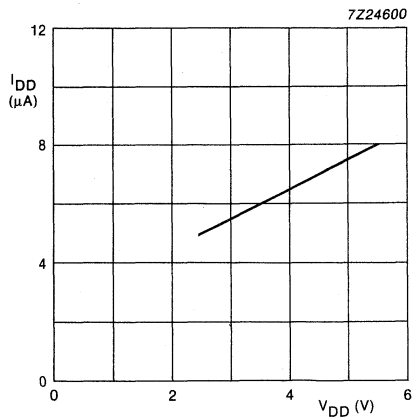


Fig.24 Typical supply current (I_{DD}) in STOP mode with LCD oscillator stopped as a function of logic supply voltage (V_{DD}).

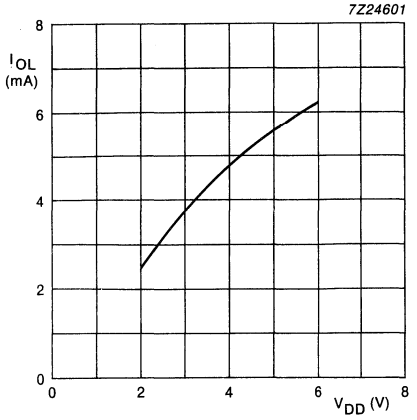
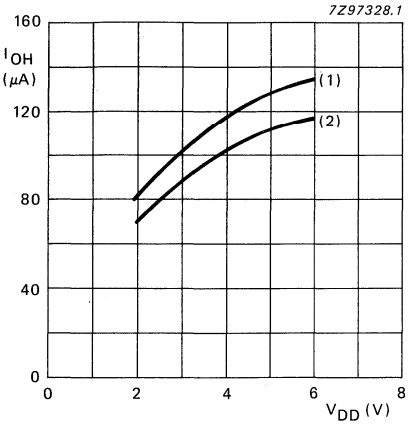


Fig.25 Typical port output sink current (I_{OL}) as a function of logic supply voltage (V_{DD}); $V_O = 0.4$ V.



(1) $V_O = V_{SS}$

(2) $V_O = 0.7\ V_{DD}$

Fig.26 Typical port output source current ($-I_{OH}$) as a function of logic supply voltage (V_{DD}).

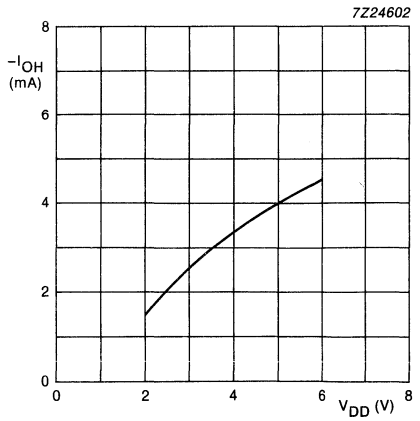


Fig.27 Typical push-pull port output source current ($-I_{OH}$) as a function of logic supply voltage (V_{DD}); $V_O = V_{DD} - 0.4$ V.

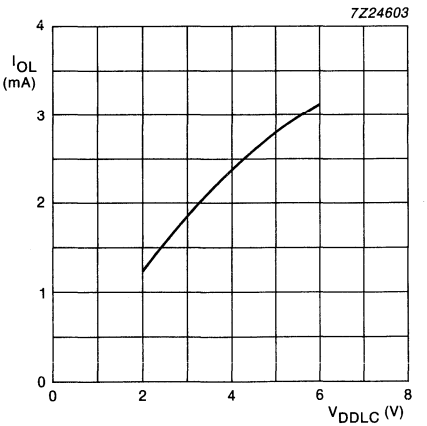


Fig.28 Typical LCD output sink current (I_{OL}) as a function of LCD supply voltage (V_{DDLC}); $V_O = 0.4$ V.

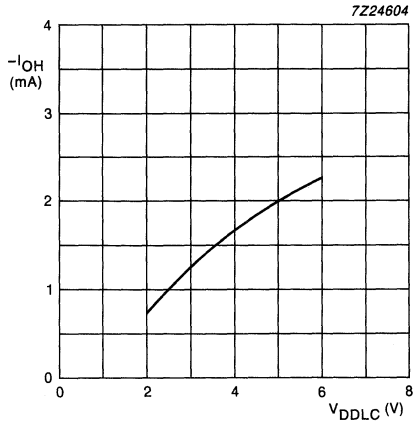


Fig.29 Typical LCD output source current ($-I_{OH}$) as a function of LCD supply voltage (V_{DDLC}); $V_O = V_{DDLC} - 0.4$ V.

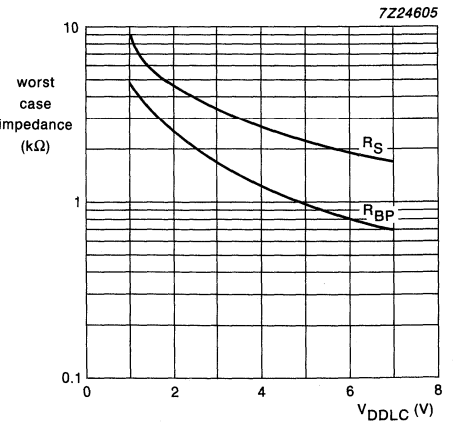


Fig.30 Typical LCD output impedance at 1/3, 1/2 and 2/3 V_{DDLC} as a function of LCD supply voltage (V_{DDLC}).

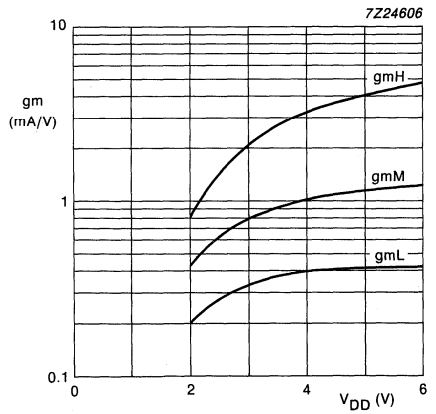


Fig.31 Typical transconductance values as a function of supply voltage (V_{DD}) for the options gmL, gmM and gmH.

SINGLE-CHIP 8-BIT MICROCONTROLLER WITH DERIVATIVE PORTS, TIMER/CAPTURE AND TIMER/COUNTER

DESCRIPTION

The PCF84C853A is a microcontroller with 33 quasi-bidirectional I/O port lines. In addition to the shared features of the PCF84CXX family of microcontrollers, the PCF84C853 includes a 16-bit timer with capture and compare registers, a 16-bit up/down counter/timer and two filtered control inputs. Together with additional derivative port lines, these powerful extensions make the device attractive for demanding realtime applications.

FEATURES

- 8-bit CPU, ROM, RAM, I/O in a single 40-lead package
- 8 K bytes ROM
- 256 bytes RAM
- Over 80 instructions (based on MAB8048) all of 1 or 2 cycles
- 33 quasi-bidirectional I/O port lines
- 8-bit programmable timer/event counter
- 16-bit derivative timer with capture and compare registers (T2)
- 16-bit up/down counter/timer (T3)
- 2 filtered input lines coupled to T2 and T3
- 3 single-level vectored interrupt: external, 8-bit programmable timer/event counter, derivative (triggered by 4 events in T2 and T3)
- 2 test inputs of which one also serves as the external interrupt input
- Stop and Idle modes
- Supply range V_{DD}: 2.5 V to 5.5 V
- Clock frequency: 1 MHz to 16 MHz
- Operating temperature range: −40 °C to + 85 °C
- Manufactured in silicon gate CMOS process

IMPORTANT

This data sheet details the specific properties of the PCF84C853A. The shared characteristics of the **84CXXX family of microcontrollers are described in 84CXXX family specification, which should be read in conjunction with this publication.**

PACKAGE OUTLINES

PCF84C853AP: 40-lead DIL; plastic (SOT129)
PCF84C853AT: 40-lead mini-pack; plastic (VSO40; SOT158A)

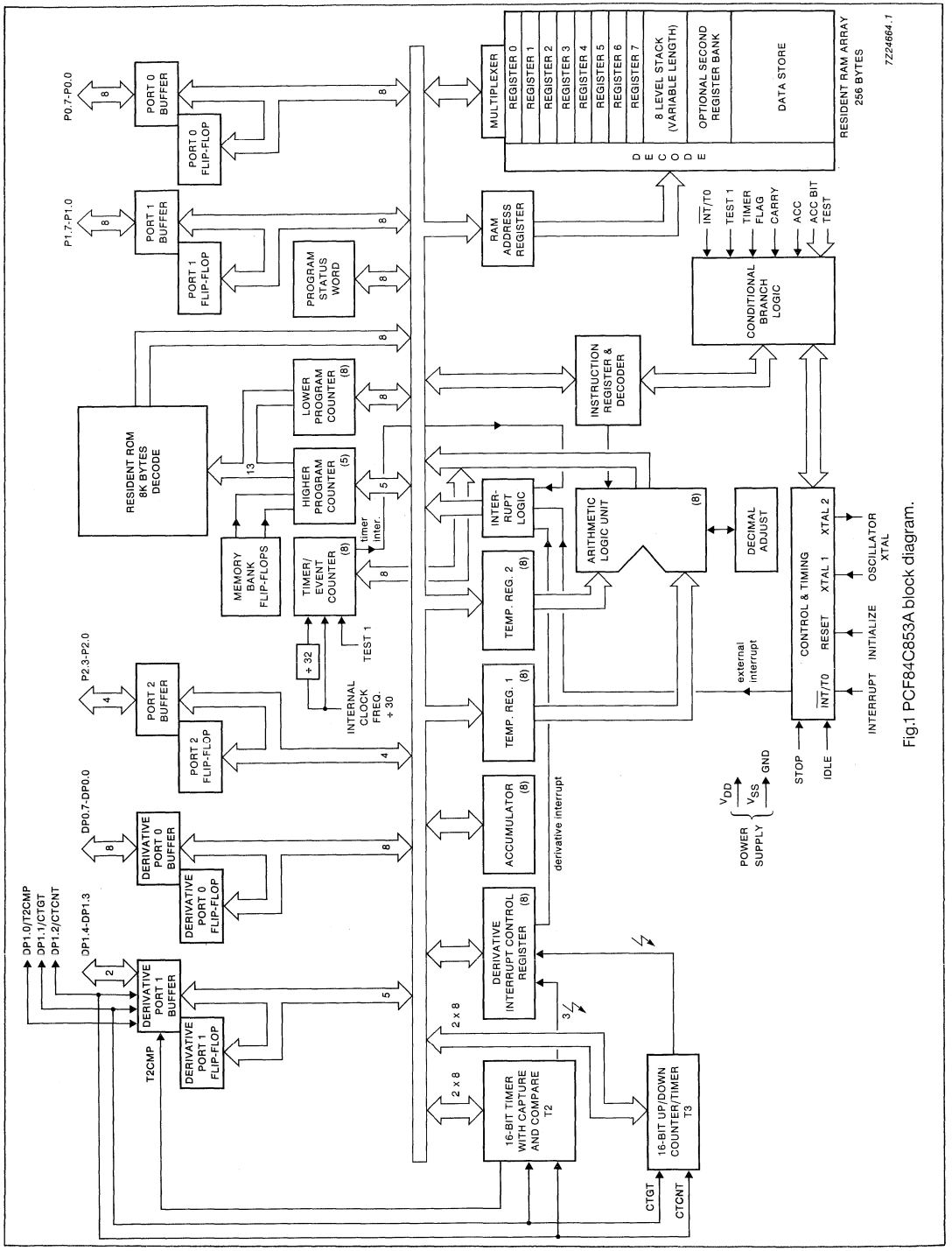


Fig.1 PCF84C853A block diagram.

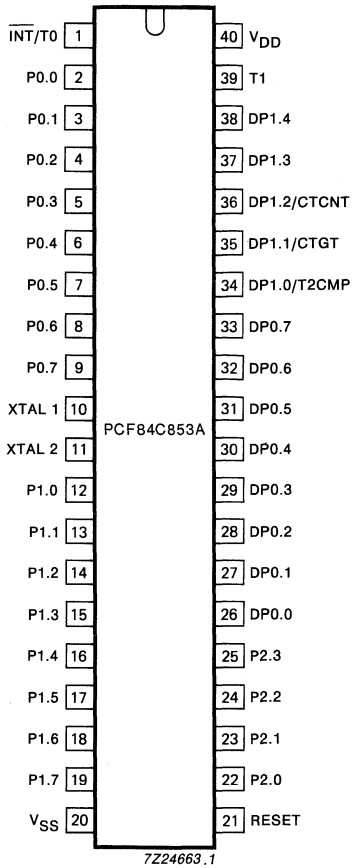


Fig.2 Pinning diagram.

Pinning

PIN	SYMBOL	TYPE	FUNCTION
1	INT/T0	I	Interrupt/Test 0
2-9	P0.0-P0.7	I/O	Port 0 : 8-bit quasi-bidirectional I/O port
10	XTAL1	I	Oscillator/external clock input
11	XTAL2	O	Oscillator output
12-19	P1.0-P1.7	I/O	Port 1: 8-bit quasi-bidirectional I/O port
20	V_{SS}	P	Ground
21	RESET	I	Reset input
22-25	P2.0-P2.3	I/O	Port 2: 4-bit quasi-bidirectional I/O port
26-33	DP0.0-DP0.7	I/O	Derivative Port 0: 8-bit quasi-bidirectional I/O port
34	DP1.0/T2CMP	I/O	Derivative Port 1: quasi-bidirectional I/O line/compare output of T2
35	DP1.1/CTGT	I/O	Derivative Port 1: quasi-bidirectional I/O line/capture input (T2) and/or gate input (T3)
36	DP1.2/CTCNT	I/O	Derivative Port 1: quasi-bidirectional I/O line/capture input (T2) and/or count input (T3)
37-38	DP1.3-DP1.4	I/O	Derivative Port 1: quasi-bidirectional I/O port
39	T1	I	Test 1/count input of 8-bit timer/event counter
40	V_{DD}	P	Supply voltage

PARALLEL PORTS

All standard quasi-bidirectional I/O ports are available:

- Port 0 parallel port of 8 lines (P0.0 to P0.7)
- Port 1 parallel port of 8 lines (P1.0 to P1.7)
- Port 2 parallel port of 4 lines (P2.0 to P2.3)

Since no serial I/O interface is provided, P2.3 is a general purpose line without restrictions.

In addition to the standard ports, two derivative I/O ports are available:

- DP0 derivative port of 8 lines (DP0.0 to DP0.7)
- DP1 derivative port of 5 lines (DP1.0/T2CPM, DP1.1/CTGT, DP1.2/CTCNT, DP1.3, DP1.4)

Three lines of derivative Port 1 are shared with signals used for timer T2 and up/down counter/timer T3; DP1.0/T2CMP, DP1.1/CTGT and DP1.2/CTCNT. Before an alternative function can be used, the output driver FET of the corresponding port-line must be turned off by writing a '1' to the port bit latch. The port pin is then pulled high by the internal pull-up (standard I/O) or is floating (open drain I/O), but may be driven by an external signal. In their non-port functions, DP1.1/CTGT and DP1.2/CTCNT serve as inputs. Therefore, they may only be configured as standard outputs (option 1) or open drain outputs (option 2). All other standard or derivative port lines are eligible for all three mask options.

The alternative functions are further controlled by the corresponding control bits. When an alternative function is used the corresponding I/O port is not disabled, so the state of the pin and latch may still be read using the MOV A,D2 instruction. Note, if a MOV D3,A instruction sets the port latch to '0' it can no longer be driven by an external signal.

Table 1 summarizes the derivative addresses of DP0 and DP1.

DP1.1/CTGT AND DP1.2/CTCNT INPUT FILTERS

The T2 capture transitions and the T3 gating and count signals are digitally filtered to remove spikes shorter than one machine cycle. The signals which are presented on DP1.1/CTGT and DP1.2/CTCNT, are sampled on the positive edge of the internal machine cycle signal ($f_{XTAL}/30$). A change on the input lines must be stable for two consecutive samples, to be accepted. Therefore the filter delay ranges from one to two machine cycles (see Fig.3). The T2 capture and a T3 counting pulses are delayed by two to three machine cycles with respect to the corresponding input signal transition (see Fig.3). The internal signals GT, CT and CNT occur only if the corresponding derivative function is enabled.

The filter delays shown in Fig.3 are only valid if the corresponding function is enabled.

Table 1 Derivative port address pairs.

Dx ADDRESS (HEX)	TYPE	DESCRIPTION
00	R	DP0 derivative port lines
01	R/W	DP0 derivative port flip-flop
02	R	DP1 derivative port lines
03	R/W	DP1 derivative port flip-flop

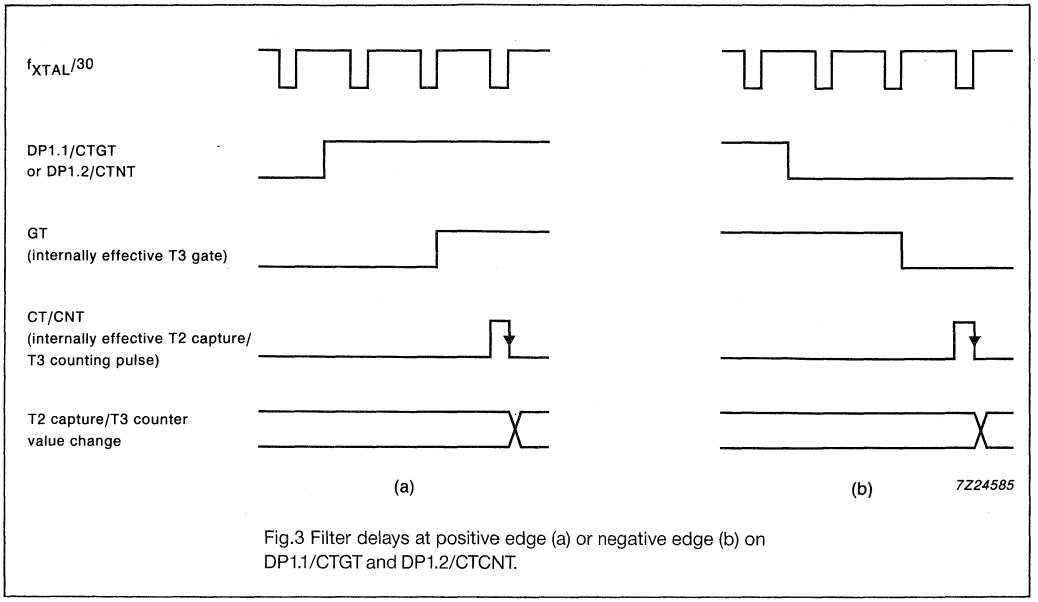


Fig.3 Filter delays at positive edge (a) or negative edge (b) on DP1.1/CTGT and DP1.2/CTCNT.

16-BIT TIMER T2

A derivative 16-bit timer (T2) with capture and compare registers is provided. Communication between the CPU and T2 is handled through derivative registers, making use of the derivative input/output instructions. 16-bit values are accessed by two consecutive byte accesses. Fig.4 gives the block diagram of the T2 section.
Typical applications of T2 are for measuring signal deviations (in conjunction with the capture register) or for generating pulse deviations (in conjunction with the compare register).

T2 derivative registers

Table 2 summarizes the derivative addresses, the register mnemonics and the access types for the T2 section.

Table 2 Derivative addresses for the T2 section.

Dx ADDRESS (HEX)	TYPE	MNEMONIC	DESCRIPTION
04	R	T2H	T2 timer register high byte
05	R	T2L	T2 timer register low byte
06	R	T2LB	T2 low byte register
07	R/W	T2CMH	T2 compare register high byte
08	R/W	T2CML	T2 compare register low byte
09	R	T2CTH	T2 capture register high byte
0A	R	T2CTL	T2 capture register low byte
0B	R/W	T2CON	T2 control register

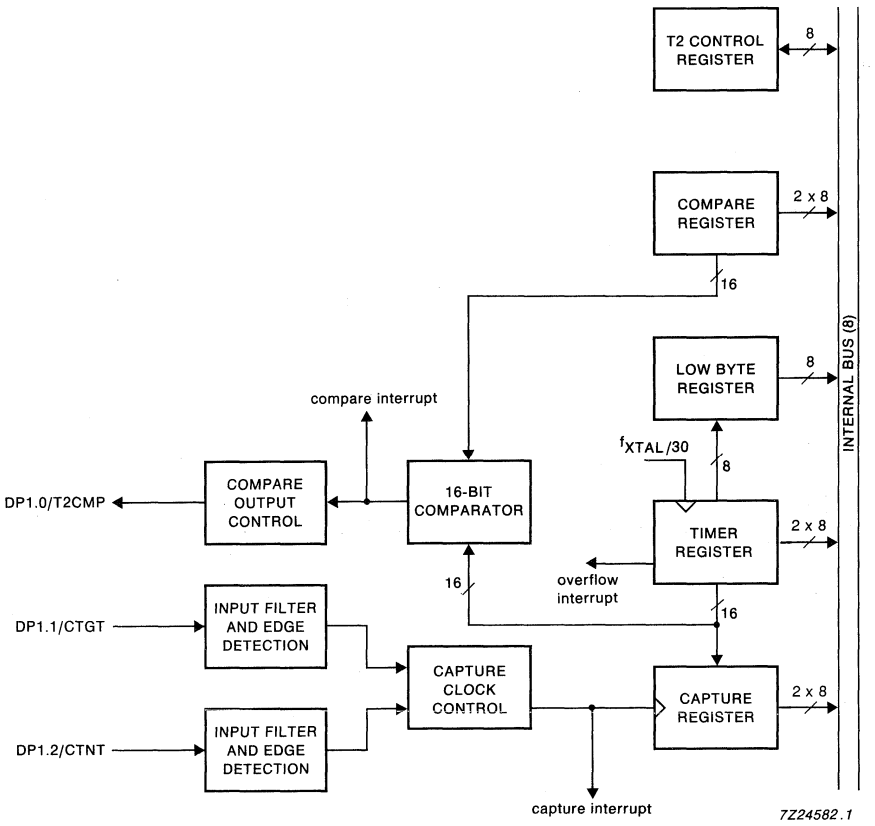


Fig.4 Block diagram of the derivative 16-bit timer T2.

Timer Register and Low Byte Register

The central unit of T2 is the 16-bit timer register. Its behaviour depends on bit ET2 and RUN in the T2 control register (see Table 3). If ET2 = 0, the timer register is disabled and cleared. Otherwise the timer register is enabled and keeps its value (RUN = 0) or increments (RUN = 1) every machine cycle (f_{XTAL}/30).

When the timer register overflows from FFFFH to 0000H, the T2 overflow flag T20V is set in the derivative interrupt control register (see section Derivative Interrupts). This event can be used to generate one of four different derivative interrupt request. The source of the interrupt is found by reading DIRCON (see section Derivative Interrupts).

The 16-bit timer register is sub-divided into two 8-bit registers. T2H contains the high byte of the timer value, T2L contains the low byte. Both are read-only. They can be read anytime without disturbing the timer function.

Whenever T2H is read, T2L is simultaneously copied into the low byte register T2LB. This allows the full 16-bit timer value to be read consistently. The value in T2LB remains fixed until T2H is read again.

Capture Function

The 16-bit capture register latches the value of the 16-bit timer register when a predefined transition on DP1.1/CTGT or DP1.2/CTCNT occurs. As CT0 and CT1 in the T2 control register (see Table 3) are used to either disable the capture function, or to select the positive, the negative or both edges of the input signal to latch the timer value into the capture register.

Bit CTS in the T2 control register selects DP1.1/CTGT (CTS = 1) or DP1.2/CTCNT (CTS = 0) as the source for the capture clock. Signals on DP1.1/CTGT and DP1.2/CTCNT are filtered (see Fig.4). For a transition to be detected, the signal on DP1.1/CTGT and DP1.2/CTCNT must be stable for at least two machine cycles ($60/f_{XTAL}$) before and after the selected edge. If several consecutive capture conditions occur, the capture register always contains the most recent value.

When the capture trigger occurs, the capture interrupt flag CTI is set in the derivative interrupt control register (see section on derivative interrupts). This event can be used to generate a derivative interrupt request. The source of the interrupt is found by reading DIRCON (see section Derivative Interrupts).

The 16-bit capture register is sub-divided into two 8-bit registers. T2CTH contains the high byte of the captured value, T2CTL contains the low byte. Both are read-only. When T2CTH is read, the capture function is inhibited until T2CTL is also read. This allows the full 16-bit capture value to be consistently read.

Compare Function

If (ET2 = 1) and (RUN = 1) in the T2 control register, a match between the timer register and the 16-bit compare register will set the compare interrupt flag CMI in the derivative interrupt control register (see section on derivative interrupts). This event can be used to generate a derivative interrupt requests. The source of the interrupt is found by reading the DIRCON (see section Derivative Interrupts).

The 16-bit compare register as sub-divided into two 8-bit registers. T2CMH contains the high byte of the compared value, T2CML contains the low byte. Both can be read from or written to. If the compare function is not used, T2CMH and T2CML may be used as storage locations. When T2CMH is written, the compare function is inhibited until T2CML is also written. This allows the full 16-bit compare value to be consistently defined.

If bit ECO = 1 in the T2 control register, a compare event can generate an output on DP1.0/T2CMP. Bits CM0 and CM1 in the T2 control register select between no change, output low, output high or output toggle. This may be useful to generate a pulse width or to signal a lapse of time.

T2 Control Register (T2CON)

The 8-bit control register defines the behaviour of the T2 section. It can be read or written. Fig.5 gives the structure of the T2 control register and Table 3 summarizes the significance of the individual control bits.

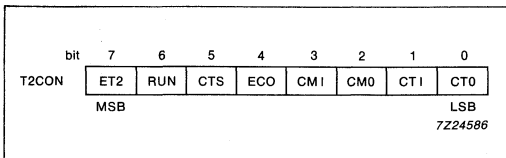


Fig.5 T2 control register (T2CON).

Table 3 Overview of T2 control register bits.

BIT	NAME	DESCRIPTION
ET2	Enable T2	ET2 = 0: timer register disabled and cleared ET2 = 1: timer register enabled (see RUN)
RUN	Run	RUN = 0: timer register value held RUN = 1: timer register increments if ET2 = 1
CTS	Capture Source	CTS = 0: DP1.2/CTCNT capture signal source CTS = 1: DP1.1/CTGT capture signal source
ECO	Enable Compare Out	ECO = 0: DP1.0/T2CMP derivative port line ECO = 1: DP1.0/T2CMP compare output
CM1 CM0	Compare 1 Compare 0	**CM1** **CM0** 0 0 no change on DP1.0/T2CMP after compare match 0 1 low level on DP1.0/T2CMP after compare match if ECO = 1 1 0 high level on DP1.0/T2CMP after compare match if ECO = 1 1 1 toggles DP1.0/T2CMP after compare match if ECO = 1
CT1 CT0	Capture 1 Capture 0	**CT1** **CT0** 0 0 capture function disabled 0 1 capture on positive edge of capture signal source 1 0 capture on negative edge of capture signal source 1 1 capture on any edge of capture signal source

16-BIT UP/DOWN COUNTER/TIMER T3

A derivative 16-bit up/down counter/timer (T3) is provided. Communication between the CPU and T3 is handled through derivative registers, making use of the derivative input/output instructions. The 16-bit counter/timer value is accessed by two consecutive byte accesses. Fig.6 gives the block diagram of the T3 section.

Typical applications of T3 are for measuring signal deviations (in timer mode in conjunction with the gating signal on DP1.1/CTGT), for interval generation (in timer mode) and for counting signal edges (in counter mode).

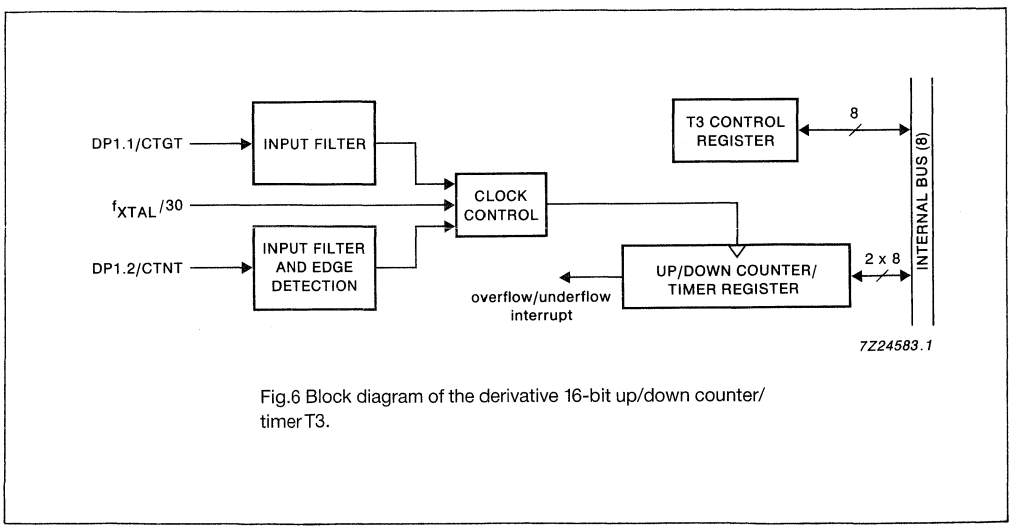


Fig.6 Block diagram of the derivative 16-bit up/down counter/timer T3.

T3 Derivative Registers

Table 4 summarizes the derivative addresses, the register mnemonics and the access types for the T3 section.

Table 4 Derivative addresses for the T3 section.

Dx ADDRESS (HEX)	TYPE	MNEMONIC	DESCRIPTION
0D	R/W	T3H	T3 counter/timer register high byte
0E	R/W	T3L	T3 counter/timer register low byte
0F	R/W	T3CON	T3 control register

Up/down Counter/Timer Register

The central unit of T3 is the 16-bit up/down counter/timer register. Its behaviour is defined by the mode bits MD0, MD1 and MD2 in the T3 control register (see Tables 5 and 6 and also Fig.7).

In timer mode (MD2 = 0), the register increments (MD1 = 1) or decrements (MD0 = 1) every machine cycle ($f_{XTAL}/30$).

In counter mode (MD2 = 1), the register increments (MD1 = 1) with the positive edge on DP1.2/CTCNT. The register decrements (MD0 = 1) with the negative edge on DP1.2/CTCNT. For a transition to be detected, the clock on DP1.2/CTCNT must be stable for at least two machine cycles ($60/f_{XTAL}$) before and after the signal change.

In both timer and counter modes, counting can be inhibited if the gating signal on DP1.1/CTGT equals zero. This feature is turned on by bit ET3GT (enable T3 gate) in the T3 control register. Being digitally filtered, the state of DP1.1/CTGT must be stable for at least two machine cycles ($60/f_{XTAL}$) before it is recognized as a valid gating signal. It is possible to simultaneously enable counting up and counting down in counter mode (MD1 = MD0 = 1). When this is done, the gating signal should be enabled by setting bit ET3GT (in DIRCON) to '1'. Then signal edges will only be counted while the gating signal (DP1.1/CTGT) is '1'.

In both timer and counter modes, an overflow (from FFFFH to 0000H) or an underflow (from 0000H to FFFFH) assert the overflow flag T30V in the derivative interrupt control register (see section on Derivative Interrupts). This event can be used to generate a derivative interrupt request. The source of the interrupt is found by reading the DIRCON register (see section Derivative Interrupts).

The 16-bit up/down counter/timer register is sub-divided into two 8-bit registers. T3H contains the high byte of the counter/timer value, T3L contains the low byte. Both can be read or written. It is the responsibility of the user software to ensure that accesses to T3H and T3L do not violate data consistency. One way to guarantee this is by disabling the T3 section for the duration of the access. If the up/down counter/timer is not used, T3H and T3L may serve as storage locations.

Table 5 Truth table T3 Up/Down Timer.

T3 CLOCK INPUT		T3 CONTROL REGISTER				T3 GATE	TIMER MODE
$f_{XTAL}/30$	DP1.2/ CTCNT	MD2	MD1	MD0	ET3GT	DP1.1/ CTGT	
⎍⎍	X	0	1	0	0	X	Timer Up
⎍⎍	X	0	1	0	1	1	Timer Up
⎍⎍	X	0	1	0	1	0	Inhibit Timer Up by DP1.1/CTGT
⎍⎍	X	0	0	1	0	X	Timer Down
⎍⎍	X	0	0	1	1	1	Timer Down
⎍⎍	X	0	0	1	1	0	Inhibit Timer Down by DP1.1/CTGT
X	X	X	0	0	X	X	Inhibit Timer

'X' denotes don't care states

Table 6 Truth table Up/Down Counter.

T3 CLOCK INPUT		T3 CONTROL REGISTER				T3 GATE	COUNTER MODE
$f_{XTAL}/30$	DP1.2/ CTCNT	MD2	MD1	MD0	ET3GT	DP1.1/ CTGT	
X	↓	1	1	0	0	X	Count Up
X	↓	1	1	0	1	1	Count Up
X	↓	1	1	0	1	0	Inhibit Count Up by DP1.1/CTGT
X	↑	1	0	1	0	X	Count Down
X	↑	1	0	1	1	1	Count Down
X	↑	1	0	1	1	0	Inhibit Count Down by DP1.1/CTGT
X	↓	1	1	1	1	1	Count Up
X	↑	1	1	1	1	1	Count Down
X	X	1	1	1	1	0	Inhibit Count Down by DP1.1/CTGT
X	X	X	0	0	X	X	Inhibit Counter

'X' denotes don't care states

T3 Control Register (T3CON)

The 8-bit T3 control register defines the behaviour of the T3 section. It can be read or written. Fig.7 gives the structure of the T3 control register and Table 7 summarizes the significance of the individual control bits.

Bits 5, 6 and 7 are fixed at zero.

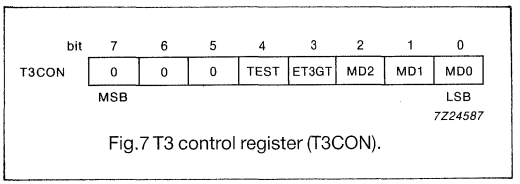


Fig.7 T3 control register (T3CON).

Table 7 Overview of T3 control register bits.

BIT	NAME	DESCRIPTION			
TEST	Test	TEST = 0: test mode disabled (state for user software) TEST = 1: test mode enabled (not allowed in user software)			
ET3GT	Enable T3 Gate	ET3GT = 0: state of DP1.1/CTGT irrelevant to T3 ET3GT = 1: counter/timer T3 inhibited/enabled if DP1.1/CTGT = 0/1			
MD2 MD1 MD0	Mode 2 Mode 1 Mode 0	**MD2**	**MD1**	**MD0**	
		X	0	0	counter/timer T3 inhibited
		0	0	1	decrementing timer mode
		0	1	0	incrementing timer mode
		0	1	1	not allowed
		1	0	1	decrementing counter mode
		1	1	0	incrementing counter mode
		1	1	1	decrementing and incrementing (use with ET3GT=1)

DERIVATIVE INTERRUPTS

Since the PCF84C853A includes no serial I/O interface, the SIO/derivative interrupt (see PCF84CXX family data sheet) reduces to a derivative interrupt. Four derivative interrupt events are defined. These events are controlled by the derivative interrupt control register (DIRCON). Table 8 summarizes the significance of the individual control bits.

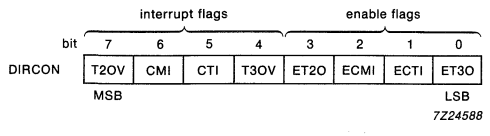


Fig.8 Derivative interrupt control register (DIRCON).

A derivative interrupt request is honoured if:-

- no interrupt routine proceeds
- no external interrupt request is pending
- the SIO/derivative interrupt is enabled
- the corresponding enable bit in the derivative interrupt control register is set

The derivative interrupt routine must include instructions that will remove the cause of the derivative interrupt by implicitly clearing the corresponding interrupt flag. Table 9 gives derivative address, the register mnemonics and the access type for the derivative interrupt control register.

DIRCON can be read or written. Read access is necessary to determine the cause of a derivative interrupt request. If the derivative interrupt is not used, the flags may be directly tested by the program. Write access is necessary to remove the cause of the derivative interrupt request. The flags may also be directly written to generate a software interrupt.

Table 8 Overview of derivative interrupt control register bits.

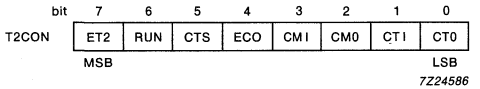
BIT	NAME	DESCRIPTION
T20V	T2 Overflow	Set: if T2 timer register overflows from FFFFH to 0000H (or by program) Reset: by program and by RESET
CMI	Compare Interrupt	Set: if T2 timer register equals T2 compare register while (ET2 = 1) and (RUN = 1) in the T2 control register (or by program) Reset: by program and by RESET
CTI	Capture Interrupt	Set: if capture trigger occurs (or by program) Reset: by program and by RESET
T30V	T3 Overflow	Set: if T3 up/down counter/timer register overflows from FFFFH to 0000H or underflows from 0000H to FFFFH (or by program) Reset: by program and by RESET
ET20	Enable T2 Overflow	ET20 = 0: T20V event cannot request interrupt ET20 = 1: T20V event requests interrupt
ECMI	Enable Compare Interrupt	ECMI = 0: CMI event cannot request interrupt ECMI = 1: CMI event requests interrupt
ECTI	Enable Capture Interrupt	ECTI = 0: CTI event cannot request interrupt ECTI = 1: CTI event requests interrupt
ET30	Enable T3 Overflow	ET30 = 0: T30V event cannot request interrupt ET30 = 1: T30V event requests interrupt

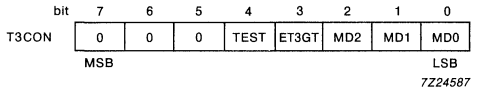
Table 9 Derivative interrupt control register address

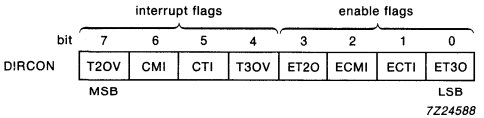
Dx ADDRESS (HEX)	TYPE	MNEMONIC	DESCRIPTION
0C	R/W	DIRCON	Derivative interrupt control register

SUMMARY OF DERIVATIVE ADDRESSES AND CONTROL REGISTERS

Dx ADDRESS (HEX)	TYPE	MNEMONIC	DESCRIPTION
00	R	DP0I	DP0 derivative port lines
01	R/W	DP0FF	DP0 derivative port flip-flop
02	R	DP1I	DP1 derivative port lines
03	R/W	DP1FF	DP1 derivative port flip-flop
04	R	T2H	T2 timer register high byte
05	R	T2L	T2 timer register low byte
06	R	T2LB	T2 low byte register
07	R/W	T2CMH	T2 compare register high byte
08	R/W	T2CML	T2 compare register low byte
09	R	T2CTH	T2 capture register high byte
0A	R	T2CTL	T2 capture register low byte
0B	R/W	T2CON	T2 control register
0C	R/W	DIRCON	Derivative interrupt control register
0D	R/W	T3H	T3 counter/timer register high byte
0E	R/W	T3L	T3 counter/timer register low byte
0F	R/W	T3CON	T3 control register
10 to FF	–	–	not used







TIMING

The PCF84C853A has been optimized for high speed. Allowed clock frequencies range from 1 MHz to a maximum which is a function of supply voltage. At VDD $\geqslant$ 4.5 V, a 16 MHz maximum clock frequency is guaranteed.

OSCILLATOR

The transconductance (gm) of the inverter stage can be mask-programmed, optimizing the oscillator for a specific frequency and resonator. Three standard transconductance options, referred to as 'LOW', 'MEDIUM' and 'HIGH' gm, can be specified by the user. Table 10 is intended as a rough selection guide for typical quartz and PXE resonators. For more details, please consult the safe operating range formulae in the PCF84CXX family data sheet.
With C1 = C2 = 8pF on-chip, external capacitors are not required for quartz oscillators. However, for adequate frequency stability, PXE resonators need external capacitors on the order of the static resonator capacitance C_O, such as external C1 = C2 = 30 to 100 pF.

RESET

The PCF84C853A does not contain a power-on reset circuit. This has the following consequences:

- RESET is an input pin only
- Passive reset always requires an external RC circuit
- When the supply voltage drops, the oscillator is not inhibited

In addition to the conditions given in the PCF84CXX family data sheet, the reset state is characterized as follows:

- all port flip-flops set to 1 (including P2.3)
- all T2 and T3 registers cleared
- derivative interrupt control register cleared

IDLE MODE

In addition to the oscillator and the 8-bit programmable timer/event counter, the T2 and T3 sections remain operative.

STOP MODE

Since the crystal oscillator is switched off, T2, T3 and the digital filters of DP1.1/CTGT and DP1.2/CTCNT receive no clock. Therefore, the complete T2 and T3 sections are frozen. After exit from STOP mode by a LOW level on INT/T0, operation proceeds from the hold state.

Table 10 Recommended transconductance options for popular quartz and PXE resonators.

OPTION	TYPICAL gm (mA/V) AT 5 V	f_{osc} (MHz) FOR QUARTZ CRYSTAL	f_{osc} (MHz) FOR PXE RESONANCE
LOW (gmL)	0.4	1 to 6	–
MEDIUM (gmM)	1.2	4 to 12	1 to 5
HIGH (gmH)	4	10 to 16	3 to 16

INSTRUCTION SET

Since no serial I/O interface is provided, the serial input/output instructions are not available except ENSI and DIS SI, which enable and disable respectively, the derivative interrupt.

RATINGS

Limiting values in accordance with the Absolute Maximum System (IEC 134)

PARAMETER	SYMBOL	MIN.	MAX.	UNIT
Supply voltage	V_{DD}	−0.8	+7	V
All input voltages	V_I	−0.5	V_{DD} +0.5	V
DC current into any input or output	I_I, I_O	–	±10	mA
Total power dissipation	P_{tot}	–	+125	mW
Storage temperature range	T_{stg}	−65	+150	°C
Operating ambient temperature range ($P_{tot} \leqslant 100$ mW)	T_{amb}	−40	+70	°C
Operating ambient temperature range ($P_{tot} \leqslant 30$ mW)	T_{amb}	−40	+85	°C
Operating junction temperature	T_j	–	+90	°C

HANDLING

Inputs and outputs are protected against electrostatic discharge in normal handling. However, it is good practice to take normal precautions appropriate to handling MOS devices (see 'Handling MOS devices', Data Handbook IC 14).

DC CHARACTERISTICS

V_{DD} = 2.5 to 5.5 V; V_{SS} = 0 V; T_{amb} = −40 to 85 °C; all voltages with respect to V_{SS}; unless otherwise specified. See Figs.9 to 18.

PARAMETER	CONDITIONS	SYMBOL	MIN.	TYP.	MAX.	UNIT
Supply voltage, operating		V_{DD}	2.5	–	5.5	V
Supply current, operating	note 1					
	V_{DD} = 5 V; f_{XTAL} = 16 MHz	I_{DD}	–	2.6	5.2	mA
	V_{DD} = 5 V; f_{XTAL} = 10 MHz	I_{DD}	–	1.6	3.2	mA
	V_{DD} = 3 V; f_{XTAL} = 3.58 MHz	I_{DD}	–	0.3	0.6	mA
IDLE mode	note 1					
	V_{DD} = 5 V; f_{XTAL} = 16 MHz	I_{DD}	–	1.3	2.6	mA
	V_{DD} = 5 V; f_{XTAL} = 10 MHz	I_{DD}	–	0.8	1.6	mA
	V_{DD} = 5 V; f_{XTAL} = 3.58 MHz	I_{DD}	–	0.15	0.4	mA
STOP	notes 1 and 2					
	V_{DD} = 2.5 V; T_{amb} = 25 °C	I_{DD}	–	1.2	2.5	µA
	V_{DD} = 2.5 V; T_{amb} = 85 °C	I_{DD}	–	–	1O	µA
Inputs						
Input voltage LOW		V_{IL}	0	–	$0.3V_{DD}$	V
Input voltage HIGH		V_{IH}	$0.7V_{DD}$	–	V_{DD}	V
Input leakage current	$V_{SS} \leqslant V_I \leqslant V_{DD}$	$\pm I_{IL}$	–	–	1	µA
Outputs						
Output sink current LOW	V_{DD} = 5 V; V_O = 0.4 V	IO_L	1.6	5.5	–	mA
Pull-up output source current HIGH	V_{DD} = 5 V; V_O = 0.7 V_{DD}	$-I_{OH}$	40	–	–	µA
	V_{DD} = 5 V; V_O = V_{SS}	$-I_{OH}$	–	–	400	µA
Push-pull output source current HIGH	V_{DD} = 5 V; V_O = V_{DD}−0.4 V	$-I_{OH}$	1.6	4	–	mA
Oscillator transconductance						
option LOW gm	V_{DD} = 5 V	gmL	0.3	0.4	–	mA/V
option MEDIUM gm	V_{DD} = 5 V	gmM	0.9	1.2	–	mA/V
option HIGH gm	V_{DD} = 5 V	gmH	3	4	–	mA/V
Oscillator feedback resistor		Rf	0.4	0.8	1.6	MΩ

Notes to characteristics

1. V_{IL} = 0, V_{IH} = V_{DD}; open drain outputs connected to V_{SS}; all other outputs open.
2. Crystal connected between XTAL1 and XTAL2; pin T1 and V_{SS}; pin INT/T0 at V_{DD}.

AC CHARACTERISTICS

V_{DD} = 2.5 to 5.5 V; V_{SS} = 0 V; T_{amb} = −40 to 85 °C. All voltages with respect to V_{SS}; unless otherwise specified. See Fig.9.

PARAMETER	CONDITIONS	SYMBOL	MIN.	TYP.	MAX.	UNIT
Rise time all outputs	V_{DD} = 5 V, T_{amb} = 25 °C, C_L = 50 pF	t_R	–	30	–	ns
Fall time all outputs	V_{DD} = 5 V, T_{amb} = 25 °C, C_L = 50 pF	t_F	–	30	–	ns
Clock frequency		f_{XTAL}	1	–	16	MHz

CHARACTERISTIC CURVES

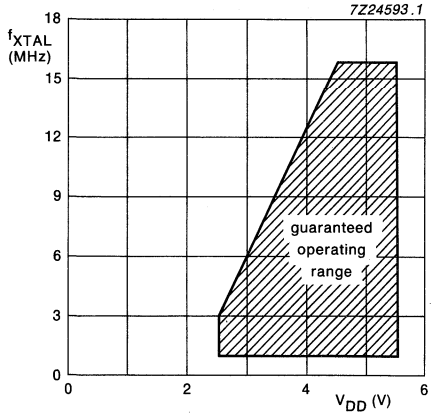
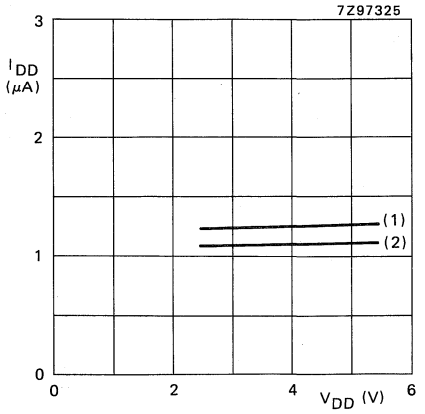
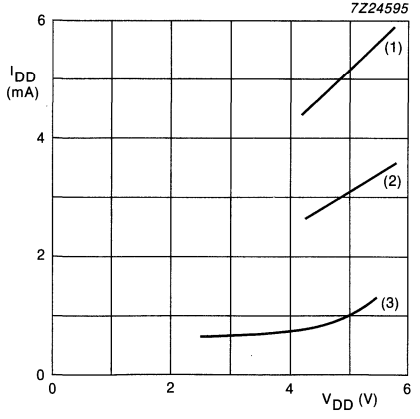


Fig.9 Maximum clock frequency (f_{XTAL}) as a function of supply voltage (V_{DD}).



(1) T_{amb} = 85 °C
(2) T_{amb} = 25 °C

Fig.10 Typical supply current (I_{DD}) in STOP mode as a function of supply voltage (V_{DD}).



(1) F_{xtal} = 16 MHz
(2) F_{xtal} = 10 MHz
(3) F_{xtal} = 3.58 MHz

Fig.11 Maximum supply current (I_{DD}) in operation as a function of supply voltage (V_{DD}).

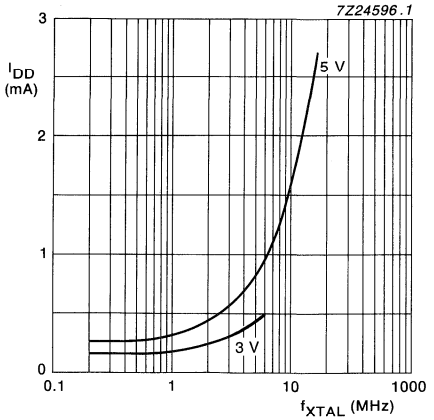
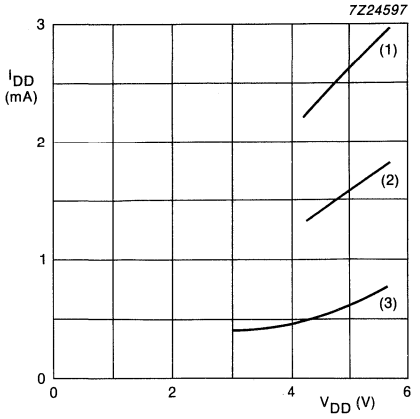


Fig.12 Typical supply current during operation as a function of frequency at V_{DD} = 3 V and V_{DD} = 5 V.



(1) F_{xtal} = 16 MHz
(2) F_{xtal} = 10 MHz
(3) F_{xtal} = 3.58 MHz

Fig.13 Maximum supply current (I_{DD}) in idle mode as a function of supply voltage (V_{DD}).

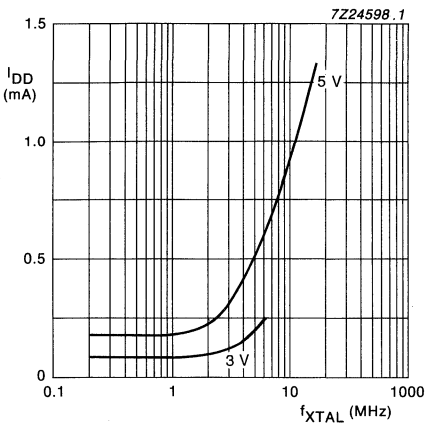


Fig.14 Typical supply current during idle mode as a function of frequency at V_{DD} = 3 V and V_{DD} = 5 V.

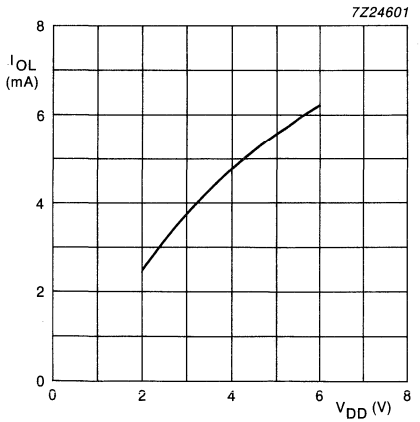
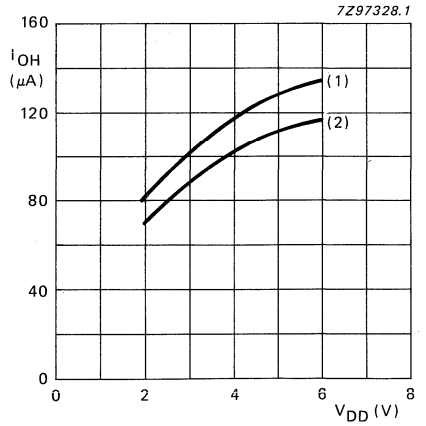


Fig.15 Typical output sink current (I_{OL}) as a function of supply voltage (V_{DD}); V_O = 0.4 V.



(1) $V_O = V_{SS}$
(2) $V_O = 0.7\ V_{DD}$

Fig.16 Typical output source current ($-I_{OH}$) as a function of supply voltage (V_{DD}).

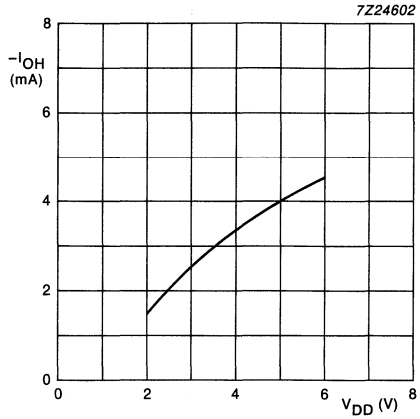


Fig.17 Typical push-pull output source current ($-I_{OH}$) as a function of supply voltage (V_{DD}); $V_O = V_{DD} - 0.4$ V.

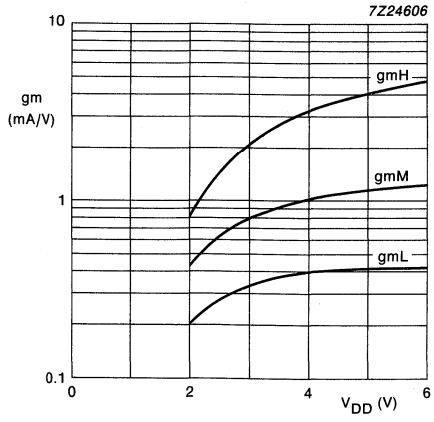


Fig.18 Typical transconductance values as a function of supply voltage (V_{DD}) for the options gmL, gmM and gmH.

Section 5 - 33XX Family and Derivatives

Single-chip 8-bit microcontroller family specification

PCD33XX

CONTENTS

Single-chip 8-bit microcontroller family specification

PCD33XX

1 INTRODUCTION

This family data sheet describes the microcontroller core which is common for all members of the PCD33XX family. For complete information of a particular microcontroller, consult both the specific microcontroller data sheet and this family data sheet.

2 FEATURES

- 8-bit CPU, ROM, RAM, I/O in a single package
- Up to 8 K bytes ROM
- Up to 256 bytes RAM
- Over 80 instructions, all instructions 1 or 2 cycles
- 8 or more quasi bi-directional I/O port lines
- 8-bit programmable timer/event counter
- 3 single-level vectored interrupts: external (chip enable), 8-bit programmable timer/counter, SIO/ derivative
- 2 Test inputs: T0 (may also be used as an interrupt) and T1 (may also be used as an input to an 8-bit counter)
- Serial I/O interface (not all devices)
- Power-on-reset
- 2 power reduction modes: Idle and Stop
- V_{DD} supply range: 1.8 V to 6 V (not all family members; for details on the supply range of a particular device, see the specific microcontroller data sheet)
- Clock frequency range: 450 kHz to 10 MHz
- Operating temperature range: – 20 °C to 70 °C
- Silicon gate CMOS fabrication process

3 GENERAL DESCRIPTION

The PCD33XX single-chip 8-bit microcontroller family consists of a wide range of derivatives containing up to 8 K bytes of on-chip mask programmable program ROM and up to 256 bytes RAM. All devices include flexible I/O ports, an 8-bit programmable timer/event counter and a choice of single-level vectored interrupts. The instruction set is based on that of the MAB8048. A number of PCD33XX family members may be used as CMOS replacements for their NMOS counterparts, where lower power consumption and higher speed are required.

The family is well supported with:

- Cross assemblers
- In-circuit emulation tools
- Window debugger
- Piggy-back versions for prototyping.

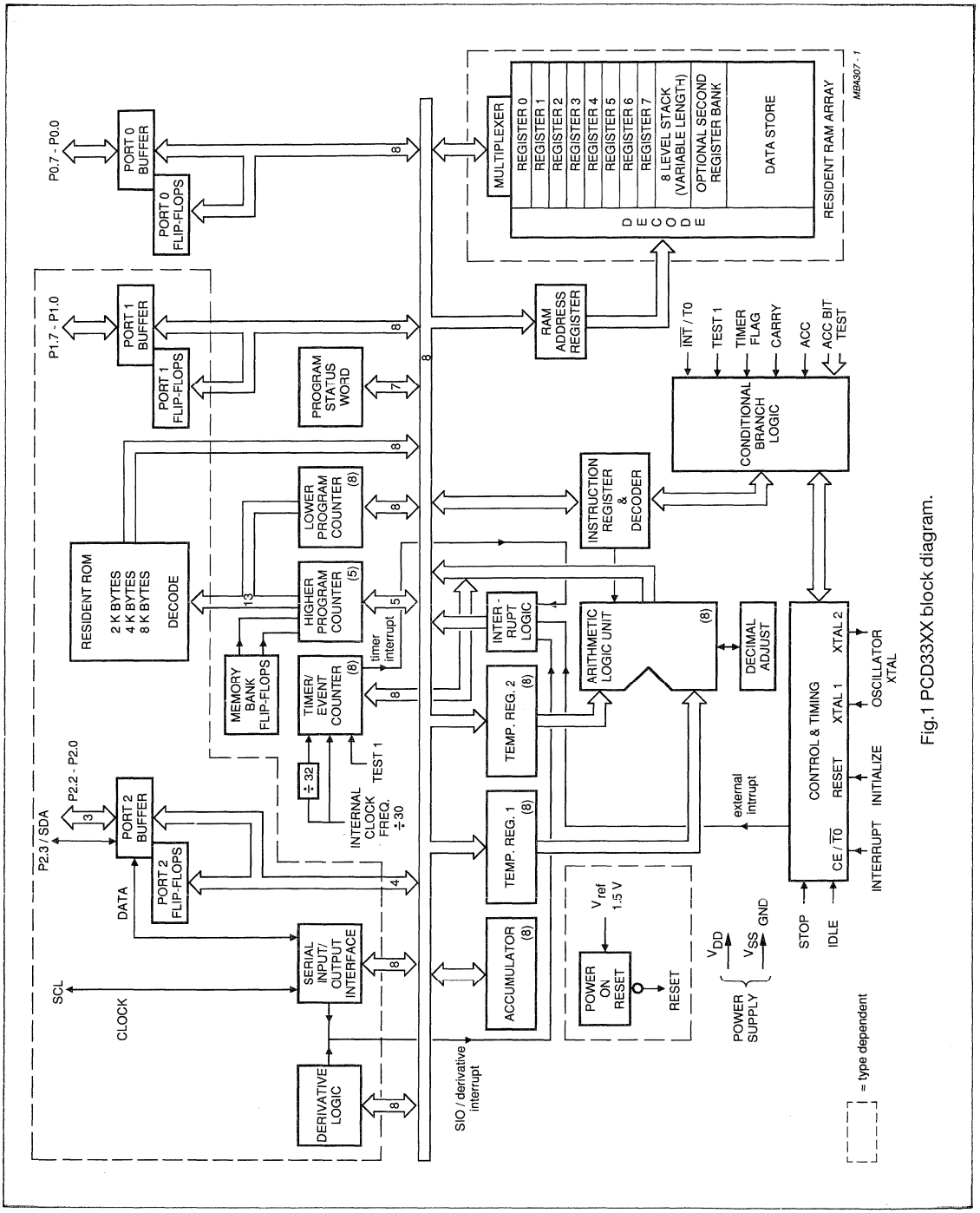


Fig.1 PCD33XX block diagram.

4 FUNCTIONAL DESCRIPTION

4.1 Program memory

Program memory consists of up to 8 K bytes of read-only memory (ROM). Each location is directly addressable by the program counter. The program memory is mask-programmed at the factory. Figure 2 shows the program memory map.

Four program memory locations are of special importance:

- Location 0: first instruction to be executed after the microcontroller is reset.
- Location 3: first instruction of an external (chip enable) interrupt ($\overline{CE}/\overline{T0}$) service routine.
- Location 5: first instruction of an SIO/derivative interrupt service routine.
- Location 7: first instruction of a timer/event counter interrupt service routine.

Of the 13-bits in the program counter, only 11 function as a counter. The two most significant bits are preset by SEL MB instructions. Thus, program memory is arranged in banks of 2 K bytes. Memory bank boundaries can only be crossed using unconditional branches (JMP) or subroutine calls (CALL) after the appropriate memory bank has been selected by a SEL MB instruction.

Each program memory bank is further divided into 8 pages of 256 bytes. Indirect (JMPP) and conditional branches can't cross page boundaries.

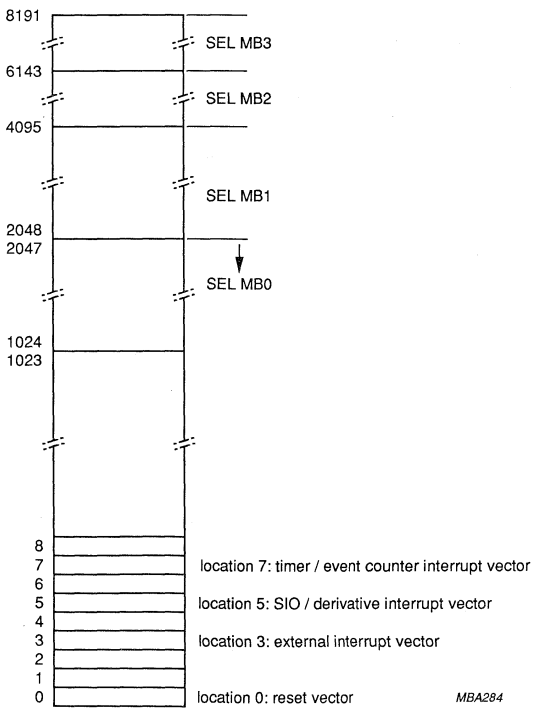


Fig.2 Program memory map.

4.2 Data memory

Data memory consists of up to 256 bytes of random access memory (RAM). All locations are indirectly addressable using RAM pointer registers. Up to 16 register locations are directly addressable. Memory also includes an 8-level program counter stack addressed by a 3-bit stack pointer. All RAM locations make efficient program loop counters when used with the 'decrement register and test' instruction DJNZ. Figure 3 shows the data memory map.

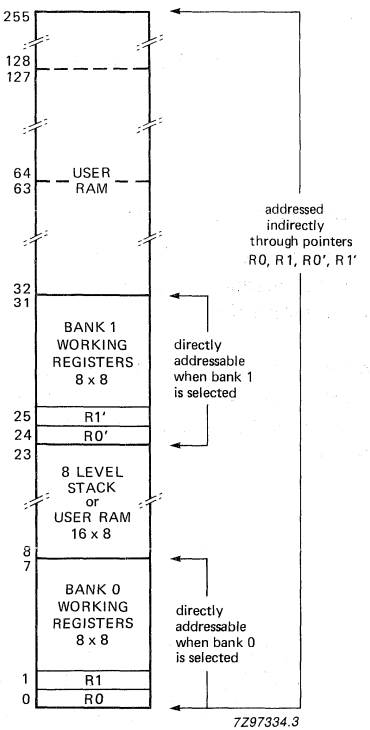


Fig.3 Data memory map.

4.2.1 Working registers

Locations 0 to 7 may be selected as working registers by the SEL RB0 instruction. These locations may then be accessed using efficient one byte, one cycle instructions. Ease of addressing, and a minimum requirement of instruction bytes to manipulate their contents make the working registers ideal for frequently accessed intermediate results.

Instead of locations 0 to 7, locations 24 to 31 may be selected as working registers by the SEL RB1 instruction. Register bank 1 may be used as an extension of register bank 0, as an alternative register bank for use by interrupt service routines, or as general purpose data memory.

The first two locations of each bank contain the RAM pointer registers (R0, R1, R0' and R1') which indirectly address all RAM locations.

4.2.2 Program counter stack

Locations 8 to 23 may be used as an 8-level program counter stack reserving 2 locations per level, or as general purpose RAM. The stack (Fig.4) saves return addresses and status during interrupt or subroutine servicing. Up to 8 levels of subroutine/interrupt nesting is possible.

A 3-bit stack pointer always points to the next free stack level. Following a device reset, the stack pointer points to level 0 (locations 8 and 9). On each subroutine call or interrupt, the contents of the 13-bit program counter and bits 4, 6 and 7 of the program status word are transferred to the level indicated by the stack pointer. The stack pointer increments and points to the next free level. Overflow from level 7 to level 0 occurs after nesting eight levels deep. Further subroutine calls and/or interrupts should be avoided since the contents of level 0 would be overwritten and lost.

The RETR instruction must be used to terminate an interrupt service routine. The RETR instruction decrements the stack pointer, and restores the program counter and status word. A subroutine should be terminated with the RET instruction. The RET instruction restores the program counter but does not restore the program status word.

As a general rule, the use of RETR in conjunction with a subroutine call is not recommended. RETR must not be used to terminate a subroutine which has been called within an interrupt routine since it would prematurely terminate the interrupt routine.

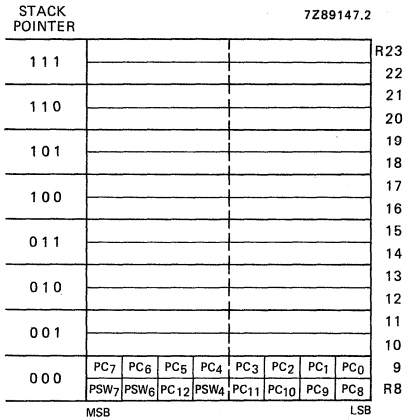


Fig.4 Program counter stack.

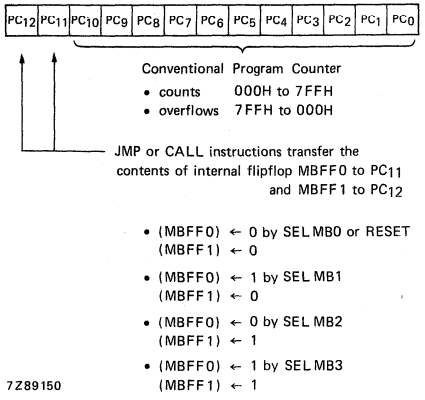


Fig.5 Program counter.

4.3 Program counter

The 13-bit program counter can address up to 8 K bytes of ROM (Fig.5). The least significant 11 bits (PC0 to PC10) are auto-incrementing. The two most significant bits, PC11 and PC12, are loaded with the contents of two internal flip flops (MBFF0 and MBFF1 respectively) when a JMP or CALL instruction is executed. The contents of these two internal flip-flops can be altered by the SEL MB instructions.

4.4 Program status word

The program status word (PSW) is an 8-bit register in the CPU which stores information about the current status of the microcontroller (Fig.6).

All bits can be read using the MOV A,PSW instruction. Only the PS bit can be written using the MOV PSW,A instruction. Table 1 describes the PSW bits and how they are affected.

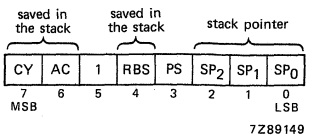


Fig.6 Program status word.

Table 1 Program status word.

BIT	NAME	FUNCTION	AFFECTED BY
7	CY	Carry, signals accumulator overflow	ADD, ADDC, DA, RLC, RRC, CLR C and CPL C instructions
6	AC	Auxiliary Carry, half carry	ADD and ADDC instructions
5	-	Not used, always 1 when read	
4	RBS	Register Bank Select 0: select register bank 0 1: select register bank 1	SEL RB instructions
3	PS	Timer Prescaler Select 0: prescaler selected (÷32) 1: prescaler not selected (÷1)	MOV PSW,A instruction
2 1 0	SP2 SP1 SP0	Stack Pointer bits 2-0	CALL, RET, RETR instructions and interrupts

Table 2 Interrupts.

INTERRUPT			INSTRUCTION	
PRIORITY	TYPE	VECTOR LOCATION	ENABLE	DISABLE
1 (highest)	external (chip enable)	003 (ROM)	EN I	DIS I
2	SIO/derivative interrupt	005 (ROM)	EN SI	DIS SI
3 (lowest)	timer/event counter	007 (ROM)	EN TCNTI	DIS TCNTI

4.5 Central processing unit

The PCD33XX instruction set provides arithmetic, logical, branching, input/output, and control facilities. The DA A, SWAP A, and XCHD instructions simplify BCD arithmetic and nibble handling. The MOVP A,@A instruction enables efficient table look-up within the current ROM page. The instruction set also has facilities for conditional branching and loop control (DJNZ).

4.6 Interrupts

The PCD33XX family handles external (chip enable), SIO/derivative and timer/event counter interrupts. The interrupt mechanism is single level. An executing interrupt routine can only be interrupted by a hardware RESET; it can't be interrupted by other interrupts (which are latched). If several interrupt requests are detected simultaneously, they are serviced in order of priority (see Table 2).

An interrupt request will only be serviced if the corresponding interrupt enable flag is set (Fig.7). When a request is serviced, the contents of the program counter, and bits 4, 6 and 7 of the program status word are saved on the program counter stack. The program counter is loaded with the appropriate interrupt vector which points to the start of the interrupt service routine. Since the accumulator is not automatically saved, it must be saved and restored by user software. The interrupt routine must be terminated by the RETR (return and restore) instruction. At least one instruction in the main program will then be executed before another interrupt routine is serviced.

To avoid erroneous real-time programs, a few words of caution:

- While the interrupt is in progress, the two most significant bits of the program counter are frozen at zero. Thus, interrupt routines and subroutines called from interrupt routines must reside (entirely) in memory bank 0.
- The SEL MB instruction should not be used within interrupt routines or in subroutines called within interrupt routines because altering the contents of MBFF0 and MBFF1 (Fig.5) may lead to erroneous JMP and CALL destinations after return from interrupt.
- Subroutines and nested subroutines called within an interrupt routine must all end with RET since RETR clears the IIP (interrupt in progress) flag (Fig.7 and Fig.8). Further pending interrupts would then interfere with the interrupt routine in progress.

4.6.1 External (chip enable) interrupt

A LOW-TO-HIGH transition on the CE/$\overline{T0}$ pin is latched in the digital filter latch if the HIGH state exceeds 7 clock periods after a LOW state of more than 4 clock periods. If the external interrupt is enabled, then the external interrupt flag (EIF) will also be asserted, constituting a valid external interrupt request. As soon as the IIP is clear, indicating that no interrupt routine is in progress, the external interrupt is invoked by a forced CALL to location 3. The EIF flag is simultaneously cleared (Fig.7 and Fig.8). The interrupt routine may acknowledge the interrupt via port lines.

Execution of a DIS I (disable external interrupt) instruction cancels a stored interrupt request by clearing both the digital filter latch and the EIF.

EI	External Interrupt
SI	SIO/Derivative Interrupt
TI	Timer/event counter Interrupt
EIF	External Interrupt Flag
TIF	Timer Interrupt Flag
PIN	Pending Interrupt Not
IIP	Interrupt In Progress Flag

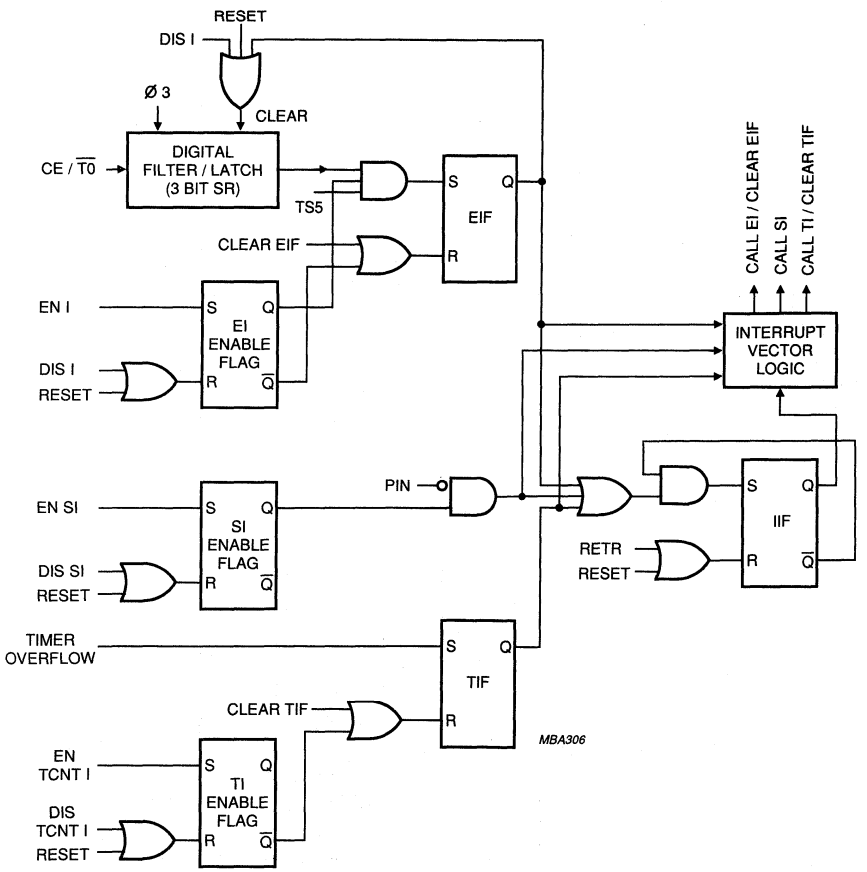


Fig.7 Simplified interrupt logic schematic (the R input overrules the S input for all flags).

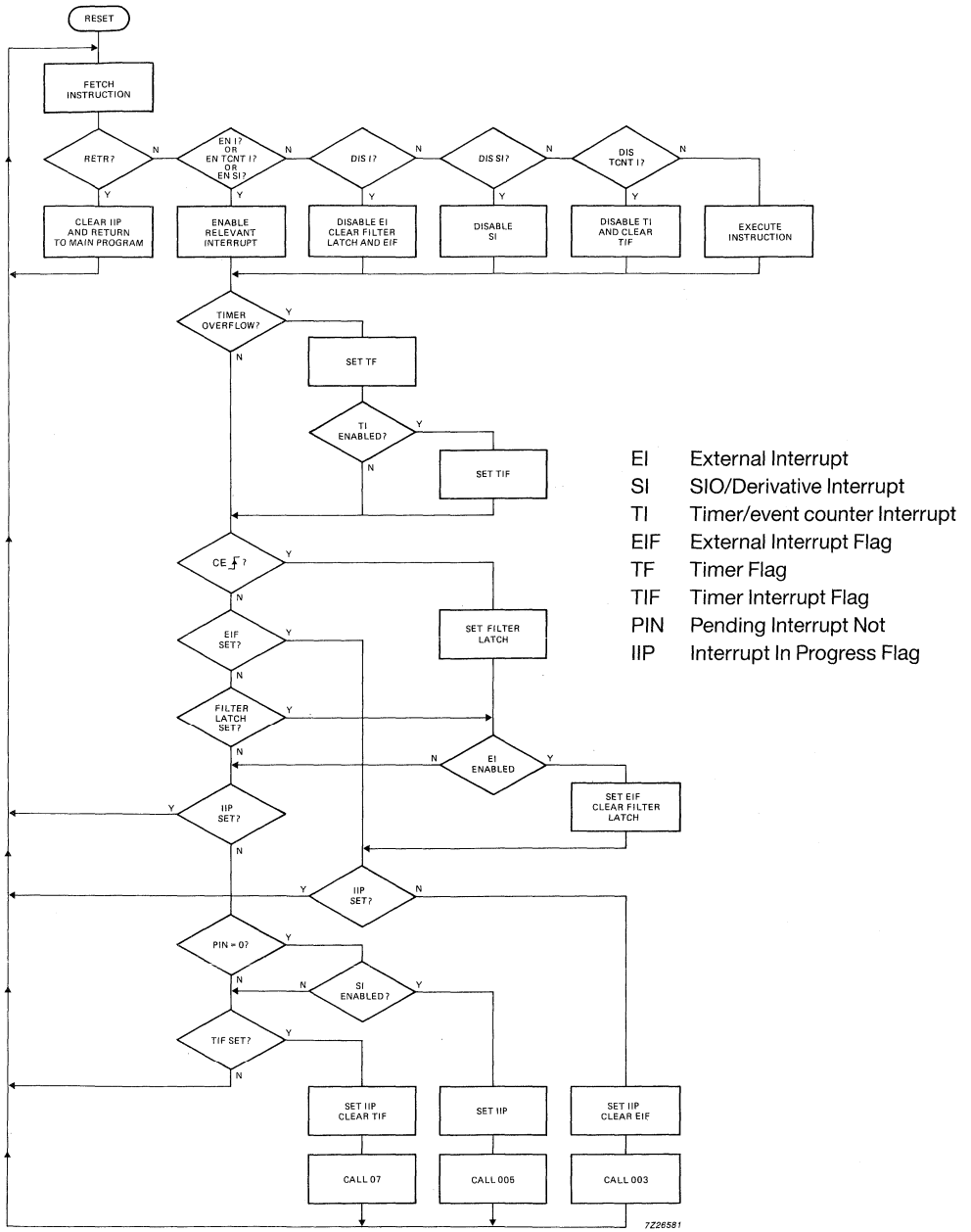


Fig.8 Flow chart illustrating CPU control in the presence of interrupts.

4.6.2 SIO/Derivative interrupt

The SIO/derivative interrupt is shared between the serial I/O interface (if available) and the derivative logic (if available). Software polling may be necessary to determine the origin of a request.

An interrupt request from the SIO or derivative logic will force the PIN flag to its active LOW state. This action is independent of the Enable SIO interrupt enable flag. If the SIO/derivative interrupt is enabled and no interrupt routine is in progress, the active LOW PIN flag will invoke the SIO/derivative interrupt routine by forcing a CALL to program memory location 5. Invoking the SIO/derivative interrupt routine does not automatically clear the PIN flag. PIN must be cleared to its inactive HIGH state by software during the interrupt routine.

More details on SIO interrupts are given in the section on the serial I/O interface. For specific information on derivative interrupts, consult the relevant data sheet.

4.6.3 Timer/event counter interrupt

When the timer/event counter interrupt is enabled, a timer/event counter overflow sets the Timer Interrupt Flag (TIF). As soon as the Interrupt in Progress (IIP) flag is clear (indicating that no interrupt routine is in progress), the timer/event counter interrupt routine is invoked by a forced CALL to program memory location 7, and the TIF flag is simultaneously cleared (Fig.7 and Fig.8).

Execution of a DIS TCNTI (disable timer/event counter interrupt) instruction cancels a stored interrupt request by clearing the TIF flag.

An additional external interrupt may be simulated by enabling the timer/event counter (EN TCNTI) and loading the counter with FFH (one less than overflow). If the event counter mode is enabled by executing the STRT CNT instruction, a rising edge on the T1 input will cause a counter overflow and set TIF.

4.7 Interrupt/Test 0 input (CE/$\overline{T0}$)

The CE/$\overline{T0}$ input may be used as:

- chip enable
- a test 0 input for branch instructions JT0 and JNT0.

When used as a test 0 input (chip enable disabled):

- the conditional branch instruction JT0 will cause a branch if CE/$\overline{T0}$ = logic 0
- the conditional branch instruction JNT0 will cause a branch if CE/$\overline{T0}$ = logic 1.

There is no internal pull-up or pull-down resistor connected to the CE/$\overline{T0}$ input. When CE/$\overline{T0}$ is not used, it must be tied to V_{DD} or V_{SS}.

4.8 Timer/event counter

The internal 8-bit up-counter may be configured to count external events, modulo-32 machine cycles, or machine cycles directly. The MOV A,T and MOV T,A instructions can be used to read and preset the counter.

After a STRT T (start timer) instruction, the counter will increment either every machine cycle (30 oscillator periods) or every 32 machine cycles. If the PS bit in the program status word is set, the counter increments every machine cycle. If the PS bit is reset, it increments every 32 machine cycles. STRT T clears the prescaler which is not otherwise accessible (see Fig.9).

After a STRT CNT (start event counter) instruction, the counter will count each LOW-to-HIGH transition on pin T1 provided that the HIGH state exceeds 4 oscillator periods after a LOW state of more than 4 oscillator periods. The maximum count rate is one increment per machine cycle ($f_{XTAL}/30$).

The timer/event counter is inhibited after reset or by executing a STOP TCNT (stop timer/event counter) instruction (see Fig.9).

When a T1 overflow occurs:

- if the timer/event counter interrupt is enabled, the Timer Interrupt Flag (TIF) is set and an interrupt request is generated
- the Timer Flag (TF) is set. TF can be tested by conditional branch instructions JTF (jump if TF = logic 1) and JNTF (jump if TF = logic 0). When a JTF or JNTF instruction is executed, TF is reset. The only other way to clear TF is to reset the microcontroller.

Table 3 Timer/event counter control.

FUNCTION	TIMER MODE	COUNTER MODE
clear	MOV T,A (A) = 0 or RESET	MOV T,A (A) = 0 or RESET
preset	MOV T,A	MOV T,A
start	STRT T	STRT CNT
stop	STOP TCNT or RESET	STOP TCNT or RESET
test	JTF/JNTF	JTF/JNTF
read	MOV A,T	MOV A,T

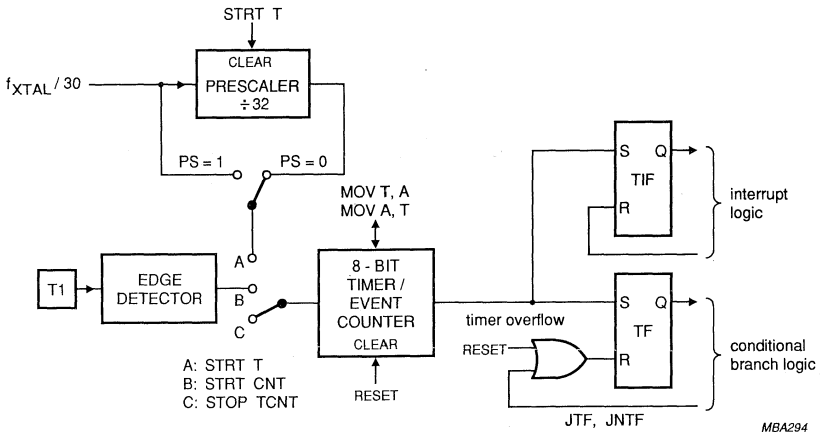


Fig.9 Timer/event counter.

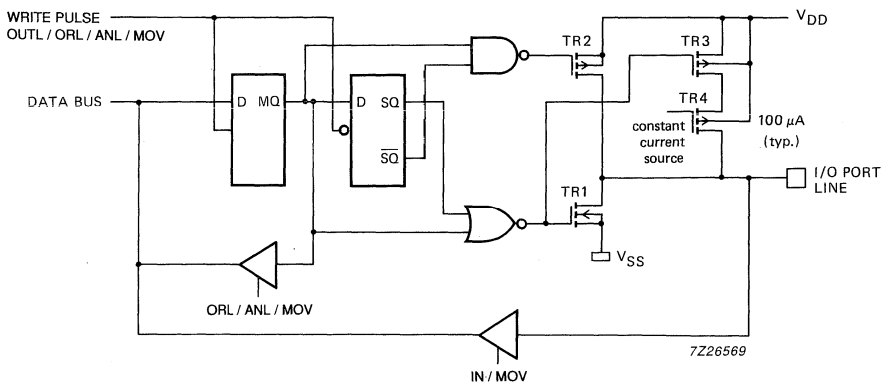


Fig.10 Standard port with switched pull-up current source (mask option 1).

4.9 Test 1/Count input (T1)

The T1 input may be used as:
- a count input to the 8-bit timer/event counter
- a test 1 input for branch instructions JT1 and JNT1.

When used as a test input:
- the conditional branch instruction JT1 will cause a branch if T1 = logic 1
- the conditional branch instruction JNT1 will cause a branch if T1 = logic 0.

There is no internal pull-up or pull-down resistor connected to the T1 input. When T1 is not used, it must be tied to V_{DD} or V_{SS}.

4.10 Parallel ports

The PCD33XX family provides up to three standard quasi-bidirectional I/O ports:
- Port 0: 8-bit parallel port (P0.0 to P0.7)
- Port 1: 8-bit parallel port (P1.0 to P1.7)
- Port 2: 4-bit parallel port (P2.0 to P2.2, P2.3/SDA)

Several members of the PCD33XX family provide all 20 I/O lines and all members contain port 0.

In addition to the standard ports, many PCD33XX microcontrollers provide a variety of derivative ports. For specific details, consult the relevant data sheet.

In general, all parallel ports lines can be individually configured as outputs or inputs. Output data written to a port is latched and remains unchanged until rewritten. Input data is not latched and must be stable when read by an input instruction.

The standard port configuration is shown in Fig.10. When a logic 0 is written to the master/slave flip-flop, TR2 and TR3 are turned off, turning off the constant current source. Current sinking is provided by TR1 which is simultaneously turned on, and the output is pulled LOW to V_{SS}.

When a logic 1 is written to the master/slave flip-flop for the first time (MQ = 1, SQ = 0), TR1 is turned off and TR3 is turned on. TR2 is also switched on for the duration of the internal write pulse (one oscillator period), driving the output rapidly to V_{DD}. TR3 turns on the constant current source TR4 which sources sufficient current for a TTL HIGH level; however, the port line can be pulled LOW by an external CMOS device, enabling the same pin to be used for both input and output. This arrangement also facilitates wired-OR applications. Subsequent writing of a logic 1 to the port line will not switch TR2 on. Booster transistor TR2 is only turned on for one oscillator period during a 0-to-1 transition at the output of the master/slave flip-flop.

To use the port line as an input, a logic one must written to the port line to turn TR1 off.

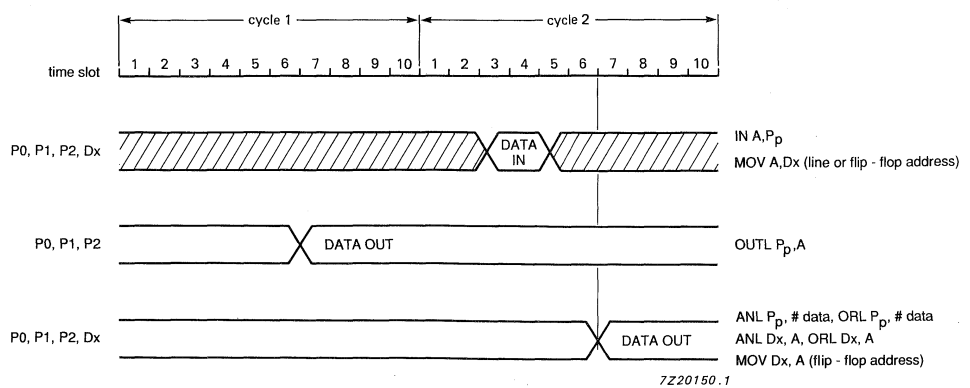


Fig.11 Input/output timing of standard and derivative ports.

Table 4 Derivative port addressing.

DERIVATIVE ADDRESS	TYPE	ACCESS
8-bit line address	R	derivative port line
8-bit flip-flop address	R/W	derivative port flip-flop

Ports 0, 1 and 2 are accessed using the parallel input/output instructions IN, OUTL, ANL and ORL. IN inputs port data to the accumulator. OUTL outputs accumulator data to the port. ANL and ORL are used to manipulate data in the port flip-flops.

Derivative ports are accessed by the derivative input/output instructions MOV, ANL and ORL. ANL and ORL are used to manipulate data in the port flip-flops. MOV is used for all data transfers between the port and the accumulator. When reading data into the accumulator, the data source can be a port line or the port flip-flop. Thus, two derivative addresses are provided per port (see Table 4).

All standard and derivative port accesses are performed by two-cycle instructions. Instruction timing is shown in Fig.11. For input operations, data is read during time slots 3 and 4 of machine cycle 2. For output operations, the data is written during time slot 7. For OUTL, data is written during machine cycle 1. For ANL, ORL and MOV Dx,A, data is transferred during machine cycle 2.

Three I/O mask options make it possible for every parallel I/O port line to be individually configured as follows:

Option 1 Standard Port: quasi-bidirectional I/O with switched pull-up current source (100 μA typ.) and p-channel booster transistor TR2. TR2 is only active for 1 clock cycle during 0-to-1 transitions (Fig.10).

Option 2 Open Drain: quasi-bidirectional I/O with only an N-channel open drain output. Application as an output requires the connection of an external pull-up resistor (Fig.12). If unused, an option 2 output should be tied to V_{SS} to prevent undesirable current flow through input stages.

Option 3 Push-Pull: outputs can sink or source 2 mA (typ.) at V_{DD} = 3 V. Push-pull lines may not be used as inputs (Fig.13).

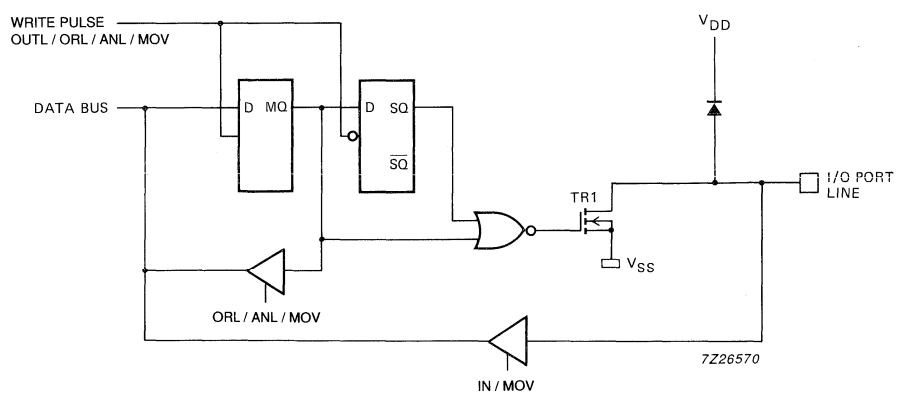


Fig.12 Open drain (mask option 2).

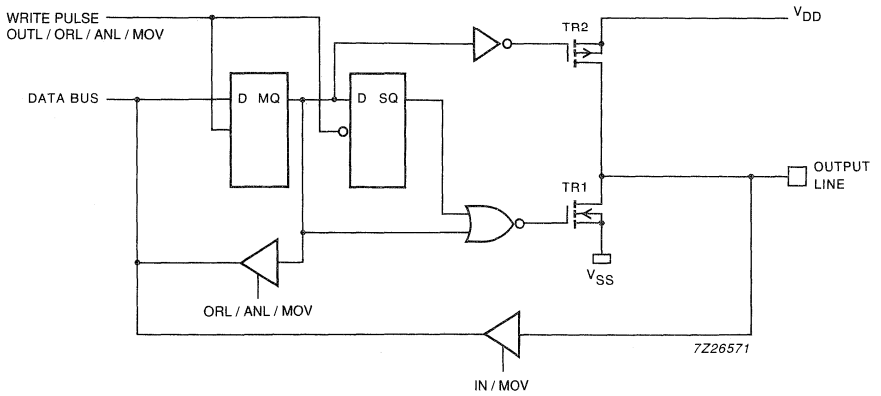


Fig.13 Push-pull (mask option 3).

For devices with a serial I/O interface, P2.3 becomes SDA and must be configured as an open drain (option 2) I/O line.

For the remaining port lines (P0.0 to P2.2), all three options are generally available. For some family members, option 3 on port 0 and port 1 lines may be restricted for emulation purposes. For specific information, refer to the relevant data sheet.

4.11 Serial I/O interface

Many members of the PCD33XX family have an on-chip serial I/O interface (I^2C). This two-line serial bus extends the microcontroller capabilities since it fully supports the PCF85XX (clips) family of I^2C-bus peripheral devices.

Microcontrollers that do not have dedicated serial I/O hardware can use port pins and software to simulate a serial interface. However, such microcontrollers must continuously monitor the serial bus, the data transfer rate is slower, and this approach may require significant processing and memory resources.

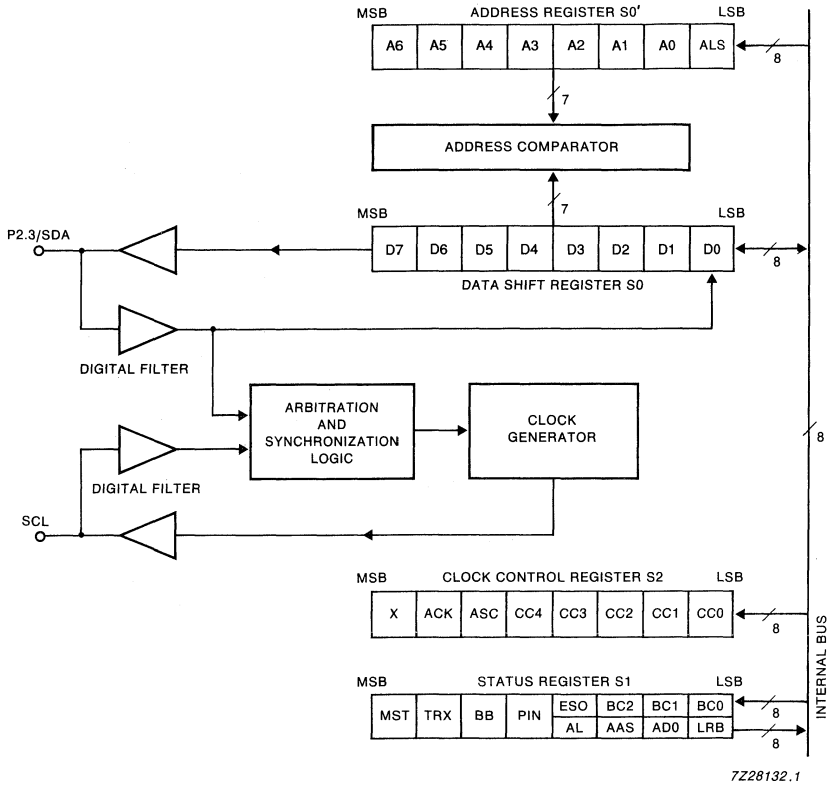
Each device on the I^2C-bus is allocated a 7-bit address. Address recognition is performed by the serial interface hardware and the microcontroller is interrupted only after a valid address (own address or general call address) has been recognized. The SIO hardware also transfers data serially and performs parallel-to serial and serial-to-parallel conversion without disrupting program execution. The microcontroller is interrupted only after a complete data byte has been transferred; the next data byte can then be written to or read from the serial I/O interface.

The serial I/O interface also facilitates the implementation of multimaster systems in which two or more microcontrollers communicate via the same I^2C-bus. An automatic arbitration procedure handles bus conflicts.

The I^2C-bus consists of a dedicated bidirectional clock line (SCL) and a bidirectional data line (P2.3/SDA) which also functions as parallel port line P2.3. When the serial I/O interface is enabled, P2.3 is disabled as a port line. Input signals on SCL and SDA are filtered for enhanced noise immunity. SCL and SDA are open drain and require external pull-up resistors if they are to be used as outputs. If unused, these two pins should be tied to V_{SS}.

The CPU and serial I/O interface communicate via the following four serial I/O interface registers (see Fig.14):

- Data Shift Register (S0)
- Address Register (S0')
- Clock Control Register (S2)
- Status Register (S1)



X = don't care

Fig.14 Serial I/O interface block diagram.

4.11.1 Data shift register (S0)

The data shift register converts serial data to parallel format and vice versa. The most significant bit is transferred first. An interrupt request is generated after a complete byte has been transferred or after a valid I²C-bus address has been detected. The MOV A,S0 instruction may be used to read S0; the MOV S0,A and MOV S0,#data instructions may be used to write to S0 if the ESO (enable serial I/O) bit in the status register (S1) is set.

4.11.2 Address register (S0')

The address register contains the device's 7-bit I²C-bus address and the ALS (always selected) bit. When ALS is zero (the recommended operating mode) bus transfers are ignored unless the START condition is immediately followed by the valid device address or the 'general call' address (00H). If ALS is set, any transfer on the bus is stored in the data shift register. The address register S0' is write-only. The MOV S0,A and MOV S0,#data instructions may be used to write to S0' if the ESO (enable serial I/O) bit in the status register (S1) is zero.

4.11.3 CLOCK CONTROL REGISTER (S2)

The clock control register is a write only register. Bits 0 to 4 of S2 define the serial clock frequency (f_{SCL}) as an integer multiple of the microcontroller clock frequency (Table 5).

Bit 5 (ASC) defines the asymmetrical clock duty cycle. If ASC is set, SCL has a duty cycle of approximately 75%. The asymmetrical clock limits the I²C-bus transmission rate to below 55 kHz. Divisors 39, 45 and 51 are not allowed if ASC is set. Resetting ASC (recommended operating mode) sets the SCL duty cycle to approximately 50%, allowing higher I²C-bus transmission rates of up to 100 kHz, and all of the divisors in Table 5 may be used.

For normal I²C-bus operation, bit 6 (ACK) must be set. After each byte transfer, an extra SCL pulse is generated during which the receiver may acknowledge reception. If ACK is reset, no acknowledge phase is available. This mode is used when a master receiver refuses to acknowledge a transfer in order to signal the 'end of transmission' to a slave transmitter.

Table 5 f_{SCL} as defined by the clock control register S2.

CC0 to CC4 (HEX)	f_{XTAL} DIVIDED BY	f_{SCL} (kHz) @ f_{XTAL} = 6 MHz	f_{SCL} (kHz) @ f_{XTAL} = 10 MHz
0	not allowed		
1	39	154*	256*
2	45	133*	222*
3	51	118*	196*
4	63	95	159*
5	75	80	133*
6	87	69	115*
7	99	61	101*
8	123	49	81
9	147	41	68
A	171	35	58
B	195	31	51
C	243	25	41
D	291	21	34
E	339	18	29
F	387	16	26
10	483	12	21
11	579	10	17
12	675	8.9	15
13	771	7.8	13.4
14	963	6.2	10.4
15	1155	5.2	8.7
16	1347	4.5	7.4
17	1539	3.9	6.5
18	1923	3.1	5.2
19	2307	2.6	4.3
1A	2691	2.2	3.7
1B	3075	2.0	3.3
1C	3843	1.6	2.6
1D	4611	1.3	2.2
1E	5379	1.1	1.9
1F	6147	1.0	1.6

* not permitted; f_{SCL} max. = 100 kHz in I²C systems

4.11.4 STATUS REGISTER (S1)

The status register controls the serial I/O interface and provides feedback about on-going bus transfers. Register S1 can be accessed by the MOV A,S1, MOV S1,A and MOV S1,#data instructions. The lower nibble of the status register is duplicated: control bits BC0-BC2 and ESO are write only, whereas feedback bits LRB, AD0, AAS and AL are read only. The status bits interact in intricate ways with each other. This must be kept in mind when developing an I²C-bus application. Table 6 describes the status bits.

4.11.4.1 Master bit (MST) and transmitter bit (TRX)

The MST and TRX bits determine the operating mode of the serial I/O interface. When not engaged in a bus transfer, MST and TRX should always be zero, placing the SIO in the slave receiver mode (see Fig.15). Return to the slave receiver mode is always performed by software. If the previous mode was a master mode, the transition (MOV S1,#D8H) involves a STOP condition, which automatically clears both MST and TRX.

The transition to the master transmitter mode is also performed by software. However, transitions to the master receiver and slave transmitter modes occur automatically if ALS = 0 (standard I²C-bus protocol). A slave receiver becomes a slave transmitter if the R/$\overline{W}$ bit in the valid address immediately following a START condition is set. A master transmitter becomes a master receiver if R/$\overline{W}$ is set in the transmitted address.

Table 6 Serial I/O status register (S1).

BIT	NAME	TYPE	DESCRIPTION
MST	Master	R/W	MST = 0: Slave (SCL input) MST = 1: Master (SCL output)
TRX	Transmitter	R/W	TRX = 0: Receiver (SDA/P2.3 input) TRX = 1: Transmitter (SDA/P2.3 output)
BB	Bus Busy	R/W	BB = 0: Bus inactive (R)/generates STOP condition (W) BB = 1: Bus busy (R)/generates START condition (W)
PIN	Pending Interrupt Not	R/W	PIN = 0: Serial interrupt pending (after byte transfer, valid address or lost arbitration); SCL forced to V_{SS} PIN = 1: No serial interrupt pending
ESO	Enable Serial I/O	W	ESO = 0: Serial I/O interface disabled; write access to S0' possible ESO = 1: Serial I/O interface enabled; write access to S0 possible
BC0, BC1, BC2	Bit Counter	W	Preset of the Bit Counter for 1 up to 8 serial data bits. (001) = 1 bit, (010) = 2 bits, etc. (000) = complete byte (8 bits) = default value.
AL	Arbitration Lost	R	AL = 1: Arbitration Lost (bus conflict) AL = 0: When corresponding serial interrupt (PIN) is cancelled
AAS	Addressed As Slave	R	AAS = 1: Following a START condition if a valid address is detected (ALS = 0), or if the first byte is received (ALS = 1) AAS = 0: When corresponding serial interrupt (PIN) is cancelled
AD0	Address Zero	R	AD0 = 1: Following a START condition if the general call address (00H) is detected AD0 = 0: After a repeated START or a STOP condition
LRB	Last Received Bit	R	Set or reset depending on last bit transferred, acknowledgement bit if ACK = 1

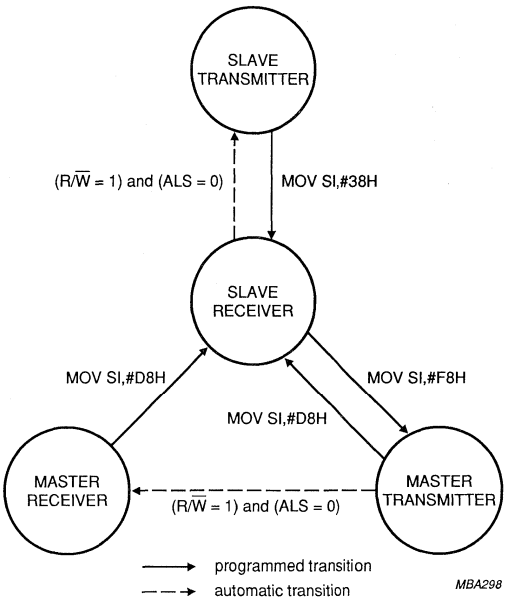


Fig.15 State diagram of the serial I/O interface.

4.11.4.2 Pending interrupt not bit (PIN)

If either MST or ALS is set, PIN is reset to zero (activated) after every byte transfer. If MST and ALS are both reset, PIN becomes zero (generating a serial interrupt request) when the valid address is detected and after each byte of the following transfer. Clock synchronization is implemented as follows: the SCL line is pulled to V_{SS} as long as PIN is zero enabling a slave to delay a master in order to read the data register (in the case of a slave receiver) or to write to the data register (in the case of a slave transmitter). PIN is automatically inactivated when S0 is accessed; it may also be inactivated by explicitly setting PIN to logic 1.

If the SIO/derivative interrupt is disabled, the serial I/O interface may be serviced by testing PIN directly in user software.

4.11.4.3 Bus busy bit (BB)

The status of the BB bit is controlled by the serial I/O interface or by bus master software. When a master clears BB (MOV S1,D8H), the serial I/O interface automatically clears MST and TRX, returning to the slave receiver mode (see Fig.15). If BB is set, write access to S1 other than accesses by the master or an addressed slave are inhibited. If BB is inadvertently set by excessive noise on the bus, the deadlock can be resolved by executing two consecutive MOV S1,18H instructions, the first of which just clears BB.

When a slave transmitter detects an end of transmission (signalled by the absence of an acknowledgment from the master receiver), it has to access S1 to inactivate PIN and become a slave receiver. However, BB should remain set. This is reflected by the MOV S1,38H instruction in Fig.15. When PIN = logic 1, clock synchronization terminates, enabling the master to generate the STOP condition.

A START condition must only be generated when BB = logic 0. Otherwise, the serial I/O interface responds as if bus arbitration had been lost.

4.11.4.4 Arbitration lost bit (AL)

The AL bit is set by the serial I/O interface when it loses bus arbitration in the master transmitter mode. Simultaneously, MST and TRX are cleared to enable the interface (now in the slave receiver mode) to determine if it is validly addressed by the device that won the arbitration. PIN is activated when the byte transfer is complete. AL will be cleared when the serial interrupt is terminated.

4.11.4.5 Addressed as slave bit (AAS)

AAS is set by the serial I/O interface following a START condition when the valid address is detected (ALS = logic 0 in register S0') or when the first byte is received (ALS = logic 1 in register S0'). AAS is cleared when the serial interrupt is terminated.

4.11.4.6 Address zero bit (AD0)

AD0 is set by the serial I/O interface independently of ALS when the 'general call' address (00H) is detected following a START condition. AD0 is cleared after a repeated START or a STOP condition.

4.11.4.7 Last received bit (LRB)

LRB contains the last bit transferred. If ACK = logic 1, LRB contains the acknowledgement bit. It remains valid as long as PIN = logic 0.

4.11.4.8 Enable serial I/O bit (ESO)

The ESO bit enables/disables the serial I/O interface. When ESO = logic 0, access to register S0' is enabled, the SCL pin is in a high impedance state and the P2.3/SDA pin is made available as a normal I/O port line.

When ESO = logic 1, the serial I/O interface is enabled and access to register S0 is possible. The remaining S1 bits can only be altered when ESO is set. The SCL and P2.3/SDA pins are enabled as serial clock and data lines respectively.

To avoid bus deadlock, ESO must be reset before a STOP instruction is executed.

4.11.4.9 Bit counter bits (BC0, BC1 and BC2)

BC0, BC1 and BC2 should all be zero for normal I^2C-bus operation. The bit counter is always cleared by a START condition; thus all eight bits of the first byte are transferred.

If a non-zero bit counter value is chosen, it is only valid for one S0transfer since the counter decrements to zero. The bit counter is useful when a master receiver signals an end of transmission by sending a negative acknowledge after the last byte received; the last byte is received with ACK = logic 0 in register S2. The negative acknowledge is then issued by setting the bit counter to one and 'receiving' one bit from the HIGH level on the SDA line. The slave transmitter interprets the same signals as a negative acknowledgement.

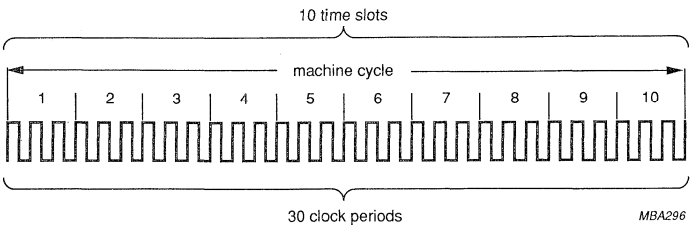


Fig.16 Machine cycle timing.

4.12 Timing

Every machine cycle consists of 10 time slots. Each time slot consists of 3 clock periods (see Fig.16).

The clock frequencies range is from 100 kHz to a maximum which is a function of supply voltage. When $V_{DD} \geq 4.5$ V, operation at 10 MHz is guaranteed.

The clock signal may be internally generated by the on-chip oscillator and an external crystal. Alternatively, an external clock may be applied to the XTAL1 pin.

4.13 Oscillator

The on-chip oscillator consists of an invertor stage including a feedback resistor and load capacitors (see Fig.17). A quartz crystal is usually connected between XTAL1 and XTAL2. Alternatively, a ceramic resonator or an inductor may be used as the timing element; however, external load capacitors should then be added for good frequency stability.

When the supply voltage drops below the power-on-reset level, the oscillator is inhibited. The internal oscillator may also be inhibited by the STOP instruction.

Oscillator start-up time depends on the external timing element. The start-up time of a quartz crystal is several milliseconds due to the narrow crystal bandwidth.

For proper oscillator start-up, the transconductance (gm) of the invertor stage must satisfy the following relationship (see Fig.17 and Fig.18):

$$4.2[R_1\omega^2(C_L + C_0 + C_f)^2 + 1/R_f] < gm < C_1C_2/[R_1(C_0 + C_f)^2 + 1/\omega^2R_f]$$

where:

R_1 = resonator series resistance
C_0 = static resonator capacitance
R_f = feedback resistor
C_L = $C_1C_2/[C_1 + C_2]$
C_f = parasitic feedback capacitance (typically 2 pF on-chip, external value depends on PC board wiring)
ω = $2\pi f_{XTAL}$

For more information on crystal oscillators and start-up conditions see the application note 'Crystal Oscillators for CMOS Circuits'.

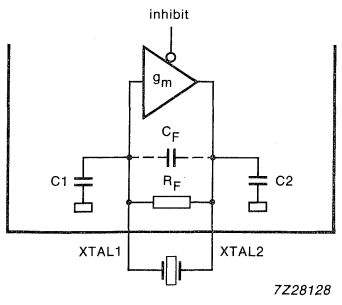


Fig.17 Oscillator with integrated elements.

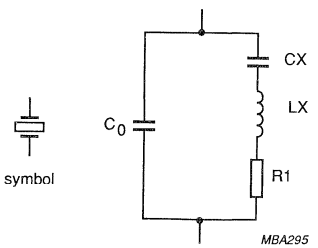


Fig.18 Crystal unit equivalent circuit.

4.14 Reset

After a falling edge on the internal reset, 1866 clock cycles are required to initialize the microcontroller to a defined state. The first instruction is then executed.

4.14.1 Passive reset

A passive reset is generated by the RC circuit shown in Fig.19. As V_{DD} rises, the discharged C_{reset} pulls the RESET pin close to V_{DD}. When V_{DD} crosses the power-on reference level V_{ref} (typically 1.5 V), the device generates a reset pulse of approximately 50 µs which helps to pull the RESET pin to V_{DD}. To ensure a correct reset, the RESET voltage should reach at least 70% of the final value of V_{DD} before C_{reset} charges through TR2 and R_{reset}. If the RESET voltage and V_{DD} rise exponentially, this requirement is satisfied when the time constant τ ($C_{reset}R_{reset}$) of the RESET pulse is greater than eight times the time constant of V_{DD}. If V_{DD} rises linearly, the requirement is satisfied when the time constant of the RESET pulse is greater than twice the time constant of V_{DD}.

In the event of a drop in the supply voltage, the diode rapidly discharges C_{reset}, ensuring reliable power-on-reset even after short supply voltage interruptions.

In battery-powered systems where V_{DD} quickly reaches its minimum operating value, passive reset can be performed without external components since the 50 µs reset pulse guarantees proper initialization.

4.14.2 Active reset

An active reset can be generated by driving the RESET pin HIGH with external logic. This pulse should be present until V_{DD} has reached its minimum operating value.

4.14.3 Reset state

After a reset, the microcontroller is initialized as follows: The program counter points to 00H. Memory bank 0, register bank 0, and stack pointer 0 (locations 8 and 9) are selected. All interrupts are disabled. The timer/event counter is stopped and cleared, the timer flag is cleared, and the timer prescaler is set to modulo 32 (PS = 0). All port flip-flops (except P2.3/SDA) are set to logic 1. P2.3/SDA and SCL are high impedance 30 clock pulses (max.) after the end of the internal reset pulse. The serial I/O interface is disabled (ESO = 0) and in the slave receiver mode 30 clock pulses (max.) after the end of internal reset pulse (S0, S0', S1 and S2 are cleared except when PIN = logic 1). The Idle and Stop modes are terminated.

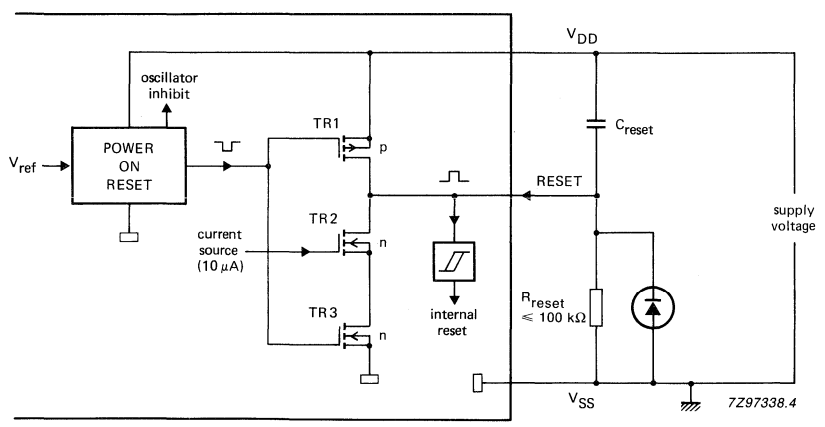


Fig.19 Passive power-on-reset.

4.15 Idle mode

The Idle mode is very useful in low power applications. When all computational tasks are completed, the device can be placed in the Idle mode instead of a power consuming waiting loop.

When the microcontroller enters the Idle mode by executing an IDLE instruction, all activity is halted except for the oscillator, the timer/event counter and the serial I/O interface(if available).

The Idle mode is terminated when an enabled interrupt (or reset) occurs. The interrupt routine is executed and program execution resumes at the instruction immediately following the IDLE instruction.

For timer/event counter interrupts and SIO/derivative interrupts, termination of the Idle mode is straightforward. However, care must be taken when the Idle mode is terminated by the external (chip enable) interrupt since CE/$\overline{T0}$ is rising-edge triggered. If CE/$\overline{T0}$ was HIGH prior to entering the Idle mode, it must go LOW before a rising edge can be generated. Fig.20 shows the exact timing for Idle mode termination with an external interrupt (CE/$\overline{T0}$).

If no interrupt is enabled, the Idle mode can only be terminated by an active signal on the RESET pin. A normal reset sequence is executed (Fig.20).

4.16 Stop mode

The Stop mode enables very low power operation. When all computational tasks are completed, the device can be virtually shut down by stopping the oscillator.

When the microcontroller enters the Stop mode by executing a STOP instruction, the oscillator is switched off. All internal states (CPU status, RAM) and I/O levels are maintained.

The Stop mode is terminated by a HIGH level on the CE/$\overline{T0}$ pin or a reset. In the latter case, a normal reset sequence is executed (see Fig.21).

Unlike the Idle mode, the microcontroller responds to a HIGH level on the CE/$\overline{T0}$ pin (i.e. not to a rising edge). If the CE/$\overline{T0}$ pin is HIGH when the STOP instruction is executed, the Stop mode will not be entered.

A rising edge on CE/$\overline{T0}$ causes program execution to continue after a delay of 1866 clock cycles. If the external interrupt is enabled, the microcontroller executes the instruction immediately following the STOP instruction before executing the interrupt routine. If the external interrupt is disabled, program execution continues with the instruction following the STOP instruction (see Fig.21).

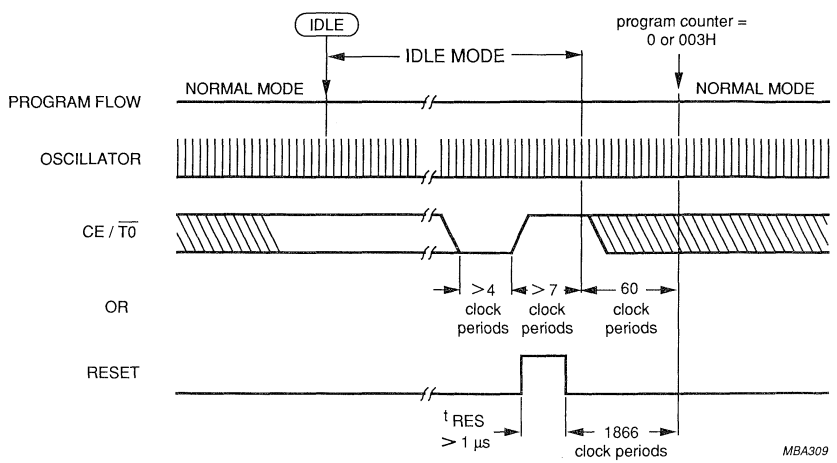


Fig.20 Entering and leaving the Idle mode.

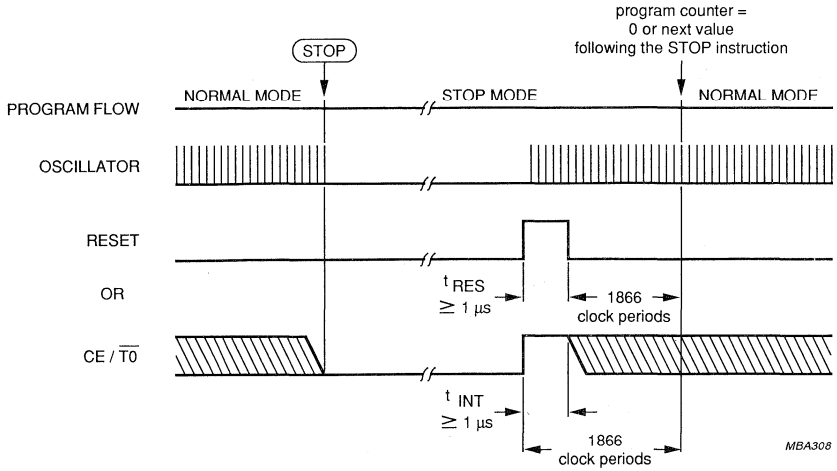


Fig.21 Entering and leaving the Stop mode.

4.17 Derivative logic

Several members of the PCD33XX family contain derivative logic. For specific information on a particular device, refer to the relevant data sheet.

The derivative registers are write only, read only or read/write (see Fig.22). They may be accessed internally via the derivative address register using the derivative input/output instructions (MOV A,Dx, MOV Dx,A, ANL Dx,A and ORL Dx,A).

Derivative interrupts share the PIN flag with the SIO interrupt (if available). When the derivative interrupt routine is executed, the PIN flag must be inactivated by software.

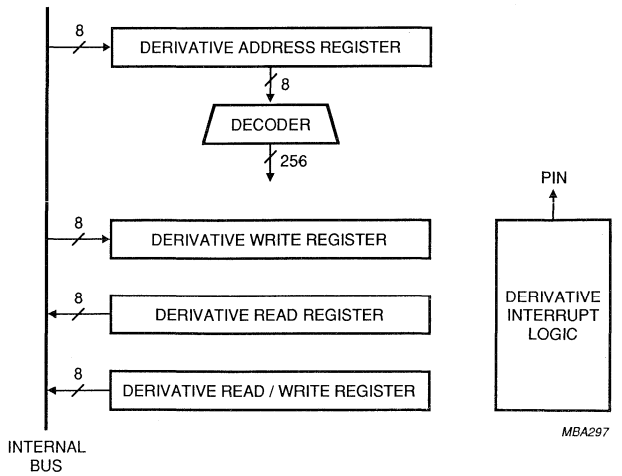


Fig.22 Block diagram of derivative logic.

5 INSTRUCTION SET

The PCD33XX family instruction set consists of over 80 one and two byte instructions. Program code efficiency is high because all RAM locations and all ROM locations on a 256 byte page require only a single byte address.

Table 8 contains the instruction set of the PCD33XX. Figure 23 shows the instruction map and Table 7 describes the symbols that are used.

Table 7 Symbols and definitions.

SYMBOL	DESCRIPTION
A	accumulator
addr	program memory address
Bb	bit designation (b = 0 to 7)
C	carry bit (bit CY)
CNT	event counter
Dx	mnemonic derivative register (x = 0 to 255)
direct	8-bit derivative register address
data	8-bit immediate data
I	interrupt
MBn	memory bank (n = 0 to 3)
MBFFn	memory bank flip-flop (n = 0 or 1)
P	mnemonic for 'in-page' operation
PC	program counter
Pp	port designation (p = 0, 1, or 2)
PS	Timer prescaler select
PSW	program status word
RB	register bank
RBS	register bank select flag
@Rr	8-bit address register (r = 0, 1)
Rr	8-bit register (r = 0 to 7)
Sn	serial I/O register (n = 0, 1, or 2)
SP	stack pointer
T	timer
TCNT	timer/event counter
TF	timer flag
T0, T1	test 0 and test 1 inputs
#	immediate data prefix
@	indirect address prefix
(X)	contents of X
((X))	contents of location addressed by X
←	is replaced by
↔	is exchanged with
< >	represents a hex digit

PCF84CXXX PCD33XX

first hexadecimal character of opcode — second hexadecimal character of opcode

	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
0	NOP	IDLE		ADD A, #data	JMP page 0	EN I	JNTF addr	DEC A	IN A,Pp 0	IN A,Pp 1	IN A,Pp 2		MOV A,Sn 0	MOV A,Sn 1		
1	INC @ Rr 0	INC @ Rr 1	JB0 addr	ADDC A,#data	CALL page 0	DIS I	JTF addr	INC A	INC Rr 0	INC Rr 1	INC Rr 2	INC Rr 3	INC Rr 4	INC Rr 5	INC Rr 6	INC Rr 7
2	XCH A, @Rr 0	XCH A, @Rr 1	STOP	MOV A, #data	JMP page 1	EN TCNTI	JNT0 addr	CLR A	XCH A,Rr 0	XCH A,Rr 1	XCH A,Rr 2	XCH A,Rr 3	XCH A,Rr 4	XCH A,Rr 5	XCH A,Rr 6	XCH A,Rr 7
3	XCHD A, @Rr 0	XCHD A, @Rr 1	JB1 addr		CALL page 1	DIS TCNTI	JT0 addr	CPL A	OUTL Pp,A 0	OUTL Pp,A 1	OUTL Pp,A 2		MOV Sn,A 0	MOV Sn,A 1	MOV Sn,A 2	
4	ORL A, @Rr 0	ORL A, @Rr 1	MOV A, T	ORL A, #data	JMP page 2	STRT CNT	JNT1 addr	SWAP A	ORL A,Rr 0	ORL A,Rr 1	ORL A,Rr 2	ORL A,Rr 3	ORL A,Rr 4	ORL A,Rr 5	ORL A,Rr 6	ORL A,Rr 7
5	ANL A, @Rr 0	ANL A, @Rr 1	JB2 addr	ANL A, #data	CALL page 2	STRT T	JT1 addr	DA A	ANL A,Rr 0	ANL A,Rr 1	ANL A,Rr 2	ANL A,Rr 3	ANL A,Rr 4	ANL A,Rr 5	ANL A,Rr 6	ANL A,Rr 7
6	ADD A, @Rr 0	ADD A, @Rr 1	MOV T, A		JMP page 3	STOP TCNT		RRC A	ADD A,Rr 0	ADD A,Rr 1	ADD A,Rr 2	ADD A,Rr 3	ADD A,Rr 4	ADD A,Rr 5	ADD A,Rr 6	ADD A,Rr 7
7	ADDC A, @Rr 0	ADDC A, @Rr 1	JB3 addr		CALL page 3			RR A	ADDC A,Rr 0	ADDC A,Rr 1	ADDC A,Rr 2	ADDC A,Rr 3	ADDC A,Rr 4	ADDC A,Rr 5	ADDC A,Rr 6	ADDC A,Rr 7
8				RET	JMP page 4	EN SI			ORL Pp,#data 0	ORL Pp,#data 1	ORL Pp,#data 2		MOV A,Dx	MOV Dx,A	ANL Dx,A	ORL Dx,A
9			JB4 addr	RETR	CALL page 4	DIS SI	JNZ addr	CLR C	ANL Pp,#data 0	ANL Pp,#data 1	ANL Pp,#data 2		MOV Sn,#data 0	MOV Sn,#data 1	MOV Sn,#data 2	
A	MOV @ Rr,A 0	MOV @ Rr,A 1		MOVP A,@A	JMP page 5	SEL MB2		CPL C	MOV Rr,A 0	MOV Rr,A 1	MOV Rr,A 2	MOV Rr,A 3	MOV Rr,A 4	MOV Rr,A 5	MOV Rr,A 6	MOV Rr,A 7
B	MOV @Rr, #data 0	MOV @Rr, #data 1	JB5 addr	JMPP @A	CALL page 5	SEL MB3			MOV Rr,#data 0	MOV Rr,#data 1	MOV Rr,#data 2	MOV Rr,#data 3	MOV Rr,#data 4	MOV Rr,#data 5	MOV Rr,#data 6	MOV Rr,#data 7
C	DEC @Rr 0	DEC @Rr 1			JMP page 6	SEL RB0	JZ addr	MOV A,PSW	DEC Rr 0	DEC Rr 1	DEC Rr 2	DEC Rr 3	DEC Rr 4	DEC Rr 5	DEC Rr 6	DEC Rr 7
D	XRL A, @Rr 0	XRL A, @Rr 1	JB6 addr	XRL A,#data	CALL page 6	SEL RB1		MOV PSW,A	XRL A,Rr 0	XRL A,Rr 1	XRL A,Rr 2	XRL A,Rr 3	XRL A,Rr 4	XRL A,Rr 5	XRL A,Rr 6	XRL A,Rr 7
E	DJNZ @ Rr,addr 0	DJNZ @ Rr,addr 1			JMP page 7	SEL MB0	JNC addr	RL A	DJNZ Rr,addr 0	DJNZ Rr,addr 1	DJNZ Rr,addr 2	DJNZ Rr,addr 3	DJNZ Rr,addr 4	DJNZ Rr,addr 5	DJNZ Rr,addr 6	DJNZ Rr,addr 7
F	MOV A, @Rr 0	MOV A, @Rr 1	JB7 addr		CALL page 7	SEL MB1	JC addr	RLC A	MOV A,Rr 0	MOV A,Rr 1	MOV A,Rr 2	MOV A,Rr 3	MOV A,Rr 4	MOV A,Rr 5	MOV A,Rr 6	MOV A,Rr 7

MBA281

Fig.23 PCD33XX instruction map.

Table 8 PCD33XX family instruction set.

MNEMONIC	OPCODE (HEX)	BYTES/ CYCLES	DESCRIPTION	FUNCTION		NOTES
ACCUMULATOR						
ADD A, Rr	6<8 + r>	1/1	Add register contents to A	(A) ← (A) + (Rr)	r = 0 to 7	1
ADD A, @Rr	6r	1/1	Add RAM data, addressed by Rr, to A	(A) ← (A) + ((Rr))	r = 0, 1	1
ADD A, #data	03 data	2/2	Add immediate data to A	(A) ← (A) + data		1
ADDC A, Rr	7<8 + r>	1/1	Add carry and register contents to A	(A) ← (A) + (Rr) + (C)	r = 0 to 7	1
ADDC A, @Rr	7r	1/1	Add carry and RAM data, addressed by Rr, to A	(A) ← (A) + ((Rr)) + (C)	r = 0, 1	1
ADDC A, #data	13 data	2/2	Add carry and immediate data to A	(A) ← (A) + data + (C)		1
ANL A, Rr	5<8 + r>	1/1	'AND' Rr with A	(A) ← (A) AND (Rr)	r = 0 to 7	
ANL A, @Rr	5r	1/1	'AND' RAM data, addressed by Rr, with A	(A) ← (A) AND ((Rr))	r = 0, 1	
ANL A, #data	53 data	2/2	'AND' immediate data with A	(A) ← (A) AND data		
ORL A, Rr	4<8 + r>	1/1	'OR' Rr with A	(A) ← (A) OR (Rr)	r = 0 to 7	
ORL A, @Rr	4r	1/1	'OR' RAM data, addressed by Rr, with A	(A) ← (A) OR ((Rr))	r = 0, 1	
ORL A, #data	43 data	2/2	'OR' immediate data with A	(A) ← (A) OR data		
XRL A, Rr	D<8 + r>	1/1	'XOR' Rr with A	(A) ← (A) XOR (Rr)	r = 0 to 7	
XRL A, @Rr	Dr	1/1	'XOR' RAM data, addressed by Rr, with A	(A) ← (A) XOR ((Rr))	r = 0, 1	
XRL A, #data	D3 data	2/2	'XOR' immediate data with A	(A) ← (A) XOR data		
INC A	17	1/1	Increment A by 1	(A) ← (A) + 1		
DEC A	07	1/1	Decrement A by 1	(A) ← (A) – 1		
CLR A	27	1/1	Clear A to zero	(A) ← 0		
CPL A	37	1/1	One's complement A	(A) ← NOT(A)		
RL A	E7	1/1	Rotate A left	$(A_{n+1}) \leftarrow (A_n)$ $(A_0) \leftarrow (A_7)$	n = 0 to 6	
RLC A	F7	1/1	Rotate A left through carry	$(A_{n+1}) \leftarrow A_n$ $(A_0) \leftarrow (C), (C) \leftarrow (A_7)$	n = 0 to 6	2
RR A	77	1/1	Rotate A right	$(A_n) \leftarrow (A_{n+1})$ $(A_7) \leftarrow (A_0)$	n = 0 to 6	
RRC A	67	1/1	Rotate A right through carry	$(A_n) \leftarrow (A_{n+1})$ $(A_7) \leftarrow (C), (C) \leftarrow (A_0)$	n = 0 to 6	2
DA A	57	1/1	Decimal adjust A	(A) ← (A) + 06H if AC = 1 or $(A_{0-3}) > 9$; (A) ← (A) + 60H if $(A_{4-7}) > 9$		2
SWAP A	47	1/1	Swap nibbles of A	$(A_{4-7}) \leftrightarrow (A_{0-3})$		2

MNEMONIC	OPCODE (HEX)	BYTES/ CYCLES	DESCRIPTION	FUNCTION		NOTES
DATA MOVES						
MOV A, Rr	F<8 + r>	1/1	Move register contents to A	(A) ← (Rr)	r = 0 to 7	
MOV A, @Rr	Fr	1/1	Move RAM data, addressed by Rr, to A	(A) ← ((Rr))	r = 0, 1	
MOV A, #data	23 data	2/2	Move immediate data to A	(A) ← data		
MOV Rr, A	A<8 + r>	1/1	Move accumulator contents to register	(Rr) ← (A)	r = 0 to 7	
MOV @Rr, A	Ar	1/1	Move accumulator contents to RAM location addressed by Rr	((Rr)) ← (A)	r = 0, 1	
MOV Rr, #data	B<8 + r> data	2/2	Move immediate data to Rr	(Rr) ← data	r = 0 to 7	
MOV @Rr, #data	Br data	2/2	Move immediate data to RAM location addressed by Rr	((R0)) ← data	r = 0, 1	
XCH A, Rr	2<8 + r>	1/1	Exchange accumulator contents with Rr	(A) ↔ (Rr)	r = 0 to 7	
XCH A, @Rr	2r	1/1	Exchange accumulator contents with RAM data addressed by Rr	(A) ↔ ((Rr))	r = 0, 1	
XCHD A, @Rr	3r	1/1	Exchange lower nibbles of A and RAM data addressed by Rr	$(A_{0-3}) \leftrightarrow ((Rr_{0-3}))$	r = 0, 1	
MOV A, PSW	C7	1/1	Move PSW contents to accumulator	(A) ← (PSW)		
MOV PSW, A	D7	1/1	Move accumulator bit 3 to PSW_3 (PS)	(PS) ← (A_3)		3
MOVP A, @A	A3	1/2	Move indirectly addressed data in current page to A	(PC_{0-7}) ← (A), (A) ← ((PC))		
CARRY FLAG						
CLR C	97	1/1	Clear carry bit	(C) ← 0		2
CPL C	A7	1/1	Complement carry bit	(C) ← NOT(C)		2
REGISTER						
INC Rr	1<8 + r>	1/1	Increment register by 1	(Rr) ← (Rr) + 1	r = 0 to 7	
INC @Rr	1r	1/1	Increment RAM data, addressed by Rr, by 1	((Rr)) ← ((Rr)) + 1	r = 0, 1	
DEC Rr	C<8 + r>	1/1	Decrement register by 1	(Rr) ← (Rr) – 1	r = 0 to 7	
DEC @Rr	Cr	1/1	Decrement RAM data, addressed by Rr, by 1	((Rr)) ← ((Rr)) – 1	r = 0, 1	

MNEMONIC	OPCODE (HEX)	BYTES/ CYCLES	DESCRIPTION	FUNCTION		NOTES
BRANCH						
JMP addr	<2n>4 addr	2/2	Unconditional jump within a 2 K bank	$(PC_{8-10}) \leftarrow n$ $(PC_{0-7}) \leftarrow$ addr $(PC_{11-12}) \leftarrow$ (MBFF 0-1)	n = 0 to 7	
JMPP @A	B3	1/2	Indirect jump within a page	$(PC_{0-7}) \leftarrow ((A))$		
DJZN Rr, addr	E<8 + r> addr	2/2	Decrement Rr by 1 and jump if not zero to addr	(Rr) ← (Rr) – 1; if (Rr) not zero, then $(PC_{0-7}) \leftarrow$ addr	r = 0 to 7	
DJNZ @Rr, addr	Er	2/2	Decrement RAM data, addressed by Rr, by 1 and jump if not zero to addr	((Rr)) ← ((Rr)) – 1; if ((Rr)) not zero, then $(PC_{0-7}) \leftarrow$ addr	r = 0, 1	
JBb addr	<2b + 1>2 addr	2/2	Jump to addr if Accumulator bit b = 1	If $(A_b) = 1$, then $(PC_{0-7}) \leftarrow$ addr	b = 0 to 7	
JC addr	F6 addr	2/2	Jump to addr if C = 1	If (C) = 1, then $(PC_{0-7}) \leftarrow$ addr		
JNC addr	E6 addr	2/2	Jump to addr if C = 0	If (C) = 0, then $(PC_{0-7}) \leftarrow$ addr		
JZ addr	C6 addr	2/2	Jump to addr if A = 0	If (A) = 0, then $(PC_{0-7}) \leftarrow$ addr		
JNZ addr	96 addr	2/2	Jump to addr if A is NOT zero	If $(A) \neq 0$, then $(PC_{0-7}) \leftarrow$ addr		
JT0 addr	36 addr	2/2	Jump to addr if T0 = 1	If T0 = 1, then $(PC_{0-7}) \leftarrow$ addr		
JNT0 addr	26 addr	2/2	Jump to addr if T0 = 0	If T0 = 0: $(PC_{0-7}) \leftarrow$ addr		
JT1 addr	56 addr	2/2	Jump to addr if T1 = 1	If T1 = 1: $(PC_{0-7}) \leftarrow$ addr		
JNT1 addr	46 addr	2/2	Jump to addr if T1 = 0	If T1 = 0: $(PC_{0-7}) \leftarrow$ addr		
JTF addr	16 addr	2/2	Jump to addr if Timer Flag = 1	If TF = 1: $(PC_{0-7}) \leftarrow$ addr		4
JNTF addr	06 addr	2/2	Jump to addr if Timer Flag = 0	If TF = 0: $(PC_{0-7}) \leftarrow$ addr		4
TIMER/EVENT COUNTER						
MOV A, T	42	1/1	Move timer/event counter contents to accumulator	(A) ← (T)		
MOV T, A	62	1/1	Move accumulator contents to timer/event counter	(T) ← (A)		
STRT CNT	45	1/1	Start event counter			
STRT T	55	1/1	Start timer			
STOP TCNT	65	1/1	Stop timer/event counter			
EN TCNTI	25	1/1	Enable timer/event counter interrupt			
DIS TCNTI	35	1/1	Disable timer/event counter interrupt			

MNEMONIC	OPCODE (HEX)	BYTES/ CYCLES	DESCRIPTION	FUNCTION		NOTES
CONTROL						
EN I	05	1/1	Enable external (chip enable) interrupt			
DIS I	15	1/1	Disable external (chip enable) interrupt			
SEL RB0	C5	1/1	Select register bank 0	(RBS) ← 0		5
SEL RB1	D5	1/1	Select register bank 1	(RBS) ← 1		5
SEL MB0	E5	1/1	Select program memory bank 0	(MBFF0) ← 0, (MBFF1) ← 0		10
SEL MB1	F5	1/1	Select program memory bank 1	(MBFF0) ← 1, (MBFF1) ← 0		10
SEL MB2	A5	1/1	Select program memory bank 2	(MBFF0) ← 0, (MBFF1) ← 1		10
SEL MB3	B5	1/1	Select program memory bank 3	(MBFF0) ← 1, (MBFF1) ← 1		10
STOP	22	1/1	Enter Stop mode			
IDLE	01	1/1	Enter Idle mode			
SUBROUTINE						
CALL addr	<2n + 1>4 addr	2/2	Jump to subroutine	((SP)) ← (PC), ($PSW_{4, 6, 7}$) (SP) ← (SP) + 1 (PC_{8-10}) ← n (PC_{0-7}) ← addr (PC_{11-12}) ← (MBFF0-1)	n = 0 to 7	6
RET	83	1/2	Return from subroutine	(SP) ← (SP) - 1 (PC) ← ((SP))		6
RETR	93	1/2	Return from interrupt and restore bits 4, 6, 7 of PSW	(SP) ← (SP) - 1 ($PSW_{4, 6, 7}$) + (PC) ← ((SP))		6
PARALLEL INPUT/OUTPUT						
IN A, Pp	08 09 0A	1/2	Input port p data to accumulator	(A) ← (P0) (A) ← (P1) (A) ← (P2)		7
OUTL Pp, A	38 39 3A	1/2	Output accumulator data to port p	(P0) ← (A) (P1) ← (A) (P2) ← (A)		
ANL Pp, #data	98 data 99 data 9A data	2/2	AND port p with immediate data	(P0) ← (P0) AND data (P1) ← (P1) AND data (P2) ← (P2) AND data		
ORL Pp, #data	88 data 89 data 8A data	2/2	OR port p data with immediate data	(P0) ← (P0) OR data (P1) ← (P1) OR data (P2) ← (P2) OR data		
DERIVATIVE INPUT/OUTPUT						
MOV A, Dx	8C direct	2/2	Move derivative register contents to accumulator	(A) ← (Dx)	x = 0 to 255	8
MOV Dx, A	8D direct	2/2	Move accumulator contents to derivative register	(Dx) ← (A)	x = 0 to 255	8
ANL Dx, A	8E direct	2/2	AND derivative register with accumulator	(Dx) ← (Dx) AND (A)	x = 0 to 255	8
ORL Dx, A	8F direct	2/2	OR derivative register with accumulator	(Dx) ← (Dx) OR (A)	x = 0 to 255	8

MNEMONIC	OPCODE (HEX)	BYTES/ CYCLES	DESCRIPTION	FUNCTION		NOTES
SERIAL INPUT/OUTPUT						
MOV A, S_n	0C 0D	1/2	Move serial I/O register contents to accumulator	(A) ← (S0) (A) ← (S1)	n = 0, 1	
MOV S_n, A	3C 3D 3E	1/2	Move accumulator contents to serial I/O register	(S0) ← (A) (S1) ← (A) (S2) ← (A)	n = 0, 1, 2	
MOV S_n, #data	9C data 9D data 9E data	2/2	Move immediate data to serial I/O register	(S0) ← data (S1) ← data (S2) ← data	n = 0, 1, 2	
EN SI	85	1/1	Enable serial I/O interrupt			
DIS SI	95	1/1	Disable serial I/O interrupt			
NOP	00	1/1	No operation	$(PC_{0-10}) \leftarrow (PC_{0-10}) + 1$		

Notes

1. PSW CY, AC affected.
2. PSW CY affected.
3. PSW PS affected.
4. Execution of a JTF or JNTF instruction resets the Timer Flag (TF).
5. PSW RBS affected.
6. PSW SP_0, SP_1, SP_2 affected.
7. (A) = 0000, P2.3, P2.2, P2.1, P2.0
8. For more information on the derivative input/output instructions of a particular microcontroller, consult the specific microcontroller data sheet.
9. (S1) has a different meaning for read and write operations. See section 4.11.4.
10. SEL MB instructions may not be used within interrupt routines.

6 LIMITING VALUES

Limiting values in accordance with the Absolute Maximum System (IEC 134).

SYMBOL	PARAMETER	MIN.	MAX.	UNIT
V_{DD}	Supply voltage range	−0.8	8.0	V
V_I	All input voltages	−0.5	V_{DD} + 0.5	V
I_I, I_O	DC current into any input or output	-	10	mA
P_{tot}	Total power dissipation	-	125	mW
T_{stg}	Storage temperature range	−65	150	°C
T_{amb}	Operating ambient temperature range	−20	70	°C
T_j	Operating junction temperature	-	90	°C

7 HANDLING

Inputs and outputs are protected against electrostatic discharge in normal handling. However, it is good practice to take normal precautions appropriate to handling MOS devices (see 'Handling MOS devices').

8 DC CHARATERISTICS

V_{DD} = 1.8 V to 6 V; V_{SS} = 0 V; T_{amb} = –20 to 70 °C; all voltages with respect to V_{SS} unless otherwise specified.

SYMBOL	PARAMETER	CONDITIONS	MIN.	TYP.	MAX.	UNIT
Supply						
V_{DD}	Operating supply voltage range	See Fig.24.	1.8	-	6.0	V
I_{DD}	Operating supply current	See Fig.26, Fig.27, and note 1; at V_{DD} = 5 V; f_{XTAL} = 10 MHz at V_{DD} = 5 V; f_{XTAL} = 6 MHz at V_{DD} = 3 V; f_{XTAL} = 3.58 MHz	 - - -	 1.6 1.0 0.3	 3.2 2.0 0.8	 mA mA mA
I_{DD}	Idle mode supply current	See Fig.28, Fig.29, and note 1; at V_{DD} = 5 V; f_{XTAL} = 10 MHz at V_{DD} = 5 V; f_{XTAL} = 6 MHz at V_{DD} = 3 V; f_{XTAL} = 3.58 MHz	 - - -	 0.8 0.5 0.15	 1.6 1.0 0.4	 mA mA mA
IDD	Stop mode supply current	See Fig.25, note 1, and note 2; at V_{DD} = 2.5 V; T_{amb} = 25 °C at V_{DD} = 2.5 V; T_{amb} = 85 °C	 - -	 1.2 -	 2.5 10.0	 µA µA
Inputs						
V_{IL}	Input voltage LOW		0	-	$0.3V_{DD}$	V
V_{IH}	Input voltage HIGH		$0.7V_{DD}$	-	V_{DD}	V
$\pm I_{IL}$	Input leakage current	$V_{SS} \leq V_I \leq V_{DD}$	-	-	1	µA
Outputs						
I_{OL}	Output sink current	All outputs except SCL, P2.3/SDA; see Fig.30. at V_{DD} = 5 V ± 10%; V_O = 0.4 V	 1.6	 3.0	 -	 mA
I_{OL}	Output sink current	SCL, P2.3/SDA; see Fig.31. at V_{DD} = 5 V ± 10%; V_O = 0.4 V	3.0	5.5	-	mA
$-I_{OH}$	Output source current	See Fig.32. at V_{DD} = 5 V ± 10%; V_O = $0.7V_{DD}$ at V_{DD} = 5 V ± 10%; V_O = V_{SS}	40 -	- -	- 400	µA µA
$-I_{OH}$	Output source current	See Fig.33. at V_{DD} = 5 V ± 10%; V_O = V_{DD} – 0.4 V	1.6	3.0	-	mA

9 AC CHARACTERISTICS

V_{DD} = 1.8 to 6 V; V_{SS} = 0 V; T_{amb} = –20 to 70 °C. All voltages with respect to V_{SS} unless otherwise specified.

SYMBOL	PARAMETER	CONDITIONS	MIN.	TYP.	MAX.	UNIT
t_R	Rise time all outputs	See note 3.	-	30	-	ns
t_F	Fall time all outputs	See note 3.	-	30	-	ns
f_{XTAL}	Clock frequency	See Fig.24.	0.45	-	10.0	MHz

10 SERIAL I/O INTERFACE CHARACTERISTICS

See Fig.34, Fig.35 and note 4.

SYMBOL	PARAMETER	CONDITIONS	SCL INPUT	SCL OUTPUT
$t_{HD;STA}$	START condition hold time		$\geq 14/f_{XTAL}$	$(DF+9)/(2f_{XTAL})$
t_{LOW}	SCL LOW time	Note 5.	$\geq 17/f_{XTAL}$	$(DF-3)/(2f_{XTAL})$
t_{HIGH}	SCL HIGH time	Note 5.	$\geq 17/f_{XTAL}$	$(DF+3)/(2f_{XTAL})$
t_{RC}	SCL rise time	Note 6.	≤ 1 µs	≤ 1 µs
t_{FC}	SCL fall time	Note 7.	≤ 0.3 µs	≤ 0.1 µs
SYMBOL	**PARAMETER**	**CONDITIONS**	**P2.3/SDA INPUT**	**P2.3/SDA OUTPUT**
t_{BUF}	Bus free time	Note 8.	$\geq 14/f_{XTAL}$	≥ 4.7 µs
$t_{SU;DAT}$	Data set-up time	Note 9.	≥ 250 ns	$\geq 15/f_{XTAL}$
$t_{HD,DAT}$	Data hold time		> 0	$\geq 9/f_{XTAL}$
t_{RD}	SDA rise time	Note 6.	≤ 1 µs	≤ 1 µs
t_{FD}	SDA fall time	Note 7.	≤ 0.3 µs	≤ 0.1 µs
$t_{SU;STO}$	Stop condition set-up time		$\geq 14/f_{XTAL}$	$(DF-3)/(2f_{XTAL})$

Notes

1. $V_{IL} = 0$; $V_{IH} = V_{DD}$; open drain outputs connected to V_{SS}; all other outputs open.
2. Crystal connected between XTAL1 and XTAL2; pin T1 at V_{SS}; pin CE/$\overline{T0}$ at V_{SS}.
3. $V_{DD} = 5$ V; $T_{amb} = 25$ °C; $C_L = 50$ pF.
4. DF is the f_{XTAL} divisor (see Table 2).
5. Values given for ASC = 0; for ASC = 1, $t_{LOW} = (DF-3)/4f_{XTAL}$, $t_{HIGH} = 3(DF+1)/4f_{XTAL}$.
6. Determined by the I^2C-bus capacitance (C_b) and the external pull-up resistor.
7. At maximum allowed I^2C-bus capacitance $C_b = 400$ pF.
8. Determined by program.
9. Independently of ASC, if $t_{LOW} \leq 24/f_{XTAL}$, $t_{SU;DAT} \geq t_{LOW} - 9/f_{XTAL}$.

11 CHARACTERISTIC CURVES

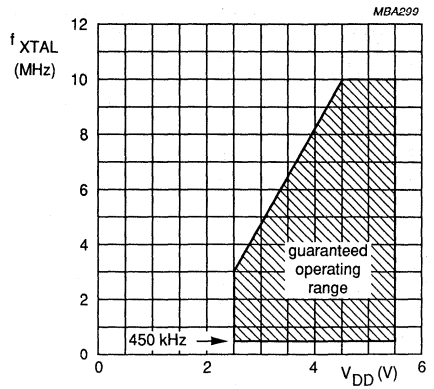
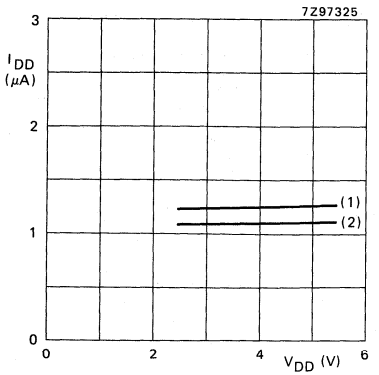
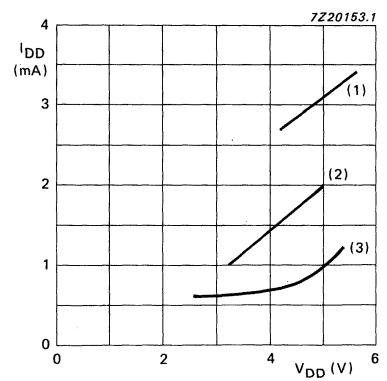


Fig.24 Maximum clock frequency (f_{XTAL}) as a function of supply voltage (V_{DD}).



(1) T_{amb} = 85° C
(2) T_{amb} = 25° C

Fig.25 Typical supply current (I_{DD}) in Stop mode as a function of supply voltage (V_{DD}).



(1) f_{XTAL} = 10.0 MHz
(2) f_{XTAL} = 6.0 MHz
(3) f_{XTAL} = 3.58 MHz

Fig.26 Maximum operating supply current (I_{DD}) as a function of supply voltage (V_{DD}).

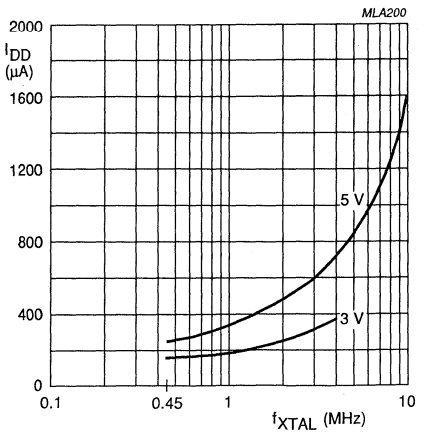
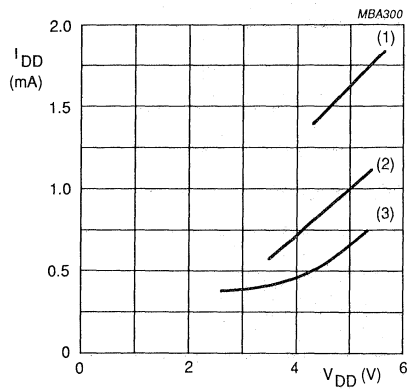


Fig.27 Typical operating supply current (I_{DD}) as a function of frequency at V_{DD} = 3 V and V_{DD} = 5 V.



(1) f_{XTAL} = 10.0 MHz
(2) f_{XTAL} = 6.0 MHz
(3) f_{XTAL} = 3.58 MHz

Fig.28 Maximum supply current (I_{DD}) in Idle mode as a function of supply voltage (V_{DD}).

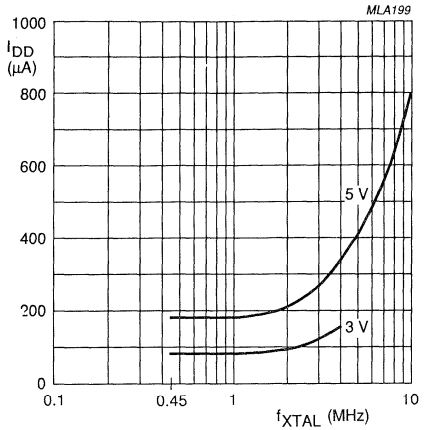


Fig.29 Typical supply current (I_{DD}) in Idle mode as a function of frequency at V_{DD} = 3 V and V_{DD} = 5 V.

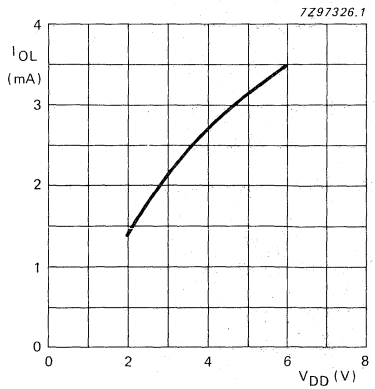


Fig.30 Typical output sink current (I_{OL}) as a function of supply voltage (V_{DD}); V_O = 0.4 V.

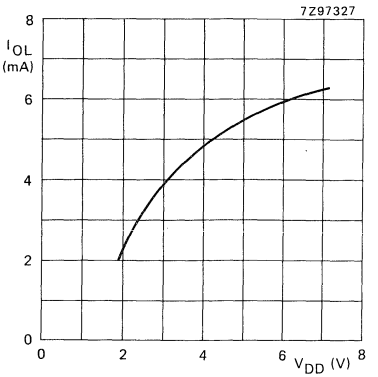
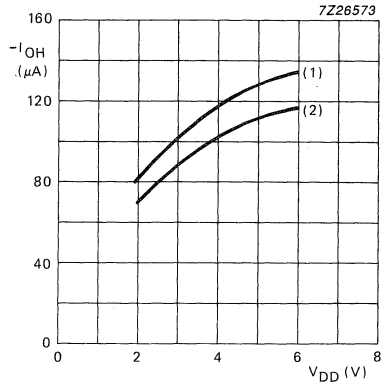


Fig.31 Typical output sink current (I_{OL}), outputs P2.3/SDA and SCL as a function of supply voltage (V_{DD}); V_O = 0.4 V.



(1) $V_O = V_{SS}$
(2) $V_O = 0.7\ V_{DD}$

Fig.32 Typical output source current ($-I_{OH}$) as a function of supply voltage (V_{DD}).

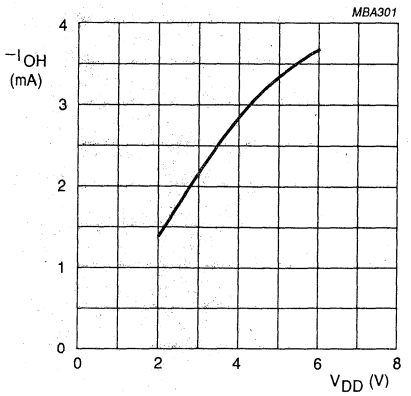


Fig.33 Typical push-pull output source current ($-I_{OH}$) as a function of supply voltage (V_{DD}); $V_O = V_{DD} - 0.4$ V.

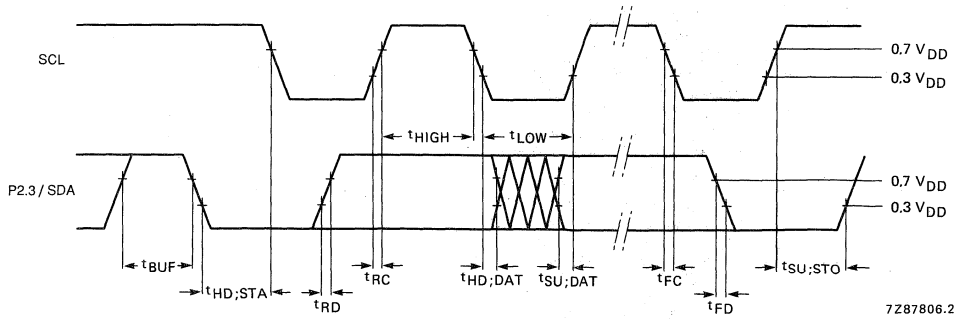


Fig.34 Slave SCL and receiver P2.3/SDA timing (SCL and P2.3/SDA input).

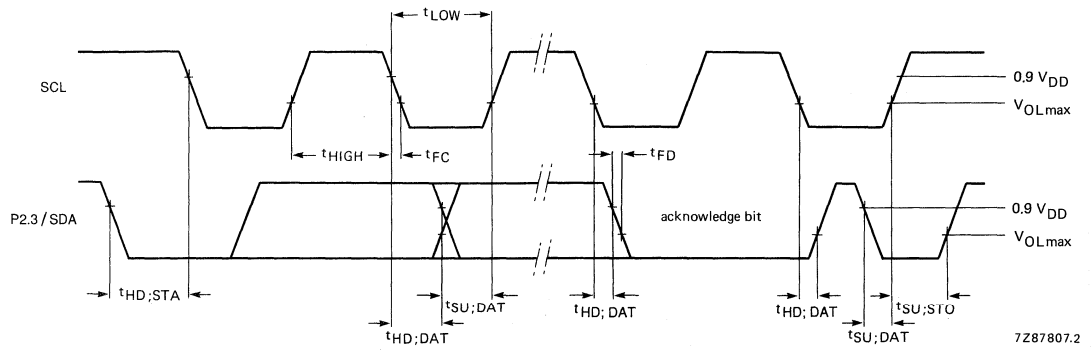


Fig.35 Master SCL and transmitter P2.3/SDA timing (SCL and P2.3/SDA output).

CMOS MICROCONTROLLER FOR TELEPHONE SETS

GENERAL DESCRIPTION

The PCD3315 is a single-chip 8-bit microcontroller fabricated in CMOS and is a member of the PCD33XX family. It has special on-chip features for application in telephone sets. For further detailed information, see PCD33XX family specification.

Features

- 8-bit CPU, ROM, RAM, I/O in a single 28-lead DIL or SO package
- 1536 ROM bytes
- 160 RAM bytes
- 20 quasi-bidirectional I/O port lines
- Two test inputs: one of which is also the external interrupt input (CE/$\overline{T0}$)
- Single-level vectored interrupts: external, timer/event counter
- 8-bit programmable timer/event counter
- Over 80 instructions (based on MAB8048)
- All instructions 1 or 2 cycles
- Clock frequency 100 kHz to 10 MHz
- Single supply voltage from 1,8 V to 6 V
- Low standby voltage and current
- STOP and IDLE mode
- On-chip oscillator with output drive capability for peripherals
- Configuration of all I/O port lines individually selected by mask: pull-up, open drain or push-pull
- Power-on-reset circuit and low supply voltage detection
- Reset state of all ports individually selected by mask
- Operating temperature range: −25 to + 70 °C

PACKAGE OUTLINES

PCD3315CP: 28-lead DIL; plastic (SOT117).
PCD3315CT: 28-lead mini-pack; plastic (SO28; SOT136A).

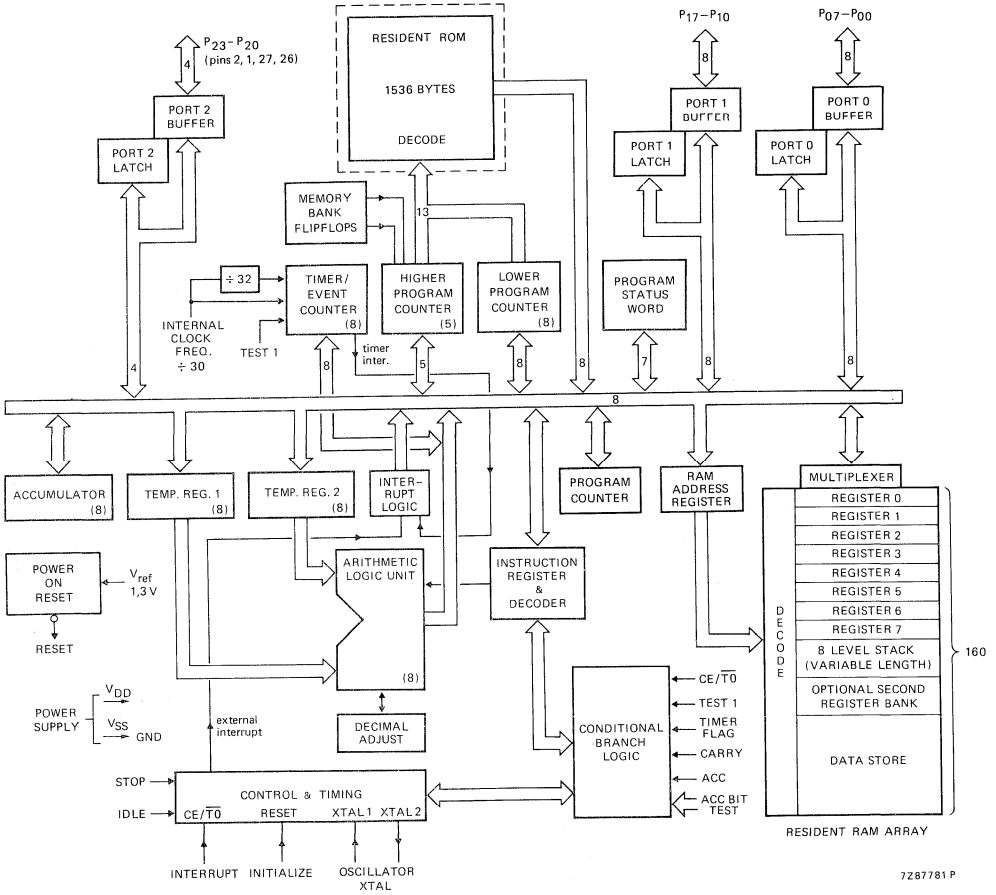
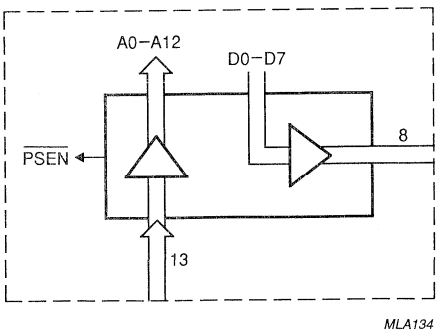


Fig. 1 Block diagram; PCD3315.



(a)

Fig. 1a Replacement of dotted part in Fig. 1, for the PCD3301B 'Piggy-back' version.

DEVELOPMENT DATA

PINNING

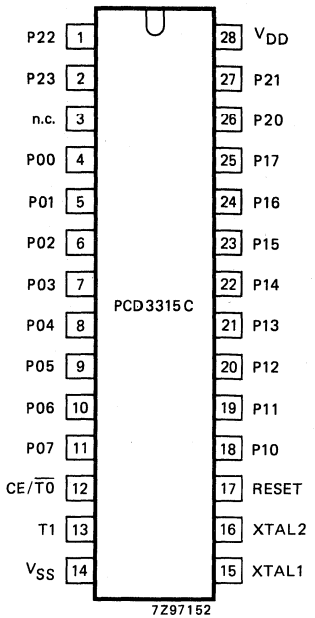


Fig. 2 Pinning diagram: PCD3315.

PIN DESIGNATION

3	n.c.	not connected
4-11	P00-P07	**Port 0**: 8-bit quasi-bidirectional I/O port.
12	CE/$\overline{T0}$	**Interrupt/Test 0**: external interrupt input (sensitive to positive-going edge edge)/test input pin; when used as a test input directly tested by conditional branch instructions JT0 and JNT0.
13	T1	**Test 1**: test input pin, directly tested by conditional branch instructions JT1 and JNT1. T1 also functions as an input to the 8-bit timer/event counter, using the STRT CNT instruction.
14	V_{SS}	**Ground**: circuit earth potential.
15	XTAL 1	**Crystal input**: connection to timing component (crystal) which determines the frequency of the internal oscillator; also the input for an external clock source.
16	XTAL 2	connection to the other side of the timing component.
17	RESET	**Reset input**: used to initialize the processor (active HIGH), or output of power-on-reset circuit.
18-25	P10-P17	**Port 1**: 8-bit quasi-bidirectional I/O port.
26, 27, 1, 2	P20-P23	**Port 2**: 4-bit quasi-bidirectional I/O port.
28	V_{DD}	**Power supply**: 1,8 V to 6 V.

D.C. CHARACTERISTICS

V_{DD} = 2,5 to 6 V; V_{SS} = 0 V; T_{amb} = −25 to + 70 °C; all voltages with respect to V_{SS}; f = 3,58 MHz with R_S = 50 Ω; unless otherwise specified.

parameter	symbol	min.	typ.	max.	unit
Supply voltage					
operating	V_{DD}	1,8	–	6	V
STOP mode for RAM retention	V_{DD}	1,0	–	6	V
Supply current					
operating at V_{DD} = 3 V	I_{DD}	–	350	–	μA
IDLE mode at V_{DD} = 3 V	I_{DD}	–	150	–	μA
STOP mode (note 1)					
at V_{DD} = 1,8 V; T_{amb} = 25 °C	I_{DD}	–	1,2	2,5	μA
at V_{DD} = 1,8 V; T_{amb} = 55 °C	I_{DD}	–	–	5	μA
at V_{DD} = 1,8 V; T_{amb} = 70 °C	I_{DD}	–	–	10	μA
RESET I/O					
Switching level	V_{RESET}	–	1,2	–	V
Sink current at $V_{DD} > V_{RESET}$	I_{OL}	–	7	–	μA
Inputs					
Input voltage LOW	V_{IL}	0	–	$0{,}3V_{DD}$	V
Input voltage HIGH	V_{IH}	$0{,}7V_{DD}$	–	V_{DD}	V
Input leakage current at $V_{SS} < V_I < V_{DD}$	$\pm I_{IL}$	–	–	1	μA
Outputs					
Output voltage LOW at V_I = V_{SS} or V_{DD}; $\lvert I_O \rvert < 1$ μA	V_{OL}	–	–	0,05	V
Output sink current LOW at V_{DD} = 3 V; V_O = 0,4 V	I_{OL}	0,6	1,5	–	mA
Pull-up output source current HIGH					
at V_{DD} = 3 V; $V_O = 0{,}9V_{DD}$	$-I_{OH}$	10	–	–	μA
at V_{DD} = 3 V; $V_O = V_{SS}$	$-I_{OH}$	–	–	200	μA
Push-pull output source current HIGH at V_{DD} = 3 V; $V_O = V_{DD}-0{,}4$ V	$-I_{OH}$	0,6	1,5	–	mA

Note 1

Crystal connected between XTAL 1 and XTAL 2; pin 2 pulled to V_{DD} via 5,6 kΩ resistor; CE and T1 at V_{SS}.

For information on the PCD3315/502/503 derivatives (CMOS redial and repertory dialler):

– please refer to PCD3315/502 data sheet.

– please refer to PCD 3315/503 data sheet.

CMOS MICROCONTROLLER FOR TELEPHONE SETS

GENERAL DESCRIPTION

The PCD3343 is a single-chip 8-bit microcontroller fabricated in CMOS and is a member of the PCD33XX family. It has special on-chip features for application in telephone sets.
The device is mask programmable, designed to provide telephone dialling facilities such as redial, repertory dial, emergency call, keyboard scan and control for liquid crystal display, pulse dial and/or DTMF dial via dedicated peripheral. For further detailed information, see PCD33XX family specification.

Features

- 8-bit CPU, ROM, RAM, I/O in a single 28-lead DIL or SO package
- 3 K ROM bytes
- 224 RAM bytes
- 20 quasi-bidirectional I/O port lines
- Two test inputs: one of which is also the external interrupt input (CE/$\overline{T0}$)
- Single-level vectored interrupts: external, timer/event counter, serial I/O
- Serial I/O which can be used in bus systems with more than one master (serial I/O data via an existing port line and clock via a dedicated line)
- 8-bit programmable timer/event counter
- Over 80 instructions (based on MAB8048)
- All instructions 1 or 2 cycles
- Clock frequency 100 kHz to 10 MHz
- Single supply voltage from 1,8 V to 6 V
- Low standby voltage and current
- STOP and IDLE mode
- On-chip oscillator with output drive capability for peripherals (e.g. PCD3312 DTMF generator)
- Configuration of all I/O port lines individually selected by mask: pull-up, open drain or push-pull
- Power-on-reset circuit and low supply voltage detection
- Reset state of all ports individually selected by mask
- Operating temperature range: −25 to + 70 °C

PACKAGE OUTLINES

PCD3343P: 28-lead DIL; plastic (SOT117).
PCD3343T: 28-lead mini-pack; plastic (SO28; SOT136A).

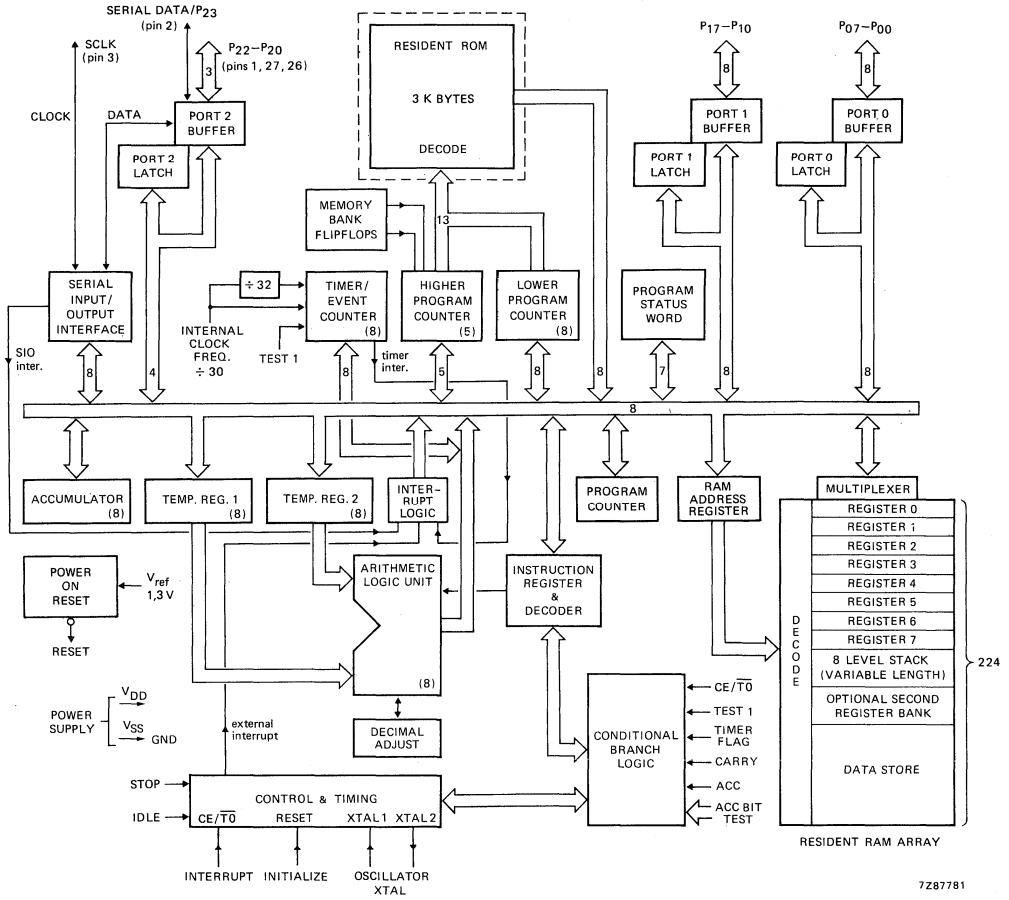
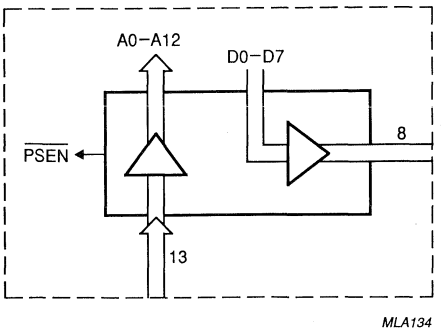


Fig. 1 Block diagram; PCD3343.

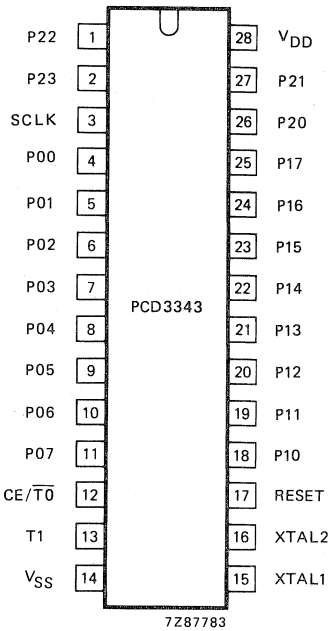


(a)

Fig. 1a Replacement of dotted part in Fig. 1, for the PCD3301B

DEVELOPMENT DATA

PINNING



Note CE/$\overline{T0}$ is labelled $\overline{INT}$/T0 on the PCD3301B and has inverted polarity.

Fig. 2 Pinning diagram: PCD3343 and bottom pinning PCD3301B.

PIN DESIGNATION

3	SCLK	**Clock**: bidirectional clock for serial I/O.
4-11	P00-P07	**Port 0**: 8-bit quasi-bidirectional I/O port.
12	CE/$\overline{T0}$	**Interrupt/Test 0**: external interrupt input (sensitive to positive-going edge)/test input pin; when used as a test input directly tested by conditional branch instructions JT0 and JNT0.
13	T1	**Test 1**: test input pin, directly tested by conditional branch instructions JT1 and JNT1. T1 also functions as an input to the 8-bit timer/event counter, using the STRT CNT instruction.
14	V_{SS}	**Ground**: circuit earth potential.
15	XTAL 1	**Crystal input**: connection to timing component (crystal) which determines the frequency of the internal oscillator; also the input for an external clock source.
16	XTAL 2	connection to the other side of the timing component.
17	RESET	**Reset input**: used to initialize the processor (active HIGH), or output of power-on-reset circuit.
18-25	P10-P17	**Port 1**: 8-bit quasi-bidirectional I/O port.
26, 27, 1, 2	P20-P23	**Port 2**: 4-bit quasi-bidirectional I/O port. P23 is the serial data input/output in serial I/O mode.
28	V_{DD}	**Power supply**: 1,8 V to 6 V.

PINNING (continued)

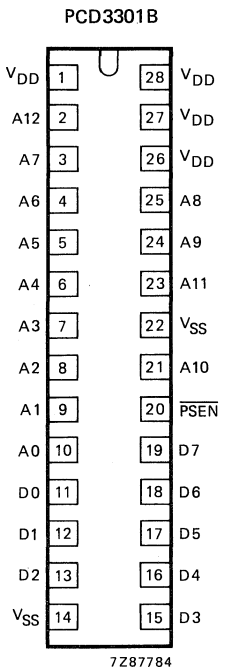


Fig. 3 Pinning diagram: PCD3301B 'Piggy-back' version top pinning; to access a 2732 or 2764 EPROM.

PIN DESIGNATION

14, 22	V_{SS}	**Ground**
1, 26-28	V_{DD}	**Power supply**
10-3, 25, 24, 21, 23, 2	A0-A12	**Address outputs**
11-13, 15-19	D0-D7	**Data**
20	$\overline{PSEN}$	**Program store enable**

Notes

1. RAM capacity of PCD3301B is 256 bytes.
2. Access time for ROMS/EPROMS to be below 7 x $1/f_{XTAL}$.
3. Pin 12 CE/$\overline{T0}$ is on the PCD3301B, inverted and labelled $\overline{INT}$/T0.

FUNCTIONAL DESCRIPTION

'Piggy-back' version PCD3301B

The PCD3301B is a special package that has standard pinning to the bottom which facilitates insertion as a mask-programmed device. An EPROM can be mounted on top in an additional socket. The total package height is greater than the standard DIL package. The RAM has 256 bytes and can also address 8 K bytes of program memory.

Program memory PCD3343

The program memory consists of 3072 bytes (8-bit words), in a read-only memory (ROM). Each location is directly addressable by the program counter. The memory is mask-programmed at the factory. Figure 4 shows the program memory map.

Four program memory locations are of special importance:

- Location 0; contains the first instruction to be executed after the processor is initialized (RESET),
- Location 3; contains the first byte of an external interrupt service subroutine,
- Location 5; contains the first byte of a serial I/O interrupt service subroutine,
- Location 7; contains the first byte of a timer/event counter interrupt service subroutine.

Program memory is arranged in banks of 2 K bytes, which are selected by SEL MB instructions. The program memory is further divided into location 'pages', each of 256 bytes. This latter division applies only for conditional branches. Memory bank boundaries can be crossed only by using the unconditional branch instructions after the appropriate memory bank has been selected. A CALL instruction can transfer control to a subroutine on any 'page'; RET and RETR instructions can transfer control from a subroutine back to the main program.

Data memory PCD3343

Data memory consists of 224 bytes (8-bit words), random-access data memory (RAM). All locations are indirectly addressable using RAM pointer registers; up to 16 designated locations are directly addressable. Memory also includes an 8-level program counter stack addressed by a 3-bit stack pointer. Figure 5 shows the data memory map.

Working registers

Locations 0 to 7 are designated as working registers, directly addressable by the direct register instructions. Ease of addressing, and a minimum requirement of instruction bytes to manipulate their contents, makes these locations suitable for storing frequently addressed intermediate results. This bank of registers can be selected by the SEL RB0 instruction.

Executing the select register bank instruction SEL RB1, designates locations 24 to 31 as working registers, instead of locations 0 to 7, and these are then directly addressable. This second bank of working registers may be used as an extension of the first or reserved for use during interrupt service subroutines saving the first bank for use in the main program. If the second bank is not used, locations 24 to 31 may serve as general purpose RAM.

The first locations of each bank contain the RAM pointer registers R0, R1, R0' and R1', which indirectly address all RAM locations.

All RAM locations make efficient program loop counters when used with the decrement register and test instruction DJNZ.

FUNCTIONAL DESCRIPTION (continued)

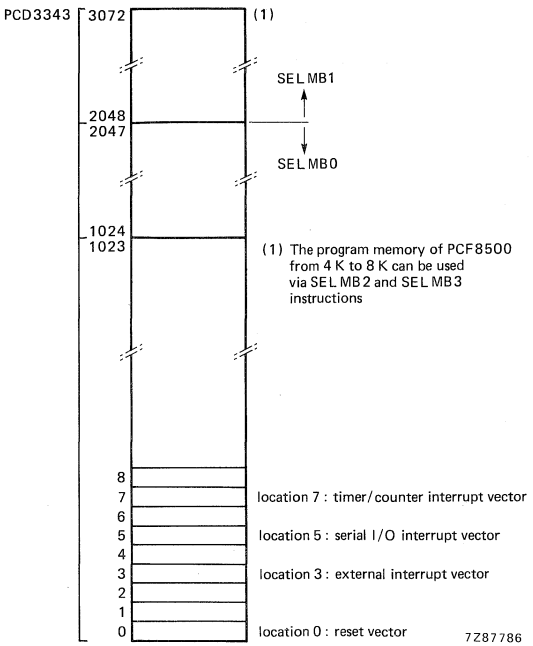


Fig. 4 Program memory map.

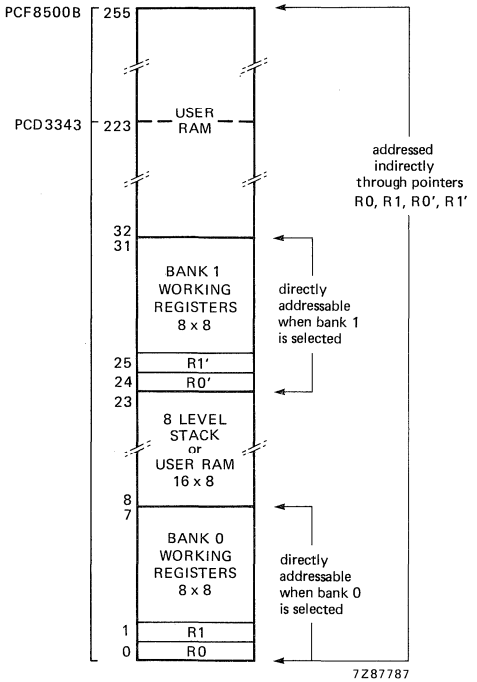


Fig. 5 Data memory map.

Program counter stack

Locations 8 to 23 may be designated as an 8-level program counter stack (2 locations per level), or as general purpose RAM. The program counter stack (Fig. 6) enables the processor to keep track of the return addresses and status generated by interrupts or CALL instructions by storing the contents of the program counter prior to servicing the subroutine. A 3-bit stack pointer determines which of the eight register pairs of the program counter stack will be loaded with next generated return address.

The stack pointer, when initialized to 000 by RESET, points to RAM locations 8 and 9. On the first subroutine CALL or interrupt, the contents of the program counter and bits 4, 6 and 7 of the program status word (PSW) are transferred to locations 8 and 9. The stack pointer increments by one and points to locations 10 and 11 ready for another CALL. Because an address may be up to 13 bits long, two bytes must be used to store each address.

At the end of a subroutine, which is signalled by a return instruction (RET or RETR), the stack pointer decrements by one and the contents of the register pair on top of the stack are transferred to the program counter. The saved PSW bits are transferred to the PSW only by the RETR instruction.

If not all 8 levels of subroutine and interrupt nesting are used, the unused portion of the stack may be used as any other indirectly addressable RAM locations. Possible locations from 32 to 223 may be used for storage of program variables or data.

Nesting of subroutines within subroutines can continue up to 8 times without overflowing the stack. If overflow does occur the deepest address stored (locations 8 and 9) will be overwritten and lost since the stack pointer overflows from 111 to 000. It also underflows from 000 to 111.

The value of the saved contents of the program counter is different for an interrupt CALL compared to a normal CALL to subroutine. With an interrupt CALL, the program counter return address is saved; with a subroutine CALL, the saved program counter value is one less than the program counter return address.

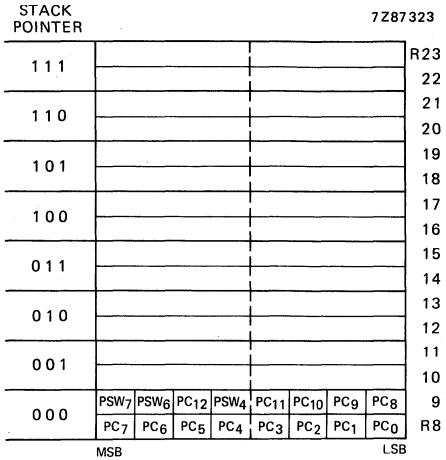


Fig. 6 Program counter stack.

IDLE and STOP modes

IDLE mode

When the microcontroller enters the IDLE mode via the IDLE instruction (H'01') the oscillator, timer/counter and serial I/O are kept running. The microcontroller exits from the IDLE mode by one of three interrupts if they are enabled or by activating a RESET. If the interrupt is not enabled the processor will remain in the IDLE mode. An active signal on the RESET pin restarts the microcontroller and a normal RESET sequence is executed (see Fig. 7).

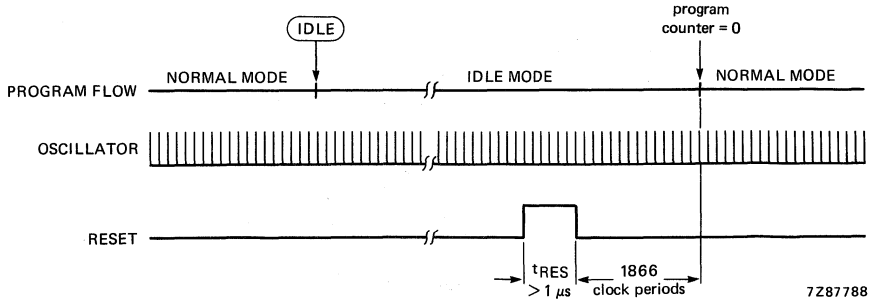


Fig. 7 Exit from IDLE mode via a RESET.

FUNCTIONAL DESCRIPTION (continued)

An active signal coming from an enabled interrupt causes the execution of the normal interrupt routine since normal interrupt scanning is still being carried out. A LOW-to-HIGH transition on the external interrupt pin (CE/$\overline{T0}$) reactivates the microcontroller. A HIGH level applied to CE/$\overline{T0}$ will reactivate the microcontroller only in the STOP mode. Thus, if CE/$\overline{T0}$ was HIGH before the microcontroller entered the IDLE mode, it must go LOW before the microcontroller can be reactivated (see Fig. 8).

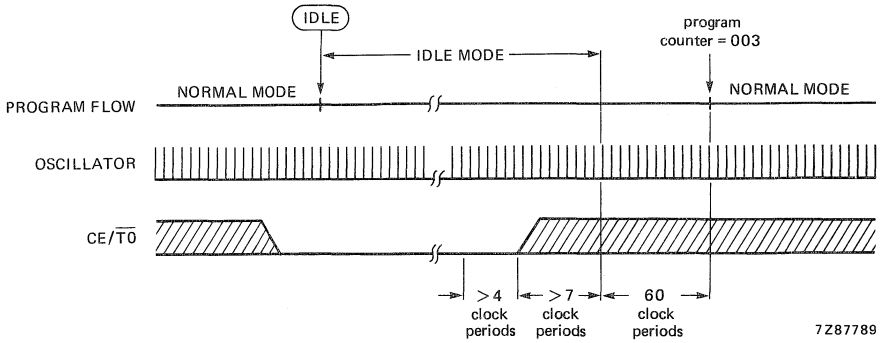


Fig. 8 Exit from IDLE mode via an interrupt.

Wake-up from the IDLE mode is ensured when CE/$\overline{T0}$ is LOW for 4 CP (clock periods) followed by a HIGH for 7 CP. After the initial forced CALL H'003' operation (60 CP) the program continues with the external interrupt service routine.

STOP mode

The microcontroller enters the STOP mode by the STOP instruction (H'22'). The oscillator is switched off. The internal status of the CPU, RAM contents and the state of I/O ports are not affected. The microcontroller can be brought-out of the STOP mode by an active signal at the external interrupt input or by an external RESET signal. When one of these two signals is applied an internal delay of 1866 CP is provided to ensure that all internal clocks are operating correctly before restart (see Fig. 9).

If the microcontroller exits from the STOP mode by activating RESET, a normal RESET sequence is executed.

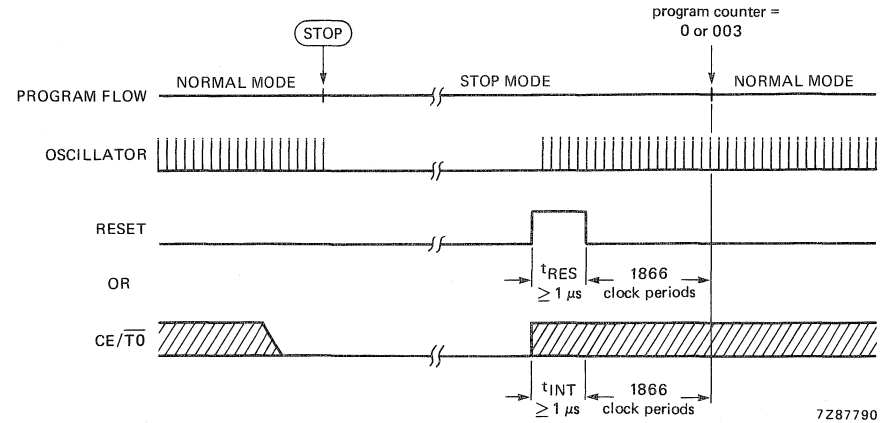


Fig. 9 Entering and exiting the STOP mode.

If the microcontroller exits the STOP mode by pulling the external interrupt input pin HIGH, an interrupt sequence is executed only if the external interrupt is enabled. In this event the microcontroller resumes the normal program sequence after returning from the interrupt routine, as in the normal mode. If the interrupt is not enabled, it continues the normal program sequence, executing the instruction following the STOP instruction.

The microcontroller is restarted by a HIGH level applied at the CE/$\overline{T0}$ pin, and not by a LOW-to-HIGH transition as in a normal interrupt mechanism.

When the CE/$\overline{T0}$ level is active during the STOP instruction then no STOP is executed.

A HIGH level on the external interrupt input of at least 1 μs will cause the microcontroller to exit the STOP mode.

I/O facilities

The PCD3343 family has 23 I/O lines arranged as:

- Port 0 parallel port of 8 lines (P00 to P07)
- Port 1 parallel port of 8 lines (P10 to P17)
- Port 2 parallel port of 4 lines (P20 to P23)
- SCLK serial I/O consisting of a data line shared with a parallel port line (P23) and a separate clock line SCLK
- CE/$\overline{T0}$ external interrupt and test input. When used as a test input can be directly tested by conditional branch instructions JT0 and JNT0
- T1 test input which can alter program sequences when tested by conditional jump instructions JT1 and JNT1. T1 also functions as an input to the 8-bit timer/event counter.

Parallel ports

All parallel ports can be used as outputs or inputs, their structure is quasi-bidirectional.
Output data written to a port is latched and remains unchanged until rewritten.
Input data is not latched and so must be present until read by an input instruction.

Input lines are fully CMOS compatible, output lines can drive one LS-TTL or CMOS load.

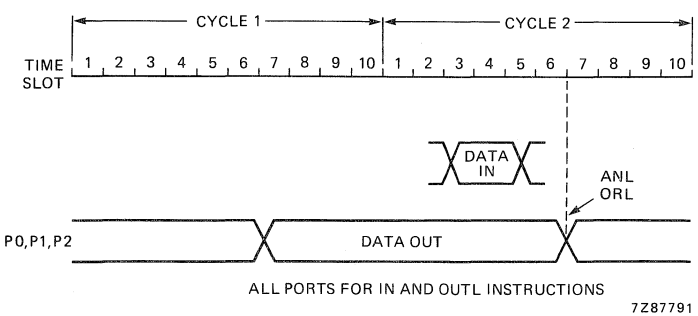


Fig. 10 Timing diagram of all ports on IN and OUTL instructions; for ANL and ORL instructions, the ports change on the time slot 7 of cycle 2.

Fig. 11 shows the quasi-bidirectional I/O interface with push-pull output and switched pull-up current source.
Each line is pulled up to V_{DD} via a constant current source (TR4), which is enabled via TR3 whenever one of the two output latches contains a logic 1. This current provides sufficient source current for a TTL HIGH level, yet can be pulled LOW by an external CMOS device, thus allowing the same pin to be used for both input and output.

FUNCTIONAL DESCRIPTION (continued)

When a logic 1 is written to the line for the first time (MQ = 1, SQ = 0), TR2 is switched on for the duration of the internal write pulse (one oscillator period), to provide a fast transition from logic 0 to logic 1. Subsequent writing of a logic 1 to the port lines will not switch TR2 on. This prevents unnecessary current through external components connected to the port lines of the same port which might be in the input mode and also connected to ground.

When a logic 0 is written to the line, TR3 switches off the current source. Current sinking capability is provided by TR1, which is now switched on. When used as an input, a logic 1 must first be written to the line, otherwise TR1 will remain low impedance.

In telephone applications this switched pull-up source may not be sufficient. Therefore the PCD3343 offers the possibility to select individually 19 of the 20 parallel port pins (not P23), by the following mask options:

Option 1- STANDARD PORT; quasi-bidirectional I/O with switched pull-up current source of 100 μA (typ.) and P-channel booster transistor TR2 (1,5 mA). TR2 is only active during 1 clock cycle (0,28 μs at 3,58 MHz).

Option 2- OPEN DRAIN; quasi-bidirectional I/O with only an N-channel open drain output. Application as an output requires connection of an external pull-up resistor (Fig. 12). When an open drain port is unused it must be connected to V_{SS}

Option 3- PUSH-PULL OUTPUT; drive capability of the output will be 1,5 mA (typ.) at V_{DD} = 3 V in both polarities. To avoid a large current flowing through the output transistors during the input mode, these push-pull pins must only be used as outputs (Fig. 13).

Also, individual mask selection of the RESET state of these port pins can be achieved by appending the following options S and R to options 1, 2 or 3.

Option S-SET; after RESET this pin will be initialized to HIGH.

Option R-RESET; after RESET this pin will be initialized to LOW.

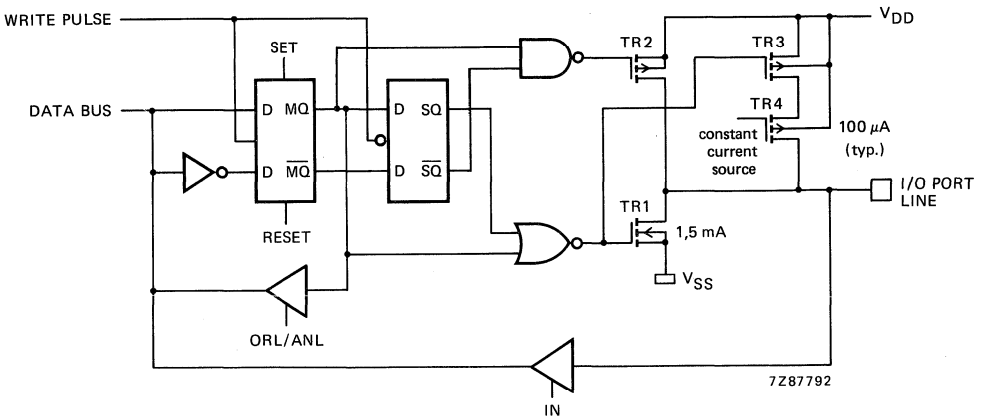


Fig. 11 Standard output with switched pull-up current source.

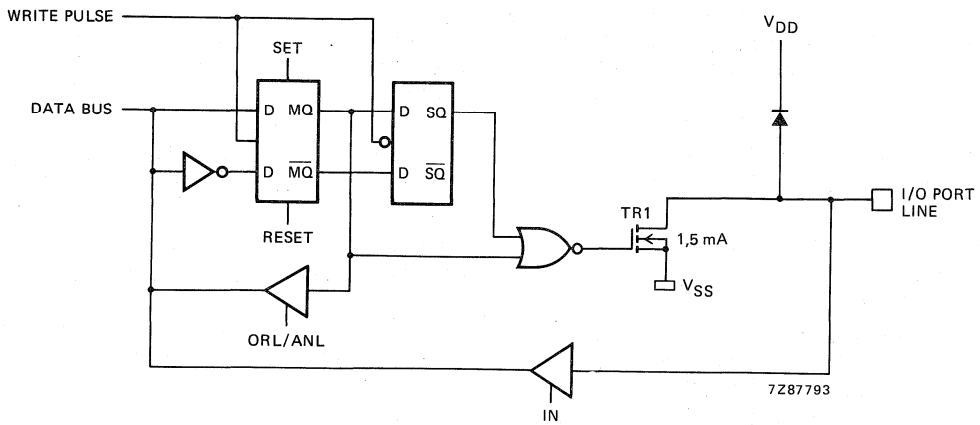


Fig. 12 Open drain output.

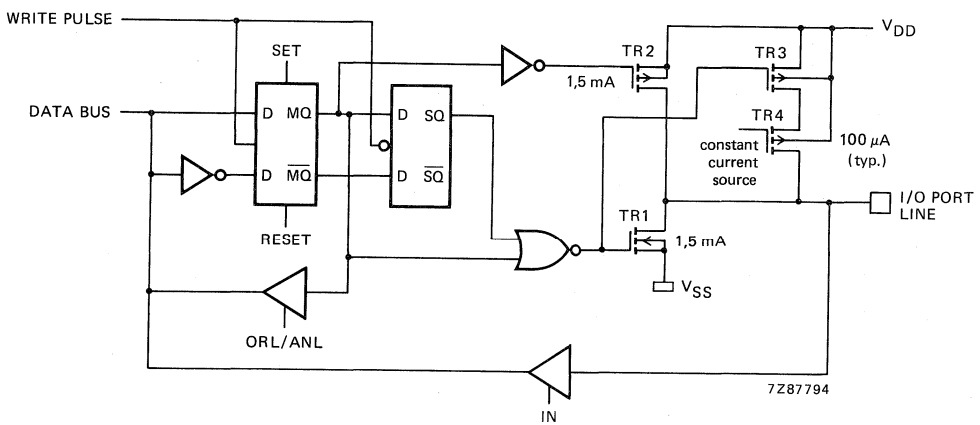


Fig. 13 Push-pull output.

FUNCTIONAL DESCRIPTION (continued)

Serial I/O (SIO)

The PCD3343 has an on-chip serial I/O interface. This SIO interface is a versatile feature in an intelligent telephone set, as shown in application diagram Fig. 32.

In this application the SIO is used to communicate with the different peripherals, such as:

- DTMF generator (PCD3312)
- LCD drivers (PCF8577)
- External RAM (PCD8571)
- Clock calendar (PCB8573)

No extra hardware is required for decoding, addressing and data processing.

Whereas a normal microcontroller must regularly monitor the serial data bus for the presence of data, the serial I/O interface detects, receives and converts the serial data stream into parallel format without interrupting the execution of the current program. An interrupt is sent to the PCD3343 only when a complete byte is received. It then reads the data byte in one instruction. Likewise during transmission the serial I/O interface performs parallel to serial conversion and subsequent serial output of the data. The microcontroller is only interrupted in the execution of its programmed tasks when a complete byte has been transmitted.

The design of the PCD3343 serial I/O system allows any number of devices from PCF8500 family (clips) to be connected via the two-line serial bus. The ability of any devices to communicate, without interrupting the operation of any other devices on the bus, is an outstanding attribute of the system. This is achieved by allocating a specific 7-bit address to each device and providing a system whereby a device reacts only to a message prefixed with its own address or the 'general CALL' address. Address recognition is performed by the interface hardware so that operation of the microcontroller need only be interrupted when a valid address has been received. This saves significant processing time and memory space compared with a conventional microcontroller employing a software serial interface. When the addressing facility is not required, for instance in a system with only two microcontrollers, direct data transfer without addressing can be performed. In multi-master systems, an automatically invoked arbitration procedure prevents two or more devices from continuing simultaneous transmission.

In NORMAL (running) and IDLE mode, the serial I/O logic remains active; its internal system clock will be switched off when there is no activity on the serial bus.

After execution of the STOP instruction, the oscillator of the PCD3343 is switched off. This means that the serial I/O logic will remain in the state it was at the occurance of the STOP instruction. To avoid "bus block" problems and to assure correct start-up of the bus after exit from the STOP mode, the user should disable the serial logic (ESO = 0) prior to the execution of the STOP instruction. This must be carried out only when the PCD3343 has finished a serial data transfer.
After a negative RESET signal the first 30 clock pulses of the 1866 pulse initialization phase set P23/SDA and SCLK HIGH. When P23/SDA or SCLK are not used, they must be connected to V_{SS}.

Serial I/O interface

Figure 14 shows the serial I/O interface. The clock line of the serial bus has exclusive use of pin 3 (SCLK) while the data line shares pin 2 (serial data) with the I/O line P23 of port 2. When the serial I/O is enabled, P23 is disabled as a parallel port line; (P23 and SCLK only open drain).

The microcontroller and interface communicate via the internal microcontroller bus and the Serial Interrupt Request line. Data and information controlling the operation of the interface are stored in four registers:

- Data shift register (S0)
- Serial I/O interface status word (S1)
- Serial clock control word (S2)
- Address register

Data shift register (S0)

Register S0 converts serial data to parallel format and vice versa. A pending interrupt is generated only after a complete byte has been transmitted, or after a complete data byte, specific address or 'general CALL' address has been received. The most significant bit is transmitted first.

Serial I/O interface status word (S1)

Register S1 provides information concerning the state of the interface and stores information from the microcontroller. Bits 0 to 3 are duplicated: control bits in these positions can only be written by the microcontroller, while interface bits can only be read.

MST and TRX (see Table 1)

These bits determine the operating mode of the serial I/O interface.

Table 1 Operating modes of the serial I/O interface

MST	TRX	operating mode
0	0	slave receiver
1	0	master receiver
0	1	slave transmitter
1	1	master transmitter

BB: Bus Busy.

This is the flag which indicates the status of the bus.

PIN: Pending Interrupt Not

PIN = '0' indicates the presence of a pending interrupt, which will cause a Serial Interrupt Request when the serial interrupt mechanism is enabled.

ESO: Enable Serial output

The ESO flag enables/disables the serial I/O interface: ESO = '1' enables, ESO = '0' disables. ESO can only be written by software.

BC0, BC1 and BC2

Bits BC0, BC1 and BC2 indicate the number of bits received or transmitted in a data stream. These bits can only be written by software.

AL: Arbitration Lost

The arbitration lost flag is set by hardware when the serial I/O interface, as master transmitter, loses a bus arbitration procedure.

AAS: Addressed As Slave

This flag is set by hardware when the interface detects either its own specific address or the 'general CALL' address as the first byte of a transfer and the interface has been programmed to operate in the address recognition mode.

AD0: Address Zero

This flag is set by hardware after detection of the 'general CALL' address when the interface is operating in the address recognition mode.

LRB: Last Received Bit

This contains either the last data bit received or, for a transmitting device in the acknowledgement mode, the acknowledgement signal from the receiving device.

Bits AL, AAS, AD0 and LRB can only be read by software.

FUNCTIONAL DESCRIPTION (continued)

Serial clock control word (S2)

Bits 0 to 4 of the clock control register S2 are used to set the frequency of the serial clock signal. When a 3,58 MHz crystal is used, the frequency of the serial clock can be varied between 92 kHz and 580 Hz (see Table 2). An asymmetrical clock with a HIGH-to-LOW ratio of 3 : 1 can be generated using bit 5. The asymmetrical clock allows a microcontroller more time per clock period for sampling the data line, making the timing of this action less critical. Bit 6 can be used to activate the acknowledge mode of the serial I/O. S2 is a write only register.

Address register

The address register contains the 7-bit address back-up latches and the bit (ALS) used to enable/disable the address recognition mode. The address register can be written using the MOV S0, A and MOV S0, # data instructions, but only when ES0 = '0'.

Serial I/O interrupt logic

An EN SI instruction enables and a DIS SI instruction disables the interrupt logic. When the logic is enabled, a pending interrupt results in a serial I/O interrupt to the processor, causing a CALL to location 5 in the ROM. When disabled, the presence of an interrupt is still indicated by PIN in S1, allowing the interrupt to be serviced. However, vectored interrupt will not occur.

Table 2 SIO clock pulse frequency control when using a 3,58 MHz crystal

hexadecimal S20-S24 code	divisor	f_{SCLK} (kHz) (approximate)
0	not allowed	
1	39	92
2	45	80
3	51	70
4	63	57
5	75	48
6	87	41
7	99	36
8	123	29
9	147	24
A	171	21
B	195	18
C	243	15
D	291	12
E	339	11
F	387	9,2
10	483	7,4
11	579	6,2
12	675	5,3
13	771	4,6
14	963	3,7
15	1155	3,1
16	1347	2,7
17	1539	2,3
18	1923	1,9
19	2307	1,6
1A	2691	1,3
1B	3075	1,2
1C	3843	0,93
1D	4611	0,78
1E	5379	0,67
1F	6147	0,58

DEVELOPMENT DATA

FUNCTIONAL DESCRIPTION (continued)

Table 3 Serial I/O addresses for telephony peripherals

type	address 7	6	5	4	3	2	1	0	description
PCF8570A	1	0	1	0	A2	A1	X	R/$\overline{W}$	2 K RAM
PCF8570	1	0	1	0	A2	A1	A0	R/$\overline{W}$	2 K RAM
PCD8571	1	0	1	0	A2	A1	A0	R/$\overline{W}$	1 K RAM
PCD3311	0	1	0	0	1	0	A0	R/$\overline{W}$	DTMF dialler
PCD3312	0	1	0	0	1	0	A0	R/$\overline{W}$	DTMF dialler
PCE2111	0	0	0	0	0	0	1	0	LCD driver *
PCD8573	1	1	0	1	0	A1	A0	R/$\overline{W}$	clock calendar
PCF8574	0	0	1	1	A2	A1	A0	R/$\overline{W}$	8-bit I/O expander
PCF8576	0	1	1	1	0	0	SA0	R/$\overline{W}$	1 : 4 LCD driver
PCF8577	0	1	1	1	0	1	0	R/$\overline{W}$	1 : 2 LCD driver

* LCD driver requires an additional enable line.

DEVELOPMENT DATA

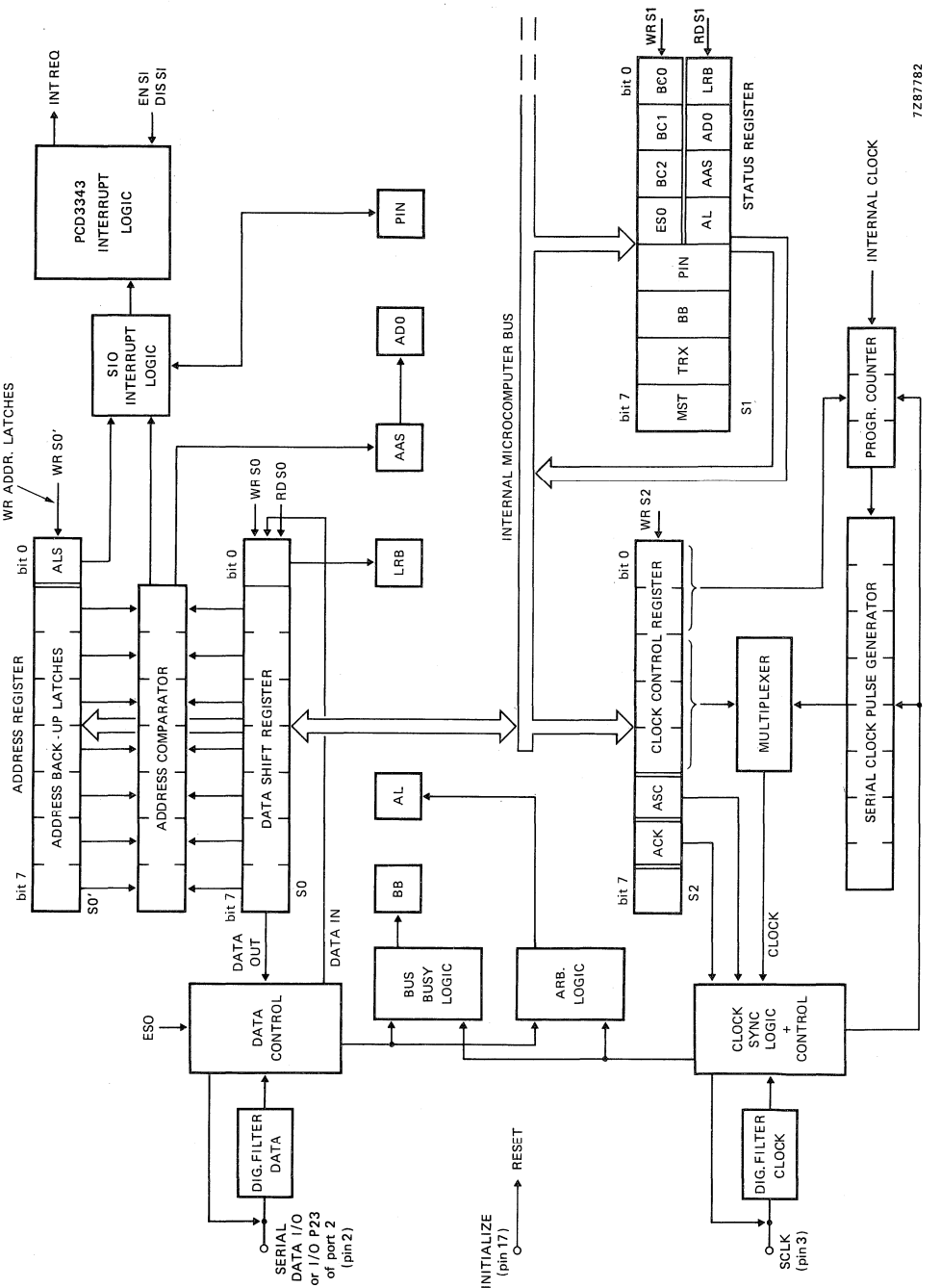


Fig. 14 Serial I/O interface.

FUNCTIONAL DESCRIPTION (continued)

Interrupts (see Fig. 15)

When the external interrupt is enabled, a LOW-to-HIGH transition on the CE/$\overline{T0}$ input initiates an external interrupt subroutine which causes a CALL to program memory location 3 following completion of the current instruction.
The interrupt must remain enabled until the interrupt instruction is completed, otherwise the next instruction of the main program will be executed. Serial I/O interrupt, when enabled, causes a CALL to location 5, and a timer/event counter overflow forces a CALL to location 7 when the timer interrupt is enabled.

When an interrupt subroutine starts, the contents of the program counter and bits 4, 6 and 7 of the PSW have been saved in the program counter stack. Accumulator contents have to be saved by software. Interrupt acknowledgement can be carried out by software via port pins. All interrupt subroutines must reside in memory bank 0.

Since the interrupt system is single level, once an interrupt is detected, all further interrupt requests are latched, but ignored, pending a RETR instruction to re-enable the interrupt input logic. After executing RETR, the program continues in the main part; this is independent of the occurance of a second interrupt during the running of the first routine. If 2 or 3 interrupts occur simultaneously, their priority is:

- (1) external
- (2) serial I/O
- (3) timer/event counter

An external interrupt can be generated by using the timer/counter in the event counter mode. The counter is first preset to (H'FF'), then EN TCNTI instruction is executed. A LOW-to-HIGH transition of the T1 input will then initiate an interrupt subroutine and cause a CALL to location 7.

On execution of a DIS I instruction, the PCD3343 always clears the digital filter/latch and the External Interrupt Flag.

The Timer Flag (TF) is reset only when the JTF or JNTF instruction is executed or after RESET.

The Timer Interrupt Flag is set when timer overflow occurs, only if the timer interrupt is enabled.

The microcontroller will exit the IDLE mode when any one of the following three interrupts is enabled:

- External
- Serial I/O
- Timer/event counter

There is no internal pull-up or pull-down device connected to the external interrupt input (pin 12). If required pin 12 must be externally connected to a resistor ($R \leqslant 100\ k\Omega$). When the external interrupt is not used pin 12 must be connected to V_{SS}.

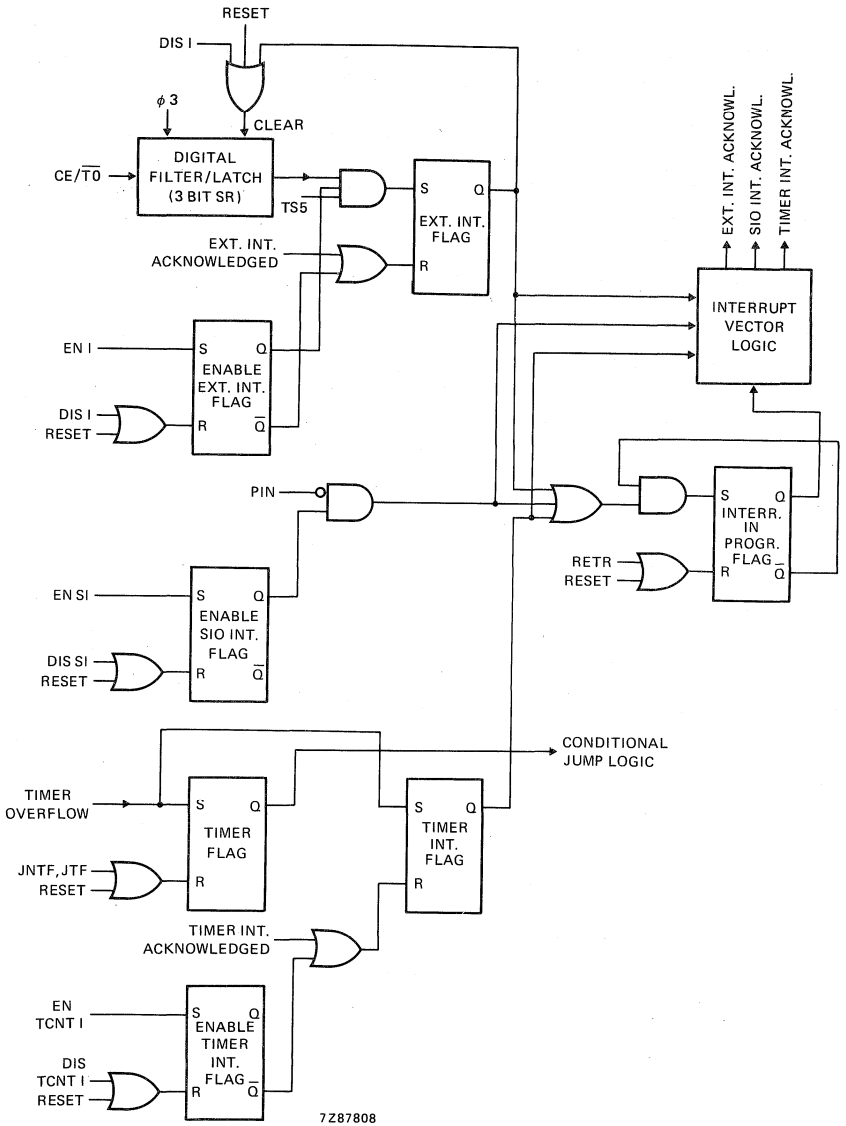


Fig. 15 Interrupt logic.

Notes to figure 15

1. CE/$\overline{T0}$ positive edge is always latched in the digital filter/latch.
2. Correct interrupt timing in ensured when CE/$\overline{T0}$ is LOW for > 4 CP followed by a HIGH for > 7 CP.
3. When the interrupt in progress flag is set, further interrupts are latched but ignored, until RETR is executed.
4. A DIS I instruction always clears a pending external interrupt.

FUNCTIONAL DESCRIPTION (continued)

Oscillator (see Fig. 16)

The 3,58 MHz oscillator can be inhibited by the STOP instruction under software control. It is also inhibited when a low-voltage condition is present to prevent discharge of a weak back-up battery.

Provided the supply voltage is within the operating range, the oscillator will be restarted after a STOP instruction by a HIGH level at either the CE/$\overline{T0}$ or RESET pin.

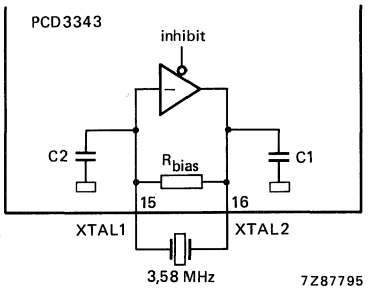


Fig. 16 Oscillator with integrated elements.

The oscillator has the output drive capability for the DTMF generator (PCD3311/3312) via pin 16 (XTAL 2). An external clock can be applied to pin 15 (XTAL 1). A machine cycle consists of 10 time slots, each time slot being 3 oscillator periods.

In telephony applications the 3,58 MHz crystal provides a 8,4 μs machine cycle. The range of the clock frequency is from 100 kHz up to a maximum which is a function of the supply voltage (see Fig. 23).

Timer/event counter (see Fig. 17)

An internal 8-bit up-counter is provided. This can count external events, modulo-32 machine cycles, or machine cycles directly. Table 4 gives the instructions that control the counter and the prescaler, and the functions performed.

When used as a timer, the input to the counter is either the overflow or input of a 5-bit prescaler. When used as an event counter, LOW-to-HIGH transitions on pin 13 (T1) are counted. The maximum rate at which the counter may be incremented is once every machine cycle (182,6 kHz for a 8,4 μs machine cycle). When the counter overflows, the timer flag is set. The flag can be tested and reset using the JTF (jump if timer flag = 1) or JNTF instruction. Overflow also generates an interrupt to the processor via setting of the Timer Interrpt Flag when the timer/event counter interrupt is enabled.

Table 4 Timer/event counter control

function	timer mode modulo-1, modulo-32*	counter mode
CLEAR	MOV T,A (A) = 0 or RESET	MOV T,A (A) = 0 or RESET
PRESET	MOV T,A	MOV T,A
START	STRT T	STRT CNT
STOP	STOP TCNT or RESET	STOP TCNT or RESET
TEST	JTF/JNTF	JTF/JNTF
READ**	MOV A,T	MOV A,T

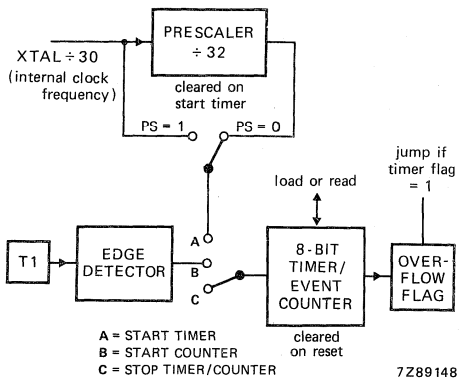


Fig. 17 Timer/event counter.

DEVELOPMENT DATA

Program status word (see Fig. 18)

The program status word (PSW) is an 8-bit word (1 byte) in the CPU which stores information about the current status of the microcontroller.
The PSW bits are:

- Bits 0 to 2 — stack pointer bits (SP_0, SP_1, SP_2)
- Bit 3 — prescaler select (PS);
  0 = modulo-32; 1 = modulo-1 (no prescaling)
- Bit 4 — working register bank select (RBS);
  0 = register bank 0; 1 = register bank 1
- Bit 5 — not used (1)
- Bit 6 — auxiliary carry (AC); half-carry bit generated by an ADD instruction and used by the decimal adjust instruction DA A
- Bit 7 — carry (CY); the carry flag indicates that previous operation has resulted in an overflow of the accumulator.

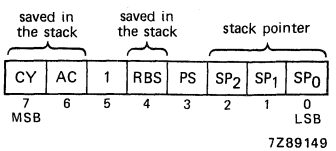


Fig. 18 Program status word.

* With prescaler select, PS = 0, the timer counts modulo-32 machine cycles, with PS = 1 it counts modulo-1 cycles (prescaler not used); prescaler cleared with STRT T, prescaler not readable.

** READ does not disturb the counting process.

FUNCTIONAL DESCRIPTION (continued)

Program status word (continued)

All bits can be read using the MOV A, PSW instruction. Bits 7 and 6 are set and cleared by CPU operation. Bit 4 can be changed by a SEL RB instruction, bit 3 by the MOV PSW, A instruction, and bits 0, 1 and 2 by the CALL, RET or RETR instructions and in the event of an interrupt. Bits 7, 6 and 4 are stored in the program counter stack during subroutine and interrupt calls. These bits are restored in the PSW with a RETR (return and restore) instruction which must be used at the end of an interrupt and can be used at the end of a normal subroutine. The RET instruction has no restore feature and cannot be used at the end of an interrupt.

Program counter (see Fig. 19)

A 13-bit program counter is used to facilitate 8 K bytes of ROM being addressed. The arrangement of the bits is shown in figure 19. During an interrupt subroutine PC_{11} and PC_{12} are forced to logic 0. All 13 bits are saved in the stack during CALL and interrupt routines.

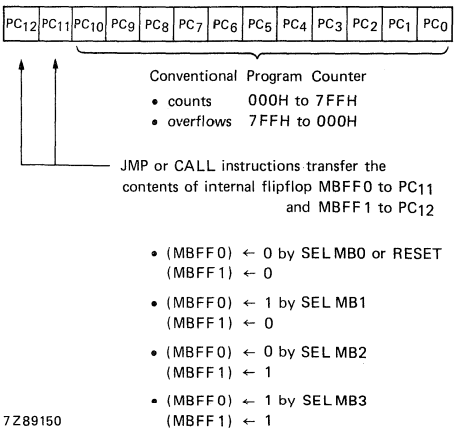


Fig. 19 Program counter.

Central processing unit

The PCD3343 has arithmetic, logical and branching capabilities. The DA A, SWAP A and XCHD instructions simplify BCD arithmetic and the handling of nibbles. The MOVP A,@A instruction permits efficient table look-up from the current ROM page.

Conditional branch logic

The conditional branch logic within the processor enables several conditions, internal and external to the processor, to be tested by the user's program. Table 5 lists the conditional jump instructions used to change the program sequence. The DJNZ instruction decrements a designated register or data memory location and branches if the contents are not zero. This instruction is useful for looping control. The JMPP@A instruction allows multiway branches to destinations determined by the contents of the accumulator.

Table 5 Conditional branches

test	jump condition	jump instruction
accumulator	all bits zero	JZ
	any bit non-zero	JNZ
accumulator bit test	1	JB0 to JB7
carry flag	1	JC
	0	JNC
timer overflow flag	1	JTF
	0	JNTF
test input T0	1	JNT0
	0	JT0*
test input T1	1	JT1
	0	JNT1
register	non-zero	DJNZ

Test input T1 (pin 13)

The T1 input line can be used as:

- A test input for branch instructions JT1 and JNT1
- An external input to the event counter

When used as a test input:

- JT1 instruction tests for logic 1 level
- JNT1 instruction tests for logic 0 level

When used as an input to the event counter, T1 must be LOW for >4 CP, followed by a HIGH for >4 CP. The transition can be recognized with a repetition rate of once per 30 oscillator clock periods (1 machine cycle).

There is no internal pull-up or pull-down resistor connected to the T1 input. If required it must be externally connected to a resistor ($R = \leqslant 100\ k\Omega$). When T1 is not used pin 13 must be connected to V_{DD} or V_{SS}.

Reset (pin 17)

A positive-going signal on the RESET input/output:

- Sets the program counter to zero
- Selects location 0 of memory band 0 and register bank 0
- Sets the stack pointer to zero (000); pointing to RAM address 8
- Disables the interrupts (external, timer and serial I/O)
- Stops the timer/event counter, then sets it to zero
- Sets the timer prescaler to modulo-32
- Resets the timer flag
- Sets all ports according to reset states
- Sets all ports except P23 to reset state *(see Serial I/O)*
- Cancels IDLE and STOP mode

A negative-going signal on the RESET input/output:

- Sets P23/SDA and SCLK to HIGH after a maximum of 30 clock pulses
- Sets the serial I/O to slave receiver mode and disables the serial I/O after 1866 clock pulses
- Starts an internal delay of 1866 clock pulses after which the microcontroller commences operation.

DEVELOPMENT DATA

FUNCTIONAL DESCRIPTION (continued)

Power-on-reset and low-voltage detection (see Fig. 20)

In telephony applications, correct operation of the PCD3343 during moments of slowly changing supply voltages and low-voltage conditions is essential. This is achieved by the addition of an internal power-on-reset and low-voltage detection circuit.
To allow an external RESET signal being fed into the PCD3343, the reset pin (pin 17) has been configured as an input/output.

While a reset condition exists in the detection circuit, pin 17 is pulled HIGH by TR1 controlled by the reset circuit.
When the reset condition is not present a pull-down current source (TR2) will be activated. TR2 forces pin 17 LOW thus removing the RESET signal from the microcontroller.

Since the level at pin 17 is recognized by the microcontroller, the reset time constant can be stretched by connecting an external capacitor between V_{DD} and pin 17 (see Fig. 22).

The signal at pin 17 can also be used as an output to reset other devices in the system.

The internal reset circuit monitors the PCD3343 supply voltage. If the voltage drops below the switching level (typ. 1,3 V), a reset (HIGH) is applied to pin 17. This reset is removed (pin 17 goes LOW), after a fixed delay (t_d), when the supply voltage rises above the switching level again. The delay ensures a complete reset even when the supply voltage quickly rises above switching level after initial switch-on.

During a low-voltage condition the oscillator is inhibited to prevent complete discharge of a weak battery. The timing of the power-on-reset and low-voltage detection circuit is shown in figure 21.

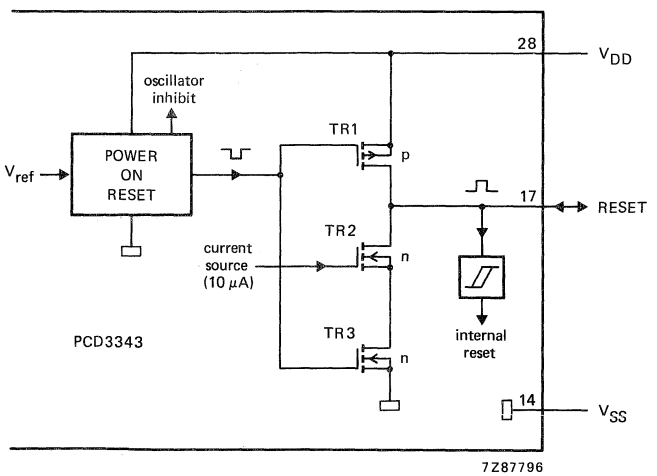
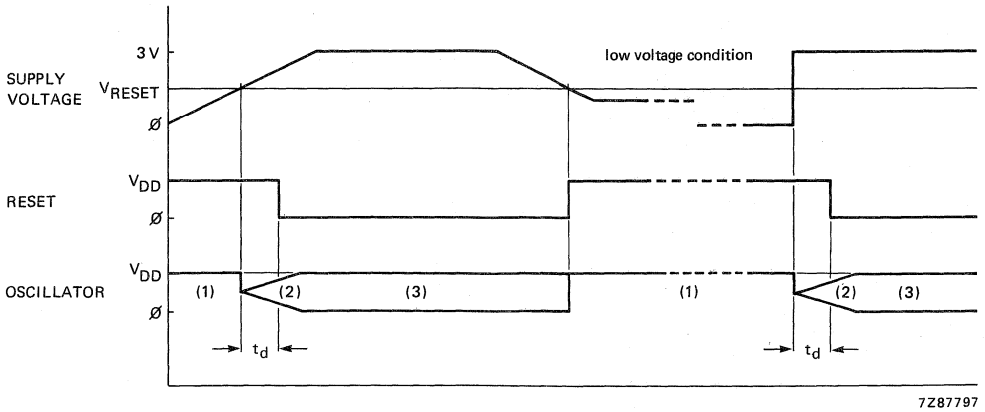


Fig. 20 Power-on-reset configuration.



Where: (1) Oscillator inhibited
(2) Oscillator starting
(3) Oscillator running, but may be stopped with a STOP condition

Fig. 21 Timing of power-on-reset and low-voltage detection.

DEVELOPMENT DATA

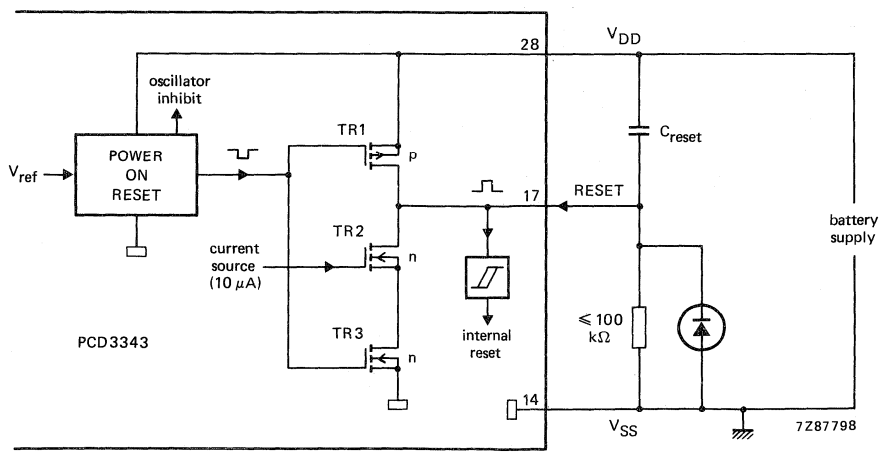


Fig. 22 Stretched power-on-reset with external capacitor.

INSTRUCTION SET

The PCD3343 instruction set consists of over 80 one and two byte instructions and is based on the MAB8048 instruction set. New instructions include those for serial I/O operation and memory bank selection. Program code efficiency is high because all RAM locations and all ROM locations on a 256 byte page require only a single byte address.

Table 8 gives the instruction set of the PCD3343. Table 7 shows the instruction map and Table 6 details the symbols and definition descriptions that are used.

Table 6 Symbols and definitions used in Table 8

symbol	definition description
A	accumulator
addr	program memory address
Bb	bit designation (b = 0-7)
RBS	register bank select
C	carry bit (bit CY)
CNT	event counter
D	mnemonic for 4-bit digit (nibble)
data	8-bit number or expression
I	interrupt
MB	memory bank
MBFF	memory bank flip-flop
P	mnemonic for 'in-page' operation
PC	program counter
Pp	port designation (p = 0, 1 or 2)
PSW	program status word
RB	register bank
Rr	register designation (r = 0-7)
Sn	serial I/O register
SP	stack pointer
T	timer
TF	timer flag
T0, T1	test 0 and 1 inputs
#	immediate data prefix
@	indirect address prefix
(X)	contents of X
((X))	contents of location addressed by X
←	is replaced by
↔	is exchanged with

Table 7 PCD3343 instruction map

first hexadecimal character of opcode ↓ / second hexadecimal character of opcode →	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
0	NOP	IDLE		ADD A,#data	JMP page 0	EN I	JNTF addr	DEC A	IN A,Pp 0 1 2				MOV A,Sn 0 1			
1	INC @Rr 0 1		JB0 addr	ADDC A,#data	CALL page 0	DIS I	JTF addr	INC A	INC Rr 0 1 2 3 4 5 6 7							
2	XCH A,@Rr 0 1		STOP	MOV A,#data	JMP page 1	EN TCNTI	JNTO addr	CLR A	XCH A,Rr 0 1 2 3 4 5 6 7							
3	XCHD A,@Rr 0 1		JB1 addr		CALL page 1	DIS TCNTI	JTO addr	CPL A	OUTL Pp,A 0 1 2				MOV Sn,A 0 1 2			
4	ORL A,@Rr 0 1		MOV A,T	ORL A,#data	JMP page 2	STRT CNT	JNT1 addr	SWAP A	ORL A,Rr 0 1 2 3 4 5 6 7							
5	ANL A,@Rr 0 1		JB2 addr	ANL A,#data	CALL page 2	STRT T	JT1 addr	DA A	ANL A,Rr 0 1 2 3 4 5 6 7							
6	ADD A,@Rr 0 1		MOV T,A		JMP page 3	STOP TCNT		RRC A	ADD A,Rr 0 1 2 3 4 5 6 7							
7	ADDC A,@Rr 0 1		JB3 addr		CALL page 3			RR A	ADDC A,Rr 0 1 2 3 4 5 6 7							
8				RET	JMP page 4	EN SI			ORL Pp,#data 0 1 2							
9			JB4 addr	RETR	CALL page 4	DIS SI	JNZ addr	CLR C	ANL Pp,#data 0 1 2				MOV Sn,#data 0 1 2			
A	MOV @Rr,A 0 1			MOVP A,@A	JMP page 5	SEL MB2		CPL C	MOV Rr,A 0 1 2 3 4 5 6 7							
B	MOV @Rr,#data 0 1		JB5 addr	JMPP @A	CALL page 5	SEL MB3			MOV Rr,#data 0 1 2 3 4 5 6 7							
C	DEC @Rr 0 1				JMP page 6	SEL RB0	JZ addr	MOV A,PSW	DEC Rr 0 1 2 3 4 5 6 7							
D	XRL A,@Rr 0 1		JB6 addr	XRL A,#data	CALL page 6	SEL RB1		MOV PSW,A	XRL A,Rr 0 1 2 3 4 5 6 7							
E	DJNZ @Rr,addr 0 1				JMP page 7	SEL MB0	JNC addr	RL A	DJNZ Rr, addr 0 1 2 3 4 5 6 7							
F	MOV A,@Rr 0 1		JB7 addr		CALL page 7	SEL MB1	JC addr	RLC A	MOV A,Rr 0 1 2 3 4 5 6 7							

INSTRUCTION SET (continued)

Table 8 Instruction set

	mnemonic	opcode (hex.)	bytes/ cycles	description	function		notes
ACCUMULATOR	ADD A, Rr	6*	1/1	Add register contents to A	(A)←(A) + (Rr)	r = 0–7	1
	ADD A, @Rr	60 61	1/1	Add RAM data, addressed by Rr, to A	(A)←(A) + ((R0)) (A)←(A) + ((R1))		1
	ADD A, #data	03 data	2/2	Add immediate data to A	(A)←(A) + data		1
	ADDC A, Rr	7*	1/1	Add carry and register contents to A	(A)←(A) + (Rr) + (C)	r = 0–7	1
	ADDC A, @Rr	70 71	1/1	Add carry and RAM data, addressed by Rr, to A	(A)←(A) + ((R0)) + (C) (A)←(A) + ((R1)) + (C)		1
	ADDC A, #data	13 data	2/2	Add carry and immediate data to A	(A)←(A) + data + (C)		1
	ANL A, Rr	5*	1/1	'AND' Rr with A	(A)←(A) AND (Rr)	r = 0–7	
	ANL A, @Rr	50 51	1/1	'AND' RAM data, addressed by Rr, with A	(A)←(A) AND ((R0)) (A)←(A) AND ((R1))		
	ANL A, #data	53 data	2/2	'AND' immediate data with A	(A)←(A) AND data		
	ORL A, Rr	4*	1/1	'OR' Rr with A	(A)←(A) OR (Rr)	r = 0–7	
	ORL A, @Rr	40 41	1/1	'OR' RAM data, addressed by Rr, with A	(A)←(A) OR ((R0)) (A)←(A) OR ((R1))		
	ORL A, #data	43 data	2/2	'OR' immediate data with A	(A)←(A) OR data		
	XRL A, Rr	D*	1/1	'XOR' Rr with A	(A)←(A) XOR (Rr)	r = 0–7	
	XRL A, @Rr	D0 D1	1/1	'XOR' RAM, addressed by Rr, with A	(A)←(A) XOR ((R0)) (A)←(A) XOR ((R1))		
	XRL A, #data	D3 data	2/2	'XOR' immediate data with A	(A)←(A) XOR data		
	INC A	17	1/1	increment A by 1	(A)←(A) + 1		
	DEC A	07	1/1	decrement A by 1	(A)←(A) −1		
	CLR A	27	1/1	clear A to zero	(A)←0		
	CPL A	37	1/1	one's complement A	(A)←NOT(A)		
	RL A	E7	1/1	rotate A left	$(A_{n+1})\leftarrow(A_n)$ $(A_0)\leftarrow(A_7)$	n = 0–6	

DEVELOPMENT DATA

ACCUMULATOR (cont.)	RLC A	F7	1/1	rotate A left through carry	$(A_{n+1})\leftarrow A_n$ $(A_0)\leftarrow(C)$, $(C)\leftarrow(A_7)$	n = 0–6	2
	RR A	77	1/1	rotate A right	$(A_n)\leftarrow(A_{n+1})$ $(A_7)\leftarrow(A_0)$	n = 0–6	
	RRC A	67	1/1	rotate A right through carry	$(A_n)\leftarrow(A_{n+1})$ $(A_7)\leftarrow(C)$, $(C)\leftarrow(A_0)$	n = 0–6	2
	DA A	57	1/1	decimal adjust A			2
	SWAP A	47	1/1	swap nibbles of A	$(A_{4-7})\leftrightarrow(A_{0-3})$		
DATA MOVES	MOV A, Rr	F*	1/1	move register contents to A	(A)←(Rr)	r = 0–7	
	MOV A, @Rr	F0 F1	1/1	move RAM data, addressed by Rr, to A	(A)←((R0)) (A)←((R1))		
	MOV A, #data	23 data	2/2	move immediate data to A	(A)←data		
	MOV Rr, A	A*	1/1	move accumulator contents to register	(Rr)←(A)	r = 0–7	
	MOV @Rr, A	A0 A1	1/1	move accumulator contents to RAM location addressed by Rr	((R0))←(A) ((R1))←(A)		
	MOV Rr, #data	B* data	2/2	move immediate data to Rr	(Rr)←data		
	MOV @Rr, #data	B0 data B1 data	2/2	move immediate data to RAM location addressed by Rr	((R0))←data ((R1))←data		
	XCH A, Rr	2*	1/1	exchange accumulator contents with Rr	(A)↔(Rr)	r = 0–7	
	XCH A, @Rr	20 21	1/1	exchange accumulator contents with RAM data addressed by Rr	(A)↔((R0)) (A)↔((R1))		
	XCHD A, @Rr	30 31	1/1	exchange lower nibbles of A and RAM data addressed by Rr	$(A_{0-3})\leftrightarrow((R0_{0-3}))$ $(A_{0-3})\leftrightarrow((R1_{0-3}))$		
	MOV A, PSW	C7	1/1	move PSW contents to accumulator	(A)←(PSW)		
	MOV PSW, A	D7	1/1	move accumulator bit 3 to PSW_3	$(PSW_3)\leftarrow(A_3)$		3
	MOVP A, @A	A3	1/2	move indirectly addressed data in current page to A	$(PC_{0-7})\leftarrow(A)$, (A)←((PC))		
FLAGS	CLR C	97	1/1	clear carry bit	(C)←0		2
	CPL C	A7	1/1	complement carry bit	(C)←NOT(C)		2

	mnemonic	opcode (hex.)	bytes/ cycles	description	function	notes
REGISTER	INC Rr	1*	1/1	increment register by 1	(Rr)←(Rr) + 1 r = 0–7	
	INC @Rr	10 11	1/1	increment RAM data, addressed by Rr, by 1	((R0))←((R0)) + 1 ((R1))←((R1)) + 1	
	DEC Rr	C*	1/1	decrement register by 1	(Rr)←(Rr) – 1 r = 0–7	
	DEC @Rr	C0 C1	1/1	decrement RAM data, addressed by Rr, by 1	((R0))←((R0)) – 1 ((R1))←((R1)) – 1	
BRANCH	JMP addr	● 4 address	2/2	unconditional jump within a 2 K bank	(PC_{8-10})←$addr_{8-10}$ (PC_{0-7})←$addr_{0-7}$ (PC_{11-12})←MBFF 0–1	
	JMPP @A	B3	1/2	indirect jump within a page	(PC_{0-7})←((A))	
	DJNZ Rr, addr	E* address	2/2	decrement Rr by 1 and jump if not zero to addr	(Rr)←(Rr) – 1 r = 0–7 if (Rr) not zero (PC_{0-7})←addr	
	DJNZ @Rr, addr	E0	2/2	decrement RAM data, addressed by Rr by 1 and jump if not zero to addr	((R0))←((R0)) – 1 if ((R0)) not zero (PC_{0-7})←addr	
		E1			((R1))←((R1)) – 1 if ((R1)) not zero (PC_{0-7})←addr	
	JBb addr	▲ 2 address	2/2	jump to addr if Acc. bit b = 1	if b = 1 : (PC_{0-7})←addr b = 0–7	
	JC addr	F6 address	2/2	jump to addr if C = 1	if C = 1 : (PC_{0-7})←addr	
	JNC addr	E6 address	2/2	jump to addr if C = 0	if C = 0 : (PC_{0-7})←addr	
	JZ addr	C6 address	2/2	jump to addr if A = 0	if A = 0 : (PC_{0-7})←addr	
	JNZ addr	96 address	2/2	jump to addr if A is NOT zero	if A≠0 : (PC_{0-7})←addr	
	JT0 addr	36 address	2/2	jump to addr if T0 = 0	if T0 = 0: (PC_{0-7})←addr	
	JNT0 addr	26 address	2/2	jump to addr if T0 = 1	if T0 = 1: (PC_{0-7})←addr	
	JT1 addr	56 address	2/2	jump to addr if T1 = 1	if T1 = 1: (PC_{0-7})←addr	
	JNT1 addr	46 address	2/2	jump to addr if T1 = 0	if T1 = 0: (PC_{0-7})←addr	
	JTF addr	16 address	2/2	jump to addr if Timer Flag = 1	if TF = 1: (PC_{0-7})←addr	4
	JNTF addr	06 address	2/2	jump to addr if Timer Flag = 0	if TF = 0: (PC_{0-7})←addr	

TIMER/EVENT COUNTER	MOV A, T	42	1/1	move timer/event counter contents to accumulator	(A)←(T)	
	MOV T, A	62	1/1	move accumulator contents to timer/event counter	(T)←(A)	
	STRT CNT	45	1/1	start event counter		
	STRT T	55	1/1	start timer		
	STOP TCNT	65	1/1	stop timer/event counter		
	EN TCNTI	25	1/1	enable timer/event counter interrupt		
	DIS TCNTI	35	1/1	disable timer/event counter interrupt		
CONTROL	EN I	05	1/1	enable external interrupt		
	DIS I	15	1/1	disable external interrupt		
	SEL RB0	C5	1/1	select register bank 0	(RBS)←0	5
	SEL RB1	D5	1/1	select register bank 1	(RBS)←1	5
	SEL MB0	E5	1/1	select program memory bank 0	(MBFF0)←0, (MBFF1)←0	
	SEL MB1	F5	1/1	select program memory bank 1	(MBFF0)←1, (MBFF1)←0	
	SEL MB2	A5	1/1	select program memory bank 2	(MBFF0)←0, (MBFF1)←1	
	SEL MB3	B5	1/1	select program memory bank 3	(MBFF0)←1, (MBFF1)←1	
	STOP	22	1/1	enter STOP mode		
	IDLE	01	1/1	enter IDLE mode		
SUBROUTINE	CALL addr	▲ 4 address	2/2	jump to subroutine	((SP))←(PC), ($PSW_{4, 6, 7}$) 6 (SP)←(SP) + 1 (PC_{8-10})←$addr_{8-10}$ (PC_{0-7})←$addr_{0-7}$ (PC_{11-12})←MBFF 0–1	6
	RET	83	1/2	return from subroutine	(SP)←(SP) – 1 (PC)←((SP))	6
	RETR	93	1/2	return from interrupt and restore bits 4, 6, 7 of PSW	(SP)←(SP) – 1 ($PSW_{4, 6, 7}$) + (PC)←((SP))	6

	mnemonic	opcode (hex.)	bytes/ cycles	description	function	notes
PARALLEL INPUT/OUTPUT	IN A, Pp	08 09 0A	1/2	input port p data to accumulator	(A)←(P0) (A)←(P1) (A)←(P2)	 7
	OUTL Pp, A	38 39 3A	1/2	output accumulator data to port p	(P0)←(A) (P1)←(A) (P2)←(A)	
	ANL Pp, #data	98 99 9A	2/2	AND port p data with immediate data	(P0)←(P0) AND data (P1)←(P1) AND data (P2)←(P2) AND data	
	ORL Pp, #data	88 89 8A	2/2	OR port p data with immediate data	(P0)←(P0) OR data (P1)←(P1) OR data (P2)←(P2) OR data	
SERIAL INPUT/OUTPUT	MOV A, S_n	0C 0D	1/2	move serial I/O register contents to accumulator	(A)←(S0) (A)←(S1)	 8
	MOV S_n, A	3C 3D 3E	1/2	move accumulator contents to serial I/O register	(S0)←(A) (S1)←(A) (S2)←(A)	 9
	MOV S_n, #data	9C 9D 9E	2/2	move immediate data to serial I/O register	(S0)←data (S1)←data (S2)←data	
	EN SI	85	1/1	enable serial I/O interrupt		
	DIS SI	95	1/1	disable serial I/O interrupt		
	NOP	00	1/1	no operation		

Notes to Table 8

1. PSW CY, AC affected
2. PSW CY affected
3. PSW PS affected
4. Execution of JTF and JNTF instructions resets the Timer Flag (TF).

* : 8, 9, A, B, C, D, E, F
● : 0, 2, 4, 6, 8, A, C, E
▲ : 1, 3, 5, 7, 9, B, D, F

5. PSW RBS affected
6. PSW SP_0, SP_1, SP_2 affected
7. (A) = 1111 P23, P22, P21, P20.
8. (S1) has a different meaning for read and write operation, see serial I/O interface.
9. (S2) is a write only register. Reading S2 will give value FFH.

RATINGS

Limiting values in accordance with the Absolute Maximum System (IEC 134)

Supply voltage (pin 28)	V_{DD}		−0,8 to + 8	V
All input voltages	V_I		0,8 to V_{DD} + 0,8	V
D.C. current into any input or output	$\pm I_I, \pm I_O$	max.	10	mA
Total power dissipation (see note)	P_{tot}	max.	500	mW
Power dissipation per output				
except P23, SCLK	P_O	max.	50	mW
P23, SCLK	P_O	max.	180	mW
Storage temperature range	T_{stg}		−65 to + 150	°C
Operating ambient temperature range	T_{amb}		−25 to + 70	°C
Operating junction temperature	T_j	max.	125	°C

Note

Thermal resistance (junction to ambient)				
for SOT117	$R_{th\ j\text{-}a}$	max.	120	K/W
for SOT136A	$R_{th\ j\text{-}a}$	max.	150	K/W

DEVELOPMENT DATA

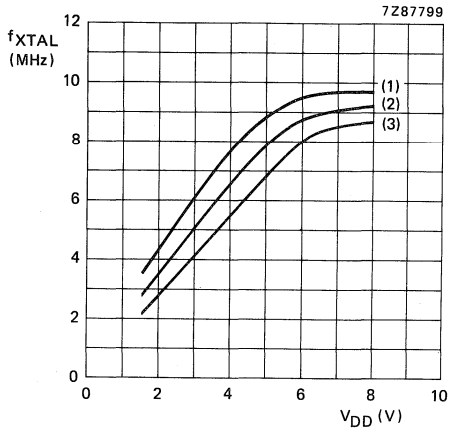
D.C. CHARACTERISTICS

V_{DD} = 2,75 to 6 V; V_{SS} = 0 V; T_{amb} = −25 to +70 °C; all voltages with respect to V_{SS}; f = 3,58 MHz with R_S = 50 Ω; unless otherwise specified.

parameter	symbol	min.	typ.	max.	unit
Supply voltage					
operating (see Fig. 23)	V_{DD}	1,8	–	6	V
STOP mode for RAM retention	V_{DD}	1,0	–	6	V
Supply current					
operating					
at V_{DD} = 3 V (see Fig. 24)	I_{DD}	–	600	–	μA
IDLE mode					
at V_{DD} = 3 V (see Fig. 25)	I_{DD}	–	300	–	μA
STOP mode (see Fig. 26 and note 1)					
at V_{DD} = 1,8 V; T_{amb} = 25 °C	I_{DD}	–	1,2	2,5	μA
at V_{DD} = 1,8 V; T_{amb} = 55 °C	I_{DD}	–	–	5	μA
at V_{DD} = 1,8 V; T_{amb} = 70 °C	I_{DD}	–	–	10	μA
RESET I/O					
Switching level	V_{RESET}	–	1,3	–	V
Sink current					
at $V_{DD} > V_{RESET}$	I_{OL}	–	7	–	μA
Inputs					
Input voltage LOW	V_{IL}	0	–	$0{,}3V_{DD}$	V
Input voltage HIGH	V_{IH}	$0{,}7V_{DD}$	–	V_{DD}	V
Input leakage current					
at $V_{SS} < V_I < V_{DD}$	$\pm I_{IL}$	–	–	1	μA
Outputs					
Output voltage LOW					
at $V_I = V_{SS}$ or V_{DD}; $\lvert I_O \rvert < 1$ μA	V_{OL}	–	–	0,05	V
Output sink current LOW					
at V_{DD} = 3 V; V_O = 0,4 V					
except P23/SDA, SCLK (see Fig. 27)	I_{OL}	0,75	1,5	–	mA
P23/SDA, SCLK (see Fig. 28)	I_{OL}	1,5	–	–	mA
Pull-up output source current HIGH (see Fig. 29)					
at V_{DD} = 3 V; $V_O = 0{,}9V_{DD}$	$-I_{OH}$	25	–	–	μA
at V_{DD} = 3 V; $V_O = V_{SS}$	$-I_{OH}$	–	–	200	μA
Push-pull output source current HIGH					
at V_{DD} = 3 V; $V_O = V_{DD}-0{,}4$ V	$-I_{OH}$	0,75	1,5	–	mA

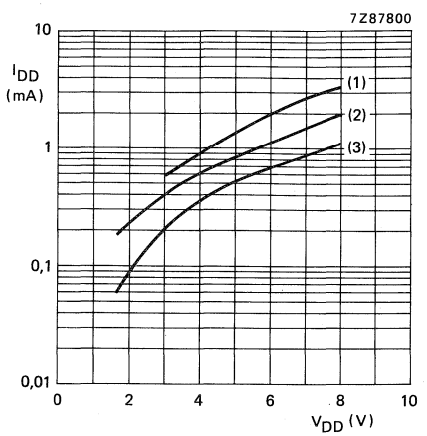
Note 1

Crystal connected between XTAL 1 and XTAL 2; SCL and SDA pulled to V_{DD} via 5,6 kΩ resistor; CE and T1 at V_{SS}.



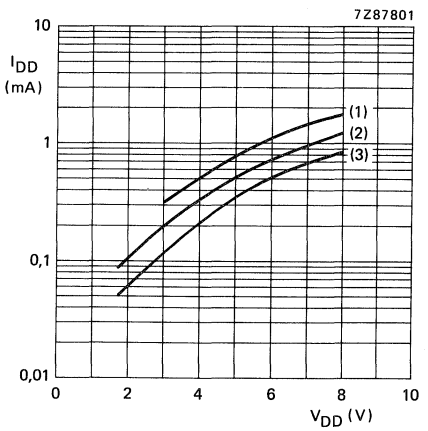
(1) $T_{amb} = -25\ ^{o}C$
(2) $T_{amb} = 25\ ^{o}C$
(3) $T_{amb} = 70\ ^{o}C$

Fig. 23 Maximum clock frequency (f_{XTAL}) as a function of the supply voltage (V_{DD}).



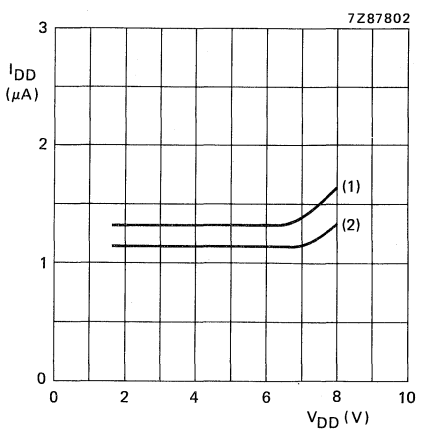
(1) clock frequency = 4 MHz
(2) clock frequency = 2 MHz
(3) clock frequency = 500 kHz

Fig. 24 Typical supply current (I_{DD}) in operating mode as a function of the supply voltage (V_{DD}); $T_{amb} = 25\ ^{o}C$.



(1) clock frequency = 4 MHz
(2) clock frequency = 2 MHz
(3) clock frequency = 500 kHz

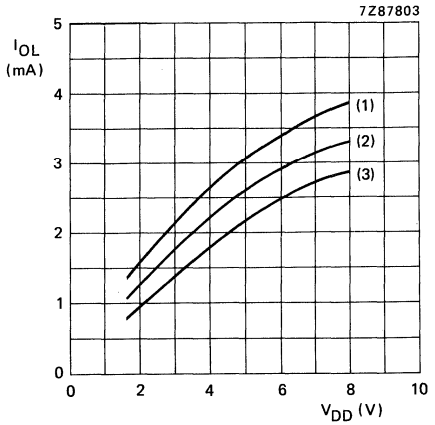
Fig. 25 Typical supply current (I_{DD}) in IDLE mode as a function of the supply voltage (V_{DD}); $T_{amb} = 25\ ^{o}C$.



(1) $T_{amb} = 70\ ^{o}C$
(2) $T_{amb} = 25\ ^{o}C$

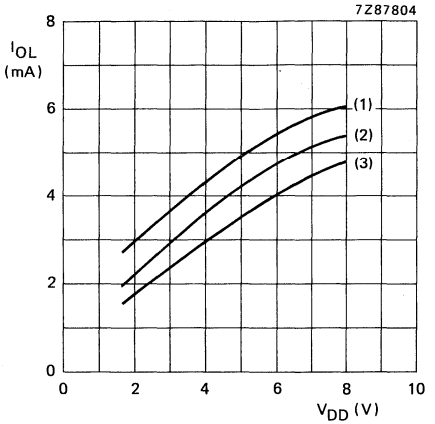
Fig. 26 Typical supply current (I_{DD}) in STOP mode as a function of the supply voltage (V_{DD}).

DEVELOPMENT DATA



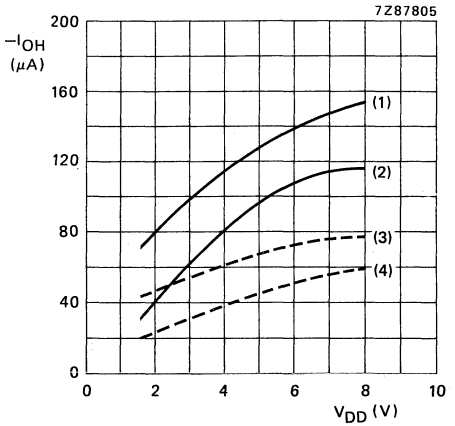
(1) $T_{amb} = -25\ ^{o}C$
(2) $T_{amb} = 25\ ^{o}C$
(3) $T_{amb} = 70\ ^{o}C$

Fig. 27 Output sink current LOW (I_{OL}), except outputs P23/SDA and SCLK, as a function of supply voltage (V_{DD}); $V_O = 0{,}4$ V.



(1) $T_{amb} = -25\ ^{o}C$
(2) $T_{amb} = +25\ ^{o}C$
(3) $T_{amb} = +70\ ^{o}C$

Fig. 28 Output current LOW (I_{OL}), outputs P23/SDA and SCLK, as a function of supply voltage (V_{DD}); $V_O = 0{,}4$ V.



(1) $T_{amb} = 25\ ^{o}C$; $V_O = V_{SS}$
(2) $T_{amb} = 25\ ^{o}C$; $V_O = 0{,}9V_{DD}$
(3) $T_{amb} = 70\ ^{o}C$; $V_O = V_{SS}$
(4) $T_{amb} = 70\ ^{o}C$; $V_O = 0{,}9V_{DD}$

Fig. 29 Output source current HIGH ($-I_{OH}$) as a function of supply voltage (V_{DD}).

DEVELOPMENT DATA

A.C. CHARACTERISTICS

Rise and fall times between 10 and 90% levels; C_L = 50 pF

parameter	symbol	at 70 °C max. value			unit
	V_{DD}	1,8	3,0	6,0	V
Fall time	t_f	200	100	70	ns
Rise time	t_r	200	100	80	ns

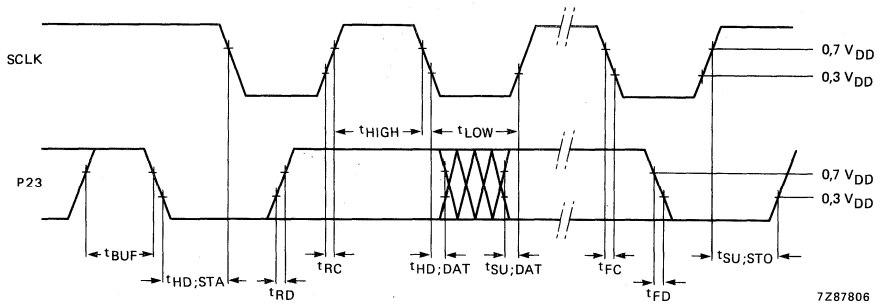


Fig. 30 PCD3343 timing requirements for the P23 and SCLK *input* signals.

Table 9 Input timing shown in figure 30

symbol	timing
t_{BUF}	$\geqslant 14t_{XTAL}$
$t_{HD;STA}$	$\geqslant 14t_{XTAL}$
t_{HIGH}	$\geqslant 17t_{XTAL}$
t_{LOW}	$\geqslant 17t_{XTAL}$
$t_{SY;STO}$	$\geqslant 14t_{XTAL}$
$t_{HD;DAT}$	> 0
$t_{SU;DAT}$	$\geqslant 250$ ns
t_{RD}	$\leqslant 1\ \mu s$
t_{RC}	$\leqslant 1\ \mu s$
t_{FD}	$\leqslant 1\ \mu s$
t_{FC}	$\leqslant 0{,}3\ \mu s$

Purchase of Philips' I²C components conveys a license under the Philips' I²C patent to use the components in the I²C-system provided the system conforms to the I²C specifications defined by Philips.

Notes to Table 9

t_{XTAL} = one period of the XTAL input frequency (f_{XTAL})
= 280 ns for f_{XTAL} = 3,58 MHz.
These figures apply to all modes.

A.C. CHARACTERISTICS (continued)

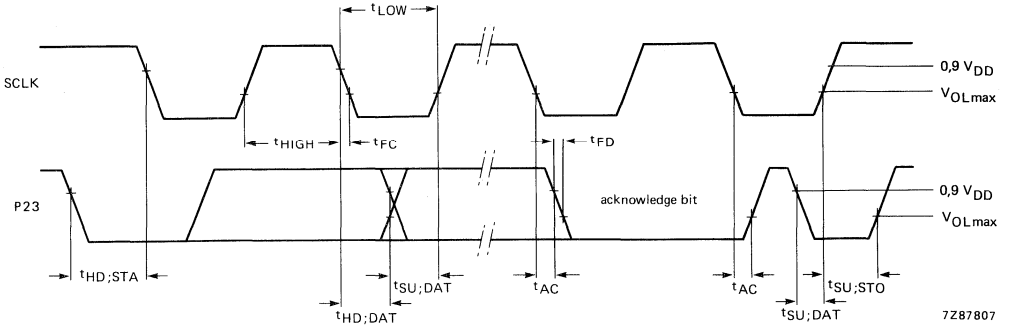


Fig. 31 PCD3343 timing requirements for the P23 and SCLK *output* signals.

Table 10 Output timing shown in figure 31

symbol	timing	
	normal mode (ASC in S2 = 0)	low-speed mode (ASC in S2 = 1)
$t_{HD;STA}$	½ (DF + 9) t_{XTAL}	¾ (DF + 9) t_{XTAL}
t_{HIGH}	½ (DF) t_{XTAL}	¾ (DF) t_{XTAL}
t_{LOW}	½ (DF) t_{XTAL}	¼ (DF) t_{XTAL}
$t_{SU;STO}$	½ (DF − 3) t_{XTAL}	¼ (DF − 3) t_{XTAL}
$t_{HD;DAT}$ (slave transmitter)		
any DF	$\geqslant 9t_{XTAL}$ $\leqslant 12t_{XTAL}$	$\geqslant 9t_{XTAL}$ $\leqslant 12t_{XTAL}$
$t_{HD;DAT}$ (master transmitter)		
for DF $\leqslant$ 51	$\geqslant 9t_{XTAL}$ $\leqslant 12t_{XTAL}$	– –
for DF $\leqslant$ 99	– –	$\geqslant 9t_{XTAL}$ $\leqslant 12t_{XTAL}$
$t_{SU;DAT}$ (master transmitter)		
for DF > 51	$\geqslant 15t_{XTAL}$ $\leqslant 24t_{XTAL}$	– –
for DF > 99	–	$\geqslant 15t_{XTAL}$ $\leqslant 24t_{XTAL}$
for DF $\leqslant$ 51	$\geqslant 9t_{XTAL}$	$\geqslant 9t_{XTAL}$
for DF $\leqslant$ 99	–	$\geqslant 9t_{XTAL}$
t_{AC}	$\geqslant 9t_{XTAL}$ $\leqslant 12t_{XTAL}$	$\geqslant 9t_{XTAL}$ $\leqslant 12t_{XTAL}$
t_{FD}, t_{FC}	$\leqslant$ 100 ns at C_b = 400 pF	$\leqslant$ 100 ns at C_b = 400 pF

Notes to Table 10

t_{XTAL} = one period of the XTAL input frequency (f_{XTAL})
= 280 ns for f_{XTAL} = 3,58 MHz.
DF = divisor (see Table 2 Serial I/O section).
C_b = the maximum bus capacitance for each line.

DEVELOPMENT DATA

APPLICATION INFORMATION

A block diagram of an electronic featurephone built around the PCD3343 is shown in figure 32. It comprises the following dedicated telephony IC's:

- TEA1060/1061 — transmission circuit for telephony
- PCD3312 — DTMF generator with Serial I/O
- PCE2111 or PCF8577 — 2 LCD drivers in LCD module MB7020160
- PCD8571 — 1 K RAM's with Serial I/O; the number of RAM's depends on the required amount of stored telephone numbers
- PCD3360/3361 — programmable multi-tone ringer

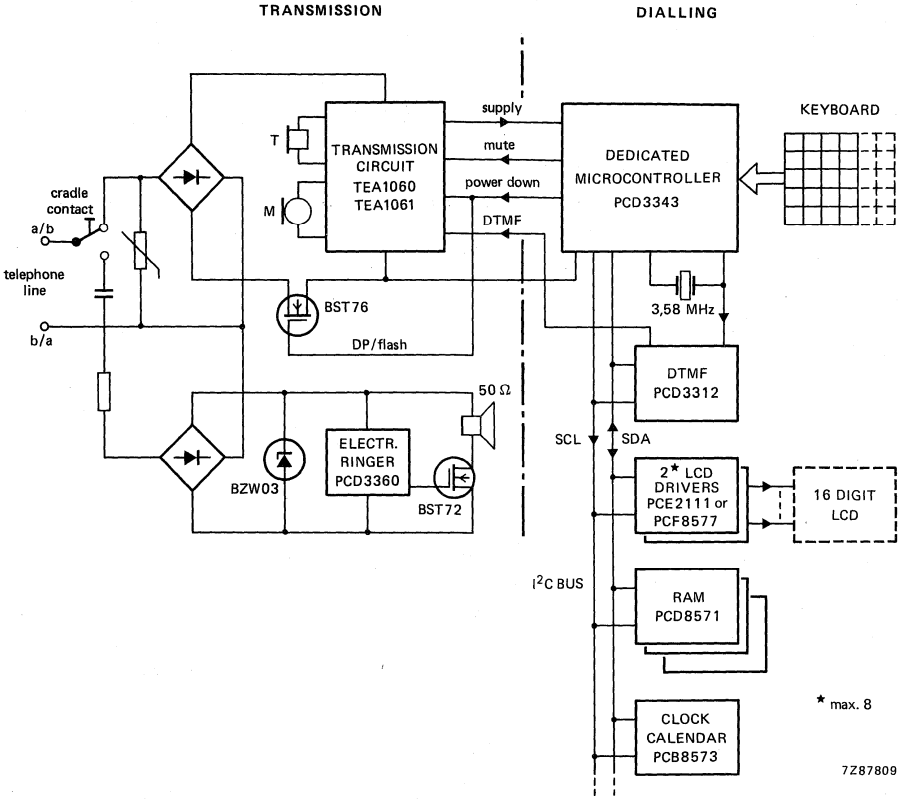


Fig. 32 Block diagram of electronic featurephone with common line interface.

A detailed application diagram of the PCD3343 with PCD3312 (DTMF), two PCD8571 (RAM) and two PCE2111 (LCD display drivers) is shown in figure 33.

Row 5 of the keyboard contains the following special keys:

- P — program and autodial
- FL — flash or register recall
- R — redial or extended redial
- AP — access pause

Row 6 contains the different diode options.

Columns 5 and 6 contain the button keys M0 to M9; single name keys for repertory telephone numbers.

APPLICATION INFORMATION (continued)

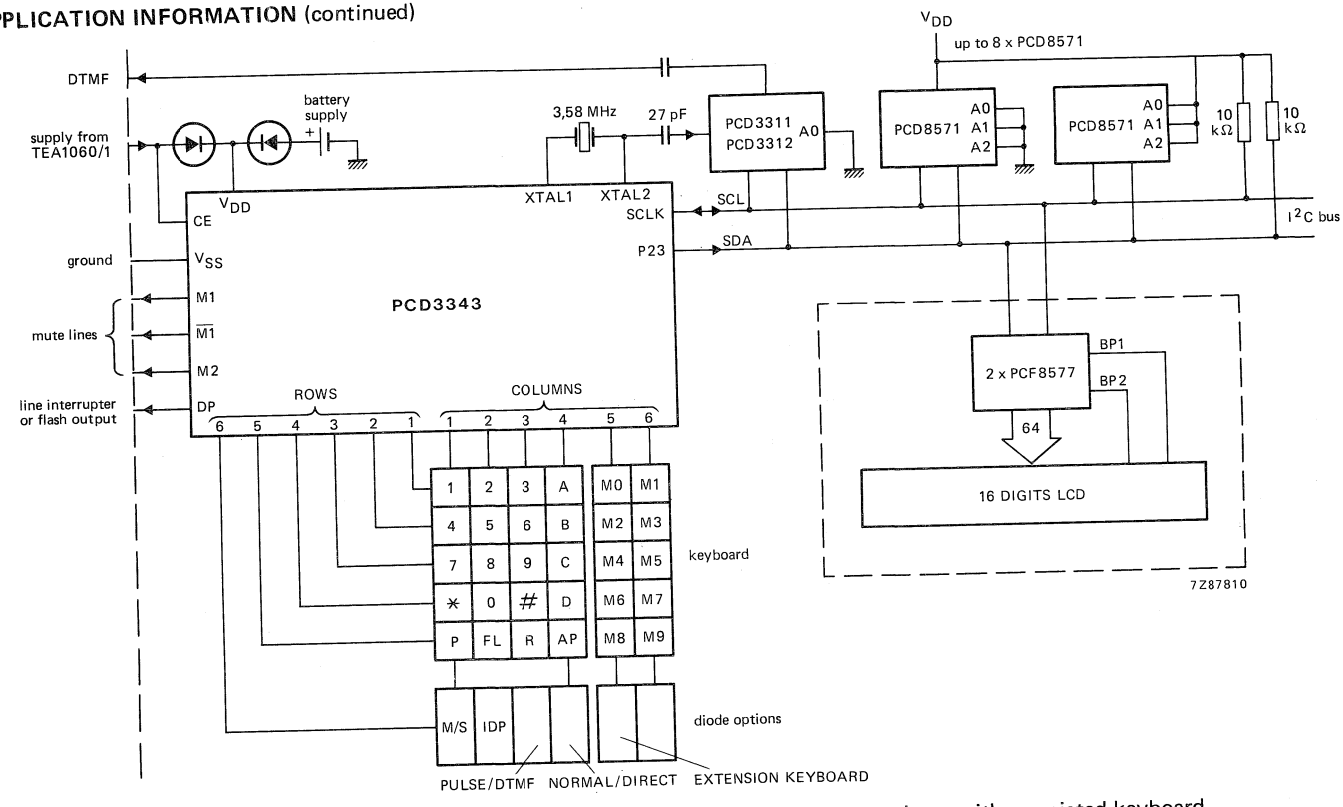


Fig. 33 Application diagram of PCD3343 for electronic featurephone with associated keyboard.

Additional information is available on request for the following:

- Serial I/O
- I^2C bus specification
- Interrupt logic
- Instruction set descriptions
- Software routines for an intelligent telephone set

CMOS MICROCONTROLLER WITH ON-CHIP DTMF GENERATOR

GENERAL DESCRIPTION

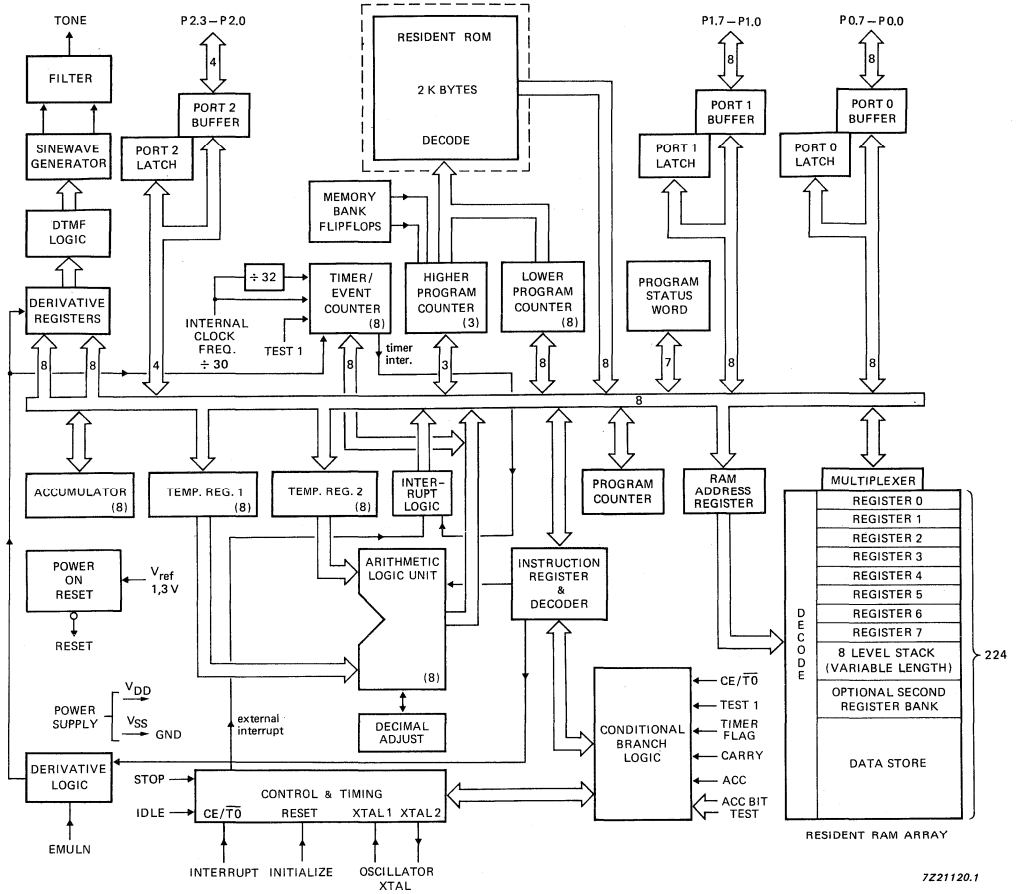
The PCD3344 is a single-chip 8-bit microcontroller fabricated in CMOS and is a member of the PCD33XX family. It has an on-chip dual tone multi-frequency (DTMF) generator and other features for application in telephone sets. For further detailed information, see PCD33XX family specification.

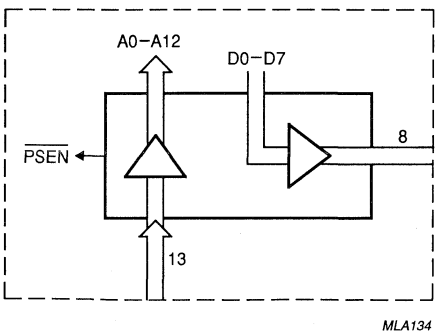
Features

- 8-bit CPU, ROM, RAM, I/O in a single 28-lead DIL or SO package
- 2048 ROM bytes
- 224 RAM bytes
- On-chip DTMF tone generator
- On-chip voltage reference for supply and temperature-independent tone output
- On-chip filtering for low output distortion (CEPT CS203 compatible)
- 20 quasi-bidirectional I/O port lines
- Two test inputs: one of which is also the external interrupt input (CE/$\overline{T0}$)
- Single-level vectored interrupts: external, timer/event counter
- 8-bit programmable timer/event counter
- Over 80 instructions (based on MAB8048)
- All instructions 1 or 2 cycles
- Clock frequency 3,58 MHz
- Single supply voltage from 2,5 V to 6 V
- Low standby voltage and current
- STOP and IDLE mode
- On-chip oscillator
- Configuration of all I/O port lines individually selected by mask: pull-up, open drain or push-pull
- Power-on-reset circuit and low supply voltage detection
- Reset state of all ports individually selected by mask
- Operating temperature range: −25 to + 70 °C

PACKAGE OUTLINES

PCD3344P: 28-lead DIL; plastic (SOT117).
PCD3344T: 28-lead mini-pack; plastic (SO28; SOT136A).





(a)

Fig. 1 PCD3344 block diagram: the function in the dotted outline is replaced as shown in (a) for the PCD3344B 'piggy-back' version.

DEVELOPMENT DATA

PINNING (for normal operation)

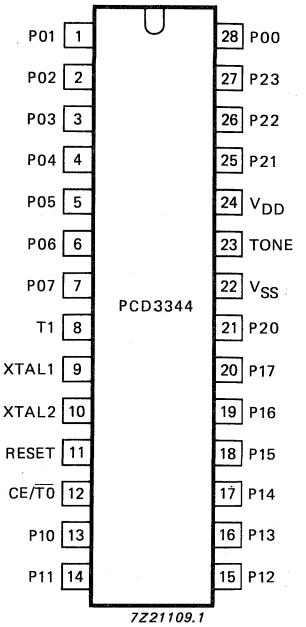


Fig. 2 Pinning diagram.

PIN DESIGNATION

28, 1-7	P00-P07	Port 0: 8-bit quasi-bidirectional I/O port.
8	T1	Test 1: test input, directly tested by conditional branch instructions JT1 and JNT1. T1 also functions as an input to the 8-bit timer/event counter using the STRT CNT instruction.
9	XTAL1	Crystal input: connection to the timing component (crystal) which determines the frequency of the internal oscillator; is also the input for an external clock source.
10	XTAL2	Connection to other side of timing component.
11	RESET	Reset input (active HIGH): used to initialize the processor or output of the power-on-reset circuit.
12	CE/$\overline{T0}$	Interrupt/Test 0: external interrupt input (sensitive to positive-going edge)/test input pin. When used as a test input is directly tested by conditional branch instructions JT0 and JNT0.
13-20	P10-P17	Port 1: 8-bit quasi-bidirectional I/O port.
21, 25-27	P20-P23	Port 2: 4-bit quasi-bidirectional I/O port.
22	V_{SS}	Ground: circuit earth potentional.
23	TONE	Tone output: single or dual tone frequency output with on-chip filtering for low output distortion (CEPT CS203 compatible). This generator is controlled via the internal processor bus.
24	V_{DD}	Power supply: 2,5 to 6 V.

FUNCTIONAL DESCRIPTION

Program memory PCD3344

The program memory comprises 2048 bytes (8-bit words) in a read-only memory (ROM). Each location is directly addressable by the program counter. The memory is mask-programmed at the factory. Figure 3 shows the program memory map.

Three program memory locations are of special importance:

- Location 0; contains the first instruction to be executed after the processor is initialized (RESET),
- Location 3; contains the first byte of an external interrupt service subroutine,
- Location 7; contains the first byte of a timer/event counter interrupt service subroutine.

The program memory is divided into location 'pages', each of 256 bytes. This division applies only for conditional branches. A CALL instruction can transfer control to a subroutine on any 'page'; RET and RETR instructions can transfer control from a subroutine back to the main program.

Data memory PCD3344

Data memory consists of 224 bytes (8-bit words) of random-access data memory (RAM). All locations are indirectly addressable using RAM pointer registers; up to 16 designated locations are directly addressable. Memory also includes an 8-level program counter stack addressed by a 3-bit stack pointer. Figure 4 shows the data memory map.

Working registers

Locations 0 to 7 are designated as working registers, directly addressable by the direct register instructions. Ease of addressing, and a minimum requirement of instruction bytes to manipulate their contents, makes these locations suitable for storing frequently-addressed intermediate results. This bank of registers can be selected by the SEL RB0 instruction.

Executing the select register bank instruction SEL RB1, designates locations 24 to 31 as working registers, instead of locations 0 to 7, and these are then directly addressable. This second bank of working registers may be used as an extension of the first or reserved for use during interrupt service subroutines saving the first bank for use in the main program. If the second bank is not used, locations 24 to 31 may serve as general purpose RAM.

The first locations of each bank contain the RAM pointer registers R0, R1, R0' and R1', which indirectly address all RAM locations.

All RAM locations make efficient program loop counters when used with the decrement register and test instruction DJNZ.

Program counter stack

Locations 8 to 23 may be designated as an 8-level program counter stack (2 locations per level), or as general purpose RAM. The program counter stack (Fig. 5) enables the processor to keep track of the return addresses and status generated by interrupts or CALL instructions by storing the contents of the program counter prior to servicing the subroutine. A 3-bit stack pointer determines which of the eight register pairs of the program counter stack will be loaded with next generated return address.

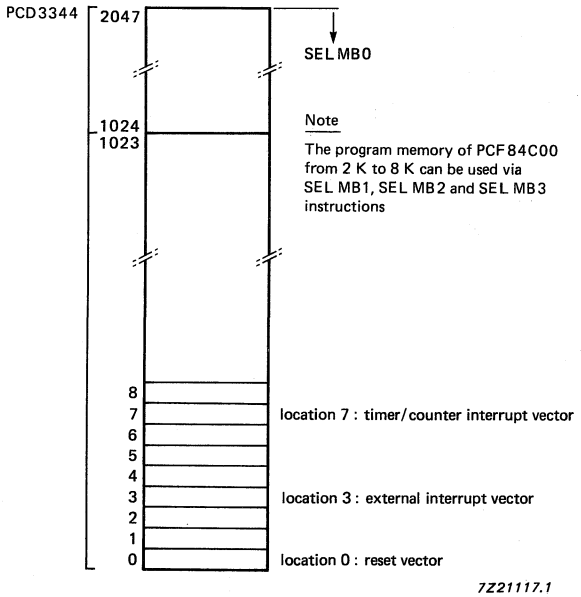


Fig. 3 Program memory map.

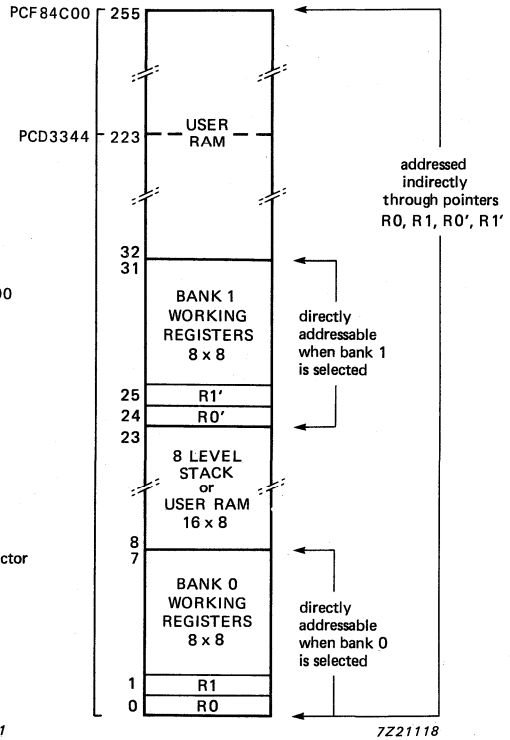


Fig. 4 Data memory map.

The stack pointer, when initialized to 000 by RESET, points to RAM locations 8 and 9. On the first subroutine CALL or interrupt, the contents of the program counter and bits 4, 6 and 7 of the program status word (PSW) are transferred to locations 8 and 9. The stack pointer increments by one and points to locations 10 and 11 ready for another CALL. Because an address may be up to 13 bits long, two bytes must be used to store each address.

At the end of a subroutine, which is signalled by a return instruction (RET or RETR), the stack pointer decrements by one and the contents of the register pair on top of the stack are transferred to the program counter. The saved PSW bits are transferred to the PSW only by the RETR instruction.

If not all 8 levels of subroutine and interrupt nesting are used, the unused portion of the stack may be used as any other indirectly addressable RAM locations. Possible locations from 32 to 223 may be used for storage of program variables or data.

Nesting of subroutines within subroutines can continue up to 8 times without overflowing the stack. If overflow does occur the deepest address stored (locations 8 and 9) will be overwritten and lost since the stack pointer overflows from 111 to 000. It also underflows from 000 to 111.

The value of the saved contents of the program counter is different for an interrupt CALL compared to a normal CALL to subroutine. With an interrupt CALL, the program counter return address is saved; with a subroutine CALL, the saved program counter value is one less than the program counter return address.

FUNCTIONAL DESCRIPTION (continued)

Program counter stack (continued)

stack pointer									
1 1 1	----------								R23/22
1 1 0	----------								R21/20
1 0 1	----------								R19/18
1 0 0	----------								R17/16
0 1 1	----------								R15/14
0 1 0	----------								R13/12
0 0 1	----------								R11/10
0 0 0	PSW7	PSW6	PC12	PSW4	PC11	PC10	PC9	PC8	R9
	PC7	PC6	PC5	PC4	PC3	PC2	PC1	PC0	R8

Fig. 5 Program counter stack.

IDLE and STOP modes

IDLE mode

When the microcontroller enters the IDLE mode via the IDLE instruction (H'01') the oscillator and timer/counter are kept running. The microcontroller exits from the IDLE mode by one of two interrupts if they are enabled or by activating a RESET. If the interrupt is not enabled the processor will remain in the IDLE mode. An active signal on the RESET pin restarts the microcontroller and a normal RESET sequence is executed (see Fig. 6).

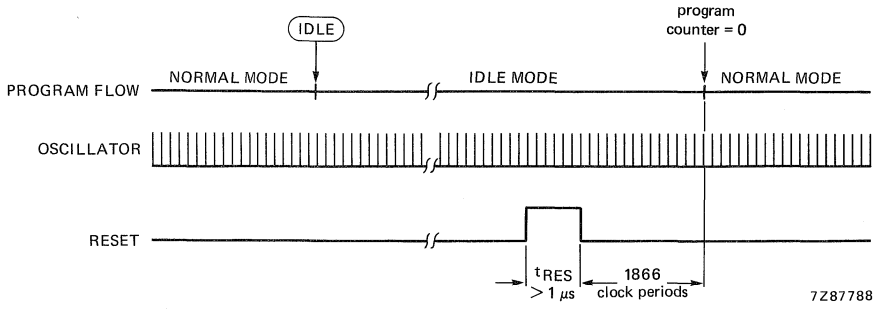


Fig. 6 Exit from IDLE mode via a RESET.

An active signal coming from an enabled interrupt causes the execution of the normal interrupt routine since normal interrupt scanning is still being carried out. A LOW-to-HIGH transition on the external interrupt pin (CE/$\overline{T0}$) reactivates the microcontroller. A HIGH level applied to CE/$\overline{T0}$ will reactivate the microcontroller only in the STOP mode. Thus, if CE/$\overline{T0}$ was HIGH before the microcontroller entered the IDLE mode, it must go LOW before the microcontroller can be reactivated (see Fig. 7).

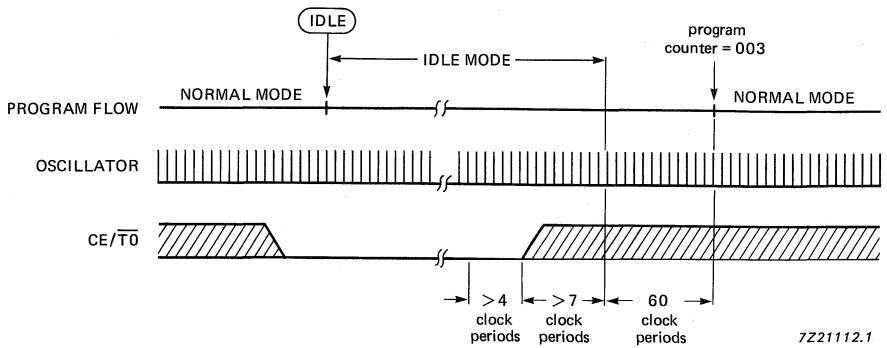


Fig. 7 Exit from IDLE mode via an interrupt.

Wake-up from the IDLE mode is ensured when CE/$\overline{T0}$ is LOW for 4 CP (clock periods) followed by a HIGH for 7 CP. After the initial forced CALL H'003' operation (60 CP) the program continues with the external interrupt service routine.

STOP mode

The microcontroller enters the STOP mode by the STOP instruction (H'22'). The oscillator is switched off. The internal status of the CPU, RAM contents and the state of I/O ports are not affected. The microcontroller can be brought-out of the STOP mode by an active signal at the external interrupt input or by an external RESET signal. When one of these two signals is applied an internal delay of 1866 CP is provided to ensure that all internal clocks are operating correctly before restart (see Fig. 8).

If the microcontroller exits from the STOP mode by activating RESET, a normal RESET sequence is executed.

If the microcontroller exits the STOP mode by pulling the external interrupt input pin HIGH, an interrupt sequence is executed only if the external interrupt is enabled. In this event the microcontroller resumes the normal program sequence after returning from the interrupt routine, as in the normal mode. If the interrupt is not enabled, it continues the normal program sequence, executing the instruction following the STOP instruction.

The microcontroller is restarted by a HIGH level applied at the CE/$\overline{T0}$ pin, and not by a LOW-to-HIGH transition as in a normal interrupt mechanism.

When the CE/$\overline{T0}$ level is active during the STOP instruction then no STOP is executed.

A HIGH level on the external interrupt input of at least 1 μs will cause the microcontroller to exit the STOP mode.

DEVELOPMENT DATA

FUNCTIONAL DESCRIPTION (continued)

STOP mode (continued)

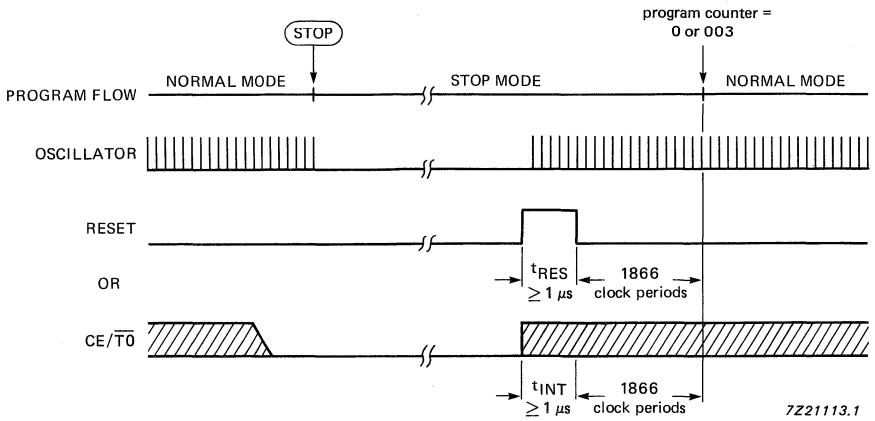


Fig. 8 Entering and exiting the STOP mode.

Tone output (DTMF mode)

Control of the sinewave generator

The on-chip sinewave oscillator is controlled by the 'derivative' registers Dx (x = H'O' to 'FF'). The instruction that controls the derivative registers is shown in Table 1.

Table 1 Derivative register control

mnemonic	opcode	description	function
MOV Dx,A	8D Dx	move accumulator contents to derivative register	(Dx) ← (A)

The instruction is 2 cycles/2 bytes. The second byte selects the derivative register to be addressed (H'O' to 'FF'). Register H'01' is for control of HIGH group frequencies, and register H'02' for control of LOW group frequencies. Thus data transport from accumulator to derivative register D01 is done by the 2-byte opcode 8D,01.

Generation of frequencies

The single and dual tones at the tone output are filtered by an on-chip switched-capacitor filter followed by an on-chip active RC low-pass filter. These ensure that the total harmonic distortion of the DTMF tones fulfil the CEPT CS 203 recommendations. An on-chip reference voltage provides output tone levels that are independent of the supply voltage.

The output frequency can be calculated as follows:

$$f_{out} = \frac{f_{XTAL}}{23\,(x+2)} \quad \text{Hz}$$

x = 60 to 255 and is the decimal value of the appropriate ROM-code (see Table 2)

Table 2 ROM-codes for DTMF applications

telephone keyboard symbol	contents of low register (hex)	contents of high register (hex)
0	A3	72
1	DD	7F
2	DD	72
3	DD	67
4	C8	7F
5	C8	72
6	C8	67
7	B5	7F
8	B5	72
9	B5	67
A	DD	5D
B	C8	5D
C	B5	5D
D	A3	5D
*	A3	7F
#	A3	67

DTMF generation is stopped by loading H'00' into both derivative registers.

I/O facilities

The PCD3344 family has 22 I/O lines arranged as:

- Port 0 parallel port of 8 lines (P00 to P07)
- Port 1 parallel port of 8 lines (P10 to P17)
- Port 2 parallel port of 4 lines (P20 to P23)
- CE/$\overline{T0}$ external interrupt and test input. When used as a test input it can be directly tested by conditional branch instructions JT0 and JNT0.
- T1 test input which can alter program sequences when tested by conditional jump instructions JT1 and JNT1. T1 also functions as an input to the 8-bit timer/event counter.

DEVELOPMENT DATA

FUNCTIONAL DESCRIPTION (continued)

Parallel ports

All parallel ports can be used as outputs or inputs, their structure is quasi-bidirectional. Output data written to a port is latched and remains unchanged until rewritten. Input data is not latched and so must be present until read by an input instruction.

Input lines are fully CMOS compatible, output lines can drive one LS-TTL or CMOS load.

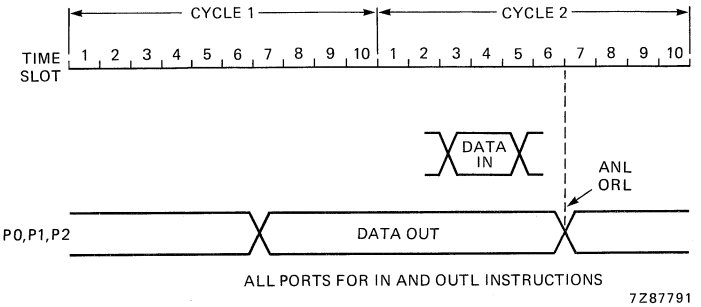


Fig. 9 Timing diagram of all ports on IN and OUTL instructions; for ANL and ORL instructions, the ports change on the time slot 7 of cycle 2.

Fig. 10 shows the quasi-bidirectional I/O interface with push-pull output and switched pull-up current source. Each line is pulled up to V_{DD} via a constant current source (TR4), which is enabled via TR3 whenever one of the two output latches contains a logic 1. This current source provides sufficient current for a TTL HIGH level, yet can be pulled LOW by an external CMOS device, thus allowing the same pin to be used for both input and output.

When a logic 1 is written to the line for the first time ($\overline{MQ}$ = 1, $\overline{SQ}$ = 0), TR2 is switched on for the duration of the internal write pulse (one oscillator period) to provide a fast transition from logic 0 to logic 1. Subsequent writing of a logic 1 to the port lines will not switch TR2 on. This prevents unnecessary current through external components connected to the port lines of the same port which might be in the input mode and also connected to ground.

When a logic 0 is written to the line, TR3 switches off the current source. Current sinking capability is provided by TR1, which is now switched on. When used as an input, a logic 1 must first be written to the line, otherwise TR1 will remain low impedance.

In telephone applications this switched pull-up source may not be sufficient. Therefore the PCD3344 offers the possibility to select individually the 20 parallel port pins by the following mask options:

Option 1 –STANDARD PORT; quasi-bidirectional I/O with switched pull-up current source of 100 μA (typ.) and P-channel booster transistor TR2 (1,5 mA). TR2 is active only during 1 clock cycle (0,28 μs at 3,58 MHz).

Option 2 –OPEN DRAIN; quasi-bidirectional I/O with only an N-channel open drain output. Application as an output requires connection of an external pull-up resistor (Fig. 11).

Option 3 –PUSH-PULL OUTPUT; drive capability of the output will be 1,5 mA (typ.) at V_{DD} = 3 V in both polarities. To avoid a large current flowing through the output transistors during the input mode, these push-pull pins must be used only as outputs (Fig. 12).

Also, individual mask selection of the RESET state of these port pins can be achieved by appending the following options S and R to options 1, 2 or 3.

Option S-SET; after RESET this pin will be initialized to HIGH

Option R-RESET; after RESET this pin will be initialized to LOW.

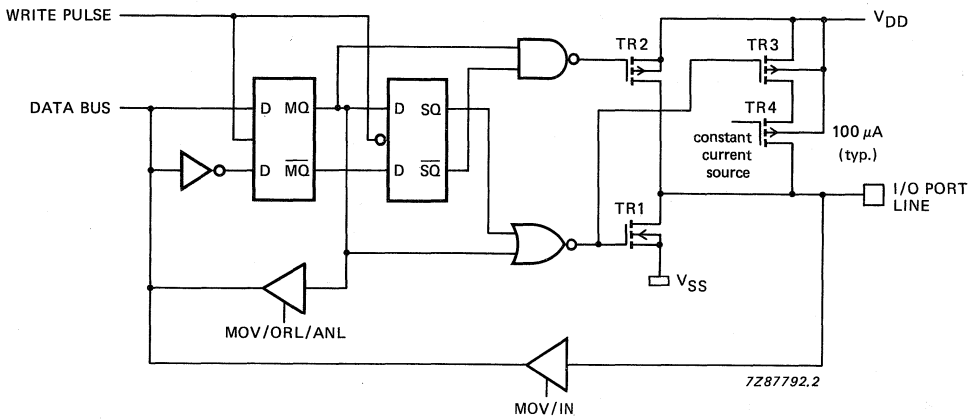


Fig. 10 Standard output with switched pull-up current source.

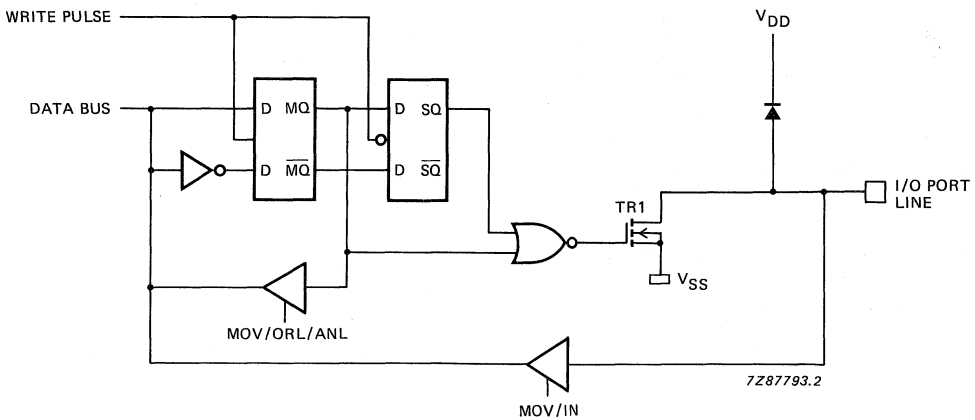


Fig. 11 Open drain output.

DEVELOPMENT DATA

FUNCTIONAL DESCRIPTION (continued)

Parallel ports (continued)

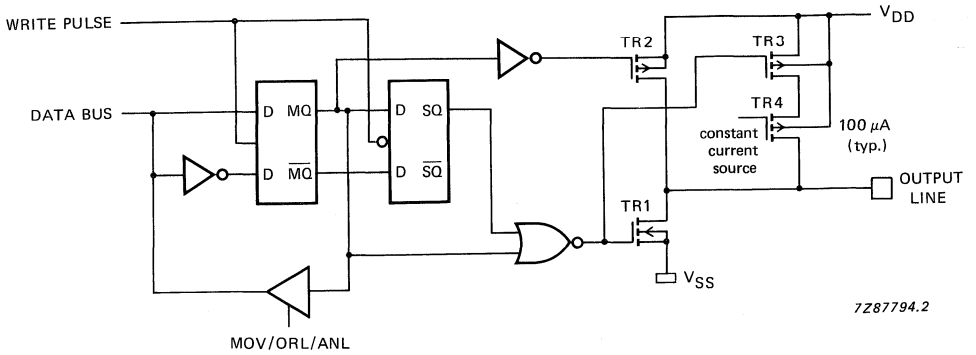


Fig. 12 Push-pull output.

Interrupts (see Fig. 13(a) and Fig. 13(b))

When an interrupt routine is entered, the contents of the program counter and bits 4, 6 and 7 of the PSW are saved in the program counter stack. The contents of the accumulator can only be saved by user software. Interrupt acknowledgement can be carried out by software via I/O ports. All interrupt routines must reside in memory bank 0; the SEL MB1, SEL MB2 and SEL MB3 instructions may not be used in an interrupt routine. An interrupt routine can only be terminated by the RETR (return and restore) instruction. During an interrupt routine, subroutine calls must be terminated by the RET instruction. Using the RETR instruction to terminate a subroutine called in an interrupt routine would terminate the interrupt routine prematurely and result in a wrong return address.

1. **External interrupt**

When the external interrupt is enabled, a HIGH-to-LOW transition on the CE/$\overline{TO}$ input initiates an external interrupt routine which forces a call to program memory location 3. The program counter points to the external interrupt vector address (003 H) between 2,6 and 3,6 machine cycles after the transition occurs. Interrupt latency depends on the instruction that is being executed when the transition occurs. External interrupts are latched in the External Interrupt Flag (EIF) even when they are not enabled. Execution of a DIS I instruction clears previously latched interrupts, the digital filter latch and the external interrupt flag.

2. **Timer/counter interrupt**

When the timer interrupt is enabled, a timer/counter overflow sets the Timer Interrupt Flag (TIF) and forces a CALL to location 7. The timer interrupts are only latched when they are enabled. The timer flag is set every time the timer/counter overflows and is not automatically reset when the timer/counter interrupt routine is called. It can only be cleared by the JTF and JNTF instructions or by a hardware RESET.

3. Simultaneous interrupts

If simultaneous interrupts occur their priority is as follows:

external (highest);
timer/counter (lowest).

An interrupt routine can only be interrupted by a hardware RESET and cannot be interrupted by other interrupts (which will be latched if enabled). When the interrupt routine is terminated by the RETR instruction, at least one instruction of the main program will be executed before another interrupt routine is entered.

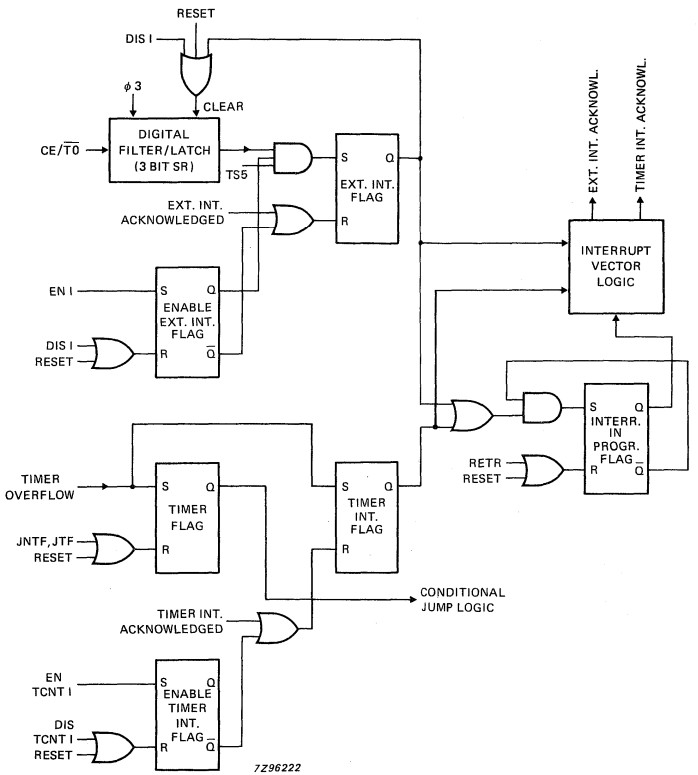


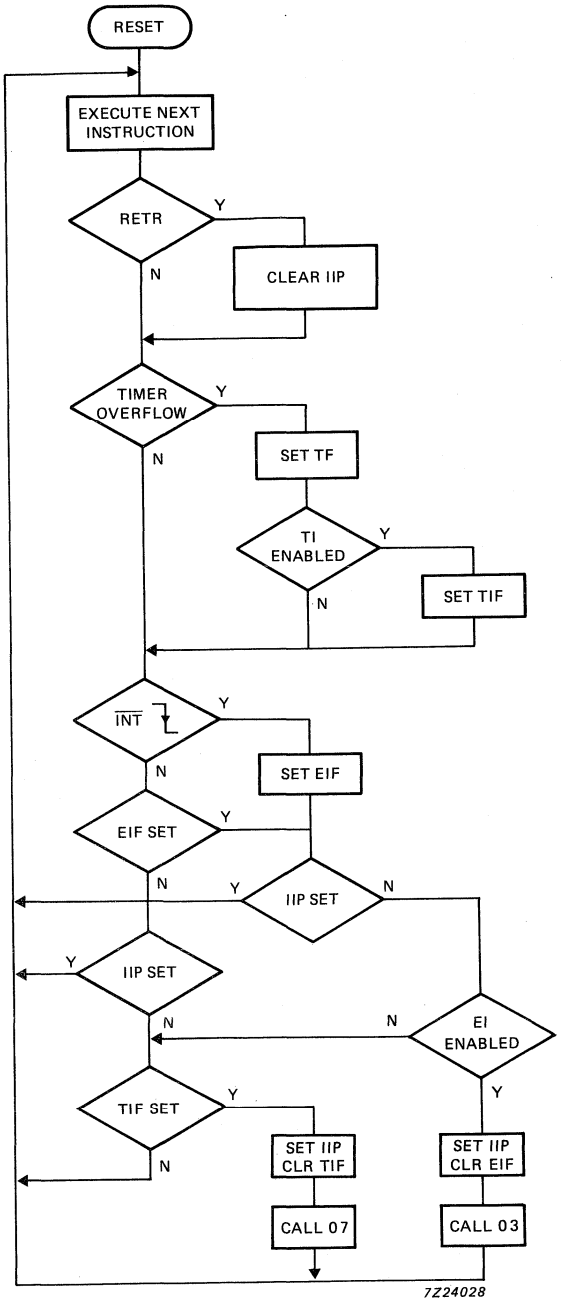
Fig. 13(a) Interrupt logic.

Notes to figure 13(a)

1. CE/$\overline{T0}$ positive edge is always latched in the digital filter/latch.
2. Correct interrupt timing is ensured when CE/$\overline{T0}$ is LOW for $>$ 4 CP followed by a HIGH for $>$ 7 CP.
3. When the interrupt in progress flag is set, further interrupts are latched but ignored, until RETR is executed.
4. A DIS I instruction always clears a pending external interrupt.
5. For all flip-flops, RESET overrules SET.

FUNCTIONAL DESCRIPTION (continued)

Interrupts (continued)



EI External Interrupt
TI Timer/counter Interrupt
EIF External Interrupt Flag
TF Timer Flag
IIP Interrupt In Progress flag

Fig. 13(b) Interrupt flowchart.

Oscillator (see Fig. 14)

The 3,58 MHz oscillator can be inhibited by the STOP instruction under software control. It is also inhibited when a low-voltage condition is present to prevent discharge of a weak back-up battery.

Provided the supply voltage is within the operating range, the oscillator will be restarted after a STOP instruction by a HIGH level at either the CE/$\overline{T0}$ or RESET pin.

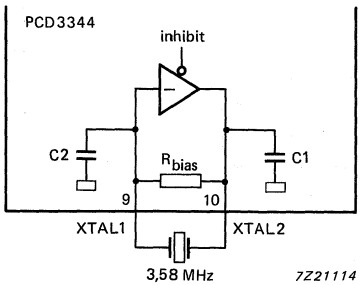


Fig. 14 Oscillator with integrated elements.

The oscillator has an output drive capability from pin 10 (XTAL2). An external clock can be applied to pin 9 (XTAL1). A machine cycle comprises 10 time slots, each time slot being 3 oscillator periods.

In telephony applications the 3,58 MHz crystal provides an 8,4 μs machine cycle. The range of the clock frequency is from 100 kHz up to a maximum which is a function of the supply voltage.

Timer/event counter (see Fig. 15)

An internal 8-bit up-counter is provided. This can count external events, modulo-32 machine cycles, or machine cycles directly. Table 3 gives the instructions that control the counter and the prescaler, and the functions performed.

When used as a timer, the input to the counter is either the overflow or input of a 5-bit prescaler. When used as an event counter, LOW-to-HIGH transitions on pin 8 (T1) are counted. The maximum rate at which the counter may be incremented is once every machine cycle (182,6 kHz for an 8,4 μs machine cycle). When the counter overflows, the timer flag is set. The flag can be tested and reset using the JTF (jump if timer flag = 1) or JNTF instruction. Overflow also generates an interrupt to the processor via setting of the Timer Interrupt Flag when the timer/event counter interrupt is enabled.

FUNCTIONAL DESCRIPTION (continued)

Timer/event counter (continued)

Table 3 Timer/event counter control

function	timer mode modulo-1, modulo-32*	counter mode
CLEAR	MOV T,A (A) = 0 or RESET	MOV T,A (A) = 0 or RESET
PRESET	MOV T,A	MOV T,A
START	STRT T	STRT CNT
STOP	STOP TCNT or RESET	STOP TCNT or RESET
TEST	JTF/JNTF	JTF/JNTF
READ**	MOV A,T	MOV A,T

* With prescaler select, PS = 0, the timer counts modulo-32 machine cycles, with PS = 1 it counts modulo-1 cycles (prescaler not used); prescaler cleared with STRT T, prescaler not readable.

** READ does not disturb the counting process.

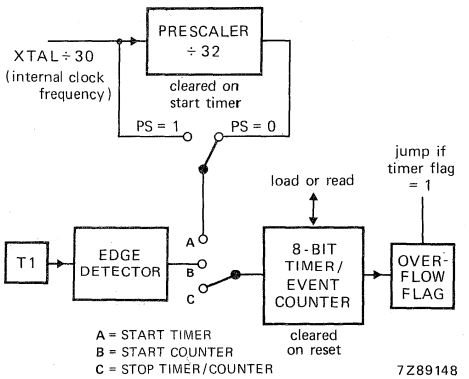


Fig. 15 Timer/event counter.

Program status word (see Fig. 16)

The program status word (PSW) is an 8-bit word (1 byte) in the CPU which stores information about the current status of the microcontroller.
The PSW bits are:

- Bits 0 to 2 — stack pointer bits (SP_0, SP_1, SP_2)
- Bit 3 — prescaler select (PS);
  0 = modulo-32; 1 = modulo-1 (no prescaling)
- Bit 4 — working register bank select (RBS);
  0 = register bank 0; 1 = register bank 1
- Bit 5 — not used (1)
- Bit 6 — auxiliary carry (AC); half-carry bit generated by an ADD instruction and used by the decimal adjust instruction DA A
- Bit 7 — carry (CY); the carry flag indicates that previous operation has resulted in an overflow of the accumulator.

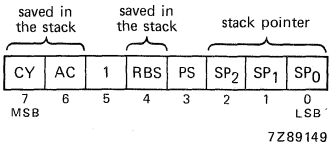


Fig. 16 Program status word.

All bits can be read using the MOV A, PSW instruction. Bits 7 and 6 are set and cleared by CPU operation. Bit 4 can be changed by a SEL RB instruction, bit 3 by the MOV PSW, A instruction, and bits 0, 1 and 2 by the CALL, RET or RETR instructions and in the event of an interrupt. Bits 7, 6 and 4 are stored in the program counter stack during subroutine and interrupt calls. These bits are restored in the PSW with a RETR (return and restore) instruction which must be used at the end of an interrupt and can be used at the end of a normal subroutine. The RET instruction has no restore feature and cannot be used at the end of an interrupt.

Program counter (see Fig. 17).

A 12-bit program counter is used to facilitate 8 K bytes of ROM being addressed. The arrangement of the bits is shown in Figure 17. During an interrupt subroutine PC_{11} and PC_{12} are forced to logic 0. All 12 bits are saved in the stack during CALL and interrupt routines.

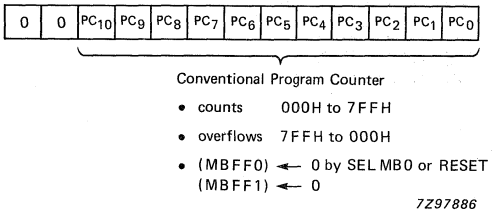


Fig. 17 Program counter.

Central processing unit

The PCD3344 has arithmetic, logical and branching capabilities. The DA A, SWAP A and XCHD instructions simplify BCD arithmetic and the handling of nibbles. The MOVP A,@A instruction permits efficient table look-up from the CURRENT ROM page.

FUNCTIONAL DESCRIPTION (continued)

Conditional branch logic

The conditional branch logic within the processor enables several conditions, internal and external to the processor, to be tested by the user's program.
Table 4 lists the conditional jump instructions used to change the program sequence. The DJNZ instruction decrements a designated register or data memory location and branches if the contents are not zero. This instruction is useful for looping control. The JMPP@A instruction allows multiway branches to destinations determined by the contents of the accumulator.

Table 4 Conditional branches

test	jump condition	jump instruction
accumulator	all bits zero	JZ
	any bit non-zero	JNZ
accumulator bit test	1	JB0 to JB7
carry flag	1	JC
	0	JNC
timer overflow flag	1	JTF
	0	JNTF
test input T0	1	JNT0
	0	JT0*
test input T1	1	JT1
	0	JNT1
register	non-zero	DJNZ

* Because of the inverted interrupt input CE/$\overline{T0}$ the conditional jump JT0 is also inverted.

Test input T1 (pin 8)

The T1 input line can be used as:

- A test input for branch instructions JT1 and JNT1
- An external input to the event counter

When used as a test input:

- JT1 instruction tests for logic 1 level
- JNT1 instruction tests for logic 0 level

When used as an input to the event counter, T1 must be LOW for $>$ 4 CP, followed by a HIGH for $>$ 4 CP. The transition can be recognized with a repetition rate of once per 30 oscillator clock periods (1 machine cycle).

There is no internal pull-up or pull-down resistor connected to the T1 input. If required it must be externally connected to a resistor ($R = \leqslant 100\ k\Omega$).
When T1 is not used pin 8 must be connected to V_{DD} or V_{SS}.

Reset (pin 11)

A positive-going signal on the RESET input/output:

- Sets the program counter to zero
- Selects location 0 of memory bank 0 and register bank 0
- Sets the stack pointer to zero (000); pointing to RAM address 8
- Disables the interrupts (external and timer)
- Stops the timer/event counter, then sets it to zero
- Sets the timer prescaler to modulo-32
- Resets the timer flag
- Sets all ports according to reset states
- Cancels IDLE and STOP mode

After the voltage is applied to RESET an internal delay of 1866 CP is introduced before the microcontroller commences operation.

Power-on reset

The internal power-on reset circuit monitors the supply voltage V_{DD}. As long as V_{DD} remains below the internal reference level V_{ref} (typically 1,3 V), the oscillator is inhibited and RESET (pin 11) has an undefined level. When V_{DD} rises above V_{ref}, the oscillator is released and RESET is pulled HIGH to V_{DD} by TR1 for a period t_D (typically 50 μs). Note that the start-up time of the oscillator is typically 10 ms because of the narrow bandwidth of the crystal.

Three modes of power-on reset are possible:

1. If V_{DD} has a fast rise time, i.e. V_{DD} reaches its minimum value before the RESET signal finishes (t_D), then no additional circuit is required (see Figs 18 and 19). Note that the first instruction is executed after the oscillator start-up time plus 1866 clock periods.
2. If V_{DD} has a slow rise time then the RESET signal should be stretched by an external CR circuit (see Figs 20 and 21). In the event of a short drop in V_{DD}, the diode path discharges the capacitor rapidly to ensure a reliable power-on reset. The RESET signal should reach at least 70% of the final value of V_{DD} to ensure a correct reset. Given that the RESET voltage and V_{DD} rise exponentially, the above requirement is satisfied when the time constant of the RESET pulse is > 8 times the time constant of V_{DD}. If V_{DD} rises linearly then a RESET time constant > 2 times the rise time of V_{DD} is required.

   When a reset is completed (RESET goes LOW) before the oscillator has started up, program execution begins after the oscillator start-up time plus 1866 clock periods (see Fig. 21). If the oscillator starts up prior to the completion of RESET then program execution begins 1866 clock periods after RESET goes LOW.
3. Fig. 22 shows an external reset applied during power-on. The external reset signal must remain HIGH until V_{DD} has reached its minimum operating value corresponding to the selected oscillator frequency. When a reset is completed (RESET goes LOW) before the oscillator has started up, program execution begins after the oscillator start-up time plus 1866 clock periods (see Fig. 23). If the oscillator starts up prior to the completion of RESET then program execution begins 1866 clock periods after RESET goes LOW.

DEVELOPMENT DATA

FUNCTIONAL DESCRIPTION (continued)

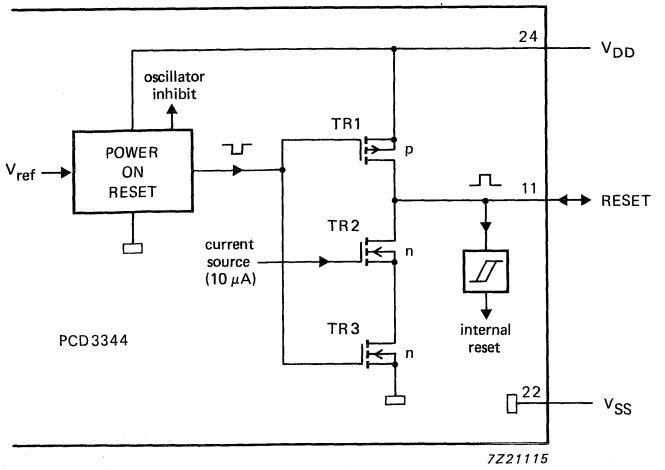


Fig. 18 Power-on reset configuration.

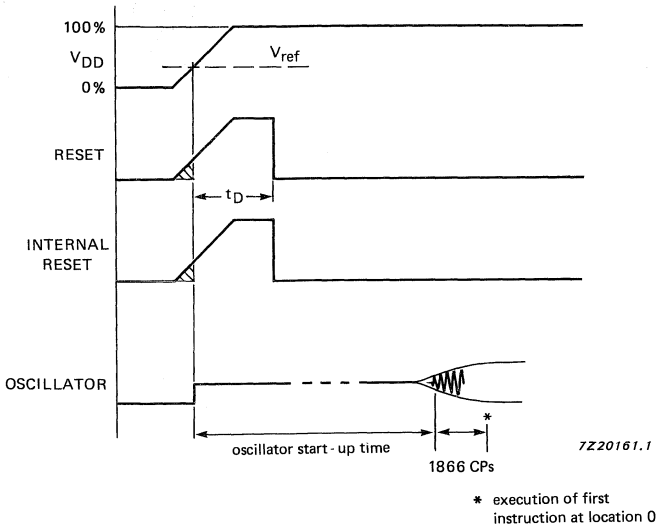


Fig. 19 Timing of power-on reset with fast rise time of V_{DD}.

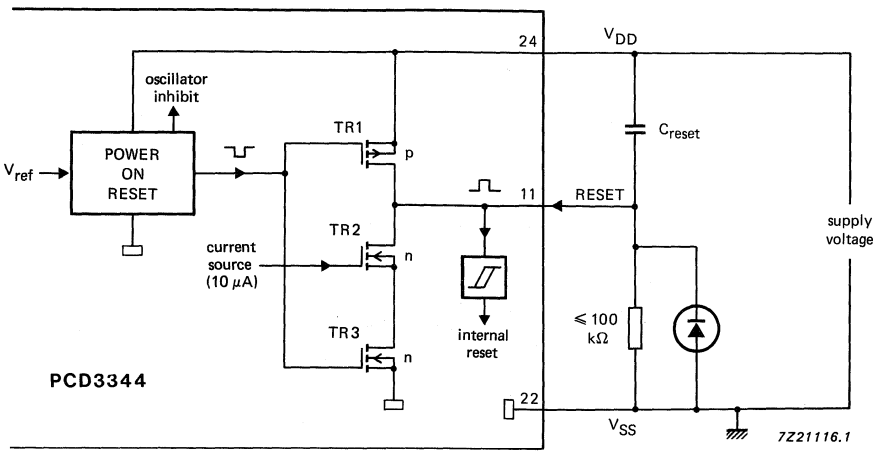


Fig. 20 Stretched power-on reset with external CR circuit.

DEVELOPMENT DATA

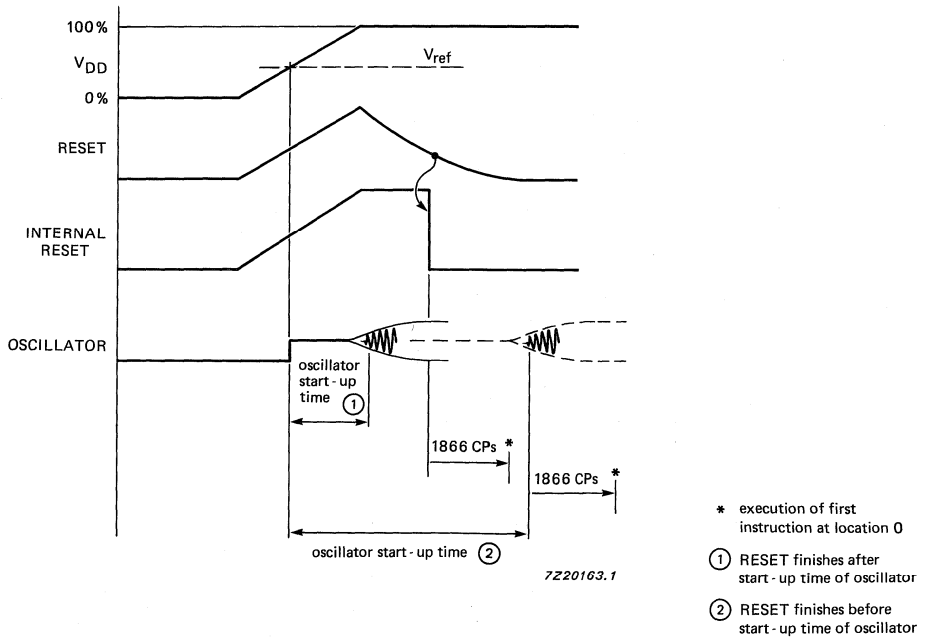


Fig. 21 Timing of power-on reset with a slowly rising V_{DD} and a stretched RESET pulse.

FUNCTIONAL DESCRIPTION (continued)

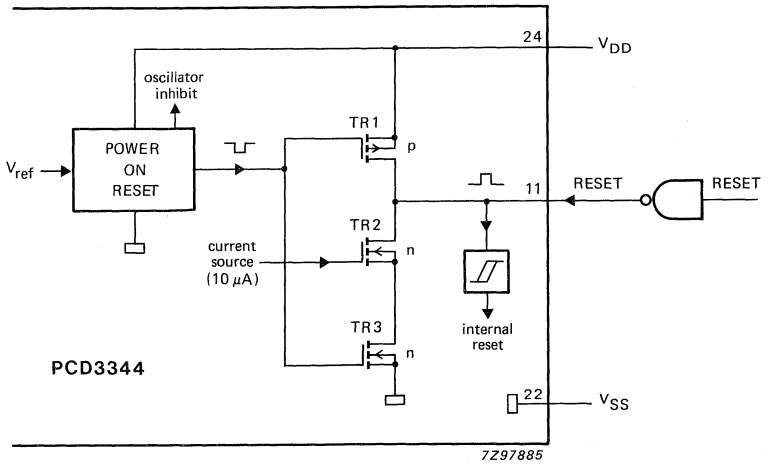


Fig. 22 External power-on reset configuration.

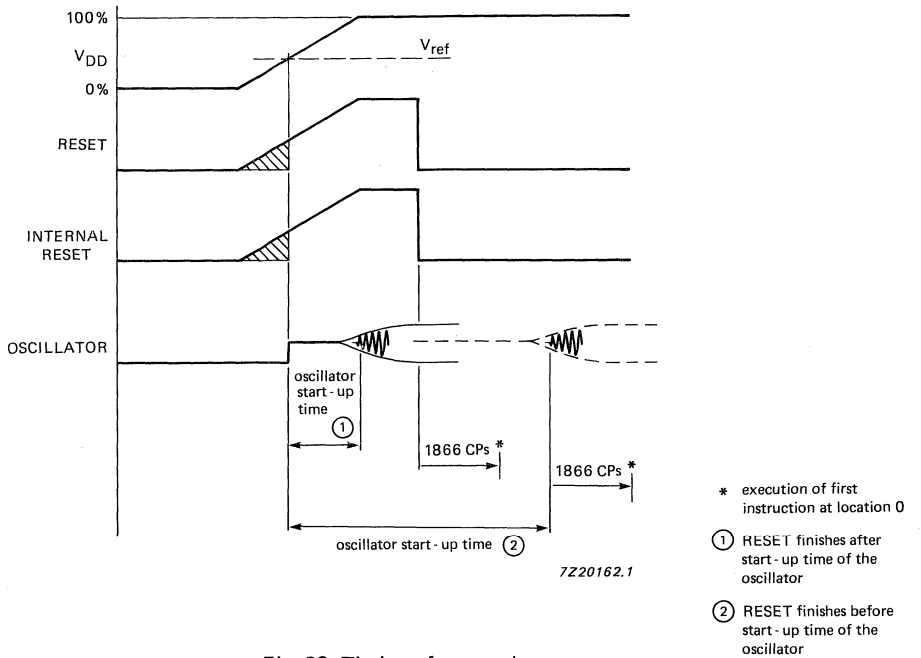


Fig. 23 Timing of external power-on reset.

INSTRUCTION SET

The PCD3344 instruction set consists of over 80 one and two byte instructions and is based on the MAB8048 instruction set. New instructions include those for memory bank selection and derivative control. Program code efficiency is high because all RAM locations and all ROM locations on a 256 byte page require only a single byte address.

Table 5 details the symbols and definition descriptions that are used in the instruction set of the PCD3344. Table 6 shows the instruction map and Table 7 gives the instruction set.

Table 5 Symbols and definitions used in Tables 6 and 7

symbol	definition description
A	accumulator
addr	program memory address
Bb	bit designation (b = 0 to 7)
RBS	register bank select
C	carry bit (bit CY)
CNT	event counter
D	mnemonic for 4-bit digit (nibble)
data	8-bit number or expression
I	interrupt
MB	memory bank
MBFF	memory bank flip-flop
P	mnemonic for 'in page' operation
PC	program counter
Pp	port designation (p = 0, 1 or 2)
PSW	program status word
RB	register bank
Rr	register designation (r = 0 to 7)
SP	stack pointer
T	timer
TF	timer flag
T0, T1	test 0 and 1 inputs
#	immediate data prefix
@	indirect address prefix
(X)	contents of X
((X))	contents of location addressed by X
Dx	derivative register (0 to H'FF')
←	is replaced by
↔	is exchanged with

DEVELOPMENT DATA

INSTRUCTION SET (continued)

Table 6 PCD3344 instruction map

first hexadecimal character of opcode ↓

second hexadecimal character of opcode →

	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
0	NOP	IDLE		ADD A, # data	JMP page 0	EN I	JNTF addr	DEC A	IN A,Pp 0	1	2					
1	INC @Rr 0	1	JB0 addr	ADDC A, # data	CALL page 0	DIS I	JTF addr	INC A	INC Rr 0	1	2	3	4	5	6	7
2	XCH A,@Rr 0	1	STOP	MOV A, # data	JMP page 1	EN TCNTI	JNT0 addr	CLR A	XCH A,Rr 0	1	2	3	4	5	6	7
3	XCHD A,@Rr 0	1	JB1 addr		CALL page 1	DIS TCNTI	JT0 addr	CPL A	OUTL Pp,A 0	1	2					
4	ORL A,@Rr 0	1	MOV A , T	ORL A, # data	JMP page 2	STRT CNT	JNT1 addr	SWAP A	ORL A,Rr 0	1	2	3	4	5	6	7
5	ANL A,@Rr 0	1	JB2 addr	ANL A, # data	CALL page 2	STRT T	JT1 addr	DA A	ANL A,Rr 0	1	2	3	4	5	6	7
6	ADD A,@Rr 0	1	MOV T , A		JMP page 3	STOP TCNT		RRC A	ADD A,Rr 0	1	2	3	4	5	6	7
7	ADDC A,@Rr 0	1	JB3 addr		CALL page 3			RR A	ADDC A,Rr 0	1	2	3	4	5	6	7
8				RET	JMP page 4				ORL Pp, # data 0	1	2			MOV Dx,A		
9			JB4 addr	RETR	CALL page 4		JNZ addr	CLR C	ANL Pp, # data 0	1	2					
A	MOV @Rr, A 0	1		MOVP A,@A	JMP page 5	SEL MB2		CPL C	MOV Rr,A 0	1	2	3	4	5	6	7
B	MOV @Rr, # data 0	1	JB5 addr	JMPP @A	CALL page 5	SEL MB3			MOV Rr, # data 0	1	2	3	4	5	6	7
C	DEC @Rr 0	1			JMP page 6	SEL RB0	JZ addr	MOV A, PSW	DEC Rr 0	1	2	3	4	5	6	7
D	XRL A,@Rr 0	1	JB6 addr	XRL A, # data	CALL page 6	SEL RB1		MOV PSW, A	XRL A,Rr 0	1	2	3	4	5	6	7
E	DJNZ @Rr,addr 0	1			JMP page 7	SEL MB0	JNC addr	RL A	DJNZ Rr, addr 0	1	2	3	4	5	6	7
F	MOV A,@Rr 0	1	JB7 addr		CALL page 7	SEL MB1	JC addr	RLC A	MOV A,Rr 0	1	2	3	4	5	6	7

DEVELOPMENT DATA

Table 7 Instruction set

	mnemonic	opcode (hex.)	bytes/ cycles	description	function		notes
ACCUMULATOR	ADD A, Rr	6*	1/1	Add register contents to A	(A)←(A) + (Rr)	r = 0–7	1
	ADD A, @Rr	60 61	1/1	Add RAM data, addressed by Rr, to A	(A)←(A) + ((R0)) (A)←(A) + ((R1))		1
	ADD A, #data	03 data	2/2	Add immediate data to A	(A)←(A) + data		1
	ADDC A, Rr	7*	1/1	Add carry and register contents to A	(A)←(A) + (Rr) + (C)	r = 0–7	1
	ADDC A, @Rr	70 71	1/1	Add carry and RAM data, addressed by Rr, to A	(A)←(A) + ((R0)) + (C) (A)←(A) + ((R1)) + (C)		1
	ADDC A, #data	13 data	2/2	Add carry and immediate data to A	(A)←(A) + data + (C)		1
	ANL A, Rr	5*	1/1	'AND' Rr with A	(A)←(A) AND (Rr)	r = 0–7	
	ANL A, @Rr	50 51	1/1	'AND' RAM data, addressed by Rr, with A	(A)←(A) AND ((R0)) (A)←(A) AND ((R1))		
	ANL A, #data	53 data	2/2	'AND' immediate data with A	(A)←(A) AND data		
	ORL A, Rr	4*	1/1	'OR' Rr with A	(A)←(A) OR (Rr)	r = 0–7	
	ORL A, @Rr	40 41	1/1	'OR' RAM data, addressed by Rr, with A	(A)←(A) OR ((R0)) (A)←(A) OR ((R1))		
	ORL A, #data	43 data	2/2	'OR' immediate data with A	(A)←(A) OR data		
	XRL A, Rr	D*	1/1	'XOR' Rr with A	(A)←(A) XOR (Rr)	r = 0–7	
	XRL A, @Rr	D0 D1	1/1	'XOR' RAM, addressed by Rr, with A	(A)←(A) XOR ((R0)) (A)←(A) XOR ((R1))		
	XRL A, #data	D3 data	2/2	'XOR' immediate data with A	(A)←(A) XOR data		
	INC A	17	1/1	increment A by 1	(A)←(A) + 1		
	DEC A	07	1/1	decrement A by 1	(A)←(A) −1		
	CLR A	27	1/1	clear A to zero	(A)←0		
	CPL A	37	1/1	one's complement A	(A)←NOT(A)		
	RL A	E7	1/1	rotate A left	$(A_{n+1})\leftarrow(A_n)$ $(A_0)\leftarrow(A_7)$	n = 0–6	

INSTRUCTION SET (continued)

ACCUMULATOR (cont.)	RLC A	F7	1/1	rotate A left through carry	$(A_{n+1})\leftarrow A_n$ $(A_0)\leftarrow(C)$, $(C)\leftarrow(A_7)$	n = 0–6	2
	RR A	77	1/1	rotate A right	$(A_n)\leftarrow(A_{n+1})$ $(A_7)\leftarrow(A_0)$	n = 0–6	
	RRC A	67	1/1	rotate A right through carry	$(A_n)\leftarrow(A_{n+1})$ $(A_7)\leftarrow(C)$, $(C)\leftarrow(A_0)$	n = 0–6	2
	DA A	57	1/1	decimal adjust A			2
	SWAP A	47	1/1	swap nibbles of A	$(A_{4-7})\leftrightarrow(A_{0-3})$		
DATA MOVES	MOV A, Rr	F*	1/1	move register contents to A	(A)←(Rr)	r = 0–7	
	MOV A, @Rr	F0 F1	1/1	move RAM data, addressed by Rr, to A	(A)←((R0)) (A)←((R1))		
	MOV A, #data	23 data	2/2	move immediate data to A	(A)←data		
	MOV Rr, A	A*	1/1	move accumulator contents to register	(Rr)←(A)	r = 0–7	
	MOV @Rr, A	A0 A1	1/1	move accumulator contents to RAM location addressed by Rr	((R0))←(A) ((R1))←(A)		
	MOV Rr, #data	B* data	2/2	move immediate data to Rr	(Rr)←data	r = 0–7	
	MOV @Rr, #data	B0 data B1 data	2/2	move immediate data to RAM location addressed by Rr	((R0))←data ((R1))←data		
	XCH A, Rr	2*	1/1	exchange accumulator contents with Rr	(A)↔(Rr)	r = 0–7	
	XCH A, @Rr	20 21	1/1	exchange accumulator contents with RAM data addressed by Rr	(A)↔((R0)) (A)↔((R1))		
	XCHD A, @Rr	30 31	1/1	exchange lower nibbles of A and RAM data addressed by Rr	$(A_{0-3})\leftrightarrow((R0_{0-3}))$ $(A_{0-3})\leftrightarrow((R1_{0-3}))$		
	MOV A, PSW	C7	1/1	move PSW contents to accumulator	(A)←(PSW)		
	MOV PSW, A	D7	1/1	move accumulator bit 3 to PSW_3	$(PSW_3)\leftarrow(A_3)$		3
	MOVP A, @A	A3	1/2	move indirectly addressed data in current page to A	$(PC_{0-7})\leftarrow(A)$, $(A)\leftarrow((PC))$		
FLAGS	CLR C	97	1/1	clear carry bit	(C)←0		2
	CPL C	A7	1/1	complement carry bit	(C)←NOT(C)		2

DEVELOPMENT DATA

	mnemonic	opcode (hex.)	bytes/ cycles	description	function	notes
REGISTER	INC Rr	1*	1/1	increment register by 1	(Rr)←(Rr) + 1 r = 0–7	
	INC @Rr	10 11	1/1	increment RAM data, addressed by Rr, by 1	((R0))←((R0)) + 1 ((R1))←((R1)) + 1	
	DEC Rr	C*	1/1	decrement register by 1	(Rr)←(Rr) – 1 r = 0–7	
	DEC @Rr	C0 C1	1/1	decrement RAM data, addressed by Rr, by 1	((R0))←((R0)) – 1 ((R1))←((R1)) – 1	
BRANCH	JMP addr	● 4 address	2/2	unconditional jump within a 2 K bank	(PC_{8-10})←$addr_{8-10}$ (PC_{0-7})←$addr_{0-7}$ (PC_{11-12})←MBFF 0–1	
	JMPP @A	B3	1/2	indirect jump within a page	(PC_{0-7})←((A))	
	DJNZ Rr, addr	E* address	2/2	decrement Rr by 1 and jump if not zero to addr	(Rr)←(Rr) – 1 r = 0–7 if (Rr) not zero (PC_{0-7})←addr	
	DJNZ @Rr, addr	E0 address	2/2	decrement RAM data, addressed by Rr by 1 and jump if not zero to addr	((R0))←((R0)) – 1 if ((R0)) not zero (PC_{0-7})←addr	
		E1 address			((R1))←((R1)) – 1 if ((R1)) not zero (PC_{0-7})←addr	
	JBb addr	▲ 2 address	2/2	jump to addr if Acc. bit b = 1	if b = 1 : (PC_{0-7})←addr b = 0–7	
	JC addr	F6 address	2/2	jump to addr if C = 1	if C = 1 : (PC_{0-7})←addr	
	JNC addr	E6 address	2/2	jump to addr if C = 0	if C = 0 : (PC_{0-7})←addr	
	JZ addr	C6 address	2/2	jump to addr if A = 0	if A = 0 : (PC_{0-7})←addr	
	JNZ addr	96 address	2/2	jump to addr if A is NOT zero	if A≠0 : (PC_{0-7})←addr	
	JT0 addr	36 address	2/2	jump to addr if T0 = 1	if T0 = 1: (PC_{0-7})←addr	
	JNT0 addr	26 address	2/2	jump to addr if T0 = 0	if T0 = 0: (PC_{0-7})←addr	
	JT1 addr	56 address	2/2	jump to addr if T1 = 1	if T1 = 1: (PC_{0-7})←addr	
	JNT1 addr	46 address	2/2	jump to addr if T1 = 0	if T1 = 0: (PC_{0-7})←addr	
	JTF addr	16 address	2/2	jump to addr if Timer Flag = 1	if TF = 1: (PC_{0-7})←addr	4
	JNTF addr	06 address	2/2	jump to addr if Timer Flag = 0	if TF = 0: (PC_{0-7})←addr	

INSTRUCTION SET (continued)

TIMER/EVENT COUNTER	MOV A, T	42	1/1	move timer/event counter contents to accumulator	(A)←(T)	
	MOV T, A	62	1/1	move accumulator contents to timer/event counter	(T)←(A)	
	STRT CNT	45	1/1	start event counter		
	STRT T	55	1/1	start timer		
	STOP TCNT	65	1/1	stop timer/event counter		
	EN TCNTI	25	1/1	enable timer/event counter interrupt		
	DIS TCNTI	35	1/1	disable timer/event counter interrupt		
CONTROL	EN I	05	1/1	enable external interrupt		
	DIS I	15	1/1	disable external interrupt		
	SEL RB0	C5	1/1	select register bank 0	(RBS)←0	5
	SEL RB1	D5	1/1	select register bank 1	(RBS)←1	5
	SEL MB0	E5	1/1	select program memory bank 0	(MBFF0)←0, (MBFF1)←0	
	SEL MB1	F5	1/1	select program memory bank 1	(MBFF0)←1, (MBFF1)←0	
	SEL MB2	A5	1/1	select program memory bank 2	(MBFF0)←0, (MBFF1)←1	
	SEL MB3	B5	1/1	select program memory bank 3	(MBFF0)←1, (MBFF1)←1	
	STOP	22	1/1	enter STOP mode		
	IDLE	01	1/1	enter IDLE mode		
SUBROUTINE	CALL addr	▲ 4 address	2/2	jump to subroutine	((SP))←(PC), ($PSW_{4, 6, 7}$) (SP)←(SP) + 1 (PC_{8-10})←$addr_{8-10}$ (PC_{0-7})←$addr_{0-7}$ (PC_{11-12})←MBFF 0–1	6
	RET	83	1/2	return from subroutine	(SP)←(SP) − 1 (PC)←((SP))	6
	RETR	93	1/2	return from interrupt and restore bits 4, 6, 7 of PSW	(SP)←(SP) − 1 ($PSW_{4, 6, 7}$) + (PC)←((SP))	6

	mnemonic	opcode (hex.)	bytes/ cycles	description	function	notes
PARALLEL INPUT/OUTPUT	IN A, Pp	08 09 0A	1/2	input port p data to accumulator	(A)←(P0) (A)←(P1) (A)←(P2)	7
	OUTL Pp, A	38 39 3A	1/2	output accumulator data to port p	(P0)←(A) (P1)←(A) (P2)←(A)	
	ANL Pp, #data	98 data 99 data 9A data	2/2	AND port p data with immediate data	(P0)←(P0) AND data (P1)←(P1) AND data (P2)←(P2) AND data	
	ORL Pp, #data	88 data 89 data 8A data	2/2	OR port p data with immediate data	(P0)←(P0) OR data (P1)←(P1) OR data (P2)←(P2) OR data	
DERIVATIVE INPUT/OUTPUT	MOV Dx, A	8D	2/2	move accumulator contents to derivative register	(Dx)←(A) x = 0 to 255	8
	NOP	00	1/1	no operation		

Notes to Table 7

1. PSW CY, AC affected
2. PSW CY affected
3. PSW PS affected
4. Execution of JTF and JNTF instructions resets the Timer Flag (TF).
5. PSW RBS affected
6. PSW SP_0, SP_1, SP_2 affected
7. (A) = 0000P23, P22, P21, P20.
8. Instructions for PCF84C00 only.

* : 8, 9, A, B, C, D, E, F
● : 0, 2, 4, 6, 8, A, C, E
▲ : 1, 3, 5, 7, 9, B, D, F

RATINGS

Limiting values in accordance with the Absolute Maximum System (IEC 134)

parameter	conditions	symbol	min.	max.	unit
Supply voltage	pin 24	V_{DD}	−0,8	+8	V
Input voltage	any pin	V_I	−0,8	V_{DD} + 0,8	V
DC current	any input or output	$\pm I_I, \pm I_O$	–	10	mA
Total power dissipation	derate according to thermal resistance	P_{tot}	–	500	mW
Power dissipation	per output	P_O	–	50	mW
Storage temperature range		T_{stg}	−65	+150	°C
Operating ambient temperature range		T_{amb}	−25	+70	°C
Operating junction temperature		T_j	–	+125	°C
Thermal resistance junction to ambient	SOT117	$R_{th\ j\text{-}a}$	–	120	K/W
	SOT136A	$R_{th\ j\text{-}a}$	–	150	K/W

DEVELOPMENT DATA

DC CHARACTERISTICS

V_{DD} = 2,5 to 6 V; V_{SS} = 0 V; T_{amb} = −25 to + 70 °C; all voltages with respect to V_{SS}; f = 3,58 MHz with R_S = 100 Ω; unless otherwise specified

parameter	conditions	symbol	min.	typ.	max.	unit
Supplies						
Supply voltage operating		V_{DD}	2,5	–	6	V
STOP mode for RAM data retention		V_{DD}	1,0	–	6	V
Supply current (Fig. 25)						
operating with tone generator on	V_{DD} = 3 V	I_{DD}	–	1,2	–	mA
operating without tone generator	V_{DD} = 3 V	I_{DD}	–	600	–	μA
IDLE mode (Fig. 26)						
with tone generator on	V_{DD} = 3 V	I_{DD}	–	900	–	μA
without tone generator	V_{DD} = 3 V	I_{DD}	–	300	–	μA
STOP mode (Fig. 27)	note 1; V_{DD} = 1,8 V					
	T_{amb} = 25 °C	I_{DD}	–	1,2	2,5	μA
	T_{amb} = 55 °C	I_{DD}	–	–	5	μA
	T_{amb} = 70 °C	I_{DD}	–	–	10	μA
RESET I/O						
Switching level		V_{RESET}	–	1,3	–	V
Sink current	$V_{DD} > V_{RESET}$	I_{OL}	–	7	–	μA
Inputs						
Input voltage LOW		V_{IL}	0	–	0,3 V_{DD}	V
Input voltage HIGH		V_{IH}	0,7 V_{DD}	–	V_{DD}	V
Input leakage current	$V_{SS} < V_I < V_{DD}$	$\pm I_{IL}$	–	–	1	μA
Outputs						
Output voltage LOW	V_I = V_{SS} or V_{DD}; $\lvert I_O \rvert < 1$ μA	V_{OL}	–	–	0,05	V
Output sink current LOW P12, P13, P14, P20, P21, P22, P23, (Fig. 28)	V_{DD} = 3 V;					
	V_O = 0,4 V	I_{OL}	0,7	1,5	–	mA
for remaining ports (Fig. 29)	V_O = 0,4 V	I_{OL}	1,5	–	–	mA
Pull-up output source current HIGH (Fig. 30)	V_{DD} = 3 V;					
	V_O = 0,9 V_{DD}	$-I_{OH}$	10	–	–	μA
	V_O = V_{SS}	$-I_{OH}$	–	–	300	μA
Push-pull output source current HIGH	V_{DD} = 3 V;					
	V_O = V_{DD} −0,4 V	$-I_{OH}$	0,6	1,5	–	mA

Note 1

Crystal connected between XTAL1 and XTAL2; CE and T1 at V_{SS}.

TONE GENERATOR CHARACTERISTICS

V_{DD} = 2,5 to 6 V; V_{SS} = 0 V; T_{amb} = −25 to + 70 °C; all voltages with respect to V_{SS}; f = 3,58 MHz with R_S = 100 Ω; unless otherwise specified

parameter	conditions	symbol	min.	typ.	max.	unit
Tone output (Fig. 24)						
DTMF output voltage levels (r.m.s. values)						
HIGH group		$V_{HG(rms)}$	158	192	205	mV
LOW group		$V_{LG(rms)}$	125	150	160	mV
Frequency deviation		$\Delta f/f$	−0,6	—	+ 0,6	%
DC voltage level		V_{dc}	—	½ V_{DD}	—	V
Output impedance		$\lvert Z_O \rvert$	—	0,1	0,5	kΩ
Pre-emphasis of group		ΔV_G	1,85	2,1	2,35	dB
Total harmonic distortion	note 2; T_{amb} = 25 °C	THD	—	−25	—	dB

Note 2

Related to the level of the LOW group frequency component (CEPT CS 203)

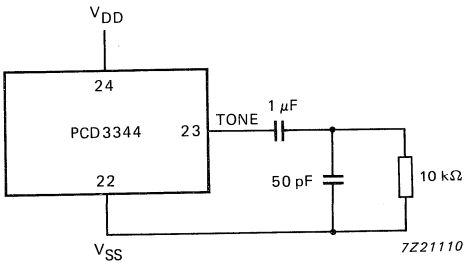


Fig. 24 Tone output test circuit.

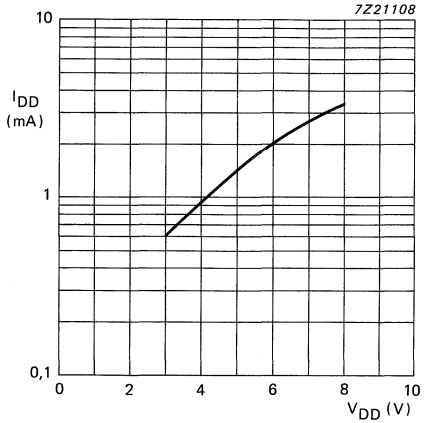


Fig. 25 Typical supply current (I_{DD}) in operating mode as a function of the supply voltage (V_{DD}); T_{amb} = 25 °C; clock frequency = 3,58 MHz; I_{DD} is increased by approximately 0,6 mA when the DTMF function is operating.

DEVELOPMENT DATA

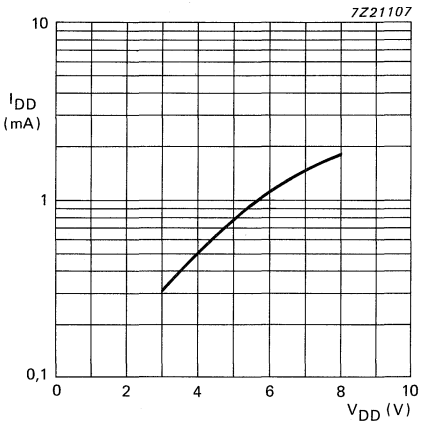
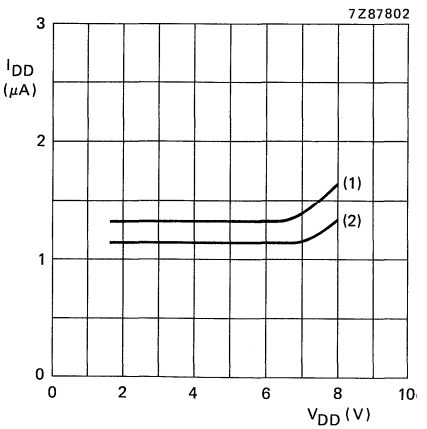
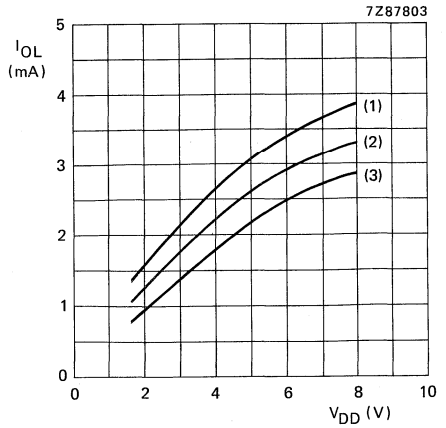


Fig. 26 Typical supply current (I_{DD}) in IDLE mode as a function of the supply voltage (V_{DD}); T_{amb} = 25 °C; clock frequency = 3,58 MHz.



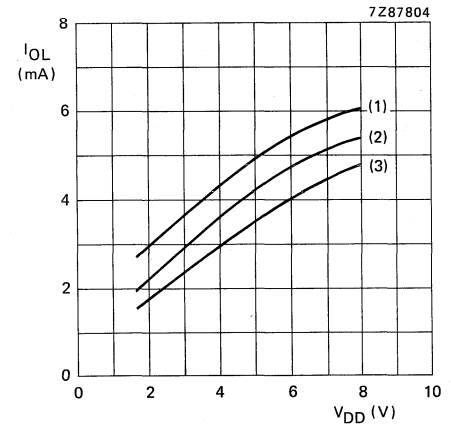
(1) T_{amb} = 25 °C
(2) T_{amb} = 70 °C

Fig. 27 Typical supply current (I_{DD}) in STOP mode as a function of the supply voltage (V_{DD}).



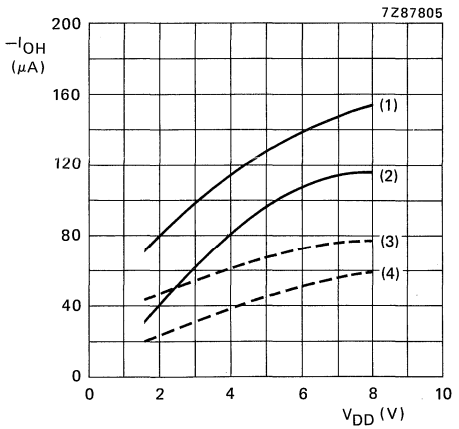
(1) $T_{amb} = -25$ °C
(2) $T_{amb} = 25$ °C
(3) $T_{amb} = 70$ °C

Fig. 28 Output sink current LOW (I_{OL}), for P12, P13, P14, P20, P21, P22, P23 as a function of supply voltage (V_{DD}); $V_O = 0,4$ V.



(1) $T_{amb} = -25$ °C
(2) $T_{amb} = +25$ °C
(3) $T_{amb} = +70$ °C

Fig. 29 Output sink current LOW (I_{OL}), for remaining ports as a function of supply voltage (V_{DD}); $V_O = 0,4$ V.



(1) $T_{amb} = 25$ °C; $V_O = V_{SS}$
(2) $T_{amb} = 25$ °C; $V_O = 0,9\ V_{DD}$
(3) $T_{amb} = 70$ °C; $V_O = V_{SS}$
(4) $T_{amb} = 70$ °C; $V_O = 0,9\ V_{DD}$

Fig. 30 Output source current HIGH ($-I_{OH}$) as a function of supply voltage (V_{DD}).

DEVELOPMENT DATA

APPLICATION INFORMATION

A block diagram of an electronic featurephone built around the PCD3344 is shown in Figure 31. It comprises the following dedicated telephony ICs:

- TEA1060/1061/1067/1068 transmission circuit for telephony
- PCF8576 or PCF8577 2 LCD drivers in LCD module MB7020160
- PCF8571 1 K RAMs with Serial I/O; the number of RAMs depends on the required amount of stored telephone numbers
- PCD3360 programmable multi-tone ringer

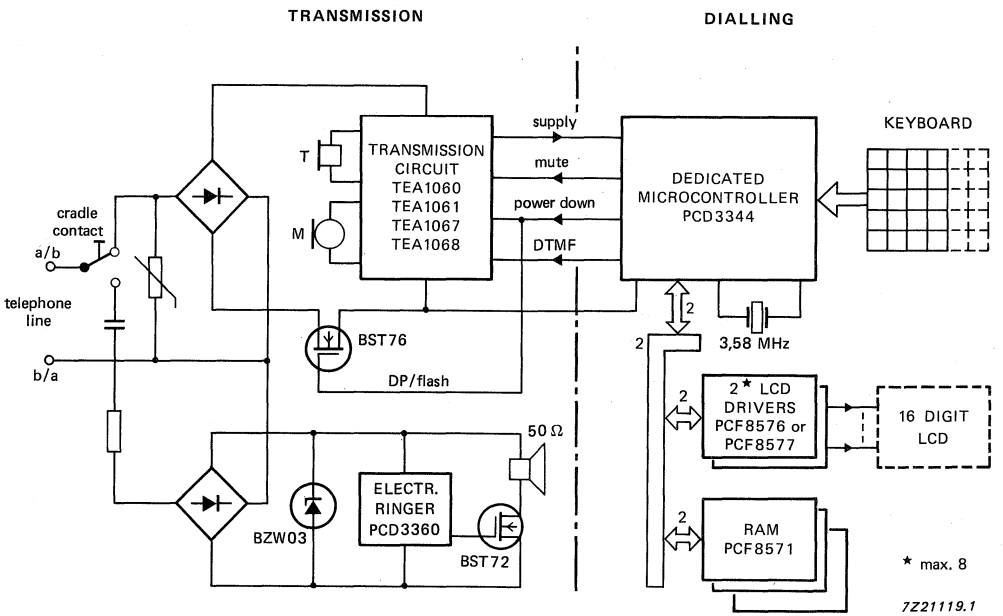


Fig. 31 Block diagram of electronic featurephone with common line interface.

SINGLE-CHIP 8-BIT MICROCONTROLLER

DESCRIPTION

The PCD3346 is a single-chip 8-bit microcontroller manufactured in CMOS technology. The PCD3346 is a member of the PCD33XX family and as such has special on-chip features for telephony applications.

The PCD3346 has 20 quasi-bidirectional I/O port lines, a serial I/O interface, a single-level vectored interrupt circuit, two 8-bit timer event counters and on-board clock oscillator and clock circuits. The PCD3346 also incorporates 256 bytes of EEPROM permitting intermediate data storage without the need for battery back-up.

The instruction set is based on that of the MAB8048 and is instruction set compatible with the MAB8400 family. The PCD3346 has bit handling abilities for both binary and BCD arithmetic.

Features

- 8-bit CPU, ROM, EEPROM, RAM, I/O in a single 28-lead DIL or SO package
- 4 K ROM bytes
- 128 RAM bytes
- 256 bytes EEPROM
- 20 quasi-bidirectional I/O port lines
- 2 x 8-bit programmable timers
- Two test inputs: one of which is also the external interrupt input (CE/$\overline{T0}$)
- Single-level vectored interrupts: external, timer/event counter, serial I/O and derivative port
- I^2C hardware interface for serial data transfer on two lines (serial I/O data via an existing port line and clock via a dedicated line)
- Clock frequency 450 kHz to 10 MHz
- Over 80 instructions (based on MAB8048) all of 1 or 2 cycles
- Single supply voltage from 2.5 V to 6.0 V
- STOP and IDLE mode
- On-chip oscillator with output drive capability for peripherals (e.g. PCD3312 DTMF generator)
- Individual mask configuration of all port lines for: pull-up, push-pull or open drain
- Power-on-reset circuit and low supply voltage detection
- Individual mask selection of reset state for all ports
- Operating temperature range: −25 to + 70 °C

LIFE SUPPORT APPLICATIONS

This product is not designed for use in life support appliances, devices, or systems where malfunction of this product can reasonably be expected to result in personal injury. Philips customers using or selling this product for use in such applications do so at their own risk and agree to fully indemnify Philips for any damages resulting from such improper use or sale.

PACKAGE OUTLINES

PCD3346P: 28-lead DIL; plastic (SOT117).
PCD3346T: 28-lead mini-pack; plastic (SO28; SOT136A).

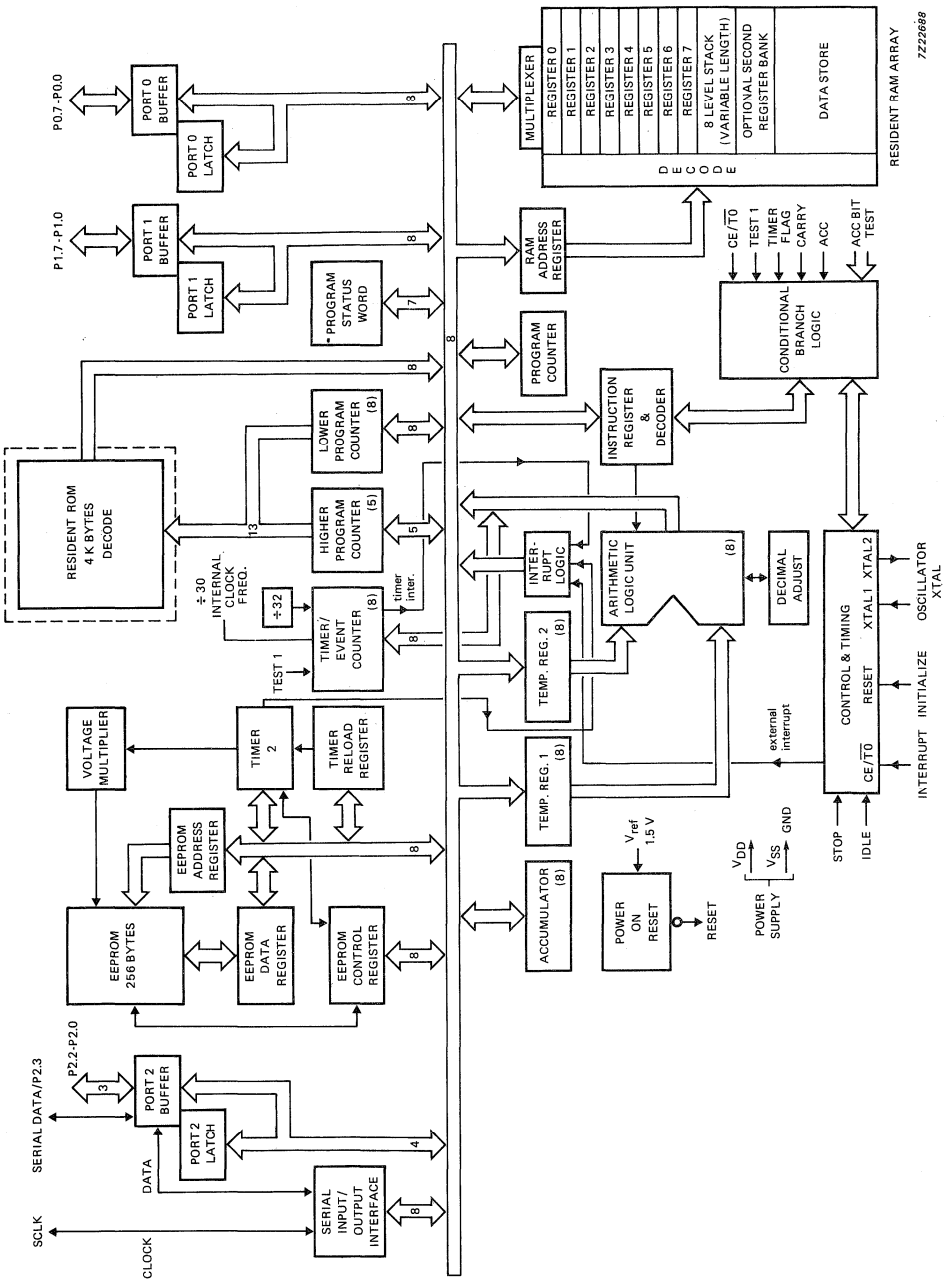


Fig. 1 Block diagram.

DEVELOPMENT DATA

PINNING

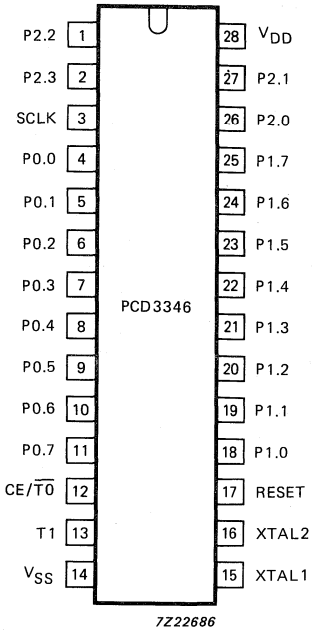


Fig. 2 Pinning diagram.

PIN DESIGNATION

Pin	symbol	type	function
3	SCLK	I/O	Clock: bidirectional clock for serial I/O.
4-11	P00-P07	I/O	Port 0: 8-bit quasi-bidirectional I/O port.
12	CE/$\overline{T0}$	I	Interrupt/Test 0: external interrupt input (sensitive to positive-going edge)/test input pin; when used as a test input directly tested by conditional branch instruction JT0 and JNT0.
13	T1	I	Test 1: test input pin, directly tested by conditional branch instructions JT1 and JNT1. T1 also functions as an input to the 8-bit timer/event counter, using the STRT CNT instruction.
14	V_{SS}	I	Ground: circuit earth potential.
15	XTAL1	I	Crystal input: connection to timing component (crystal) which determines the internal oscillator frequency; also the input for an external clock source.
16	XTAL2	I/O	Connection to other side of timing component.
17	RESET	I/O	Reset input: used to initialize the processor (active HIGH), or output of power-on-reset circuit.
18-25	P10-P17	I/O	Port 1: 8-bit quasi-bidirectional I/O port.
26,27, 1,2	P20-P23	I/O	Port 2: 4-bit quasi-bidirectional I/O port. P23 is the serial data input/output in serial I/O mode.
28	V_{DD}	I	Power supply: 2.5 V to 6.0 V.

FUNCTIONAL DESCRIPTION

Emulation of PCD3346

Emulation of the PCD3346 can be achieved using the piggyback version PCA3346B.

Program memory

The program memory consists of 4 K bytes, in a read-only memory (ROM). Each location is directly addressable by the program counter. The memory is mask-programmed at the factory. Figure 3 shows the program memory map.

Four program memory locations are of special importance:

- Location 0; contains the first instruction to be executed after the processor is initialized (RESET),
- Location 3; contains the first byte of an external interrupt service subroutine,
- Location 5; contains the first byte of a serial I/O and derivative interrupt service subroutine,
- Location 7; contains the first byte of a timer/event counter interrupt service subroutine.

Program memory is arranged in banks of 2 K bytes, which are selected by SEL MB instructions. The program memory is further divided into location 'pages', each of 256 bytes. This latter division applies only for conditional branches. Memory bank boundaries can be crossed only by using the unconditional branch instructions after the appropriate memory bank has been selected. A CALL instruction can transfer control to a subroutine on any 'page'; RET and RETR instructions can transfer control from a subroutine back to the main program.

Data memory

Data memory consists of 128 bytes, random-access data memory (RAM). Allocations are indirectly addressable using RAM pointer registers; up to 16 designated locations are directly addressable. Memory also includes an 8-level program counter stack addressed by a 3-bit stack pointer. Figure 4 shows the data memory map.

Working registers

Locations 0 to 7 are designated as working registers, directly addressable by the direct register instructions. Ease of addressing, and a minimum requirement of instruction bytes to manipulate their contents, makes these locations suitable for storing frequently addressed intermediate results. This bank of registers can be selected by the SEL RB0 instruction.

Executing the select register bank instruction SEL RB1, designates locations 24 to 31 as working registers, instead of locations 0 to 7, and these are then directly addressable. This second bank of working registers may be used as an extension of the first or reserved for use during interrupt service subroutines saving the first bank for use in the main program. If the second bank is not used, locations 24 to 31 may serve as general purpose RAM.

The first locations of each bank contain the RAM pointer registers R0, R1, R0' and R1', which indirectly address all RAM locations.

All RAM locations make efficient program loop counters when used with the decrement register and test instruction DJNZ.

DEVELOPMENT DATA

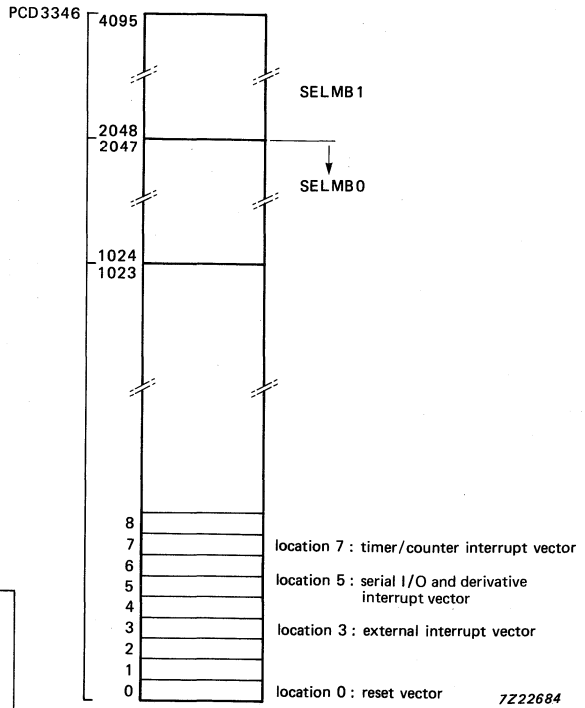


Fig. 3 Program memory map.

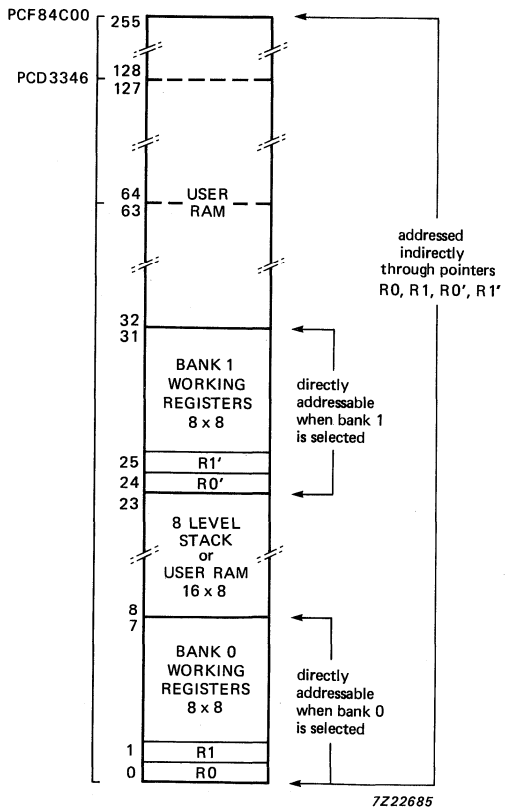


Fig. 4 Data memory map.

FUNCTIONAL DESCRIPTION (continued)

Program counter stack

Locations 8 to 23 may be designated as an 8-level program counter stack (2 locations per level), or as general purpose RAM. The program counter stack (Fig. 5) enables the microcontroller to keep track of the return addresses and status prior to interrupts or CALL instructions by storing the contents of the program counter before servicing the subroutine. A 3-bit stack pointer determines which of the eight register pairs of the program counter stack will be loaded with the next return address and status.

The stack pointer, when initialized to 000 by RESET, points to RAM locations 8 and 9. On the first subroutine CALL or interrupt, the contents of the program counter and bits 4, 6 and 7 of the program status word (PSW) are transferred to locations 8 and 9. The stack pointer increments by one and points to locations 10 and 11, ready for another CALL. Because an address may be up to 13 bits long, two bytes must be used to store each address.

At the end of a subroutine, which is signalled by a return instruction (RET or RETR), the stack pointer decrements by one and the contents of the register pair on top of the stack are transferred to the program counter. The saved PSW bits are transferred to the PSW only by the RETR instruction.

If all 8 levels of subroutine and interrupt nesting are not used, the unused portion of the stack may be used as any other indirectly addressable RAM locations.

Nesting of subroutines within subroutines can continue up to 8 times without overflowing the stack. If overflow does occur the deepest address stored (locations 8 and 9) will be overwritten and lost since the stack pointer overflows from 111 to 000. It also underflows from 000 to 111.

The value of the saved contents of the program counter during an interrupt CALL is not the same as the value saved during a normal CALL to subroutine. During an interrupt CALL, the program counter return address is saved; during a subroutine CALL, the saved program counter value is one less than the program counter return address.

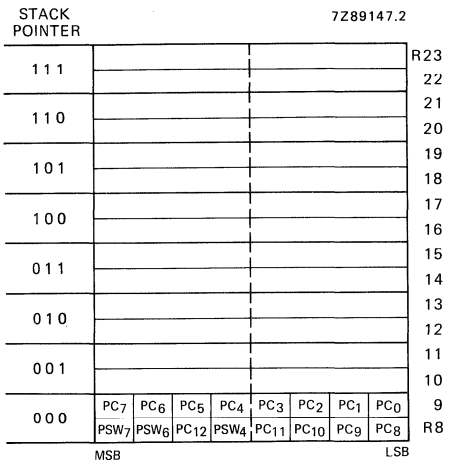


Fig. 5 Program counter stack.

EEPROM memory

The PCD3346 has 256 bytes of Electrically Eraseable Programmable Read Only Memory on-chip. On-chip EEPROM makes register retention possible without battery back-up. The 256 bytes of EEPROM are arranged as 64 pages. Each page is a single row of 32 bits, i.e. 4 bytes. This organisation makes it possible to read and erase/write in single page and byte mode. Addressing is performed on-chip using an 8-bit address decoder; the upper six bits select the page and the lowest two a byte on that page. The EEPROM address register is auto-incremented within the page which simplifies the read/write operation. Auto-increment is active during read and write functions, but is inactive during erase mode. Overflow during auto-increment leaves the register at 00000000.

The EEPROM is interfaced with the CPU via the derivative registers. Table 1 shows the instructions that control the derivative registers.

Table 1 Derivative register instruction set.

The PCD3346 has an additional 4 instructions to handle the derivative registers, these are:

Mnemonic	description	opcode (hex)	function x = 1 to 6
MOV A,Dx	move derivative register contents to accumulator	8C Dx	(A) <– (Dx)
MOV Dx,A	move accumulator contents to derivative register	8D Dx	(Dx) <– (A)
ANL Dx,A	AND derivative register with accumulator	8E Dx	(Dx) <– (Dx) AND (A)
ORL Dx,A	OR derivative register with accumulator	8F Dx	(Dx) <– (Dx) OR (A)

Each instruction is 2 cycles, 2 bytes. The second byte selects the derivative register to be manipulated (1 to 6) e.g. data transport from accumulator to derivative register D01 is performed by the two byte opcode 8 D 01.

For the PCD3346 six derivative registers are required and these are given in Table 2.

Table 2 PCD3346 derivative registers

name	derivative address	function
ADD1	01H*	contains the EEPROM array address
DATR	03H	contains the read or write data
EPCR	04H	EEPROM control register
RELR	05H	timer T2 reload
T2	06H	contents of timer T2

* For larger EEPROMs a second address register (ADD2) is necessary to accommodate the higher part of the EEPROM address. To ensure uniformity of derivative registers, a derivative address (02H) should be reserved for the high address part.

FUNCTIONAL DESCRIPTION (continued)

Table 3 EEPROM control register

EPCR (04H)	STT2	EIT2	TF2	EWP	MC3	MC2	MC1	MC0
	7	6	5	4	3	2	1	0

EPCR is a derivative R/W register containing the status of the EEPROM interface and timer T2.

Abbreviations used

Bit		Function
STT2	start timer T2	Setting this bit clears the pre-scaler and starts timer T2 in the general purpose mode. Timer T2 in the general purpose mode does not affect the EEPROM interface. T2 is loaded with the value in the RELR register at both start up and underflow. Setting STT2 to zero, stops Timer T2 and the pre-scaler, clears the pre-scaler and then reloads Timer T2.
EIT2	enable interrupt	When this bit is set and the SIO enabled, a derivative interrupt request is generated when TF2 is set. No interrupt request will be generated if EIT2 is set to zero.
TF2	timer flag T2	This bit is set by hardware when Timer T2 underflows and a derivative interrupt is requested if the SI is enabled (Fig. 6). TF2 must be cleared by software.
EWP	erase/write cycle in progress	This bit is set by software at the beginning of an erase or write cycle in the byte mode. It must be set by software in the page mode. It is cleared by hardware when the erase or write cycle has expired.
MCn	mode control bits	Four bits which control the working mode of the EEPROM. The truth table is given in Table 4.

Table 4 Truth table for EEPROM interface control bits

EWP	MC3	MC2	MC1	MC0	function
0	0	0	0	0	read byte
1	0	1	1	0	write byte
1	1	0	0	1	erase/write byte
1	1	1	0	0	erase page
0/1*	0	1	0	1	write page
1	1	0	1	0	erase bulk

Other bit patterns have no effect upon the interface. Bits MC3 to MC0 can be read and written by software, but are also cleared by hardware after each erase or write cycle has expired.

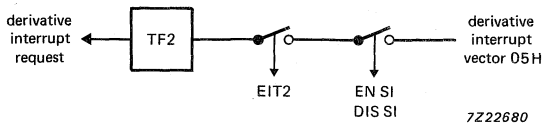


Fig. 6 Organization of a derivative interrupt.

Timer T2

Timer T2 determines the period of the erase and write timing cycles. Both times are the same, and are required to be of a fixed duration independent of the oscillator frequency. This requirement is fulfilled by T2 which is programmable timer with reload register RELR. Timer T2 is clocked via a 4-bit pre-scaler, which in turn is serviced by the processor clock ($f_{osc}/30$).

This means that T2 is decremented every 480 oscillator periods.
Erase and write times are both of 10 ms duration. The value for the RELR is to be calculated according to the following formula. Table 5 gives examples of the RELR values for different XTAL frequencies.

$$(RELR) = \frac{f_{XTAL}}{480} \cdot 10\ ms$$

RELR is a R/W derivative register. At the beginning of a read/write cycle, Timer 2 is loaded with the contents of RELR and the count started. Underflow of the timer signals that the erase cycle has finished and that Timer T2 is reloaded once more with the contents of RELR. The next underflow will signify the end of the write cycle and the counter will be switched off. Erase and write times are both of 10 ms duration.

* EWP = 0 selects write page mode, EWP = 1 executes write page.

FUNCTIONAL DESCRIPTION (continued)

Table 5 RELR values for various XTAL frequencies

f_{xtal}	timer 2 clocking (after 4-bit pre-scaler)	RELR value for 10 ms duration
450 kHz	1.1 ms	10
1 MHz	0.48 ms	21
3.58 MHz	0.134 ms	75
10 MHz	0.048 ms	209

Timer T2 may also be used as a general purpose timer. It is started by setting the STT2 bit in register EPCR, which loads the contents of RELR into the timer. At overflow, the interrupt flag T2 is set and the timer auto-reloaded with RELR. Timer T2 can be read "on-the-fly" and is addressed as derivative register T2. The following instructions demonstrate control of the timer in the general purpose mode.

```
                          ; TF2 must be cleared
MOV A, #_timevalue        ; time to be down-counted
MOV RELR, A               ; load derivative register RELR
EN SI                     ; enable SI interrupt
MOV A, #STT2+EIT2         ; start counter and enable derivative interrupt
ORL EPCR, A               ; load status in derivative control register
```

When the timer underflows, an interrupt call to ROM address 5 will be generated (SI interrupt) and appropriate actions will be decided by software. The content of timer can only be read "on-the-fly" using the instruction MOV A, D6.

Read EEPROM

Reading data from the EEPROM is possible only when no erase/write action is in progress. The read operation of a data byte is performed as follows:

- load EEPROM address
- load EPCR (not necessary when the MCn bits are already set to read mode)
- read data register

A program example could be:

```
MOV A, #_address          ; load EEPROM read address
MOV ADD1, A               ; send address to ADD1 register
MOV A, DATR               ; load data from EEPROM to accumulator
MOV Rm, A                 ; store data
```

After executing the third instruction, the ADD1 address register is auto-incremented and the next byte can be read by repeating the last two instructions.

The erase/write operation

There are five erase/write modes controlled by the MCn bits in register EPCR:

- write byte
- erase/write byte
- erase page
- write page
- erase bulk

Each write or erase action takes 10 ms, this value must be loaded into the RELR register before starting any write/erase action, e.g.:

```
MOV A, #_RELR value     ; load appropriate 10 ms value according to fXTAL
                        ; frequency into accumulator
MOV RELR, A             ; load RELR register
```

A new erase or write operation can be started if no previous erase or write operation is in progress.

Write byte

A write operation may be started only if the addressed byte has been erased. The write operation is as follows:

- load EEPROM address
- load data
- load EPCR

The last instruction sets the operation mode and starts Timer T2, the address register is incremented and an interrupt is generated when the 10 ms count is reached.

Erase/write byte

The erase/write operation clears the addressed byte and writes the new. The erase/write operation is as follows:

- load EEPROM address
- load data
- load EPCR

The last instruction sets the operation mode and starts Timer T2, the address register is incremented and an interrupt is generated when the 2 x 10 ms count is reached.

Erase page

In this mode a complete four byte page will be erased. This operation requires only that the first byte of the page be addressed, after which the entire page will be erased. The erase page operation is as follows:

- load EEPROM with address of first byte of any page
- load EPCR

The last instruction starts Timer T2 and an interrupt is generated after the 10 ms period has elapsed.

FUNCTIONAL DESCRIPTION (continued)

* Piggybacks and SDS probes only (it is recommended to do it always, also for ROM code).

If the EPCR-register is checked before any erase/write operation, a dummy read of the EEPROM data register is neccessary for technical reasons. Please note that the EEPROM address register ADD1 will be autoincremented by one due to the dummy read.

Example.

```
LABEL1      MOV A, D4                    ; EPCR check
            test EWP- bit, jump to LABEL1 ; busy wait
            MOV A, D3                    ; dummy read
            .
            .
            MOV A, value                 ; choose e/w operation
            MOV D4, A                    ; start e/w operation
```

Write page

A write page operation can only be started if the addressed page is empty (erased). If the page is not empty an erase page must be executed beforehand. The write page operation is as follows:

- load EPCR (select write page mode)
- load ADD1 with address of first byte of any page
- load four data bytes (EEPROM address register auto-increments after each load except after fourth byte*)
- load EPCR (start write page)

The last instruction starts Timer T2 and an interrupt is generated after the 10 ms period has elapsed.

Given that an erase page cycle has been executed, a write page program is as follows:

```
MOV A, #MC2+MC0          ;
MOV EPCR, A              ; select write page mode
MOV A, #_address         ; EEPROM start address to accumulator
MOV ADD1, A              ; address to derivative address register

MOV A, R1                ; byte to be written to accumulator
MOV DATR, A              ; write 1st byte, address register auto-incremented
MOV A, R2                ; byte to be written to accumulator
MOV DATR, A              ; write 2nd byte, address register auto-incremented
MOV A, R3                ; byte to be written to accumulator
MOV DATR, A              ; write 3rd byte, address register auto-incremented
MOV A, R4                ; byte to be written to accumulator
MOV DATR, A              ; write 4th byte

MOV A, #EWP+MC2+MC0      ; set write page mode and execute bit in accumulator
MOV EPCR,A               ; start cycle
```

The last instruction sets the operation mode and starts Timer T2, the address register is incremented and an interrupt is generated at Timer T2 underflow.

* Note that data cannot be addressed (read or write) over a page limit.

Bulk erase

In the bulk erase mode the complete EEPROM is erased. The bulk erase operation is as follows:

– Load EPCR.

After start of bulk erase Timer T2 is started and an interrupt is generated at underflow. Below is an example of a program for a bulk erase:

```
MOV A,#EWP+MC3+MC1      ; set the bulk erase mode in accumulator
MOV EPCR,A              ; load EPCR and start erase cycle
```

Timer T2 is started and an interrupt is generated at Timer T2 underflow.

FUNCTIONAL DESCRIPTION (continued)

IDLE and STOP modes

IDLE mode

When the microcontroller enters the IDLE mode via the IDLE instruction (01H), the oscillator, timer/counter and serial I/O are kept running. The microcontroller exits from the IDLE mode via a RESET or one of three interrupts if they are enabled. If the interrupt is not enabled, the microcontroller will remain in the IDLE mode. An active signal on the RESET pin restarts the microcontroller and a normal RESET sequence is executed (see Fig. 7).

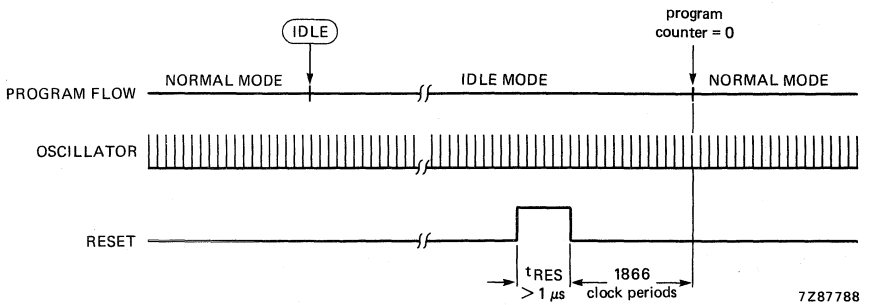


Fig. 7 Exit from IDLE mode via a RESET.

Any erase/write cycle in progress during IDLE mode will be completed. The interrupt request generated at the end of the erase/write cycle, if enabled, wakes up the microcontroller returning it to the normal mode.

An active signal coming from an enabled interrupt causes the execution of the normal interrupt routine since normal interrupt scanning is still being carried out. A LOW to HIGH transition on the external interrupt pin (CE/$\overline{TO}$) reactivates the microcontroller. A HIGH level applied to CE/$\overline{TO}$ will reactivate the microcontroller only in the STOP mode. Thus if CE/$\overline{TO}$ was HIGH before the microcontroller entered the IDLE mode, it must go LOW before the microcontroller can be reactivated (see fig. 8).

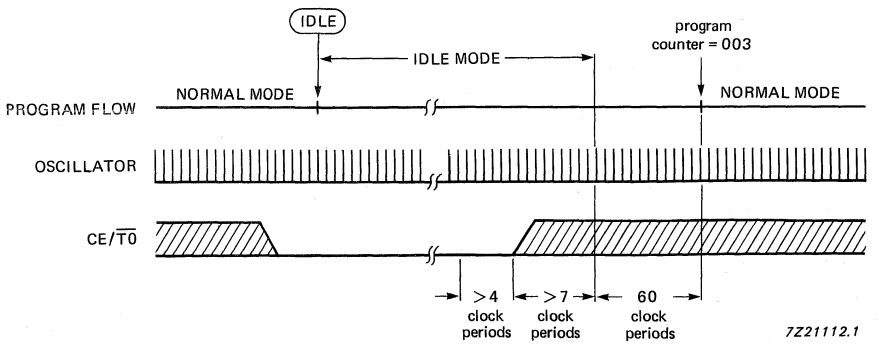


Fig. 8 Exit from IDLE mode via an interrupt.

Wake-up from the IDLE mode is ensured when CE/$\overline{T0}$ is LOW for at least 4 CP (clock periods) and then HIGH for 7 CP. After the initial forced CALL 003H operation (60 CP), the program continues with the external interrupt service routine. During the STOP mode, the address of the instruction immediately following the instruction that caused the microcontroller to enter the IDLE mode is present on the address bus.

STOP mode

Before entering the STOP mode bit EWP of the EPCR register must be set to logic 0.

The microcontroller enters the STOP mode via the STOP instruction (22H). The oscillator is switched off and the internal status of the CPU, RAM contents and the state of I/O ports are not affected. The microcontroller can be brought out of the STOP mode by an active signal at the external interrupt input or by an external RESET signal. When one of these two signals is applied, an internal delay of 1866 CP is provided to ensure that all internal clocks are operating correctly before restart (see Fig. 9).

Note: the start-up time of a crystal oscillator is measured in milliseconds, and the 1866 CP count begins after this start-up time.

If the microcontroller exits from the STOP mode by activating RESET, a normal RESET sequence is executed.

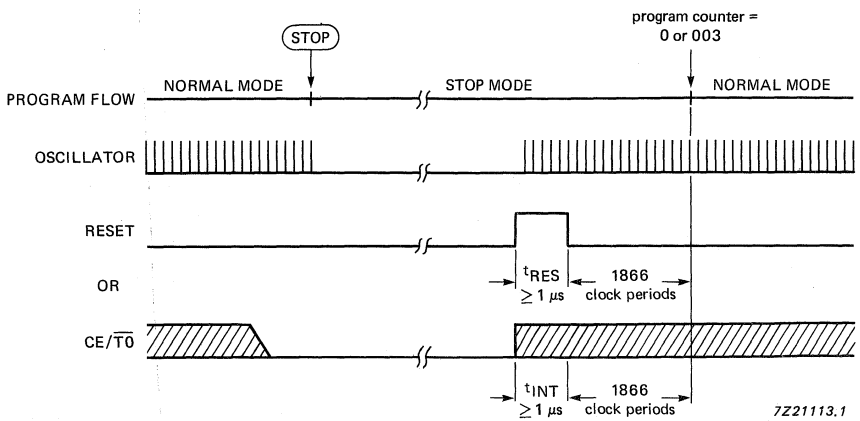


Fig. 9 Entering and exiting the STOP mode.

If the microcontroller exits the STOP mode by pulling the external interrupt input pin HIGH, an interrupt sequence is executed only if the external interrupt is enabled. In this event the microcontroller resumes the normal program sequence after returning from the interrupt routine, as in the normal mode. If the interrupt is not enabled, it continues the normal program sequence, executing the instruction following the STOP instruction.

The microcontroller is restarted by a HIGH level applied at the CE/$\overline{T0}$ pin.
Note: When exiting the STOP mode via an interrupt, a further instruction in the main program series is executed prior to entering the interrupt routine.

If the CE/$\overline{T0}$ level is active during the STOP instruction then no STOP is executed.

A HIGH level on the external interrupt input of at least 1 μs will cause the microcontroller to exit the STOP mode. During the STOP mode, the address of the instruction immediately following the last STOP instruction is present on the address bus.

FUNCTIONAL DESCRIPTION (continued)

I/O facilities

The PCD3346 has 23 I/O lines arranged as:

- Port 0 8-bit parallel port (P0.0 to P0.7)
- Port 1 8-bit parallel port (P1.0 to P1.7)
- Port 2 4-bit parallel port (P2.0 to P2.3)
- SCLK serial I/O clock line
- CE/$\overline{T0}$ external interrupt and test input. When used as a test input can be directly tested by conditional branch instructions JT0 and JNT0
- T1 test input which can alter program sequences when tested by conditional jump instructions JT1 and JNT1. T1 also functions as an input to the 8-bit timer/event counter.

Parallel ports

Parallel port lines can be individually configured as outputs or inputs; their structure is quasi-bidirectional. Output data written to a port is latched and remains unchanged until rewritten. Input data is not latched and must be present until read by an input instruction.

Input lines are fully CMOS compatible; output lines can drive one TTL or CMOS load.

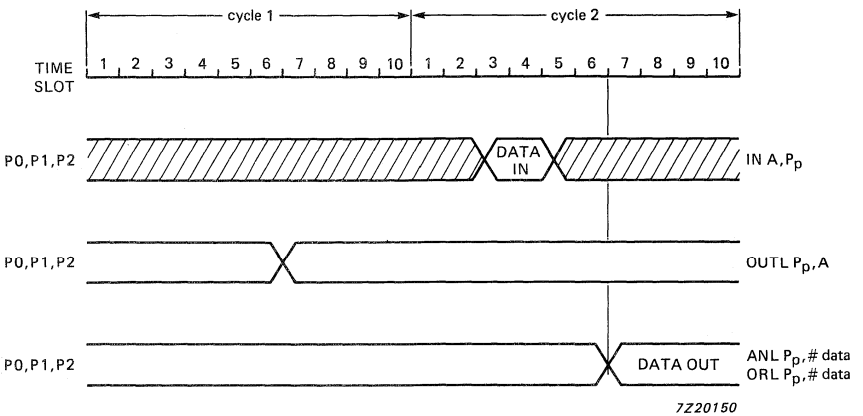


Fig. 10 Timing diagram for all ports using IN, OUTL, ANL and ORL instructions.

Figure 10 shows the timing diagram for all ports using IN, OUTL, ANL and ORL instructions. For the OUTL instruction data changes on time slot 7 of cycle 1. For the ANL and ORL instructions, the ports change on time slot 7 of cycle 2.

Figure 11 shows the quasi-bidirectional I/O interface with push-pull output and switched pull-up current source.

Each line is pulled up to V_{DD} via a constant current source (TR4), which is enabled via TR3 whenever one of the two output latches contains a logic 1. This current source is sufficient for a TTL HIGH level, yet can be pulled LOW by an external CMOS device, thus allowing the same pin to be used for both input and output.

When a logic 1 is written to a port line for the first time (MQ = 1, SQ = 0), TR2 is switched on for the duration of the internal write pulse (one oscillator period), to provide a fast transition from logic 0 to logic 1. Subsequent writing of a logic 1 to a port line will not switch TR2 on. This ensures that the port lines can be pulled LOW when configured as inputs.

When a logic 0 is written to a port line, TR3 switches off the current source. Current sinking capability is provided by TR1, which is now switched on. When used as an input, a logic 1 must first be written to the port line; otherwise TR1 will remain low impedance.

PCD3346 mask options make it possible for each individual port line to be configured as one of the following:

1. STANDARD PORT: quasi-bidirectional I/O with switched pull-up current source of 100 μA (typ.) and P-channel booster transistor TR2. TR2 is only active during 1 clock cycle (Fig. 11).
2. OPEN DRAIN: quasi-bidirectional I/O with only an N-channel open drain output. Application as an output requires connection of an external pull-up resistor (Fig. 12). When an open drain port is unused it must be connected to V_{SS}.
3. PUSH-PULL OUTPUT: the outputs can sink or source 1,6 mA (min.) at V_{DD} = 5 V. These pins may not be used as inputs (Fig. 13). The contents of the port latch may be read using the IN A,pp instruction.

Individual mask selection of the pin's state following a RESET can be achieved by appending option S or R to options 1, 2 or 3.

Option S-SET; following RESET the pin will be set HIGH.

Option R-RESET; following RESET the pin will be set LOW.

DEVELOPMENT DATA

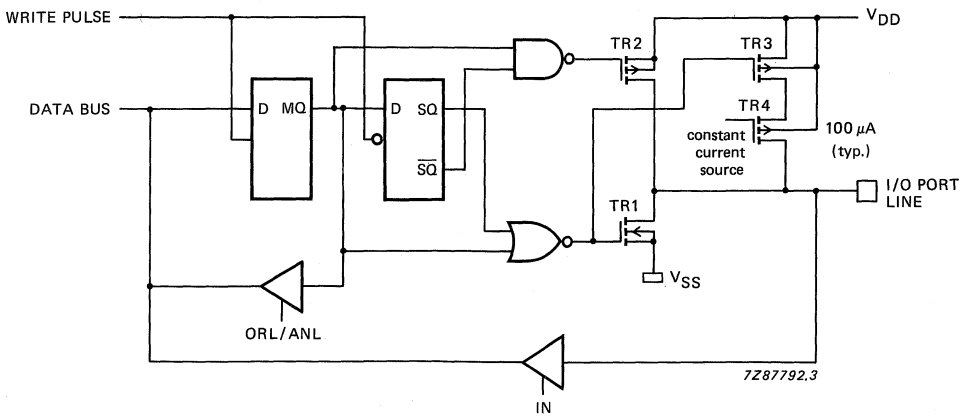


Fig. 11 Standard output with switched pull-up current source.

FUNCTIONAL DESCRIPTION (continued)

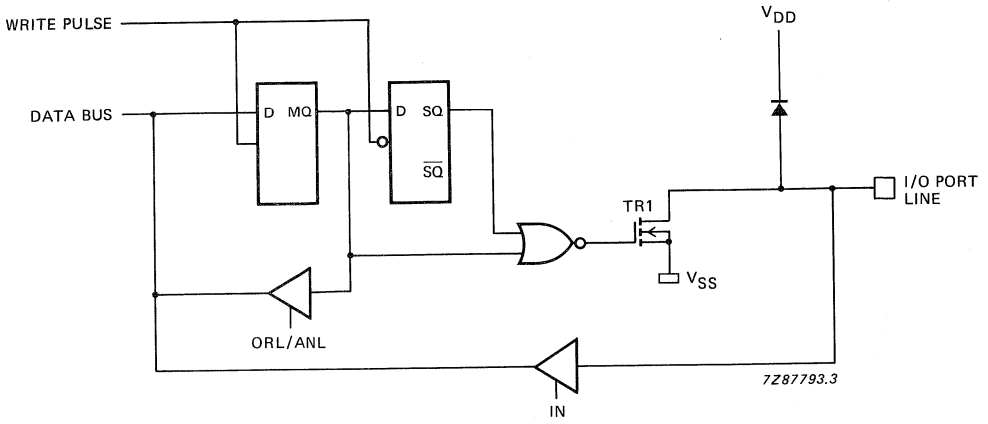


Fig. 12 Open drain output.

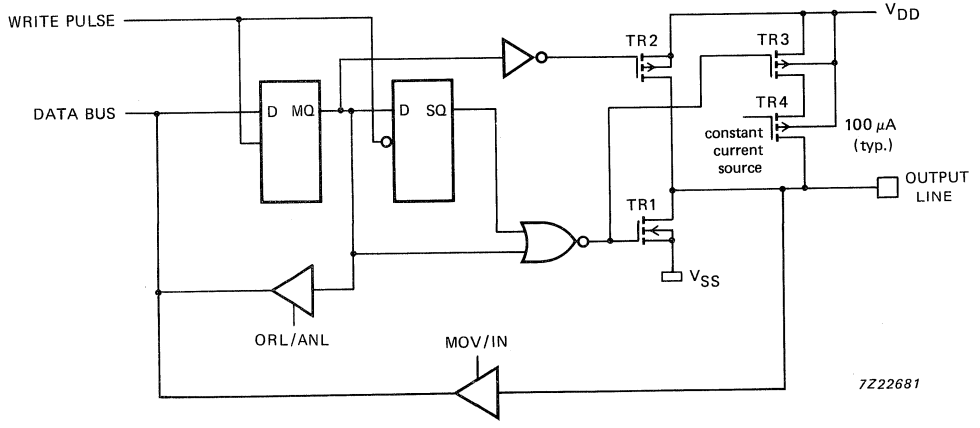


Fig. 13 Push-pull output.

Serial I/O (SIO)

The PCD3346 has an on-chip serial I/O interface tailored for I^2C-bus communications. The SIO interface is a versatile feature in intelligent telephone circuitry.

Where a normal microcontroller must regularly monitor the serial data bus for the presence of data, this serial I/O interface detects, receives and converts the serial data stream into parallel format without interrupting the execution of the current program. An interrupt is sent to the CPU only when a complete byte is received. It then reads the data byte in one instruction.

During transmission the serial I/O interface performs parallel to serial conversion and subsequent serial output of the data. The microcontroller is only interrupted in the execution of its programmed tasks when a complete byte has been transmitted.

The design of the PCD3346 serial I/O system allows any number of devices from the PCF8500 family (clips) to be connected via the two-line serial bus. The ability of any devices to communicate, without interrupting the operation of any other devices on the bus, is an outstanding attribute of the system. This is achieved by allocating a specific 7-bit address to each device and providing a system whereby a device reacts only to a message prefixed with its own address or the 'general CALL' address. Address recognition is performed by the interface hardware so that operation of the microcontroller need only be interrupted when a valid address has been received. This saves significant processing time and memory space compared with a conventional microcontroller employing a software serial interface. When the addressing facility is not required, for instance in a system with only two microcontrollers, direct data transfer without addressing can be performed. In multi-master systems, an automatically invoked arbitration procedure prevents two or more devices from continuing simultaneous transmission.

In NORMAL (running) and IDLE mode, the serial I/O logic remains active; its internal system clock will be switched off when there is no activity on the serial bus.

After execution of the STOP instruction, the oscillator of the PCD3346 is switched off. This means that the serial I/O logic will remain in the state it was in when the STOP mode was entered. To avoid "bus block" problems and to assure correct start-up of the bus after exit from the STOP mode, the user should disable the serial logic (ESO = 0) prior to the execution of the STOP instruction.

After a reset, the first 30 clock pulses of the 1866 pulse initiation phase reset the serial I/O interface and set P2.3/SDA and SCLK HIGH.

Serial I/O interface

Figure 14 shows the serial I/O interface. The clock line of the serial bus has exclusive use of the SCLK pin while the data line shares the SERIAL DATA pin with I/O line P2.3. When the serial I/O is enabled, P2.3 is disabled as a parallel port line; (P2.3 and SCLK are open drain; when unused these lines must be connected to V_{SS}).

FUNCTIONAL DESCRIPTION
(continued)

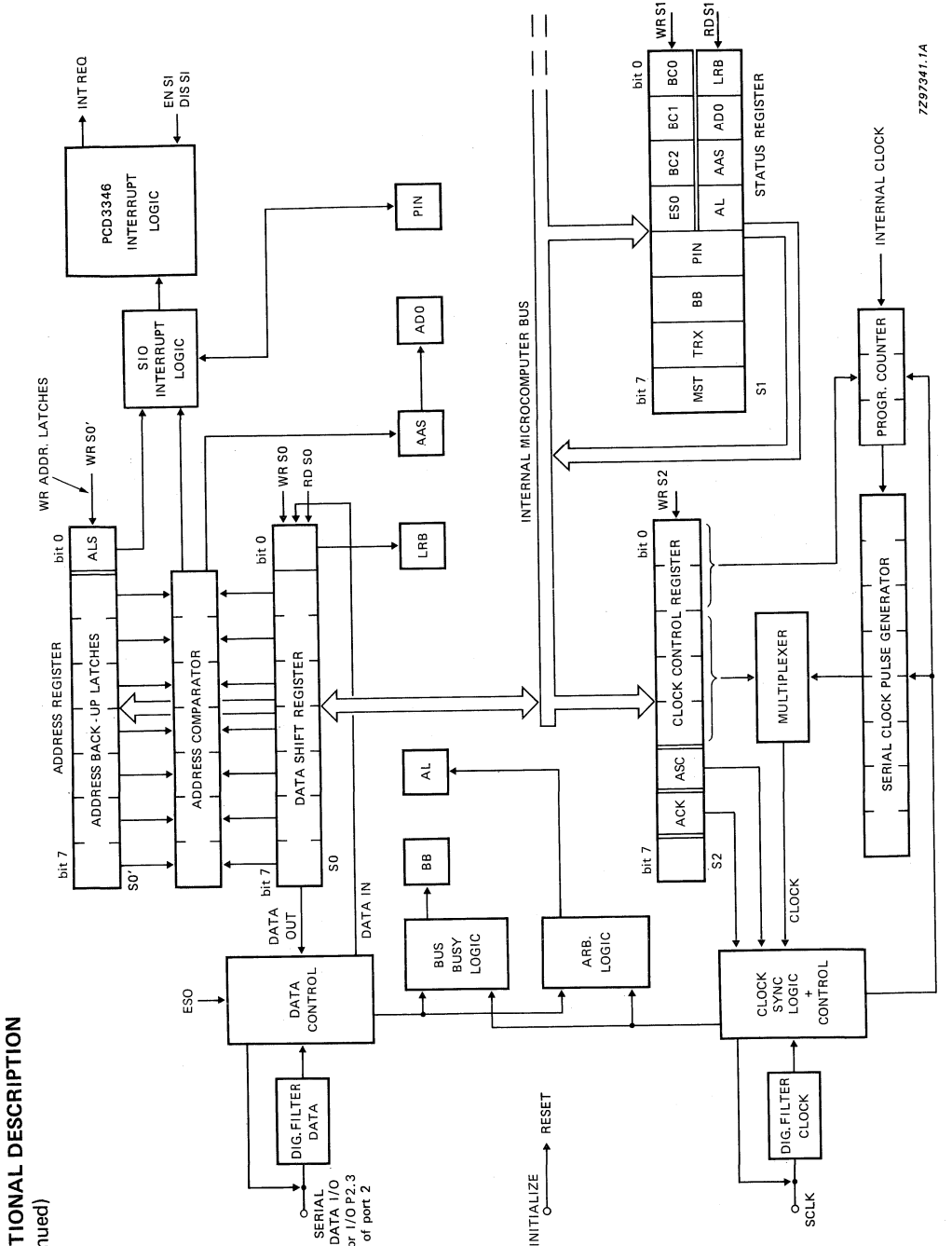


Fig. 14 Serial I/O interface.

The microcontroller and interface communicate via the internal microcontroller bus and the Serial Interrupt Request line. Data and information controlling the operation of the interface are stored in four registers:

- Data shift register (S0)
- Serial I/O interface status word (S1)
- Serial clock control word (S2)
- Address register (SO')

Data shift register (S0)

Register S0 converts serial data to parallel format and vice versa. An interrupt is generated only after a complete byte has been transmitted, or after a complete data byte, specific address or 'general CALL' address has been received. The most significant bit is transmitted first.

Serial I/O interface status word (S1)

Register S1 provides information concerning the state of the interface and stores information from the microcontroller. Bits 0 to 3 are duplicated: control bits in these positions can only be written by the microcontroller, while status bits can only be read.

MST and TRX (see Table 6)

These bits determine the operating mode of the serial I/O interface.

Table 6 Operating modes of the serial I/O interface

MST	TRX	operating mode
0	0	slave receiver
1	0	master receiver
0	1	slave transmitter
1	1	master transmitter

BB: Bus Busy.

This flag indicates the status of the bus.

PIN: Pending Interrupt NOT

PIN = '0' indicates the presence of a pending interrupt, which will cause a Serial Interrupt Request when the serial interrupt mechanism is enabled.

ESO: Enable Serial output

The ESO flag enables/disables the serial I/O interface: ESO = '1' enables, ESO = '0' disables. ESO can only be written by software.

BC0, BC1 and BC2

Bits BC0, BC1 and BC2 indicate the number of bits received or transmitted in a data stream. These bits can only be written by software.

AL: Arbitration Lost

The arbitration lost flag is set by hardware when the serial I/O interface, as master transmitter, loses a bus arbitration procedure.

FUNCTIONAL DESCRIPTION (continued)

AAS: Addressed As Slave

This flag is set by hardware when the interface detects either its own specific address or the 'general CALL' address as the first byte of a transfer and the interface has been programmed to operate in the address recognition mode.

AD0: Address Zero

This flag is set by hardware after detection of the 'general CALL' address when the interface is operating in the address recognition mode.

LRB: Last Received Bit

This contains either the last data bit received or, for a transmitting device in the acknowledgement mode, the acknowledgement signal from the receiving device.

Bits AL, AAS, AD0 and LRB can only be read by software.

Serial clock control word (S2)

Bits 0 to 4 of the clock register S2 determine the frequency of the serial clock signal. When a 6 MHz crystal is used, the frequency of the serial clock can be varied between 1 kHz and 154 kHz (see Table 7). An asymmetrical clock with a HIGH-to-LOW ratio of 3 : 1 can be generated using bit 5. The asymmetrical clock allows a microcontroller more time per clock period for sampling the data line, making the timing of this action less critical. Bit 6 can be used to activate the acknowledge mode of the serial I/O. S2 is a write only register.

Address register (S0)

The address register contains the 7-bit address back-up latches and the bit (ALS) used to enable/disable the address recognition mode. The address register can be written using the MOV S0, A and MOV S0, # data instructions, but only when ESO = logic 0.

Serial I/O interrupt logic

An EN SI instruction enables and a DIS SI instruction disables the interrupt logic. When enabled, a pending interrupt results in a serial I/O interrupt to the processor, causing a CALL to location 5 in the ROM. When disabled, the presence of an interrupt request is still indicated by PIN in S1, but the interrupt will not be serviced.

Table 7 SIO clock pulse frequency control when using a 3.58 MHz, 6 MHz or 10 MHz crystal

hexadecimal S20-S24 code	divisor	f_{XTAL} (3.58 MHz) f_{SCLK} (kHz) **	f_{XTAL} (6 MHz) f_{SCLK} (kHz) **	f_{XTAL} (10 MHz) f_{SCLK} (kHz) **
0	not allowed			
1	39	92	*154	*256
2	45	80	*133	*222
3	51	70	*118	*196
4	63	57	95	*159
5	75	48	80	*133
6	87	41	69	*115
7	99	36	61	*101
8	123	29	49	81
9	147	24	41	68
A	171	21	35	58
B	195	18	31	51
C	243	15	25	41
D	291	12	21	34
E	339	11	18	29
F	387	9.2	16	26
10	483	7.4	12	21
11	579	6.2	10	17
12	675	5.3	8.9	15
13	771	4.6	7.8	13.4
14	963	3.7	6.2	10.4
15	1155	3.1	5.2	8.7
16	1347	2.7	4.5	7.4
17	1539	2.3	3.9	6.5
18	1923	1.9	3.1	5.2
19	2307	1.6	2.6	4.3
1A	2691	1.3	2.2	3.7
1B	3075	1.2	2.0	3.3
1C	3843	0.93	1.6	2.6
1D	4611	0.78	1.3	2.2
1E	5379	0.67	1.1	1.9
1F	6147	0.58	1.0	1.6

DEVELOPMENT DATA

* Not permitted for I^2C operation.

** Maximum clock frequency for the I^2C-bus is 100 kHz.

FUNCTIONAL DESCRIPTION (continued)

Interrupts

When an interrupt routine is entered, the contents of the program counter and bits 4, 6 and 7 of the PSW are saved in the program counter stack. The contents of the accumulator can only be saved by user software. Interrupt acknowledgement may be carried out by software via I/O ports. All interrupt routines must reside in memory bank 0; the SEL MB instructions may not be used within an interrupt routine. An interrupt routine can only be terminated by the RETR (return and restore) instruction. During an interrupt routine, subroutine calls must be terminated by the RET instruction. Using the RETR instruction to terminate a subroutine called in an interrupt routine would terminate the interrupt routine prematurely and result in a wrong return address.

External Interrupts

When the external interrupt is enabled and no interrupt routine is in progress, a LOW-to-HIGH transition on the CE/$\overline{\text{TO}}$ pin sets the External Interrupt Flag (EIF) and initiates the external interrupt routine by forcing a CALL to program memory location 3.* The program counter points to the external interrupt vector address (003 H) between 2,6 and 3,6 machine cycles after the transition occurs. Interrupt latency depends on the instruction that is being executed when the transition occurs. If an interrupt routine is already in progress, an external interrupt request is stored in the External Interrupt Flag (EIF). When the external interrupt has been disabled, the request is still latched into the digital filter. Execution of a DIS I instruction cancels stored interrupt requests by clearing both the digital filter latch and the external interrupt flag.

An additional external interrupt can be created by enabling the timer/counter interrupt and loading FFH into the counter (one less than overflow). If the event counter mode is enabled by executing the STRT CNT instruction, a LOW-to-HIGH transition on the T1 input will then initiate an interrupt routine by forcing a call to the timer/counter interrupt vector (location 7).

SIO Interrupt

An interrupt request from the SIO hardware will set the PIN flag to its active LOW state. This action is independent of the Enable SIO interrupt flag. When the SIO interrupt is enabled, the active LOW PIN flag will invoke the SIO interrupt routine by forcing a CALL to program memory location 5. After the SIO interrupt is initiated, the PIN flag is not automatically reset HIGH. PIN must be cleared by software during the interrupt routine.

Timer/Counter Interrupt

When no interrupt is in progress and the timer/counter is enabled, a timer/counter overflow sets the Timer Interrupt Flag (TIF). This initiates the timer interrupt routine by forcing a CALL to program memory location 7.** If an interrupt routine is in progress, the interrupt request is stored in the timer interrrupt flag only if the timer interrupt has been enabled. Execution of a DIS TCNTI instruction cancels a previously stored interrupt request. The timer flag (TF) is set every time the timer/counter overflows and is not automatically reset when the timer/counter interrupt routine is called. It can only be cleared by the JTF and JNTF instructions or by a hardware RESET.

* This CALL clears the EIF flag.
** This CALL clears the TIF flag.

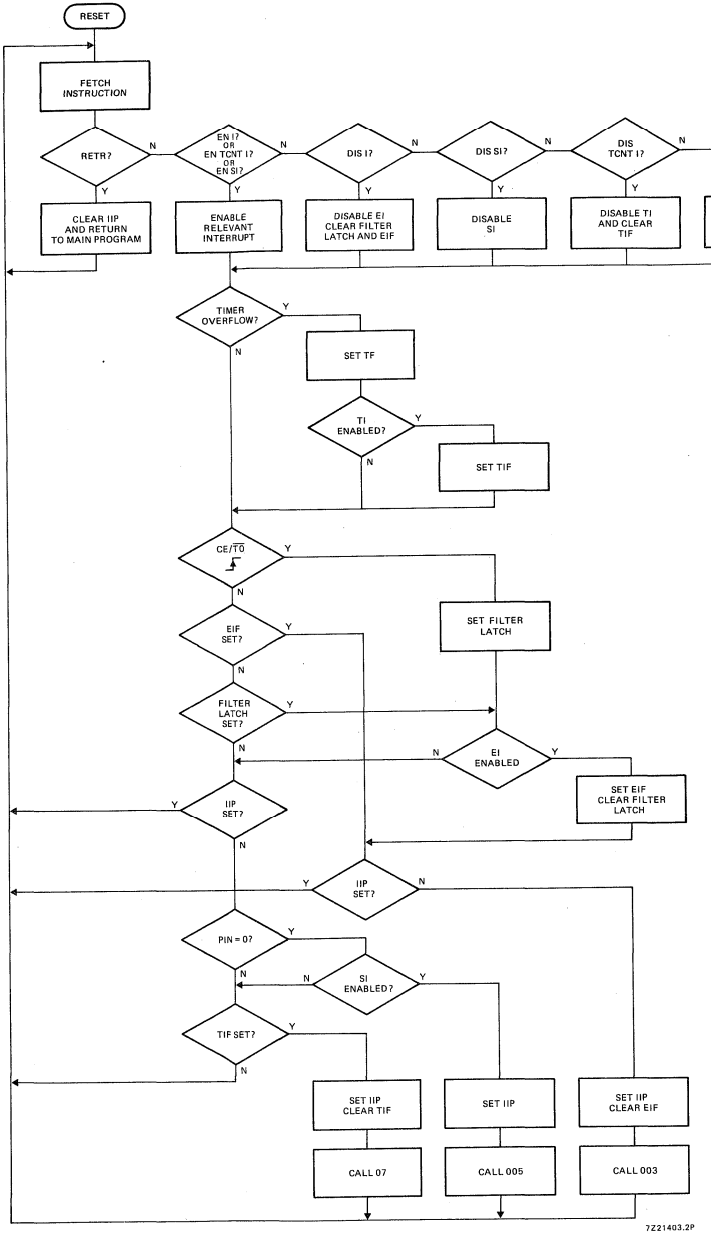
Interrupt Priority

If simultaneous interrupts occur, their priority is as follows:

External (highest)
SIO
Timer/Counter (lowest)

An interrupt routine can only be interrupted by a hardware RESET and cannot be interrupted by other interrupts (which will be latched). When the interrupt routine is terminated by the RETR instruction, at least one instruction of the main program will be executed before another interrupt routine is entered.

FUNCTIONAL DESCRIPTION (continued)



EI	External Interrupt	TF	Timer Flag
SI	Serial Interrupt	TIF	Timer Interrupt Flag
TI	Timer/counter Interrupt	PIN	Pending Interrupt Not (SIO)
EIF	External Interrupt Flag	IIP	Interrupt In Progress Flag

Fig. 15 (a) Flow chart illustrating the interrupt handling sequence.

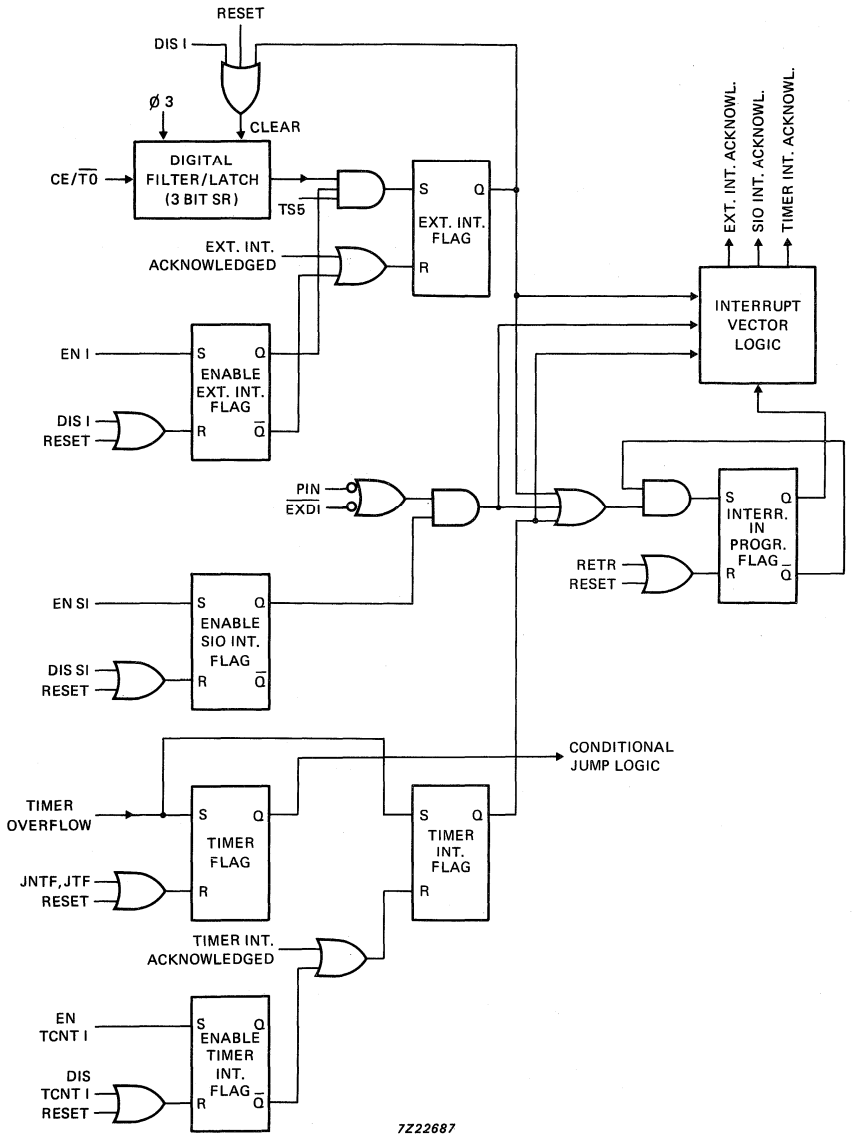


Fig. 15 (b) Simplified schematic of interrupt logic, to be used in conjunction with the functional description.

Notes to Figure 15

1. The positive edge of CE/$\overline{T0}$ is always latched in the digital filter/latch.
2. Correct interrupt timing is ensured when CE/$\overline{TO}$ is LOW for >4 CP followed by a HIGH for >7 CP.
3. When the interrupt in progress flag is set, further external and timer interrupts are latched but ignored, until RETR is executed.
4. A DIS I instruction always clears a pending external interrupt.
5. For all flip-flops, RESET overrules SET.

FUNCTIONAL DESCRIPTION (continued)

Oscillator (see Fig. 16)

The oscillator can be inhibited by the STOP instruction under software control. It is also inhibited when a low-supply voltage condition is present to prevent discharge of a weak back-up battery. Provided the supply voltage is within the operating range, the oscillator will be restarted after a STOP instruction by a HIGH level at the CE/$\overline{T0}$ pin or a HIGH level at the RESET pin.

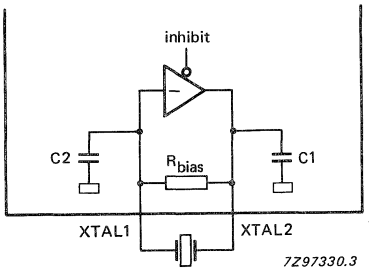


Fig. 16 Oscillator with integrated elements.

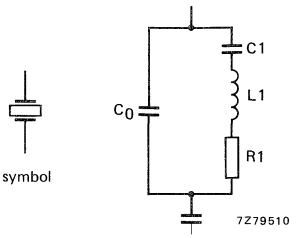


Fig. 17 Crystal unit equivalent circuit.

The values of crystal series resistance R1 and the crystal's total load capacitance C_L (C_0 + wiring + external capacitors) must not be above the curve (Fig. 18) for the corresponding frequency.
Note: if external capacitors are connected to XTAL1 and XTAL2, they must be of equal value.

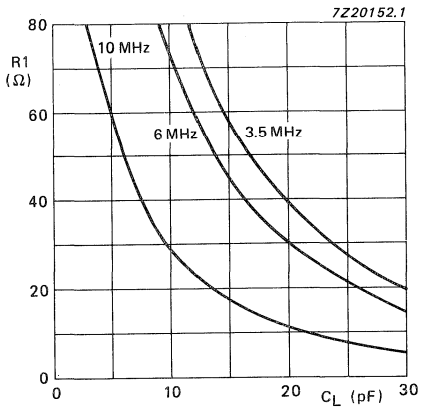


Fig. 18 Crystal circuit criteria.

XTAL2 is the output of the inverting amplifier. An external clock can be applied to XTAL1.
A machine cycle consists of 10 time slots; each time slot is 3 oscillator periods.

In telephony applications the 3.58 MHz crystal provides an 8.4 μs machine cycle.

DEVELOPMENT DATA

Timer/event counter (see Fig. 19)

An internal 8-bit up-counter is provided. This can count external events, modulo-32 machine cycles, or machine cycles directly. Table 8 shows the instructions that control the counter and the prescaler, and the functions performed.

When used as a timer, the input to the counter is either the overflow or input of a 5-bit prescaler. When used as an event counter, LOW-to-HIGH transitions on pin T1 are counted. The counter is incremented during a machine cycle only if the falling edge occurs during the first 7 time slots; otherwise it is incremented during the next cycle. The maximum rate at which the counter may be incremented is once every machine cycle. When the counter overflows, the Timer flag is set. The flag can be tested and reset using the JTF (jump if Timer flag = logic 1) or JNTF (jump if Timer flag = logic 0) instruction. Overflow also generates an interrupt request to the microcontroller by setting the Timer Interrupt Flag when the timer/event counter interrupt is enabled.

Table 8 Timer/event counter control

function	timer mode modulo-1, modulo-32*	counter mode
CLEAR	MOV T,A (A) = 0 or RESET	MOV T,A (A) = 0 or RESET
PRESET	MOV T,A	MOV T,A
START	STRT T	STRT CNT
STOP	STOP TCNT or RESET	STOP TCNT or RESET
TEST	JTF/JNTF	JTF/JNTF
READ**	MOV A,T	MOV A,T

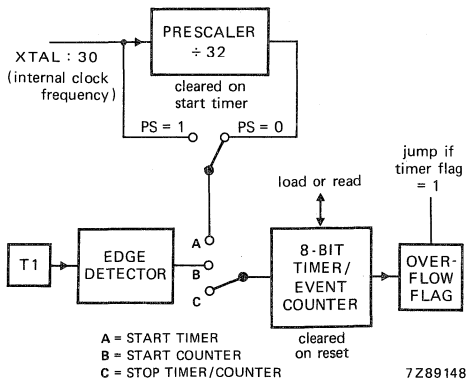


Fig. 19 Timer/event counter.

* With prescaler select (PS) = logic 0, the timer is incremented every 32 machine cycles; with PS = logic 1 the timer is incremented every machine cycle (prescaler not used), the prescaler is cleared by the STRT T instruction and is not readable.

** READ does not disturb the counting process.

FUNCTIONAL DESCRIPTION (continued)

Program status word (see Fig. 20)

The program status word (PSW) is an 8-bit word (1 byte) in the CPU which stores information about the current status of the microcontroller.
The PSW bits are:

- Bits 0 to 2 — stack pointer bits (SP_0, SP_1, SP_2)
- Bit 3 — prescaler select (PS);
  0 = modulo-32; 1 = modulo-1 (no prescaling)
- Bit 4 — working register bank select (RBS);
  0 = register bank 0; 1 = register bank 1
- Bit 5 — not used (1)
- Bit 6 — auxiliary carry (AC); half-carry bit generated by an ADD instruction and used by the decimal adjust instruction DA A
- Bit 7 — carry (CY); the carry flag indicates that the previous operation resulted in an overflow of the accumulator

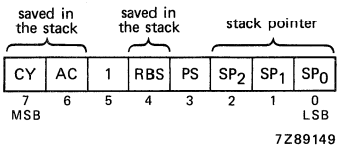


Fig. 20 Program status word.

All bits can be read using the MOV A, PSW instruction. Bits 7 and 6 are set and cleared by CPU operation. Bit 4 can be changed by the SEL RB instructions, bit 3 by the MOV PSW, A instruction, and bits 0, 1 and 2 by the CALL, RET or RETR instructions and in the event of an interrupt. Bits 7, 6 and 4 are stored in the program counter stack during subroutine and interrupt calls. These bits are restored in the PSW with a RETR (return and restore) instruction which must be used at the end of an interrupt service routine and can be used at the end of a normal subroutine. The RET instruction has no restore feature and cannot be used at the end of an interrupt service routine.

Program counter (see Fig. 21)

The 13-bit program counter is able to address 8 K bytes of ROM. The arrangement of the bits is shown in Fig. 21. During an interrupt subroutine PC 11 and PC 12 are forced to logic 0. All 13 bits are saved in the stack during CALL and interrupt routines.

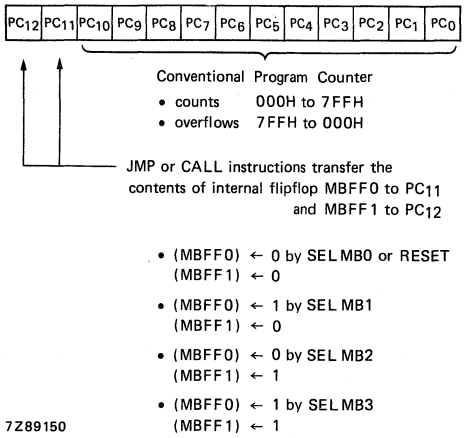


Fig. 21 Program counter.

DEVELOPMENT DATA

Central processing unit

The PCD3346 has arithmetic, logical and branching capabilities. The DA A, SWAP A and XCHD instructions simplify BCD arithmetic and the handling of nibbles. The MOVP A,@A instruction permits efficient table look-up from the current ROM page.

Conditional branch logic

The conditional branch logic within the microcontroller enables several conditions, internal and external to the microcontroller, to be tested by the user's program. Table 9 lists the conditional jump instructions used to change the program sequence. The DJNZ instruction decrements a designated register or data memory location and branches if the contents are not zero. This instruction is useful for looping control. The JMPP@A instruction allows multiway branches to destinations determined by the contents of the accumulator.

Table 9 Conditional branches

test	jump condition	jump instruction
accumulator	all bits zero	JZ
	any bit non-zero	JNZ
accumulator bit test	1	JB0 to JB7
carry flag	1	JC
	0	JNC
timer overflow flag	1	JTF
	0	JNTF
test input T0	1	JNT0
	0	JT0 *
test input T1	1	JT1
	0	JNT1
register	non-zero	DJNZ

* Because of the inverted interrupt input CE/$\overline{T0}$ the conditional jump JT0 is also inverted.

FUNCTIONAL DESCRIPTION (continued)

Test input T1

The T1 input line can be used as:

- A test input for branch instructions JT1 and JNT1
- An external input to the event counter

When used as a test input:

- JT1 instruction tests for logic 1 level
- JNT1 instruction tests for logic 0 level

When used as an input to the event counter, T1 must be LOW for $>$ 4 CP, and then HIGH for $>$ 4 CP. A transition can be recognized every 30 oscillator clock periods (1 machine cycle).

There is no internal pull-up or pull-down resistor connected to the T1 input. When T1 is not used, it must be tied to V_{DD} or V_{SS}.

Power-on-reset

The internal power-on reset circuit monitors the supply voltage V_{DD}. As long as the supply voltage remains below the internal reference level V_{ref} (typically 1.5 V), the oscillator is inhibited and RESET has an undefined level. When V_{DD} rises above the internal reference level, the oscillator is released and RESET is pulled high to V_{DD} by TR1 for a period t_D (typically 50 μs).

N.B. Because of the narrow bandwidth of the crystal, the start-up time of the oscillator is typically 10 ms.

Three applications of power-on-reset are possible:

1. If V_{DD} can be switched with a fast rise time i.e. V_{DD} reaches its minimum operating value (corresponding to the selected oscillator frequency) before the RESET signal has finished (t_D), then no extra components are required (see Figs 22 and 23). Note that the first instruction is executed after the oscillator start-up time plus 1866 clock periods have elapsed.
2. If V_{DD} has a slow rise time then the RESET signal should be stretched by an external RC circuit (see Figs 24 and 25). In the event of a short drop in the supply voltage, the diode path rapidly discharges the capacitor to ensure a reliable power-on-reset. To ensure a correct reset, the RESET signal should reach at least 70% of the final value of V_{DD}. Given that the RESET voltage and V_{DD} rise exponentially, the above requirement is satisfied when the time constant τ of the RESET pulse is $>$ 8 times the time constant of V_{DD}. If V_{DD} rises linearly, then a RESET time constant $>$ 2 times the rise time of V_{DD} is required.

   When a reset is completed (RESET goes LOW) before the oscillator has started up, program execution begins after the oscillator start-up time plus 1866 clock periods have elapsed (see Fig. 25). If the oscillator is started up prior to the completion of RESET, then program execution begins 1866 clock periods after RESET goes LOW.
3. Fig. 26 shows an external reset during power-on. The external reset signal must remain HIGH until V_{DD} has reached its minimum operating value corresponding to the selected oscillator frequency. When a reset is completed (RESET goes LOW) before the oscillator has started up, program execution begins after the oscillator start-up time plus 1866 clock periods have elapsed (see Fig. 27). If the oscillator is started-up prior to the completion of RESET, then program execution begins 1866 clock periods after RESET goes LOW.

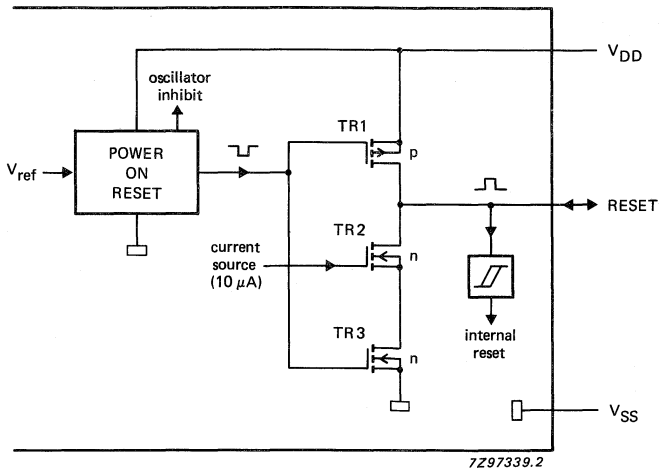


Fig. 22 Power-on-reset configuration.

DEVELOPMENT DATA

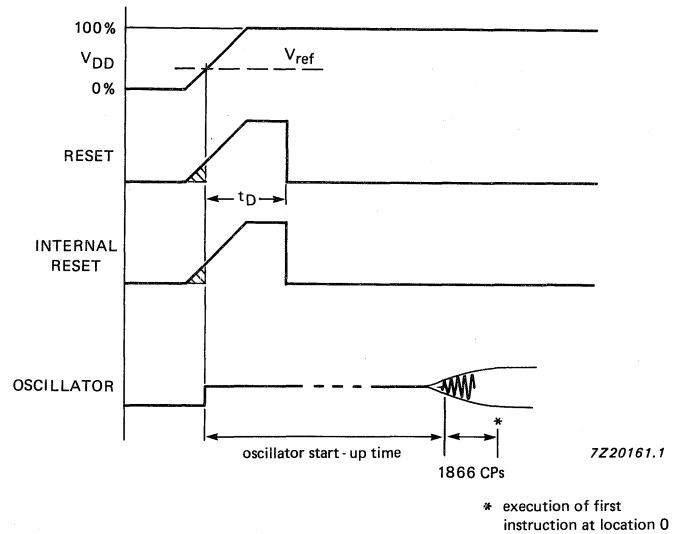


Fig. 23 Timing of power-on-reset with fast rise time.

FUNCTIONAL DESCRIPTION (continued)

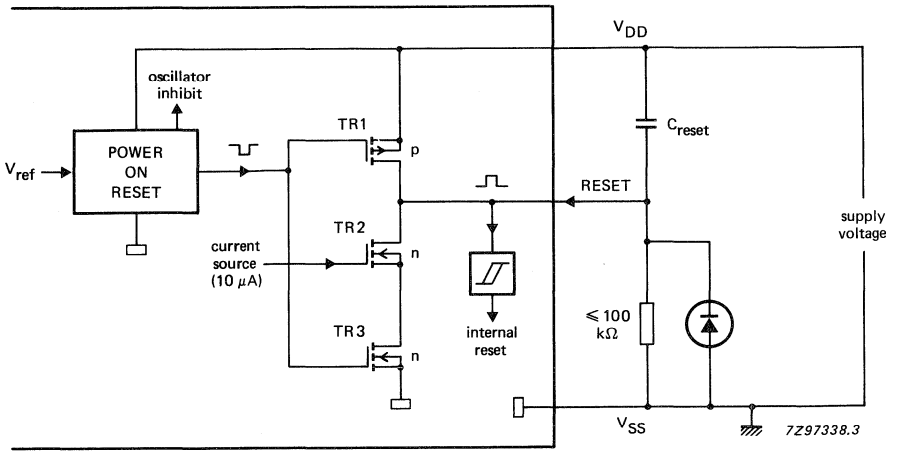


Fig. 24 Stretched power-on-reset with external components.

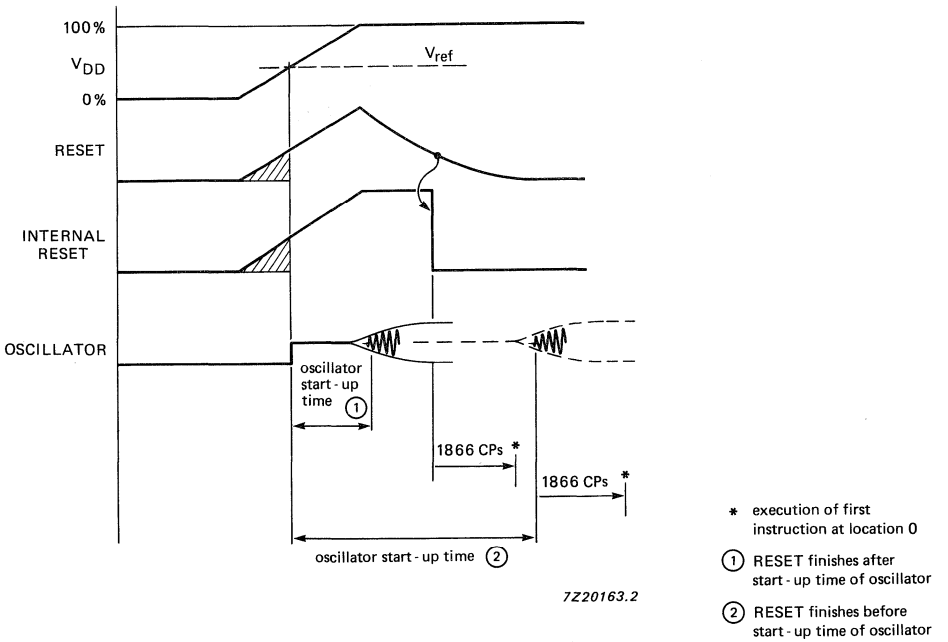


Fig. 25 Timing of power-on-reset with a slowly rising V_{DD} and a stretched RESET pulse.

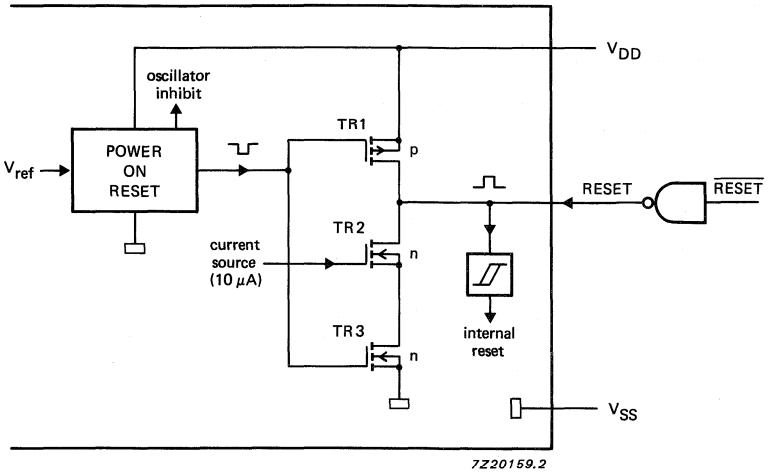


Fig. 26 External power-on-reset configuration.

DEVELOPMENT DATA

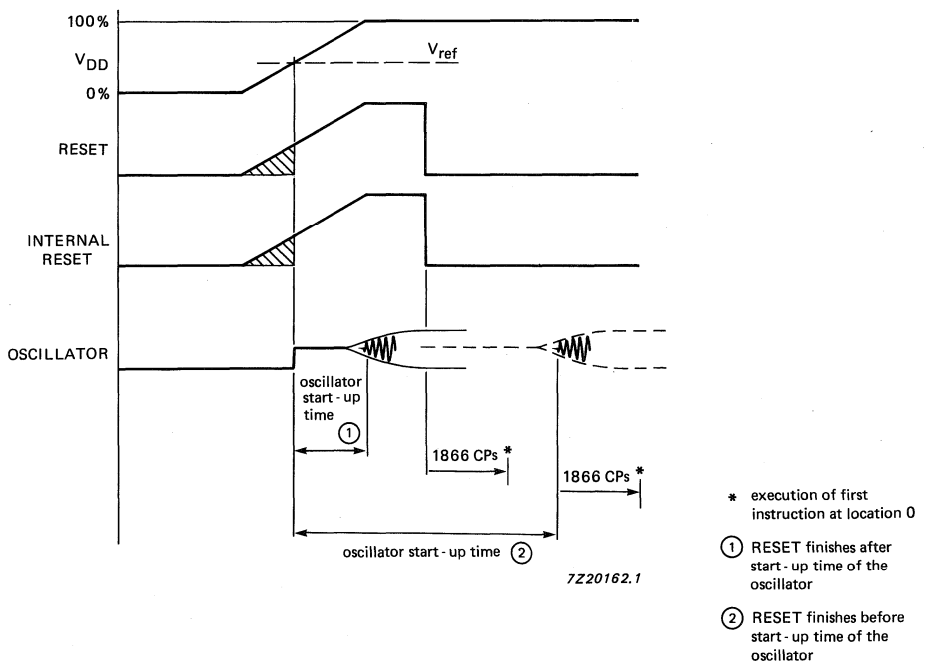


Fig. 27 Timing of external power-on-reset.

INSTRUCTION SET

The instruction set consists of over 80 one and two byte instructions. Program code efficiency is high because all RAM locations and all ROM locations on a 256 byte page require only a single byte address.

Table 12 contains the instruction set. Table 11 shows the instruction map and Table 10 details the symbols that are used.

Table 10 Symbols and definitions used in Table 12

symbol	description
A	accumulator
addr	program memory address
Bb	bit designation (b = 0-7)
RBS	register bank select
C	carry bit (bit CY)
CNT	event counter
Dx	mnemonic derivative register (x = 0 ... 255)
direct	8-bit derivative register address
data	8-bit number or expression
I	interrupt
MB	memory bank
MBFF	memory bank flip-flop
P	mnemonic for 'in-page' operation
PC	program counter
Pp	port designation (p = 0, 1 or 2)
PSW	program status word
RB	register bank
Rr	register designation (r = 0-7)
Sn	serial I/O register
SP	stack pointer
T	timer
TF	timer flag
T0, T1	test 0 and 1 inputs
#	immediate data prefix
@	indirect address prefix
(X)	contents of X
((X))	contents of location addressed by X
←	is replaced by
↔	is exchanged with

Table 11 Instruction map

first hexadecimal character of opcode ↓ / second hexadecimal character of opcode →	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
0	NOP	IDLE		ADD A, # data	JMP page 0	EN I	JNTF addr	DEC A	IN A,Pp 0	IN A,Pp 1	IN A,Pp 2		MOV A,Sn 0	MOV A,Sn 1		
1	INC @Rr 0	INC @Rr 1	JB0 addr	ADDC A, # data	CALL page 0	DIS I	JTF addr	INC A	0	1	2	INC Rr 3	4	5	6	7
2	XCH A,@Rr 0	XCH A,@Rr 1	STOP	MOV A, # data	JMP page 1	EN TCNTI	JNT0 addr	CLR A	0	1	2	XCH A,Rr 3	4	5	6	7
3	XCHD A,@Rr 0	XCHD A,@Rr 1	JB1 addr		CALL page 1	DIS TCNTI	JT0 addr	CPL A	OUTL Pp,A 0	OUTL Pp,A 1	OUTL Pp,A 2		MOV Sn,A 0	MOV Sn,A 1	MOV Sn,A 2	
4	ORL A,@Rr 0	ORL A,@Rr 1	MOV A, T	ORL A, # data	JMP page 2	STRT CNT	JNT1 addr	SWAP A	0	1	2	ORL A,Rr 3	4	5	6	7
5	ANL A,@Rr 0	ANL A,@Rr 1	JB2 addr	ANL A, # data	CALL page 2	STRT T	JT1 addr	DA A	0	1	2	ANL A,Rr 3	4	5	6	7
6	ADD A,@Rr 0	ADD A,@Rr 1	MOV T, A		JMP page 3	STOP TCNT		RRC A	0	1	2	ADD A,Rr 3	4	5	6	7
7	ADDC A,@Rr 0	ADDC A,@Rr 1	JB3 addr		CALL page 3			RR A	0	1	2	ADDC A,Rr 3	4	5	6	7
8				RET	JMP page 4	EN SI			ORL Pp, # data 0	ORL Pp, # data 1	ORL Pp, # data 2		MOV A, Dx	MOV Dx, A	ANL Dx, A	ORL Dx, A
9			JB4 addr	RETR	CALL page 4	DIS SI	JNZ addr	CLR C	ANL Pp,# data 0	ANL Pp,# data 1	ANL Pp,# data 2		MOV Sn,# data 0	MOV Sn,# data 1	MOV Sn,# data 2	
A	MOV @Rr,A 0	MOV @Rr,A 1		MOVP A,@A	JMP page 5	SEL MB2		CPL C	0	1	2	MOV Rr,A 3	4	5	6	7
B	MOV @Rr, #data 0	MOV @Rr, #data 1	JB5 addr	JMPP @A	CALL page 5	SEL MB3			0	1	2	MOV Rr, # data 3	4	5	6	7
C	DEC @Rr 0	DEC @Rr 1			JMP page 6	SEL RB0	JZ addr	MOV A, PSW	0	1	2	DEC Rr 3	4	5	6	7
D	XRL A,@Rr 0	XRL A,@Rr 1	JB6 addr	XRL A,# data	CALL page 6	SEL RB1		MOV PSW, A	0	1	2	XRL A,Rr 3	4	5	6	7
E	DJNX @Rr, addr 0	DJNX @Rr, addr 1			JMP page 7	SEL MB0	JNC addr	RL A	0	1	2	DJNZ Rr, addr 3	4	5	6	7
F	MOV A,@Rr 0	MOV A,@Rr 1	JB7 addr		CALL page 7	SEL MB1	JC addr	RLC A	0	1	2	MOV A,Rr 3	4	5	6	7

INSTRUCTION SET (continued)

Table 12 Instruction set

	mnemonic	opcode (hex.)	bytes/ cycles	description	function		notes
ACCUMULATOR	ADD A, Rr	6*	1/1	Add register contents to A	(A)←(A) + (Rr)	r = 0–7	1
	ADD A, @Rr	60 61	1/1	Add RAM data, addressed by Rr, to A	(A)←(A) + ((R0)) (A)←(A) + ((R1))		1
	ADD A, #data	03 data	2/2	Add immediate data to A	(A)←(A) + data		1
	ADDC A, Rr	7*	1/1	Add carry and register contents to A	(A)←(A) + (Rr) + (C)	r = 0–7	1
	ADDC A, @Rr	70 71	1/1	Add carry and RAM data, addressed by Rr, to A	(A)←(A) + ((R0)) + (C) (A)←(A) + ((R1)) + (C)		1
	ADDC A, #data	13 data	2/2	Add carry and immediate data to A	(A)←(A) + data + (C)		1
	ANL A, Rr	5*	1/1	'AND' Rr with A	(A)←(A) AND (Rr)	r = 0–7	
	ANL A, @Rr	50 51	1/1	'AND' RAM data, addressed by Rr, with A	(A)←(A) AND ((R0)) (A)←(A) AND ((R1))		
	ANL A, #data	53 data	2/2	'AND' immediate data with A	(A)←(A) AND data		
	ORL A, Rr	4*	1/1	'OR' Rr with A	(A)←(A) OR (Rr)	r = 0–7	
	ORL A, @Rr	40 41	1/1	'OR' RAM data, addressed by Rr, with A	(A)←(A) OR ((R0)) (A)←(A) OR ((R1))		
	ORL A, #data	43 data	2/2	'OR' immediate data with A	(A)←(A) OR data		
	XRL A, Rr	D*	1/1	'XOR' Rr with A	(A)←(A) XOR (Rr)	r = 0–7	
	XRL A, @Rr	D0 D1	1/1	'XOR' RAM, addressed by Rr, with A	(A)←(A) XOR ((R0)) (A)←(A) XOR ((R1))		
	XRL A, #data	D3 data	2/2	'XOR' immediate data with A	(A)←(A) XOR data		
	INC A	17	1/1	increment A by 1	(A)←(A) + 1		
	DEC A	07	1/1	decrement A by 1	(A)←(A) – 1		
	CLR A	27	1/1	clear A to zero	(A)←0		
	CPL A	37	1/1	one's complement A	(A)←NOT(A)		
	RL A	E7	1/1	rotate A left	$(A_{n+1})\leftarrow(A_n)$ $(A_0)\leftarrow(A_7)$	n = 0–6	

DEVELOPMENT DATA

ACCUMULATOR (cont.)	RLC A	F7	1/1	rotate A left through carry	$(A_{n+1}) \leftarrow A_n$ $(A_0) \leftarrow (C), (C) \leftarrow (A_7)$	n = 0–6	2
	RR A	77	1/1	rotate A right	$(A_n) \leftarrow (A_{n+1})$ $(A_7) \leftarrow (A_0)$	n = 0–6	
	RRC A	67	1/1	rotate A right through carry	$(A_n) \leftarrow (A_{n+1})$ $(A_7) \leftarrow (C), (C) \leftarrow (A_0)$	n = 0–6	2
	DA A	57	1/1	decimal adjust A			2
	SWAP A	47	1/1	swap nibbles of A	$(A_{4-7}) \leftrightarrow (A_{0-3})$		
DATA MOVES	MOV A, Rr	F*	1/1	move register contents to A	(A)←(Rr)	r = 0–7	
	MOV A, @Rr	F0 F1	1/1	move RAM data, addressed by Rr, to A	(A)←((R0)) (A)←((R1))		
	MOV A, #data	23 data	2/2	move immediate data to A	(A)←data		
	MOV Rr, A	A*	1/1	move accumulator contents to register	(Rr)←(A)	r = 0–7	
	MOV @Rr, A	A0 A1	1/1	move accumulator contents to RAM location addressed by Rr	((R0))←(A) ((R1))←(A)		
	MOV Rr, #data	B* data	2/2	move immediate data to Rr	(Rr)←data		
	MOV @Rr, #data	B0 data B1 data	2/2	move immediate data to RAM location addressed by Rr	((R0))←data ((R1))←data		
	XCH A, Rr	2*	1/1	exchange accumulator contents with Rr	(A)↔(Rr)	r = 0–7	
	XCH A, @Rr	20 21	1/1	exchange accumulator contents with RAM data addressed by Rr	(A)↔((R0)) (A)↔((R1))		
	XCHD A, @Rr	30 31	1/1	exchange lower nibbles of A and RAM data addressed by Rr	$(A_{0-3}) \leftrightarrow ((R0_{0-3}))$ $(A_{0-3}) \leftrightarrow ((R1_{0-3}))$		
	MOV A, PSW	C7	1/1	move PSW contents to accumulator	(A)←(PSW)		
	MOV PSW, A	D7	1/1	move accumulator bit 3 to PSW_3	$(PSW_3) \leftarrow (A_3)$		3
	MOVP A, @A	A3	1/2	move indirectly addressed data in current page to A	$(PC_{0-7}) \leftarrow (A), (A) \leftarrow ((PC))$		
FLAGS	CLR C	97	1/1	clear carry bit	(C)←0		2
	CPL C	A7	1/1	complement carry bit	(C)←NOT(C)		2

INSTRUCTION SET (continued)

	mnemonic	opcode (hex.)	bytes/ cycles	description	function	notes
REGISTER	INC Rr	1*	1/1	increment register by 1	(Rr)←(Rr) + 1 r = 0–7	
	INC @Rr	10 11	1/1	increment RAM data, addressed by Rr, by 1	((R0))←((R0)) + 1 ((R1))←((R1)) + 1	
	DEC Rr	C*	1/1	decrement register by 1	(Rr)←(Rr) − 1 r = 0–7	
	DEC @Rr	C0 C1	1/1	decrement RAM data, addressed by Rr, by 1	((R0))←((R0)) − 1 ((R1))←((R1)) − 1	
BRANCH	JMP addr	• 4 addr	2/2	unconditional jump within a 2 K bank	(PC_{8-10})←$addr_{8-10}$ (PC_{0-7})←$addr_{0-7}$ (PC_{11-12})←MBFF 0–1	
	JMPP @A	B3	1/2	indirect jump within a page	(PC_{0-7})←((A))	
	DJNZ Rr, addr	E* addr	2/2	decrement Rr by 1 and jump if not zero to addr	(Rr)←(Rr) − 1 r = 0–7 if (Rr) not zero (PC_{0-7})←addr	
	DJNZ @Rr, addr	E0	2/2	decrement RAM data, addressed by Rr, by 1 and jump if not zero to addr	((R0))←((R0)) − 1 if ((R0)) not zero (PC_{0-7})←addr	
		E1			((R1))←((R1)) − 1 if ((R1)) not zero (PC_{0-7})←addr	
	JBb addr	▲ 2 addr	2/2	jump to addr if Acc. bit b = 1	if b = 1 : (PC_{0-7})←addr b = 0–7	
	JC addr	F6 addr	2/2	jump to addr if C = 1	if C = 1 : (PC_{0-7})←addr	
	JNC addr	E6 addr	2/2	jump to addr if C = 0	if C = 0 : (PC_{0-7})←addr	
	JZ addr	C6 addr	2/2	jump to addr if A = 0	if A = 0 : (PC_{0-7})←addr	
	JNZ addr	96 addr	2/2	jump to addr if A is NOT zero	if A ≠ 0 : (PC_{0-7})←addr	
	JT0 addr	36 addr	2/2	jump to addr if T0 = 1	if T0 = 1: (PC_{0-7})←addr	
	JNT0 addr	26 addr	2/2	jump to addr if T0 = 0	if T0 = 0: (PC_{0-7})←addr	
	JT1 addr	56 addr	2/2	jump to addr if T1 = 1	if T1 = 1: (PC_{0-7})←addr	
	JNT1 addr	46 addr	2/2	jump to addr if T1 = 0	if T1 = 0: (PC_{0-7})←addr	
	JTF addr	16 addr	2/2	jump to addr if Timer Flag = 1	if TF = 1: (PC_{0-7})←addr	4
	JNTF addr	06 addr	2/2	jump to addr if Timer Flag = 0	if TF = 0: (PC_{0-7})←addr	

TIMER/EVENT COUNTER	MOV A, T	42	1/1	move timer/event counter contents to accumulator	(A)←(T)	
	MOV T, A	62	1/1	move accumulator contents to timer/event counter	(T)←(A)	
	STRT CNT	45	1/1	start event counter		
	STRT T	55	1/1	start timer		
	STOP TCNT	65	1/1	stop timer/event counter		
	EN TCNTI	25	1/1	enable timer/event counter interrupt		
	DIS TCNTI	35	1/1	disable timer/event counter interrupt		
CONTROL	EN I	05	1/1	enable external interrupt		
	DIS I	15	1/1	disable external interrupt		
	SEL RB0	C5	1/1	select register bank 0	(RBS)←0	5
	SEL RB1	D5	1/1	select register bank 1	(RBS)←1	5
	SEL MB0	E5	1/1	select program memory bank 0	(MBFF0)←0, (MBFF1)←0	
	SEL MB1	F5	1/1	select program memory bank 1	(MBFF0)←1, (MBFF1)←0	10
	SEL MB2	A5	1/1	select program memory bank 2	(MBFF0)←0, (MBFF1)←1	10
	SEL MB3	B5	1/1	select program memory bank 3	(MBFF0)←1, (MBFF1)←1	10
	STOP	22	1/1	enter STOP mode		
	IDLE	01	1/1	enter IDLE mode		
SUBROUTINE	CALL addr	▲ 4 addr	2/2	jump to subroutine	$((SP))\leftarrow(PC), (PSW_{4, 6, 7})$ $(SP)\leftarrow(SP) + 1$ $(PC_{8-10})\leftarrow addr_{8-10}$ $(PC_{0-7})\leftarrow addr_{0-7}$ $(PC_{11-12})\leftarrow$MBFF 0–1	6
	RET	83	1/2	return from subroutine	$(SP)\leftarrow(SP) - 1$ $(PC)\leftarrow((SP))$	6
	RETR	93	1/2	return from interrupt and restore bits 4, 6, 7 of PSW	$(SP)\leftarrow(SP) - 1$ $(PSW_{4, 6, 7}) + (PC)\leftarrow((SP))$	6

INSTRUCTION SET (continued)

	mnemonic	opcode (hex.)	bytes/ cycles	description	function	notes
PARALLEL INPUT/OUTPUT	IN A, Pp	08 09 0A	1/2	input port p data to accumulator	(A)←(P0) (A)←(P1) (A)←(P2)	7
	OUTL Pp, A	38 39 3A	1/2	output accumulator data to port p	(P0)←(A) (P1)←(A) (P2)←(A)	
	ANL Pp, #data	98 data 99 data 9A data	2/2	AND port p data with immediate data	(P0)←(P0) AND data (P1)←(P1) AND data (P2)←(P2) AND data	
	ORL Pp, #data	88 data 89 data 8A data	2/2	OR port p data with immediate data	(P0)←(P0) OR data (P1)←(P1) OR data (P2)←(P2) OR data	
DERIVATIVE INPUT/OUTPUT	MOV A, Dx	8C direct	2/2	move derivative register contents to accumulator	(A)←(Dx) x = 1 to 6	
	MOV Dx, A	8D direct	2/2	move accumulator contents to derivative register	(Dx)←(A) x = 1 to 6	
	ANL Dx, A	8E direct	2/2	AND derivative register with accumulator	(Dx)←(Dx) AND (A) x = 1 to 6	
	ORL Dx, A	8F direct	2/2	OR derivative register with accumulator	(Dx)←(Dx) OR (A) x = 1 to 6	

SERIAL INPUT/OUTPUT	MOV A, S_n	0C 0D	1/2	move serial I/O register contents to accumulator	(A)←(S0) (A)←(S1)	9
	MOV S_n, A	3C 3D 3E	1/2	move accumulator contents to serial I/O register	(S0)←(A) (S1)←(A) (S2)←(A)	
	MOV S_n, #data	9C data 9D data 9E data	2/2	move immediate data to serial I/O register	(S0)←data (S1)←data (S2)←data	
	EN SI	85	1/1	enable serial I/O interrupt		
	DIS SI	95	1/1	disable serial I/O interrupt		
	NOP	00	1/1	no operation		

Notes to Table 12

1. PSW CY, AC affected
2. PSW CY affected
3. PSW PS affected
4. Execution of a JTF or JNTF instruction resets the Timer Flag (TF).

* : 8, 9, A, B, C, D, E, F
● : 0, 2, 4, 5, 6, A, C, E
▲ : 1, 3, 5, 7, 9, B, D, F

5. PSW RBS affected
6. PSW SP_0, SP_1, SP_2 affected
7. (A) = 0000, P2.3, P2.2, P2.1, P2.0.
8. Instructions for 84C00T only.
9. (S1) has a different meaning for read and write operation, see serial I/O interface.
10. SEL MB1, SEL MB2 and SEL MB3 may not be used within interrupt routines.

RATINGS

Limiting values in accordance with the Absolute Maximum System (IEC 134)

parameter	symbol	min.	max.	unit
Supply voltages	V_{DD}	−0.8	+ 8	V
All input voltages	V_I	0.5	V_{DD} + 0.5	V
DC current into any input or output	$\pm I_I$ $\pm I_O$	–	10	mA
Total power dissipation (see note)	P_{tot}	–	500	mW
Power dissipation per output	P_O	–	50	mW
Storage temperature range	T_{stg}	−65	+ 150	°C
Operating ambient temperature range	T_{amb}	−25	+ 70	°C
Operating junction temperature	T_j	–	125	°C

HANDLING

Inputs and outputs are protected against electrostatic discharge in normal handling. However, to be totally safe it is desirable to take normal precautions appropriate to handling MOS devices (see 'Handling MOS devices').

THERMAL RESISTANCE

From junction to ambient

SOT117	$R_{th\ j\text{-}a}$	=	120 K/W
SOT136A	$R_{th\ j\text{-}a}$	=	150 K/W

DC CHARACTERISTICS

V_{DD} = 2.5 to 6.0 V; V_{SS} = 0 V; T_{amb} = −25 to + 70 °C; all voltages with respect to V_{SS}; unless otherwise specified

parameter	symbol	min.	typ.	max.	unit
Supply voltage operating (see Fig. 28)	V_{DD}	2.5	–	6	V
Supply current operating (see Fig. 29 and note 2)					
at V_{DD} = 5 V; f_{XTAL} = 10 MHz	I_{DD}	–	2.0	3.5	mA
at V_{DD} = 5 V; f_{XTAL} = 6 MHz	I_{DD}	–	1.2	2.4	mA
at V_{DD} = 3 V; f_{XTAL} = 3.58 MHz	I_{DD}	–	0.4	0.8	mA
IDLE mode (see Fig. 30 and note 2)					
at V_{DD} = 5 V; f_{XTAL} = 10 MHz	I_{DD}	–	1.0	2.0	mA
at V_{DD} = 5 V; f_{XTAL} = 6 MHz	I_{DD}	–	0.7	1.2	mA
at V_{DD} = 3 V; f_{XTAL} = 3.58 MHz	I_{DD}	–	0.25	0.4	mA
STOP mode (see Fig. 36 and notes 1 and 2)					
at V_{DD} = 2.5 V; T_{amb} = 70 °C	I_{DD}	–	–	10	μA
EEPROM Erase/Write cycle limitation*		10^4			
EEPROM data retention time	T_{RET}	10	–	–	years
RESET I/O					
Switching level	V_{RESET}	–	1.5	–	V
Sink current at $V_{DD} > V_{RESET}$	I_{OL}	–	7	–	μA
Inputs					
Input voltage LOW	V_{IL}	0	–	0.3 V_{DD}	V
Input voltage HIGH	V_{IH}	0.7 V_{DD}	–	V_{DD}	V
Input leakage current at $V_{SS} < V_I < V_{DD}$	$\pm I_{IL}$	–	–	1	μA
Outputs					
Output voltage LOW at V_I = V_{SS} or V_{DD}; $\|I_O\| < 1$ μA	V_{OL}	–	–	0.05	V
Output sink current LOW at V_{DD} = 3 V; V_O = 0.4 V					
except P2.3/SDA, SCLK (see Fig. 32)	I_{OL}	0.7	1.5	–	mA
P2.3/SDA, SCLK (see Fig. 33)	I_{OL}	1.5	–	–	mA
Pull-up output source current HIGH (see Fig. 34)					
at V_{DD} = 3 V; V_O = 0.9 V_{DD}	$-I_{OH}$	10	–	–	μA
at V_{DD} = 3 V; V_O = V_{SS}	$-I_{OH}$	–	–	300	μA
Push-pull output source current HIGH at V_{DD} = 3 V; V_O = V_{DD} − 0.4 V	$-I_{OH}$	0.7	1.5	–	mA

* Verified on sampling bases.

Notes to the DC characteristics

1. Crystal connected between XTAL1 and XTAL2; T1 and CE both tied to V_{SS}.
2. $V_{IL} = V_{SS}$; $V_{IH} = V_{DD}$; all outputs disconnected, all open-drain outputs connected to V_{SS}.

AC CHARACTERISTICS

V_{DD} = 2.5 to 5.5 V; V_{SS} = 0 V; T_{amb} = −40 to + 85 °C. All voltages with respect to V_{SS} unless otherwise specified

parameter	symbol	min.	typ.	max.	unit
Rise time all outputs (note 1)	t_R	–	30	–	ns
Fall time all outputs (note 1)	t_F	–	30	–	ns
Cycle time (= 30 CP; note 2)	t_{CY}	3	–	67	μs

Notes to the AC characteristics

1. At V_{DD} = 5 V; T_{amb} = + 25 °C; C_L = 50 pF.
2. 1 Time slot (TS) = 3 CP, 1 clock pulse (CP) = $1/f_{XTAL}$.

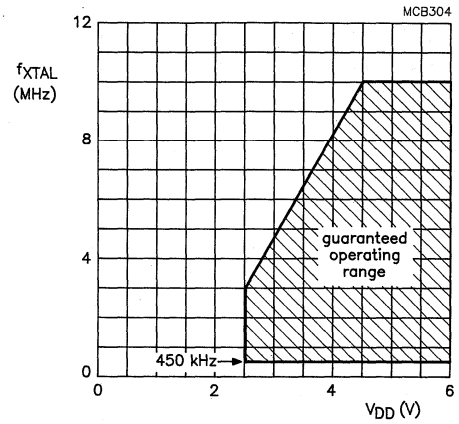
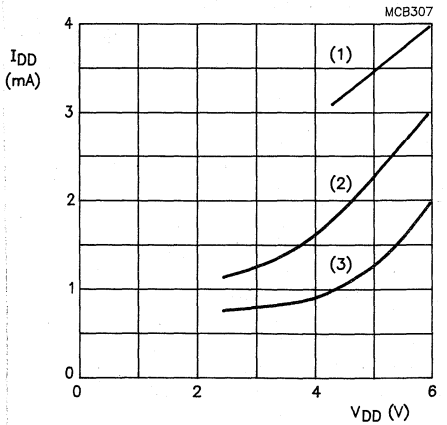


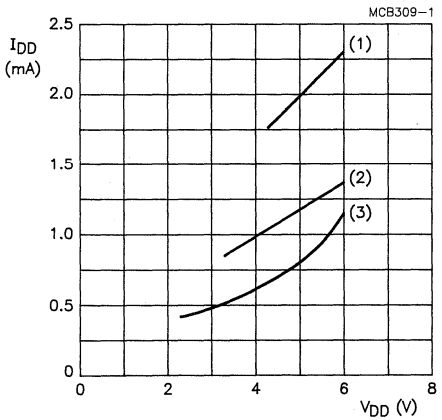
Fig. 28 Maximum clock frequency (F_{XTAL}) as a function of the supply voltage (V_{DD}).



(1) clock frequency = 10 MHz
(2) clock frequency = 6 MHz
(3) clock frequency = 3.58 MHz

Fig. 29 Maximum supply current (I_{DD}) in operation mode as a function of the supply voltage (V_{DD}).

DEVELOPMENT DATA



(1) clock frequency = 10 MHz
(2) clock frequency = 6 MHz
(3) clock frequency = 3.58 MHz

Fig. 30 Maximum supply current (I_{DD}) in IDLE mode as a function of the supply voltage (V_{DD}).

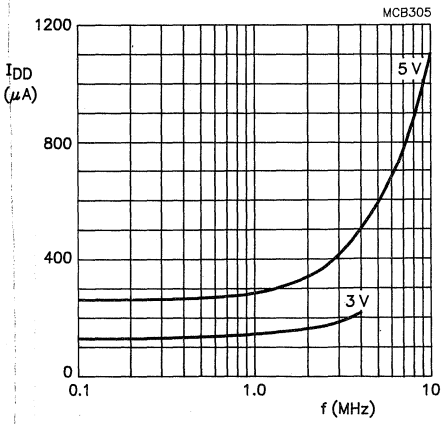


Fig. 31 Typical supply current during IDLE mode as a function of frequency at V_{DD} = 3 V and V_{DD} = 5 V.

AC CHARACTERISTICS (continued)

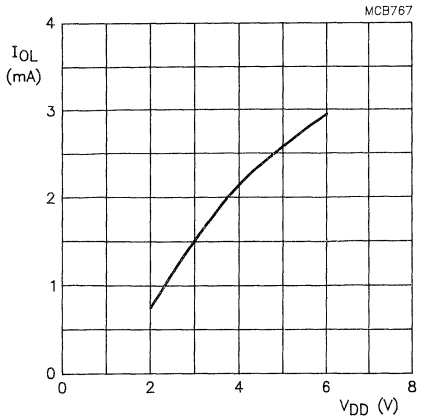


Fig. 32 Typical output sink current (I_{OL}), outputs P0.0 to P0.7 and P2.0 to P2.2, as a function of the supply voltage (V_{DD}); V_O = 0.4 V.

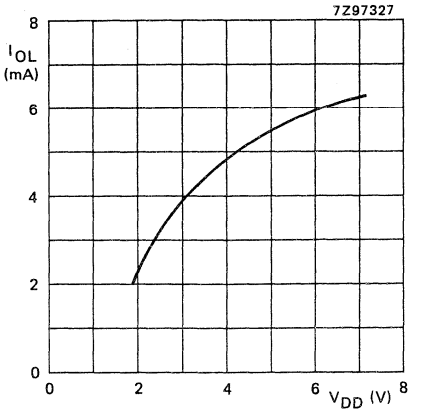
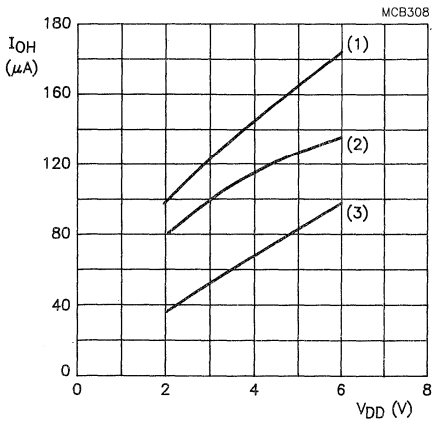


Fig. 33 Typical output sink current (I_{OL}), outputs P2.3/SDA and SCLK, as a function of the supply voltage (V_{DD}); V_O = 0.4 V.



(1) $V_O = V_{SS}$
(2) $V_O = 0.7\ V_{DD}$
(3) $V_O = 0.9\ V_{DD}$

Fig. 34 Typical output source current ($-I_{OH}$) as a function of the supply voltage (V_{DD}).

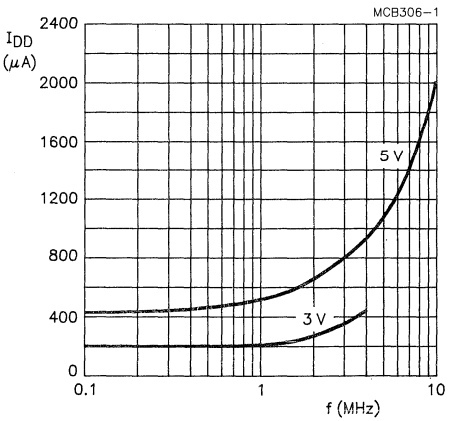
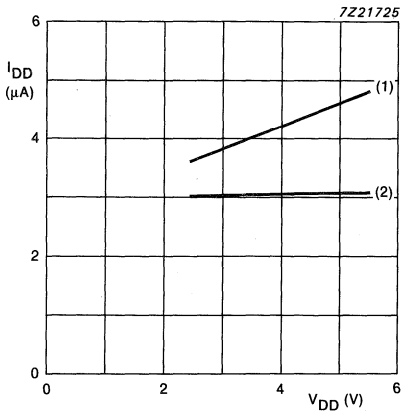


Fig. 35 Typical supply current during operating mode as a function of frequency at V_{DD} = 3 V and V_{DD} = 5 V.



(1) T_{amb} = 85 °C
(2) T_{amb} = 25 °C

Fig. 36 Typical supply current (I_{DD}) in STOP modes as a function of the supply voltage (V_{DD}).

Table 13 Input timing shown in Fig. 37

symbol	timing
t_{BUF}	$\geqslant 14t_{XTAL}$
$t_{HD;STA}$	$\geqslant 14t_{XTAL}$
t_{HIGH}	$\geqslant 17t_{XTAL}$
t_{LOW}	$\geqslant 17t_{XTAL}$
$t_{SU;STO}$	$\geqslant 14t_{XTAL}$
$t_{HD;DAT}$	> 0
$t_{SU;DAT}$	$\geqslant 250$ ns
t_{RD}	$\leqslant 1$ μs
t_{RC}	$\leqslant 1$ μs
t_{FD}	$\leqslant 1$ μs
t_{FC}	$\leqslant 0.3$ μs

Notes to Table 13

t_{XTAL} = one period of the XTAL input frequency (f_{XTAL})
= 167 ns for f_{XTAL} = 6 MHz
These figures apply to all modes.

DEVELOPMENT DATA

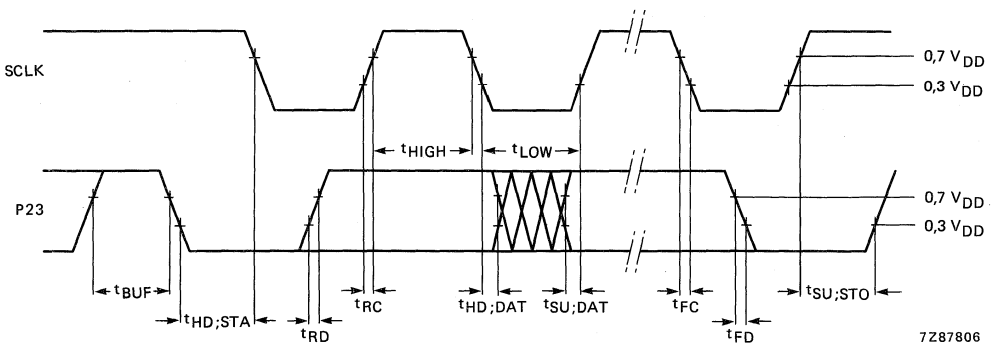


Fig. 37 Timing requirements for the P2.3 and SCLK *input* signals.

Purchase of Philips' I²C components conveys a license under the Philips' I²C patent to use the components in the I²C-system provided the system conforms to the I²C specifications defined by Philips.

AC CHARACTERISTICS (continued)

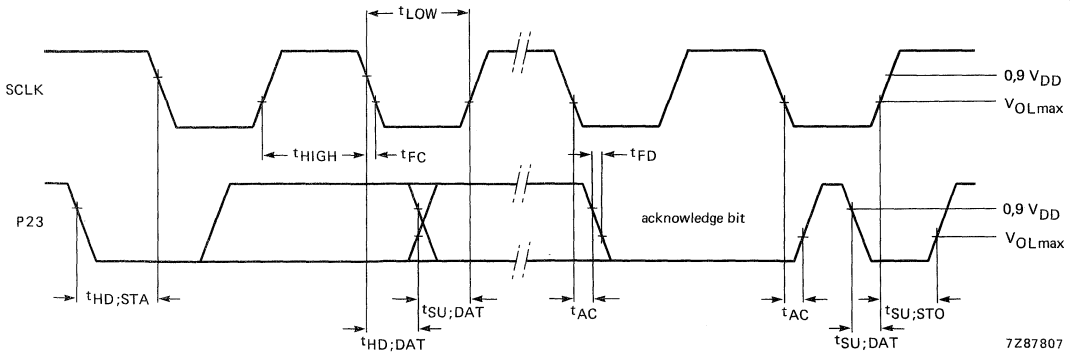


Fig. 38 Timing requirements for the P2.3 and SCLK *output* signals.

Table 14 Output timing shown in Fig. 38

symbol	timing	
	normal mode (ASC in S2 = 0)	low-speed mode (ASC in S2 = 1)
$t_{HD;STA}$	½ (DF + 9) t_{XTAL}	¾ (DF + 9) t_{XTAL}
t_{HIGH}	½ (DF) t_{XTAL}	¾ (DF) t_{XTAL}
t_{LOW}	½ (DF) t_{XTAL}	¼ (DF) t_{XTAL}
$t_{SU;STO}$	½ (DF − 3) t_{XTAL}	¼ (DF − 3) t_{XTAL}
$t_{HD;DAT}$ (slave transmitter) any DF	$\geqslant 9t_{XTAL}$ $\leqslant 12t_{XTAL}$	$\geqslant 9t_{XTAL}$ $\leqslant 12t_{XTAL}$
$t_{HD;DAT}$ (master transmitter) for DF $\leqslant$ 51) for DF $\leqslant$ 99	$\geqslant 9t_{XTAL}$ $\leqslant 12t_{XTAL}$ – –	– – $\geqslant 9t_{XTAL}$ $\leqslant 12t_{XTAL}$
$t_{SU;DAT}$ (master transmitter) for DF > 51 for DF > 99	$\geqslant 15t_{XTAL}$ $\leqslant 24t_{XTAL}$ – –	– – $\geqslant 15t_{XTAL}$ $\leqslant 24t_{XTAL}$
t_{AC}	$\geqslant 9t_{XTAL}$ $\leqslant 12t_{XTAL}$	$\geqslant 9t_{XTAL}$ $\leqslant 12t_{XTAL}$
t_{FD}, t_{FC}	$\leqslant$ 100 ns at C_b = 400 pF	$\leqslant$ 100 ns at C_b = 400 pF

Notes to Table 14

t_{XTAL} = one period of the XTAL input frequency (f_{XTAL}) = 167 ns for f_{XTAL} = 6 MHz.
DF = divisor (see Table 7 Serial I/O section).
C_b = the maximum bus capacitance for each line.

CMOS MICROCONTROLLER WITH ON-CHIP DTMF GENERATOR

GENERAL DESCRIPTION

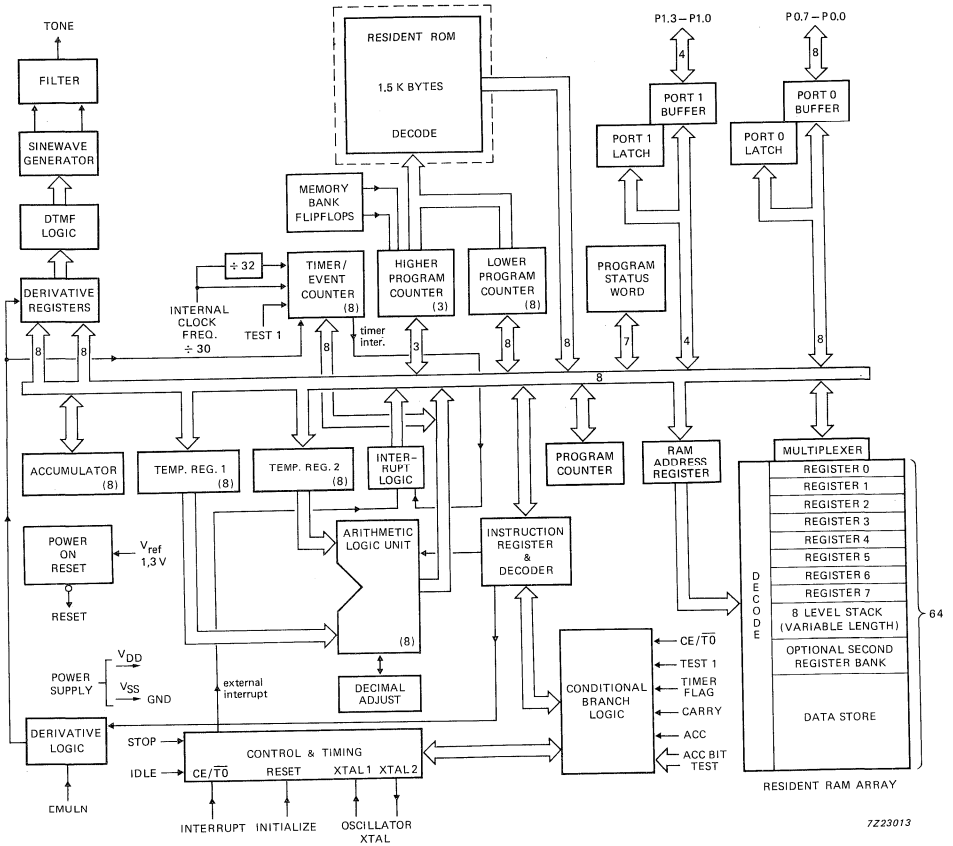
The PCD3347 is a single-chip 8-bit microcontroller fabricated in CMOS and is a member of the PCD33XX family. It has an on-chip dual tone multi-frequency (DTMF) generator and other features for application in telephone sets. For further detailed information, see PCD33XX family specification.

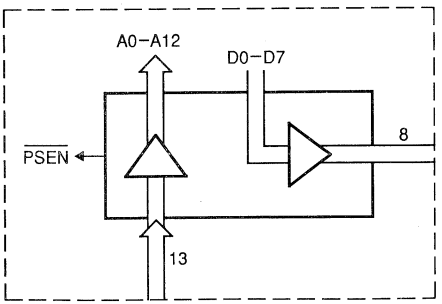
Features

- 8-bit CPU, ROM, RAM, I/O in a single 20-lead DIL or SO package
- 1536 ROM bytes
- 64 RAM bytes
- On-chip DTMF tone generator
- On-chip voltage reference for supply and temperature-independent tone output
- On-chip filtering for low output distortion (CEPT CS203 compatible)
- 12 quasi-bidirectional I/O port lines
- Two test inputs: one of which is also the external interrupt input (CE/$\overline{T0}$)
- Single-level vectored interrupts: external, timer/event counter
- 8-bit programmable timer/event counter
- Over 80 instructions (based on MAB8048)
- All instructions 1 or 2 cycles
- Clock frequency 3,58 MHz
- Single supply voltage from 2,5 V to 6 V
- Low standby voltage and current
- STOP and IDLE mode
- On-chip oscillator
- Configuration of all I/O port lines individually selected by mask: pull-up, open drain or push-pull
- Power-on-reset circuit and low supply voltage detection
- Reset state of all ports individually selected by mask
- Operating temperature range: −25 to + 70 °C

PACKAGE OUTLINES

PCD3347P: 20-lead DIL; plastic (SOT146).
PCD3347T: 20-lead mini-pack; plastic (SO20; SOT163A).





(a)

Fig. 1 PCD3347 block diagram: the function in the dotted outline is replaced as shown in (a) for the PCD3344B 'piggy-back' version.

DEVELOPMENT DATA

PINNING (for normal operation)

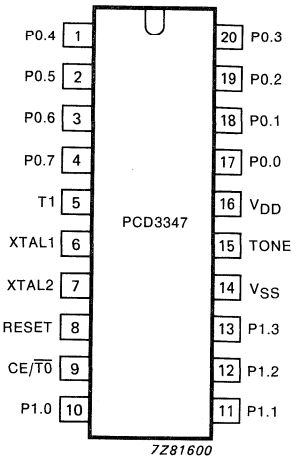


Fig. 2 Pinning diagram.

PIN DESIGNATION

17-20, 1-4	P0.0-P0.7	Port 0: 8-bit quasi-bidirectional I/O port.
5	T1	Test 1: test input, directly tested by conditional branch instructions JT1 and JNT1. T1 also functions as an input to the 8-bit timer/event counter using the STRT CNT instruction.
6	XTAL1	Crystal input: connection to the timing component (crystal) which determines the frequency of the internal oscillator; is also the input for an external clock source.
7	XTAL2	Connection to other side of timing component.
8	RESET	Reset input (active HIGH): used to initialize the processor or output of the power-on-reset circuit.
9	CE/$\overline{T0}$	Interrupt/Test 0: external interrupt input (sensitive to positive-going edge)/test input pin. When used as a test input is directly tested by conditional branch instructions JT0 and JNT0.
10-13	P1.0-P1.3	Port 1: 4-bit quasi-bidirectional I/O port.
14	V_{SS}	Ground: circuit earth potentional.
15	TONE	Tone output: single or dual tone frequency output with on-chip filtering for low output distortion (CEPT CS203 compatible). This generator is controlled via the internal processor bus.
16	V_{DD}	Power supply: 2,5 to 6 V.

FUNCTIONAL DESCRIPTION

Program memory PCD3347

The program memory comprises 1536 bytes (8-bit words) in a read-only memory (ROM). Each location is directly addressable by the program counter. The memory is mask-programmed at the factory. Figure 3 shows the program memory map.

Three program memory locations are of special importance:

- Location 0; contains the first instruction to be executed after the processor is initialized (RESET),
- Location 3; contains the first byte of an external interrupt service subroutine,
- Location 7; contains the first byte of a timer/event counter interrupt service subroutine.

The program memory is divided into location 'pages', each of 256 bytes. This division applies only for conditional branches. A CALL instruction can transfer control to a subroutine on any 'page'; RET and RETR instructions can transfer control from a subroutine back to the main program.

Data memory PCD3347

Data memory consists of 64 bytes (8-bit words) of random-access data memory (RAM). All locations are indirectly addressable using RAM pointer registers; up to 16 designated locations are directly addressable. Memory also includes an 8-level program counter stack addressed by a 3-bit stack pointer. Figure 4 shows the data memory map.

Working registers

Locations 0 to 7 are designated as working registers, directly addressable by the direct register instructions. Ease of addressing, and a minimum requirement of instruction bytes to manipulate their contents, makes these locations suitable for storing frequently-addressed intermediate results. This bank of registers can be selected by the SEL RB0 instruction.

Executing the select register bank instruction SEL RB1, designates locations 24 to 31 as working registers, instead of locations 0 to 7, and these are then directly addressable. This second bank of working registers may be used as an extension of the first or reserved for use during interrupt service subroutines saving the first bank for use in the main program. If the second bank is not used, locations 24 to 31 may serve as general purpose RAM.

The first locations of each bank contain the RAM pointer registers R0, R1, R0' and R1', which indirectly address all RAM locations.

All RAM locations make efficient program loop counters when used with the decrement register and test instruction DJNZ.

Program counter stack

Locations 8 to 23 may be designated as an 8-level program counter stack (2 locations per level), or as general purpose RAM. The program counter stack (Fig. 5) enables the processor to keep track of the return addresses and status generated by interrupts or CALL instructions by storing the contents of the program counter prior to servicing the subroutine. A 3-bit stack pointer determines which of the eight register pairs of the program counter stack will be loaded with next generated return address.

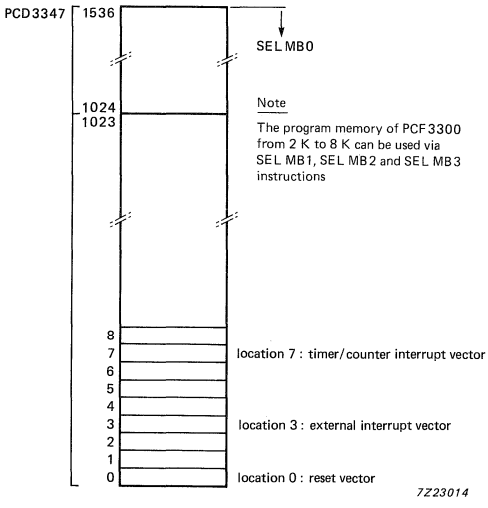


Fig. 3 Program memory map.

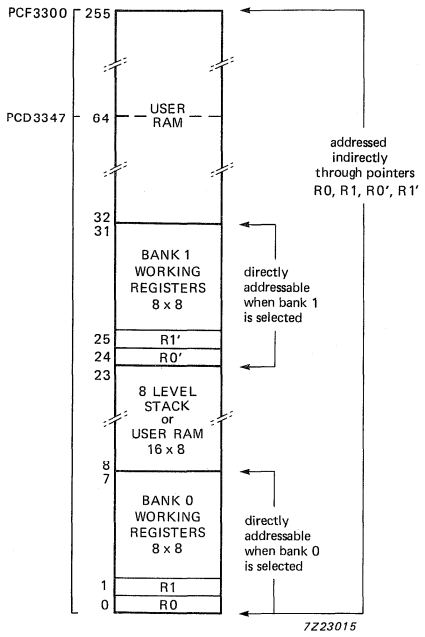


Fig. 4 Data memory map.

DEVELOPMENT DATA

The stack pointer, when initialized to 000 by RESET, points to RAM locations 8 and 9. On the first subroutine CALL or interrupt, the contents of the program counter and bits 4, 6 and 7 of the program status word (PSW) are transferred to locations 8 and 9. The stack pointer increments by one and points to locations 10 and 11 ready for another CALL. Because an address may be up to 13 bits long, two bytes must be used to store each address.

At the end of a subroutine, which is signalled by a return instruction (RET or RETR), the stack pointer decrements by one and the contents of the register pair on top of the stack are transferred to the program counter. The saved PSW bits are transferred to the PSW only by the RETR instruction.

If not all 8 levels of subroutine and interrupt nesting are used, the unused portion of the stack may be used as any other indirectly addressable RAM locations. Possible locations from 32 to 64 may be used for storage of program variables or data.

Nesting of subroutines within subroutines can continue up to 8 times without overflowing the stack. If overflow does occur the deepest address stored (locations 8 and 9) will be overwritten and lost since the stack pointer overflows from 111 to 000. It also underflows from 000 to 111.

The value of the saved contents of the program counter is different for an interrupt CALL compared to a normal CALL to subroutine. With an interrupt CALL, the program counter return address is saved; with a subroutine CALL, the saved program counter value is one less than the program counter return address.

FUNCTIONAL DESCRIPTION (continued)

Program counter stack (continued)

stack pointer									
1 1 1	----------								R23/22
1 1 0	----------								R21/20
1 0 1	----------								R19/18
1 0 0	----------								R17/16
0 1 1	----------								R15/14
0 1 0	----------								R13/12
0 0 1	----------								R11/10
0 0 0	PSW7	PSW6	PC12	PSW4	PC11	PC10	PC9	PC8	R9
	PC7	PC6	PC5	PC4	PC3	PC2	PC1	PC0	R8

Fig. 5 Program counter stack.

IDLE and STOP modes

IDLE mode

When the microcontroller enters the IDLE mode via the IDLE instruction (H'01') the oscillator and timer/counter are kept running. The microcontroller exits from the IDLE mode by one of two interrupts if they are enabled or by activating a RESET. If the interrupt is not enabled the processor will remain in the IDLE mode. An active signal on the RESET pin restarts the microcontroller and a normal RESET sequence is executed (see Fig. 6).

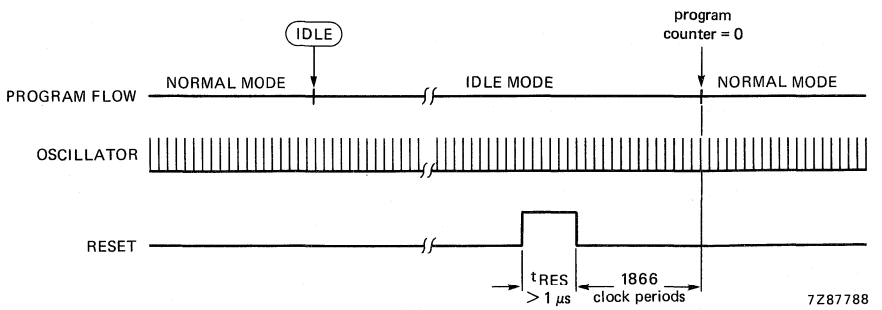


Fig. 6 Exit from IDLE mode via a RESET.

An active signal coming from an enabled interrupt causes the execution of the normal interrupt routine since normal interrupt scanning is still being carried out. A LOW-to-HIGH transition on the external interrupt pin (CE/$\overline{T0}$) reactivates the microcontroller. A HIGH level applied to CE/$\overline{T0}$ will reactivate the microcontroller only in the STOP mode. Thus, if CE/$\overline{T0}$ was HIGH before the microcontroller entered the IDLE mode, it must go LOW before the microcontroller can be reactivated (see Fig. 7).

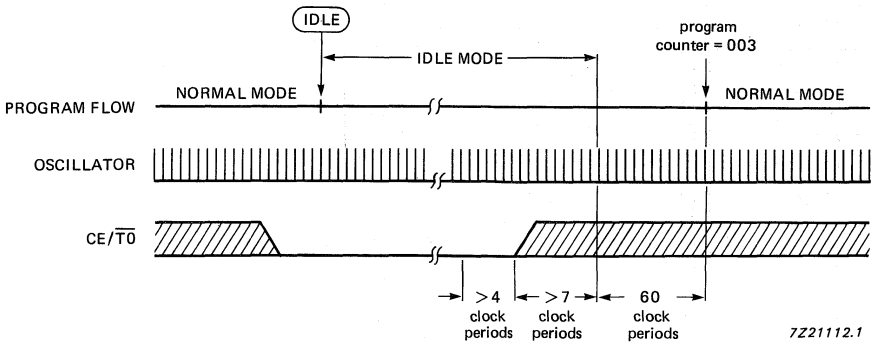


Fig. 7 Exit from IDLE mode via an interrupt.

Wake-up from the IDLE mode is ensured when CE/$\overline{T0}$ is LOW for 4 CP (clock periods) followed by a HIGH for 7 CP. After the initial forced CALL H'003' operation (60 CP) the program continues with the external interrupt service routine.

STOP mode

The microcontroller enters the STOP mode by the STOP instruction (H'22'). The oscillator is switched off. The internal status of the CPU, RAM contents and the state of I/O ports are not affected. The microcontroller can be brought-out of the STOP mode by an active signal at the external interrupt input or by an external RESET signal. When one of these two signals is applied an internal delay of 1866 CP is provided to ensure that all internal clocks are operating correctly before restart (see Fig. 8).

If the microcontroller exits from the STOP mode by activating RESET, a normal RESET sequence is executed.

If the microcontroller exits the STOP mode by pulling the external interrupt input pin HIGH, an interrupt sequence is executed only if the external interrupt is enabled. In this event the microcontroller resumes the normal program sequence after returning from the interrupt routine, as in the normal mode. If the interrupt is not enabled, it continues the normal program sequence, executing the instruction following the STOP instruction.

The microcontroller is restarted by a HIGH level applied at the CE/$\overline{T0}$ pin, and not by a LOW-to-HIGH transition as in a normal interrupt mechanism.

When the CE/$\overline{T0}$ level is active during the STOP instruction then no STOP is executed.

A HIGH level on the external interrupt input of at least 1 μs will cause the microcontroller to exit the STOP mode.

DEVELOPMENT DATA

FUNCTIONAL DESCRIPTION (continued)

STOP mode (continued)

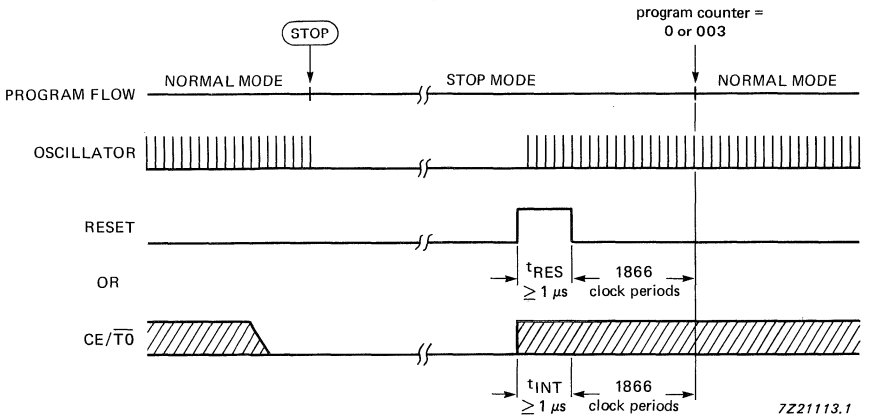


Fig. 8 Entering and exiting the STOP mode.

Tone output (DTMF mode)

Control of the sinewave generator

The on-chip sinewave oscillator is controlled by the 'derivative' registers Dx (x = H'O' to 'FF'). The instruction that controls the derivative registers is shown in Table 1.

Table 1 Derivative register control

mnemonic	opcode	description	function
MOV Dx,A	8D Dx	move accumulator contents to derivative register	(Dx) ← (A)

The instruction is 2 cycles/2 bytes. The second byte selects the derivative register to be addressed (H'O' to 'FF'). Register H'01' is for control of HIGH group frequencies, and register H'02' for control of LOW group frequencies. Thus data transport from accumulator to derivative register D01 is done by the 2-byte opcode 8D,01.

Generation of frequencies

The single and dual tones at the tone output are filtered by an on-chip switched-capacitor filter followed by an on-chip active RC low-pass filter. These ensure that the total harmonic distortion of the DTMF tones fulfil the CEPT CS 203 recommendations. An on-chip reference voltage provides output tone levels that are independent of the supply voltage.

The output frequency can be calculated as follows:

$$f_{out} = \frac{f_{XTAL}}{23\,(x + 2)} \quad \text{Hz}$$

x = 60 to 255 and is the decimal value of the appropriate ROM-code (see Table 2)

Table 2 ROM-codes for DTMF applications

telephone keyboard symbol	contents of low register (hex)	contents of high register (hex)
0	A3	72
1	DD	7F
2	DD	72
3	DD	67
4	C8	7F
5	C8	72
6	C8	67
7	B5	7F
8	B5	72
9	B5	67
A	DD	5D
B	C8	5D
C	B5	5D
D	A3	5D
*	A3	7F
#	A3	67

DTMF generation is stopped by loading H'00' into both derivative registers.

I/O facilities

The PCD3344 family has 14 I/O lines arranged as:

- Port 0 parallel port of 8 lines (P0.0 to P0.7)
- Port 1 parallel port of 4 lines (P1.0 to P1.3)
- CE/$\overline{T0}$ external interrupt and test input. When used as a test input it can be directly tested by conditional branch instructions JT0 and JNT0.
- T1 test input which can alter program sequences when tested by conditional jump instructions JT1 and JNT1. T1 also functions as an input to the 8-bit timer/event counter.

DEVELOPMENT DATA

FUNCTIONAL DESCRIPTION (continued)

Parallel ports

All parallel ports can be used as outputs or inputs, their structure is quasi-bidirectional. Output data written to a port is latched and remains unchanged until rewritten. Input data is not latched and so must be present until read by an input instruction.

Input lines are fully CMOS compatible, output lines can drive one LS-TTL or CMOS load.

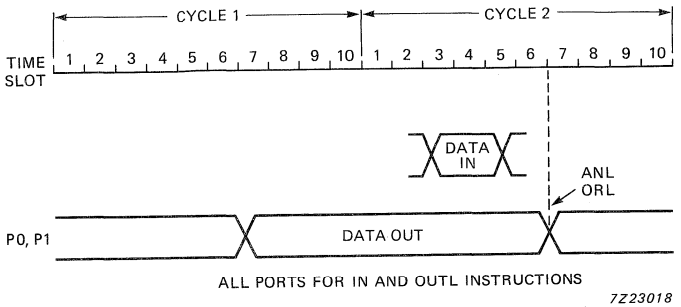


Fig. 9 Timing diagram of all ports on IN and OUTL instructions; for ANL and ORL instructions, the ports change on the time slot 7 of cycle 2.

Fig. 10 shows the quasi-bidirectional I/O interface with push-pull output and switched pull-up current source. Each line is pulled up to V_{DD} via a constant current source (TR4), which is enabled via TR3 whenever one of the two output latches contains a logic 1. This current source provides sufficient current for a TTL HIGH level, yet can be pulled LOW by an external CMOS device, thus allowing the same pin to be used for both input and output.

When a logic 1 is written to the line for the first time (MQ = 1, SQ = 0), TR2 is switched on for the duration of the internal write pulse (one oscillator period) to provide a fast transition from logic 0 to logic 1. Subsequent writing of a logic 1 to the port lines will not switch TR2 on. This prevents unnecessary current through external components connected to the port lines of the same port which might be in the input mode and also connected to ground.

When a logic 0 is written to the line, TR3 switches off the current source. Current sinking capability is provided by TR1, which is now switched on. When used as an input, a logic 1 must first be written to the line, otherwise TR1 will remain low impedance.

In telephone applications this switched pull-up source may not be sufficient. Therefore the PCD3347 offers the possibility to select individually the 12 parallel port pins by the following mask options:

Option 1 – STANDARD PORT; quasi-bidirectional I/O with switched pull-up current source of 100 μA (typ.) and P-channel booster transistor TR2 (2,5 mA). TR2 is active only during 1 clock cycle (0,28 μs at 3,58 MHz).

Option 2 – OPEN DRAIN; quasi-bidirectional I/O with only an N-channel open drain output. Application as an output requires connection of an external pull-up resistor (Fig. 11).

Option 3 – PUSH-PULL OUTPUT; drive capability of the output will be 2,5 mA (typ.) at V_{DD} = 3 V in both polarities. To avoid a large current flowing through the output transistors during the input mode, these push-pull pins must be used only as outputs (Fig. 12).

Also, individual mask selection of the RESET state of these port pins can be achieved by appending the following options S and R to options 1, 2 or 3.

Option S-SET; after RESET this pin will be initialized to HIGH

Option R-RESET; after RESET this pin will be initialized to LOW.

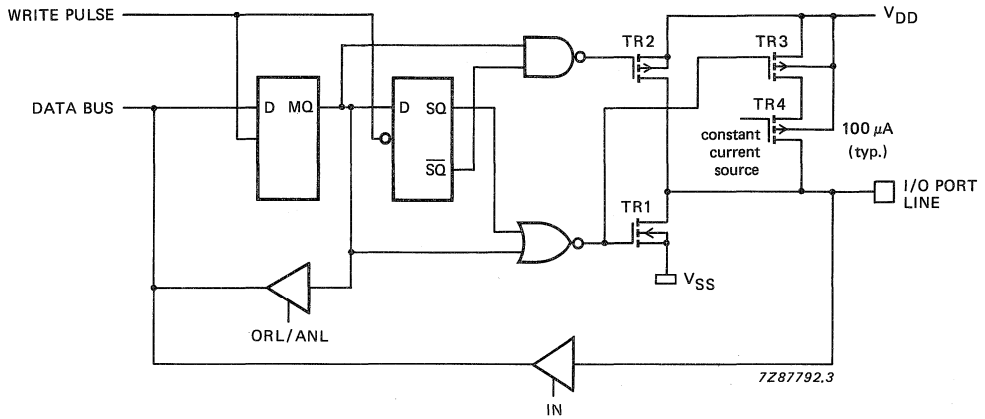


Fig. 10 Standard output with switched pull-up current source.

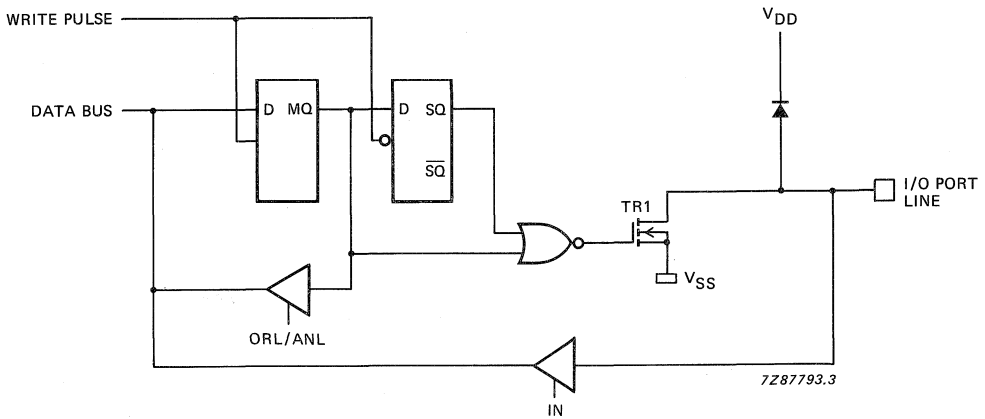


Fig. 11 Open drain output.

DEVELOPMENT DATA

FUNCTIONAL DESCRIPTION (continued)

Parallel ports (continued)

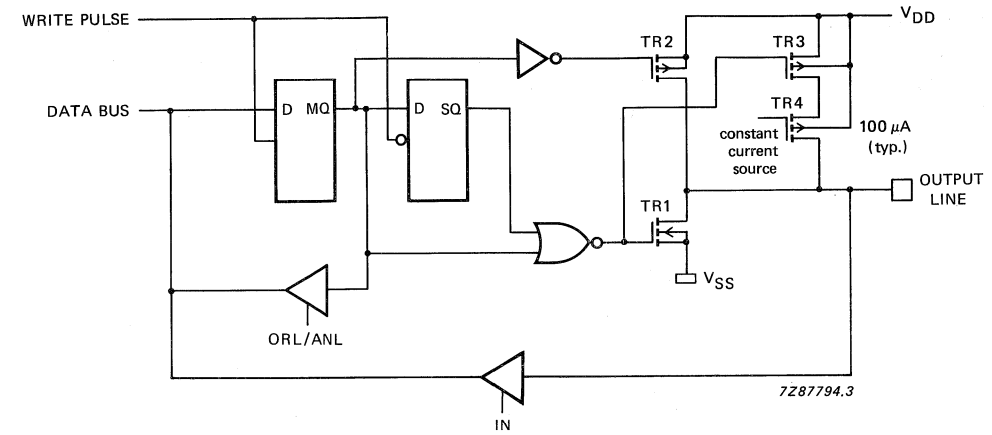


Fig. 12 Push-pull output.

Interrupts (see Fig. 13(a) and Fig. 13(b))

When an interrupt routine is entered, the contents of the program counter and bits 4, 6 and 7 of the PSW are saved in the program counter stack. The contents of the accumulator can only be saved by user software. Interrupt acknowledgement can be carried out by software via I/O ports. All interrupt routines must reside in memory bank 0; the SEL MB1, SEL MB2 and SEL MB3 instructions may not be used in an interrupt routine. An interrupt routine can only be terminated by the RETR (return and restore) instruction. During an interrupt routine, subroutine calls must be terminated by the RET instruction. Using the RETR instruction to terminate a subroutine called in an interrupt routine would terminate the interrupt routine prematurely and result in a wrong return address.

1. **External interrupt**

When the external interrupt is enabled, a HIGH-to-LOW transition on the CE/$\overline{TO}$ input initiates an external interrupt routine which forces a call to program memory location 3. The program counter points to the external interrupt vector address (003 H) between 2,6 and 3,6 machine cycles after the transition occurs. Interrupt latency depends on the instruction that is being executed when the transition occurs. External interrupts are latched in the External Interrupt Flag (EIF) even when they are not enabled. Execution of a DIS I instruction clears previously latched interrupts, the digital filter latch and the external interrupt flag.

2. **Timer/counter interrupt**

When the timer interrupt is enabled, a timer/counter overflow sets the Timer Interrupt Flag (TIF) and forces a CALL to location 7. The timer interrupts are only latched when they are enabled. The timer flag is set every time the timer/counter overflows and is not automatically reset when the timer/counter interrupt routine is called. It can only be cleared by the JTF and JNTF instructions or by a hardware RESET.

3. Simultaneous interrupts

If simultaneous interrupts occur their priority is as follows:

external (highest);
timer/counter (lowest).

An interrupt routine can only be interrupted by a hardware RESET and cannot be interrupted by other interrupts (which will be latched if enabled). When the interrupt routine is terminated by the RETR instruction, at least one instruction of the main program will be executed before another interrupt routine is entered.

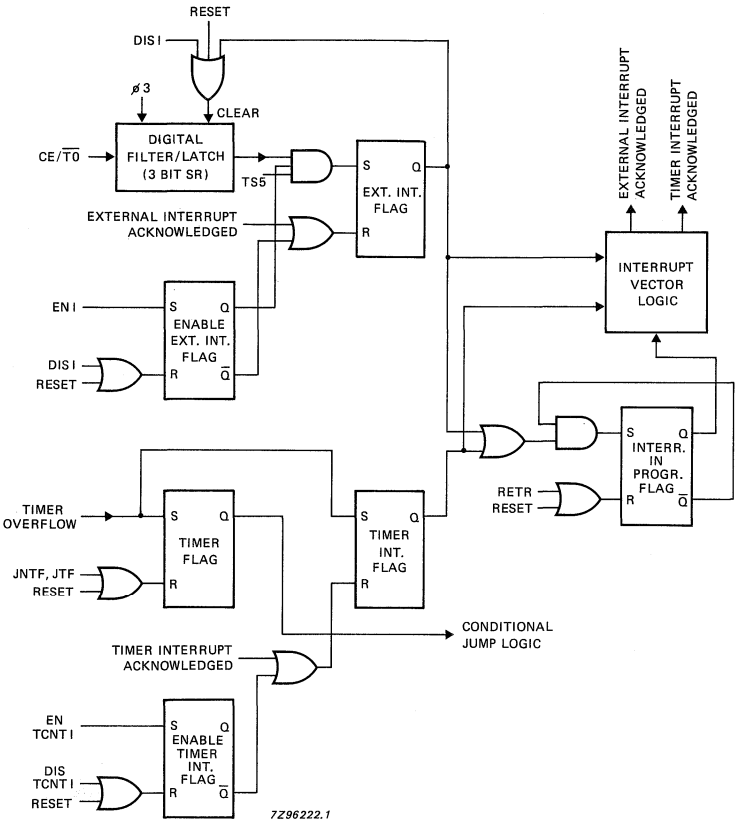


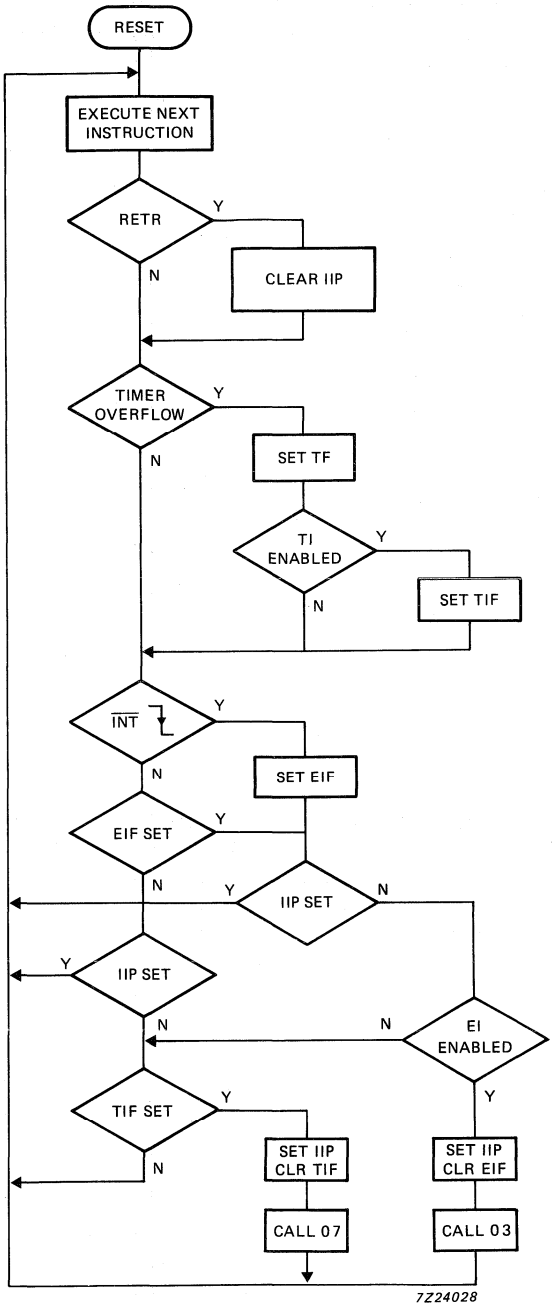
Fig. 13(a) Interrupt logic.

Notes to figure 13(a)

1. CE/$\overline{T0}$ positive edge is always latched in the digital filter/latch.
2. Correct interrupt timing is ensured when CE/$\overline{T0}$ is LOW for $>$ 4 CP followed by a HIGH for $>$ 7 CP.
3. When the interrupt in progress flag is set, further interrupts are latched but ignored, until RETR is executed.
4. A DIS I instruction always clears a pending external interrupt.
5. For all flip-flops, RESET overrules SET.

FUNCTIONAL DESCRIPTION (continued)

Interrupts (continued)



EI External Interrupt
TI Timer/counter Interrupt
EIF External Interrupt Flag
TF Timer Flag
IIP Interrupt In Progress flag

Fig. 13(b) Interrupt flowchart.

Oscillator (see Fig. 14)

The 3,58 MHz oscillator can be inhibited by the STOP instruction under software control. It is also inhibited when a low-voltage condition is present to prevent discharge of a weak back-up battery.

Provided the supply voltage is within the operating range, the oscillator will be restarted after a STOP instruction by a HIGH level at either the CE/$\overline{T0}$ or RESET pin.

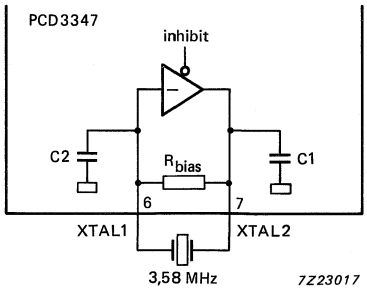


Fig. 14 Oscillator with integrated elements.

The oscillator has an output drive capability from pin 7 (XTAL2). An external clock can be applied to pin 6 (XTAL1). A machine cycle comprises 10 time slots, each time slot being 3 oscillator periods.

In telephony applications the 3,58 MHz crystal provides an 8,4 μs machine cycle. The range of the clock frequency is from 100 kHz up to a maximum which is a function of the supply voltage.

Timer/event counter (see Fig. 15)

An internal 8-bit up-counter is provided. This can count external events, modulo-32 machine cycles, or machine cycles directly. Table 3 gives the instructions that control the counter and the prescaler, and the functions performed.

When used as a timer, the input to the counter is either the overflow or input of a 5-bit prescaler. When used as an event counter, LOW-to-HIGH transitions on pin 8 (T1) are counted. The maximum rate at which the counter may be incremented is once every machine cycle (182,6 kHz for an 8,4 μs machine cycle). When the counter overflows, the timer flag is set. The flag can be tested and reset using the JTF (jump if timer flag = 1) or JNTF instruction. Overflow also generates an interrupt to the processor via setting of the Timer Interrupt Flag when the timer/event counter interrupt is enabled.

FUNCTIONAL DESCRIPTION (continued)

Timer/event counter (continued)

Table 3 Timer/event counter control

function	timer mode modulo-1, modulo-32*	counter mode
CLEAR	MOV T,A (A) = 0 or RESET	MOV T,A (A) = 0 or RESET
PRESET	MOV T,A	MOV T,A
START	STRT T	STRT CNT
STOP	STOP TCNT or RESET	STOP TCNT or RESET
TEST	JTF/JNTF	JTF/JNTF
READ**	MOV A,T	MOV A,T

* With prescaler select, PS = 0, the timer counts modulo-32 machine cycles, with PS = 1 it counts modulo-1 cycles (prescaler not used); prescaler cleared with STRT T, prescaler not readable.

** READ does not disturb the counting process.

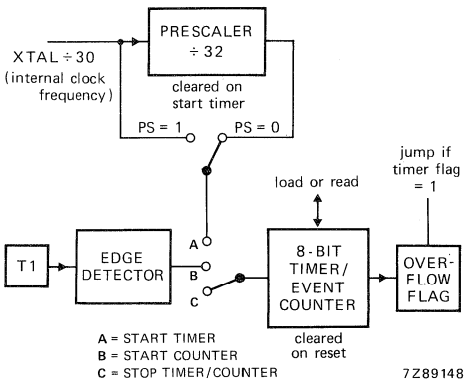


Fig. 15 Timer/event counter.

Program status word (see Fig. 16)

The program status word (PSW) is an 8-bit word (1 byte) in the CPU which stores information about the current status of the microcontroller.
The PSW bits are:

- Bits 0 to 2 — stack pointer bits (SP_0, SP_1, SP_2)
- Bit 3 — prescaler select (PS); 0 = modulo-32; 1 = modulo-1 (no prescaling)
- Bit 4 — working register bank select (RBS); 0 = register bank 0; 1 = register bank 1
- Bit 5 — not used (1)
- Bit 6 — auxiliary carry (AC); half-carry bit generated by an ADD instruction and used by the decimal adjust instruction DA A
- Bit 7 — carry (CY); the carry flag indicates that previous operation has resulted in an overflow of the accumulator.

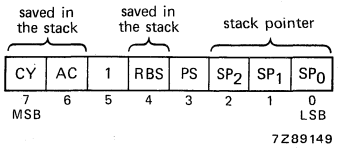


Fig. 16 Program status word.

All bits can be read using the MOV A, PSW instruction. Bits 7 and 6 are set and cleared by CPU operation. Bit 4 can be changed by a SEL RB instruction, bit 3 by the MOV PSW, A instruction, and bits 0, 1 and 2 by the CALL, RET or RETR instructions and in the event of an interrupt. Bits 7, 6 and 4 are stored in the program counter stack during subroutine and interrupt calls. These bits are restored in the PSW with a RETR (return and restore) instruction which must be used at the end of an interrupt and can be used at the end of a normal subroutine. The RET instruction has no restore feature and cannot be used at the end of an interrupt.

Program counter (see Fig. 17).

A 12-bit program counter is used to facilitate 8 K bytes of ROM being addressed. The arrangement of the bits is shown in Figure 17. During an interrupt subroutine PC_{11} and PC_{12} are forced to logic 0. All 12 bits are saved in the stack during CALL and interrupt routines.

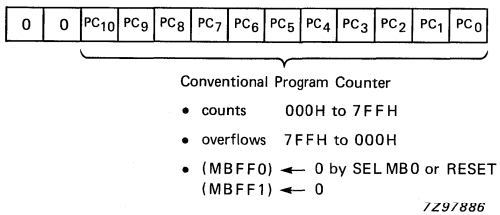


Fig. 17 Program counter.

Central processing unit

The PCD3347 has arithmetic, logical and branching capabilities. The DA A, SWAP A and XCHD instructions simplify BCD arithmetic and the handling of nibbles. The MOVP A,@A instruction permits efficient table look-up from the CURRENT ROM page.

FUNCTIONAL DESCRIPTION (continued)

Conditional branch logic

The conditional branch logic within the processor enables several conditions, internal and external to the processor, to be tested by the user's program.
Table 4 lists the conditional jump instructions used to change the program sequence. The DJNZ instruction decrements a designated register or data memory location and branches if the contents are not zero. This instruction is useful for looping control. The JMPP@A instruction allows multiway branches to destinations determined by the contents of the accumulator.

Table 4 Conditional branches

test	jump condition	jump instruction
accumulator	all bits zero	JZ
	any bit non-zero	JNZ
accumulator bit test	1	JB0 to JB7
carry flag	1	JC
	0	JNC
timer overflow flag	1	JTF
	0	JNTF
test input T0	1	JNT0
	0	JT0*
test input T1	1	JT1
	0	JNT1
register	non-zero	DJNZ

* Because of the inverted interrupt input CE/$\overline{T0}$ the conditional jump JT0 is also inverted.

Test input T1 (pin 8)

The T1 input line can be used as:

- A test input for branch instructions JT1 and JNT1
- An external input to the event counter

When used as a test input:

- JT1 instruction tests for logic 1 level
- JNT1 instruction tests for logic 0 level

When used as an input to the event counter, T1 must be LOW for $>$ 4 CP, followed by a HIGH for $>$ 4 CP. The transition can be recognized with a repetition rate of once per 30 oscillator clock periods (1 machine cycle).

There is no internal pull-up or pull-down resistor connected to the T1 input. If required it must be externally connected to a resistor ($R = \leqslant 100\ k\Omega$).
When T1 is not used pin 8 must be connected to V_{DD} or V_{SS}.

Reset (pin 11)

A positive-going signal on the RESET input/output:

- Sets the program counter to zero
- Selects location 0 of memory bank 0 and register bank 0
- Sets the stack pointer to zero (000); pointing to RAM address 8
- Disables the interrupts (external and timer)
- Stops the timer/event counter, then sets it to zero
- Sets the timer prescaler to modulo-32
- Resets the timer flag
- Sets all ports according to reset states
- Cancels IDLE and STOP mode

After the voltage is applied to RESET an internal delay of 1866 CP is introduced before the microcontroller commences operation.

Power-on reset

The internal power-on reset circuit monitors the supply voltage V_{DD}. As long as V_{DD} remains below the internal reference level V_{ref} (typically 1,3 V), the oscillator is inhibited and RESET (pin 8) has an undefined level. When V_{DD} rises above V_{ref}, the oscillator is released and RESET is pulled HIGH to V_{DD} by TR1 for a period t_D (typically 50 μs). Note that the start-up time of the oscillator is typically 10 ms because of the narrow bandwidth of the crystal.

Three modes of power-on reset are possible:

1. If V_{DD} has a fast rise time, i.e. V_{DD} reaches its minimum value before the RESET signal finishes (t_D), then no additional circuit is required (see Figs 18 and 19). Note that the first instruction is executed after the oscillator start-up time plus 1866 clock periods.
2. If V_{DD} has a slow rise time then the RESET signal should be stretched by an external CR circuit (see Figs 20 and 21). In the event of a short drop in V_{DD}, the diode path discharges the capacitor rapidly to ensure a reliable power-on reset. The RESET signal should reach at least 70% of the final value of V_{DD} to ensure a correct reset. Given that the RESET voltage and V_{DD} rise exponentially, the above requirement is satisfied when the time constant of the RESET pulse is >8 times the time constant of V_{DD}. If V_{DD} rises linearly then a RESET time constant >2 times the rise time of V_{DD} is required.

   When a reset is completed (RESET goes LOW) before the oscillator has started up, program execution begins after the oscillator start-up time plus 1866 clock periods (see Fig. 21). If the oscillator starts up prior to the completion of RESET then program execution begins 1866 clock periods after RESET goes LOW.
3. Fig. 22 shows an external reset applied during power-on. The external reset signal must remain HIGH until V_{DD} has reached its minimum operating value corresponding to the selected oscillator frequency. When a reset is completed (RESET goes LOW) before the oscillator has started up, program execution begins after the oscillator start-up time plus 1866 clock periods (see Fig. 23). If the oscillator starts up prior to the completion of RESET then program execution begins 1866 clock periods after RESET goes LOW.

DEVELOPMENT DATA

FUNCTIONAL DESCRIPTION (continued)

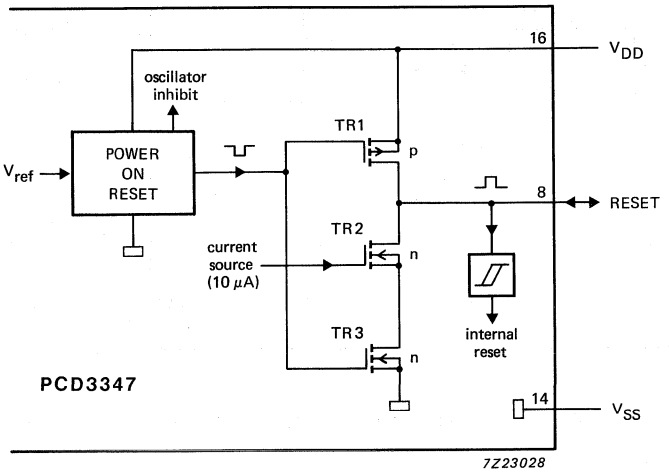


Fig. 18 Power-on reset configuration.

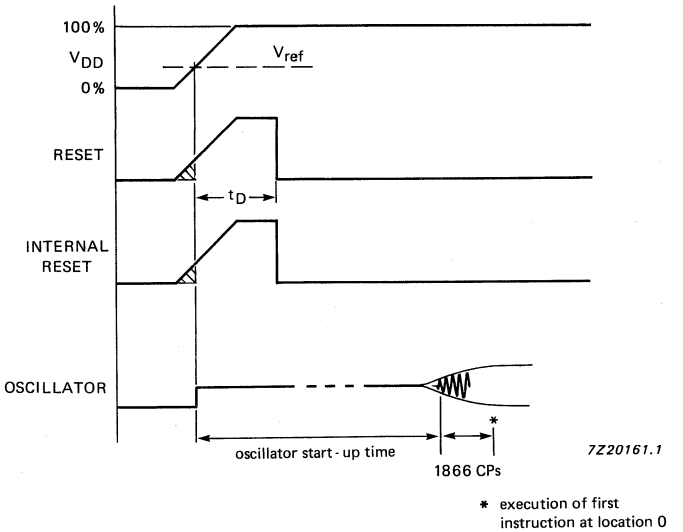


Fig. 19 Timing of power-on reset with fast rise time of V_{DD}.

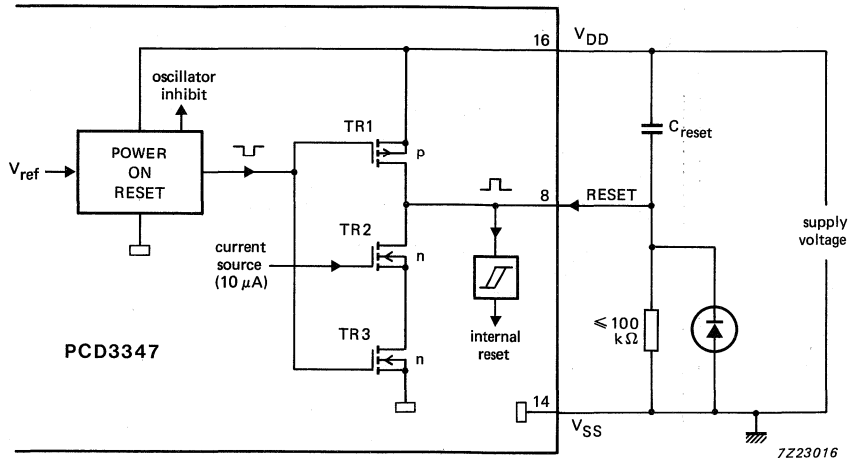


Fig. 20 Stretched power-on reset with external CR circuit.

DEVELOPMENT DATA

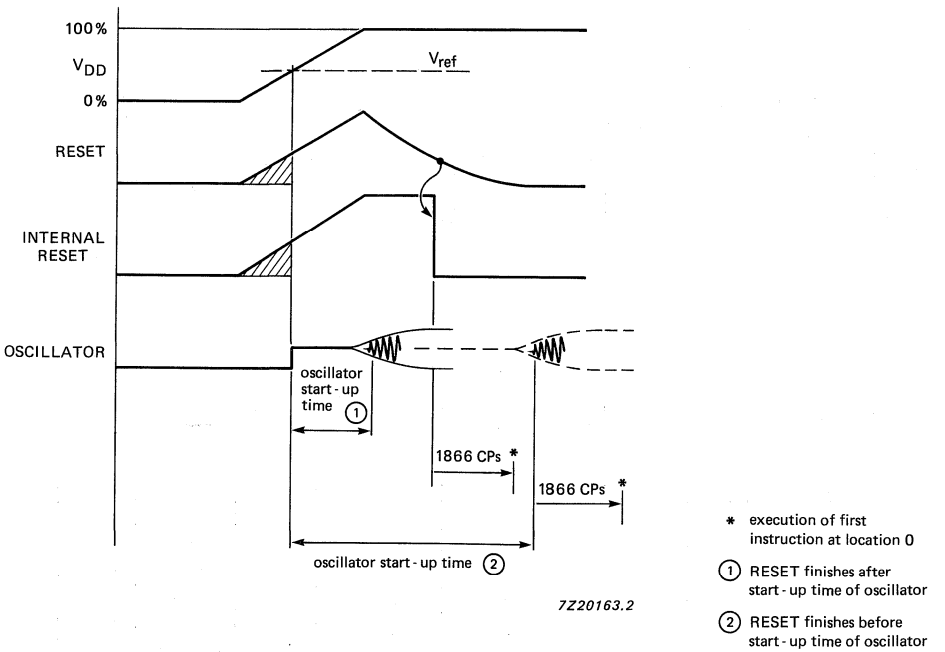


Fig. 21 Timing of power-on reset with a slowly rising V_{DD} and a stretched RESET pulse.

FUNCTIONAL DESCRIPTION (continued)

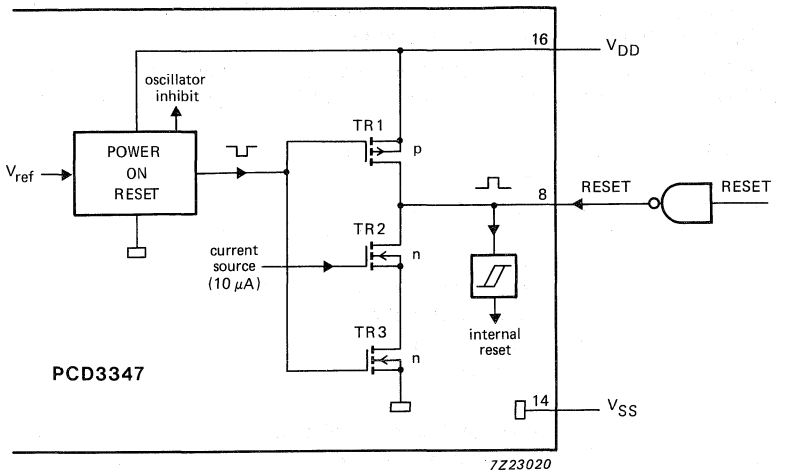


Fig. 22 External power-on reset configuration.

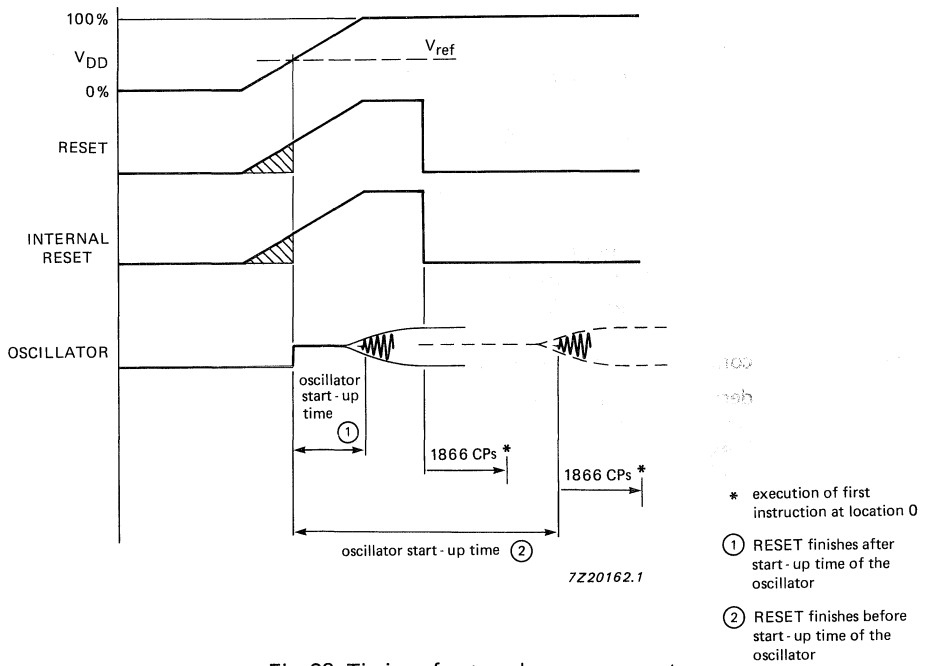


Fig. 23 Timing of external power-on reset.

INSTRUCTION SET

The PCD3347 instruction set consists of over 80 one and two byte instructions and is based on the MAB8048 instruction set. New instructions include those for memory bank selection and derivative control. Program code efficiency is high because all RAM locations and all ROM locations on a 256 byte page require only a single byte address.

Table 5 details the symbols and definition descriptions that are used in the instruction set of the PCD3347. Table 6 shows the instruction map and Table 7 gives the instruction set.

Table 5 Symbols and definitions used in Tables 6 and 7

symbol	definition description
A	accumulator
addr	program memory address
Bb	bit designation (b = 0 to 7)
RBS	register bank select
C	carry bit (bit CY)
CNT	event counter
D	mnemonic for 4-bit digit (nibble)
data	8-bit number or expression
I	interrupt
MB	memory bank
MBFF	memory bank flip-flop
P	mnemonic for 'in page' operation
PC	program counter
Pp	port designation (p = 0 or 1)
PSW	program status word
RB	register bank
Rr	register designation (r = 0 to 7)
SP	stack pointer
T	timer
TF	timer flag
T0, T1	test 0 and 1 inputs
#	immediate data prefix
@	indirect address prefix
(X)	contents of X
((X))	contents of location addressed by X
Dx	derivative register (0 to H'FF')
←	is replaced by
↔	is exchanged with

INSTRUCTION SET (continued)

Table 6 PCD3347 instruction map

first hexadecimal character of opcode ↓

second hexadecimal character of opcode →

	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
0	NOP	IDLE		ADD A, # data	JMP page 0	EN I	JNTF addr	DEC A	IN A,Pp 0	1	2					
1	INC @Rr 0	1	JB0 addr	ADDC A, # data	CALL page 0	DIS I	JTF addr	INC A	INC Rr 0	1	2	3	4	5	6	7
2	XCH A,@Rr 0	1	STOP	MOV A, # data	JMP page 1	EN TCNTI	JNT0 addr	CLR A	XCH A,Rr 0	1	2	3	4	5	6	7
3	XCHD A,@Rr 0	1	JB1 addr		CALL page 1	DIS TCNTI	JT0 addr	CPL A	OUTL Pp,A 0	1	2					
4	ORL A,@Rr 0	1	MOV A , T	ORL A, # data	JMP page 2	STRT CNT	JNT1 addr	SWAP A	ORL A,Rr 0	1	2	3	4	5	6	7
5	ANL A,@Rr 0	1	JB2 addr	ANL A, # data	CALL page 2	STRT T	JT1 addr	DA A	ANL A,Rr 0	1	2	3	4	5	6	7
6	ADD A,@Rr 0	1	MOV T , A		JMP page 3	STOP TCNT		RRC A	ADD A,Rr 0	1	2	3	4	5	6	7
7	ADDC A,@Rr 0	1	JB3 addr		CALL page 3			RR A	ADDC A,Rr 0	1	2	3	4	5	6	7
8				RET	JMP page 4				ORL Pp, # data 0	1	2			MOV Dx,A		
9			JB4 addr	RETR	CALL page 4		JNZ addr	CLR C	ANL Pp, # data 0	1	2					
A	MOV @Rr, A 0	1		MOVP A,@A	JMP page 5	SEL MB2		CPL C	MOV Rr,A 0	1	2	3	4	5	6	7
B	MOV @Rr, # data 0	1	JB5 addr	JMPP @A	CALL page 5	SEL MB3			MOV Rr, # data 0	1	2	3	4	5	6	7
C	DEC @Rr 0	1			JMP page 6	SEL RB0	JZ addr	MOV A, PSW	DEC Rr 0	1	2	3	4	5	6	7
D	XRL A,@Rr 0	1	JB6 addr	XRL A, # data	CALL page 6	SEL RB1		MOV PSW, A	XRL A,Rr 0	1	2	3	4	5	6	7
E	DJNZ @Rr,addr 0	1			JMP page 7	SEL MB0	JNC addr	RL A	DJNZ Rr, addr 0	1	2	3	4	5	6	7
F	MOV A,@Rr 0	1	JB7 addr		CALL page 7	SEL MB1	JC addr	RLC A	MOV A,Rr 0	1	2	3	4	5	6	7

Table 7 Instruction set

	mnemonic	opcode (hex.)	bytes/ cycles	description	function		notes
ACCUMULATOR	ADD A, Rr	6*	1/1	Add register contents to A	(A)←(A) + (Rr)	r = 0–7	1
	ADD A, @Rr	60 61	1/1	Add RAM data, addressed by Rr, to A	(A)←(A) + ((R0)) (A)←(A) + ((R1))		1
	ADD A, #data	03 data	2/2	Add immediate data to A	(A)←(A) + data		1
	ADDC A, Rr	7*	1/1	Add carry and register contents to A	(A)←(A) + (Rr) + (C)	r = 0–7	1
	ADDC A, @Rr	70 71	1/1	Add carry and RAM data, addressed by Rr, to A	(A)←(A) + ((R0)) + (C) (A)←(A) + ((R1)) + (C)		1
	ADDC A, #data	13 data	2/2	Add carry and immediate data to A	(A)←(A) + data + (C)		1
	ANL A, Rr	5*	1/1	'AND' Rr with A	(A)←(A) AND (Rr)	r = 0–7	
	ANL A, @Rr	50 51	1/1	'AND' RAM data, addressed by Rr, with A	(A)←(A) AND ((R0)) (A)←(A) AND ((R1))		
	ANL A, #data	53 data	2/2	'AND' immediate data with A	(A)←(A) AND data		
	ORL A, Rr	4*	1/1	'OR' Rr with A	(A)←(A) OR (Rr)	r = 0–7	
	ORL A, @Rr	40 41	1/1	'OR' RAM data, addressed by Rr, with A	(A)←(A) OR ((R0)) (A)←(A) OR ((R1))		
	ORL A, #data	43 data	2/2	'OR' immediate data with A	(A)←(A) OR data		
	XRL A, Rr	D*	1/1	'XOR' Rr with A	(A)←(A) XOR (Rr)	r = 0–7	
	XRL A, @Rr	D0 D1	1/1	'XOR' RAM, addressed by Rr, with A	(A)←(A) XOR ((R0)) (A)←(A) XOR ((R1))		
	XRL A, #data	D3 data	2/2	'XOR' immediate data with A	(A)←(A) XOR data		
	INC A	17	1/1	increment A by 1	(A)←(A) + 1		
	DEC A	07	1/1	decrement A by 1	(A)←(A) −1		
	CLR A	27	1/1	clear A to zero	(A)←0		
	CPL A	37	1/1	one's complement A	(A)←NOT(A)		
	RL A	E7	1/1	rotate A left	$(A_{n+1})\leftarrow(A_n)$ $(A_0)\leftarrow(A_7)$	n = 0–6	

INSTRUCTION SET (continued)

ACCUMULATOR (cont.)	RLC A	F7	1/1	rotate A left through carry	$(A_{n+1}) \leftarrow A_n$ $(A_0) \leftarrow (C), (C) \leftarrow (A_7)$	n = 0–6	2
	RR A	77	1/1	rotate A right	$(A_n) \leftarrow (A_{n+1})$ $(A_7) \leftarrow (A_0)$	n = 0–6	
	RRC A	67	1/1	rotate A right through carry	$(A_n) \leftarrow (A_{n+1})$ $(A_7) \leftarrow (C), (C) \leftarrow (A_0)$	n = 0–6	2
	DA A	57	1/1	decimal adjust A			2
	SWAP A	47	1/1	swap nibbles of A	$(A_{4-7}) \leftrightarrow (A_{0-3})$		
DATA MOVES	MOV A, Rr	F*	1/1	move register contents to A	(A)←(Rr)	r = 0–7	
	MOV A, @Rr	F0 F1	1/1	move RAM data, addressed by Rr, to A	(A)←((R0)) (A)←((R1))		
	MOV A, #data	23 data	2/2	move immediate data to A	(A)←data		
	MOV Rr, A	A*	1/1	move accumulator contents to register	(Rr)←(A)	r = 0–7	
	MOV @Rr, A	A0 A1	1/1	move accumulator contents to RAM location addressed by Rr	((R0))←(A) ((R1))←(A)		
	MOV Rr, #data	B* data	2/2	move immediate data to Rr	(Rr)←data	r = 0–7	
	MOV @Rr, #data	B0 data B1 data	2/2	move immediate data to RAM location addressed by Rr	((R0))←data ((R1))←data		
	XCH A, Rr	2*	1/1	exchange accumulator contents with Rr	(A)↔(Rr)	r = 0–7	
	XCH A, @Rr	20 21	1/1	exchange accumulator contents with RAM data addressed by Rr	(A)↔((R0)) (A)↔((R1))		
	XCHD A, @Rr	30 31	1/1	exchange lower nibbles of A and RAM data addressed by Rr	$(A_{0-3}) \leftrightarrow ((R0_{0-3}))$ $(A_{0-3}) \leftrightarrow ((R1_{0-3}))$		
	MOV A, PSW	C7	1/1	move PSW contents to accumulator	(A)←(PSW)		
	MOV PSW, A	D7	1/1	move accumulator bit 3 to PSW_3	$(PSW_3) \leftarrow (A_3)$		3
	MOVP A, @A	A3	1/2	move indirectly addressed data in current page to A	$(PC_{0-7}) \leftarrow (A), (A) \leftarrow ((PC))$		
FLAGS	CLR C	97	1/1	clear carry bit	(C)←0		2
	CPL C	A7	1/1	complement carry bit	(C)←NOT(C)		2

DEVELOPMENT DATA

	mnemonic	opcode (hex.)	bytes/ cycles	description	function	notes
REGISTER	INC Rr	1*	1/1	increment register by 1	(Rr)←(Rr) + 1 r = 0–7	
	INC @Rr	10 11	1/1	increment RAM data, addressed by Rr, by 1	((R0))←((R0)) + 1 ((R1))←((R1)) + 1	
	DEC Rr	C*	1/1	decrement register by 1	(Rr)←(Rr) – 1 r = 0–7	
	DEC @Rr	C0 C1	1/1	decrement RAM data, addressed by Rr, by 1	((R0))←((R0)) – 1 ((R1))←((R1)) – 1	
BRANCH	JMP addr	● 4 address	2/2	unconditional jump within a 2 K bank	(PC_{8-10})←$addr_{8-10}$ (PC_{0-7})←$addr_{0-7}$ (PC_{11-12})←MBFF 0–1	
	JMPP @A	B3	1/2	indirect jump within a page	(PC_{0-7})←((A))	
	DJNZ Rr, addr	E* address	2/2	decrement Rr by 1 and jump if not zero to addr	(Rr)←(Rr) – 1 r = 0–7 if (Rr) not zero (PC_{0-7})←addr	
	DJNZ @Rr, addr	E0 address	2/2	decrement RAM data, addressed by Rr by 1 and jump if not zero to addr	((R0))←((R0)) – 1 if ((R0)) not zero (PC_{0-7})←addr	
		E1 address			((R1))←((R1)) – 1 if ((R1)) not zero (PC_{0-7})←addr	
	JBb addr	▲ 2 address	2/2	jump to addr if Acc. bit b = 1	if b = 1 : (PC_{0-7})←addr b = 0–7	
	JC addr	F6 address	2/2	jump to addr if C = 1	if C = 1 : (PC_{0-7})←addr	
	JNC addr	E6 address	2/2	jump to addr if C = 0	if C = 0 : (PC_{0-7})←addr	
	JZ addr	C6 address	2/2	jump to addr if A = 0	if A = 0 : (PC_{0-7})←addr	
	JNZ addr	96 address	2/2	jump to addr if A is NOT zero	if A≠0 : (PC_{0-7})←addr	
	JT0 addr	36 address	2/2	jump to addr if T0 = 1	if T0 = 1: (PC_{0-7})←addr	
	JNT0 addr	26 address	2/2	jump to addr if T0 = 0	if T0 = 0: (PC_{0-7})←addr	
	JT1 addr	56 address	2/2	jump to addr if T1 = 1	if T1 = 1: (PC_{0-7})←addr	
	JNT1 addr	46 address	2/2	jump to addr if T1 = 0	if T1 = 0: (PC_{0-7})←addr	
	JTF addr	16 address	2/2	jump to addr if Timer Flag = 1	if TF = 1: (PC_{0-7})←addr	4
	JNTF addr	06 address	2/2	jump to addr if Timer Flag = 0	if TF = 0: (PC_{0-7})←addr	

INSTRUCTION SET (continued)

TIMER/EVENT COUNTER	MOV A, T	42	1/1	move timer/event counter contents to accumulator	(A)←(T)	
	MOV T, A	62	1/1	move accumulator contents to timer/event counter	(T)←(A)	
	STRT CNT	45	1/1	start event counter		
	STRT T	55	1/1	start timer		
	STOP TCNT	65	1/1	stop timer/event counter		
	EN TCNTI	25	1/1	enable timer/event counter interrupt		
	DIS TCNTI	35	1/1	disable timer/event counter interrupt		
CONTROL	EN I	05	1/1	enable external interrupt		
	DIS I	15	1/1	disable external interrupt		
	SEL RB0	C5	1/1	select register bank 0	(RBS)←0	5
	SEL RB1	D5	1/1	select register bank 1	(RBS)←1	5
	SEL MB0	E5	1/1	select program memory bank 0	(MBFF0)←0, (MBFF1)←0	
	SEL MB1	F5	1/1	select program memory bank 1	(MBFF0)←1, (MBFF1)←0	
	SEL MB2	A5	1/1	select program memory bank 2	(MBFF0)←0, (MBFF1)←1	
	SEL MB3	B5	1/1	select program memory bank 3	(MBFF0)←1, (MBFF1)←1	
	STOP	22	1/1	enter STOP mode		
	IDLE	01	1/1	enter IDLE mode		
SUBROUTINE	CALL addr	▲ 4 address	2/2	jump to subroutine	$((SP))\leftarrow(PC), (PSW_{4,6,7})$ $(SP)\leftarrow(SP)+1$ $(PC_{8-10})\leftarrow addr_{8-10}$ $(PC_{0-7})\leftarrow addr_{0-7}$ $(PC_{11-12})\leftarrow$MBFF 0–1	6
	RET	83	1/2	return from subroutine	$(SP)\leftarrow(SP)-1$ $(PC)\leftarrow((SP))$	6
	RETR	93	1/2	return from interrupt and restore bits 4, 6, 7 of PSW	$(SP)\leftarrow(SP)-1$ $(PSW_{4,6,7}) + (PC)\leftarrow((SP))$	6

DEVELOPMENT DATA

	mnemonic	opcode (hex.)	bytes/ cycles	description	function	notes
PARALLEL INPUT/OUTPUT	IN A, Pp	08 09 0A	1/2	input port p data to accumulator	(A)←(P0) (A)←(P1) (A)←(P2)	 7
	OUTL Pp, A	38 39 3A	1/2	output accumulator data to port p	(P0)←(A) (P1)←(A) (P2)←(A)	 7
	ANL Pp, #data	98 data 99 data 9A data	2/2	AND port p data with immediate data	(P0)←(P0) AND data (P1)←(P1) AND data (P2)←(P2) AND data	 7
	ORL Pp, #data	88 data 89 data 8A data	2/2	OR port p data with immediate data	(P0)←(P0) OR data (P1)←(P1) OR data (P2)←(P2) OR data	 7
DERIVATIVE INPUT/OUTPUT	MOV Dx, A	8D	2/2	move accumulator contents to derivative register	(Dx)←(A) x = 0 to 255	8
	NOP	00	1/1	no operation		

Notes to Table 7

1. PSW CY, AC affected
2. PSW CY affected
3. PSW PS affected
4. Execution of JTF and JNTF instructions resets the Timer Flag (TF).
5. PSW RBS affected
6. PSW SP_0, SP_1, SP_2 affected
7. (A) = 0000P23, P22, P21, P20 PCD3347 Port 2 is not bonded.
8. Instructions for PCF3300 only.

* : 8, 9, A, B, C, D, E, F
● : 0, 2, 4, 6, 8, A, C, E
▲ : 1, 3, 5, 7, 9, B, D, F

RATINGS

Limiting values in accordance with the Absolute Maximum System (IEC 134)

parameter	conditions	symbol	min.	max.	unit
Supply voltage	pin 16	V_{DD}	−0,8	+8	V
Input voltage	any pin	V_I	−0,8	V_{DD} + 0,8	V
DC current	any input or output	$\pm I_I, \pm I_O$	–	10	mA
Total power dissipation	derate according to thermal resistance	P_{tot}	–	500	mW
Power dissipation	per output	P_O	–	50	mW
Storage temperature range		T_{stg}	−65	+150	°C
Operating ambient temperature range		T_{amb}	−25	+70	°C
Operating junction temperature		T_j	–	+125	°C
Thermal resistance junction to ambient	SOT146	$R_{th\ j\text{-}a}$	–	120	K/W
	SOT163A	$R_{th\ j\text{-}a}$	–	150	K/W

DEVELOPMENT DATA

DC CHARACTERISTICS

V_{DD} = 2,5 to 6 V; V_{SS} = 0 V; T_{amb} = −25 to + 70 °C; all voltages with respect to V_{SS}; f = 3,58 MHz with R_S = 100 Ω; unless otherwise specified

parameter	conditions	symbol	min.	typ.	max.	unit
Supplies						
Supply voltage						
operating		V_{DD}	2,5	–	6	V
STOP mode for RAM data retention		V_{DD}	1,0	–	6	V
Supply current (Fig. 25)						
operating with tone generator on	V_{DD} = 3 V	I_{DD}	–	800	–	μA
operating without tone generator	V_{DD} = 3 V	I_{DD}	–	400	–	μA
IDLE mode (Fig. 26)						
with tone generator on	V_{DD} = 3 V	I_{DD}	–	600	–	μA
without tone generator	V_{DD} = 3 V	I_{DD}	–	200	–	μA
STOP mode (Fig. 27)	note 1; V_{DD} = 1,8 V					
	T_{amb} = 25 °C	I_{DD}	–	1,5	2,0	μA
	T_{amb} = 55 °C	I_{DD}	–	–	5	μA
	T_{amb} = 70 °C	I_{DD}	–	–	10	μA
RESET I/O						
Switching level		V_{RESET}	–	1,3	–	V
Sink current	$V_{DD} > V_{RESET}$	I_{OL}	–	7	–	μA
Inputs						
Input voltage LOW		V_{IL}	0	–	0,3 V_{DD}	V
Input voltage HIGH		V_{IH}	0,7 V_{DD}	–	V_{DD}	V
Input leakage current	$V_{SS} < V_I < V_{DD}$	$\pm I_{IL}$	–	–	1	μA
Outputs						
Output voltage LOW	V_I = V_{SS} or V_{DD}; $\lvert I_O \rvert < 1$ μA	V_{OL}	–	–	0,05	V
Output sink current LOW for all remaining ports	V_{DD} = 3 V; V_O = 0,4 V	I_{OL}	1,5	2,5	–	mA
Pull-up output source current HIGH (Fig. 29)	V_{DD} = 3 V; V_O = 0,7 V_{DD}	$-I_{OH}$	10	–	–	μA
	V_O = V_{SS}	$-I_{OH}$	–	–	300	μA
Push-pull output source current HIGH	V_{DD} = 3 V; V_O = V_{DD} −0,4 V	$-I_{OH}$	0,6	1,5	–	mA

Note 1

Crystal connected between XTAL1 and XTAL2; CE and T1 at V_{SS}.

TONE GENERATOR CHARACTERISTICS

V_{DD} = 2,5 to 6 V; V_{SS} = 0 V; T_{amb} = −25 to + 70 °C; all voltages with respect to V_{SS}; f = 3,58 MHz with R_S = 100 Ω; unless otherwise specified

parameter	conditions	symbol	min.	typ.	max.	unit
Tone output (Fig. 24)						
DTMF output voltage levels (r.m.s. values)						
HIGH group		$V_{HG(rms)}$	158	192	205	mV
LOW group		$V_{LG(rms)}$	125	150	160	mV
Frequency deviation		$\Delta f/f$	−0,6	–	+ 0,6	%
DC voltage level		V_{dc}	–	½ V_{DD}	–	V
Output impedance		$\|Z_O\|$	–	0,1	0,5	kΩ
Load resistance		R_L	10	–	–	kΩ
Pre-emphasis of group		ΔV_G	1,85	2,1	2,35	dB
Total harmonic distortion	note 2; T_{amb} = 25 °C	THD	–	−25	–	dB

Note 2

Related to the level of the LOW group frequency component (CEPT CS 203)

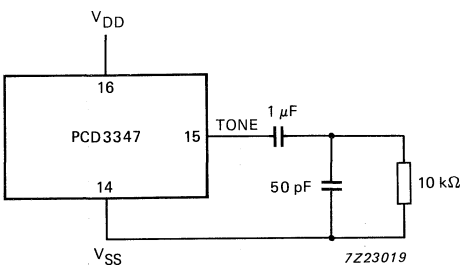


Fig. 24 Tone output test circuit.

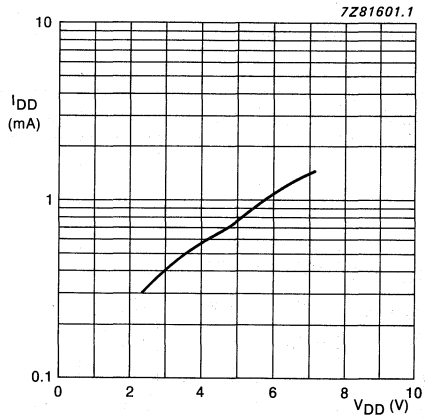


Fig. 25 Typical supply current (I_{DD}) in operating mode as a function of the supply voltage (V_{DD}); T_{amb} = 25 °C; clock frequency = 3,58 MHz; I_{DD} is increased by approximately 0,6 mA when the DTMF function is operating.

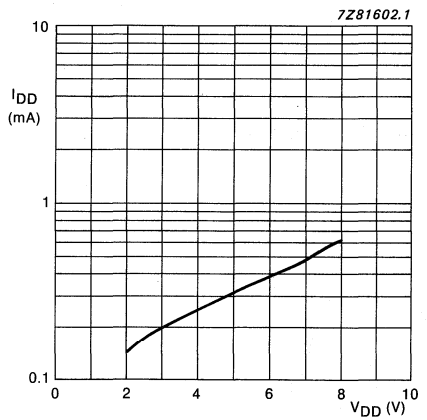
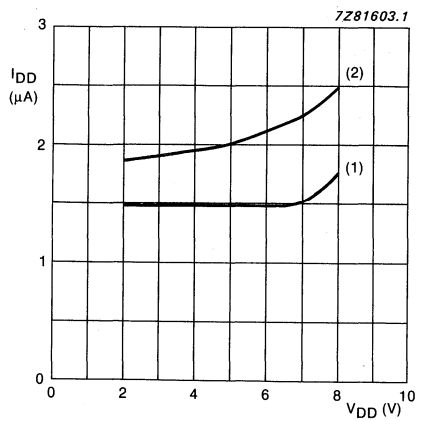
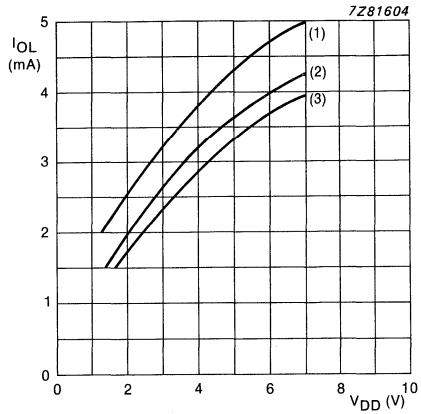


Fig. 26 Typical supply current (I_{DD}) in IDLE mode as a function of the supply voltage (V_{DD}); T_{amb} = 25 °C; clock frequency = 3,58 MHz.



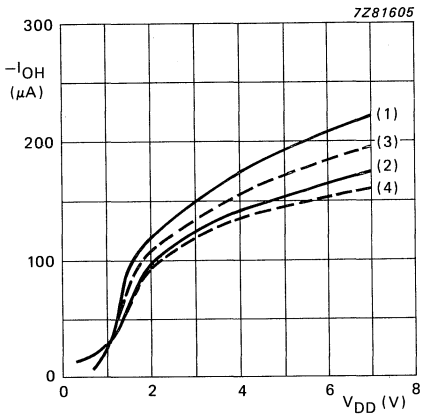
(1) T_{amb} = 25 °C
(2) T_{amb} = 70 °C

Fig. 27 Typical supply current (I_{DD}) in STOP mode as a function of the supply voltage (V_{DD}).



(1) $T_{amb} = -25\ ^{o}C$
(2) $T_{amb} = 25\ ^{o}C$
(3) $T_{amb} = 70\ ^{o}C$

Fig. 28 Output sink current LOW (I_{OL}), as a function of supply voltage (V_{DD}); $V_O = 0{,}4$ V.



(1) $T_{amb} = 25\ ^{o}C$; $V_O = V_{SS}$
(2) $T_{amb} = 25\ ^{o}C$; $V_O = 0{,}7\ V_{DD}$
(3) $T_{amb} = 70\ ^{o}C$; $V_O = V_{SS}$
(4) $T_{amb} = 70\ ^{o}C$; $V_O = 0{,}7\ V_{DD}$

Fig. 29 Output source current HIGH ($-I_{OH}$) as a function of supply voltage (V_{DD}).

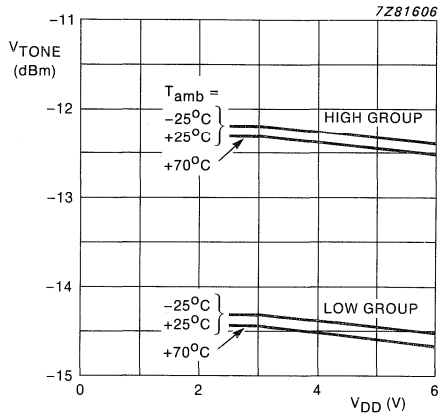


Fig. 30 DTMF output voltage levels as a function of operating supply voltage; R_L = 1 MΩ.

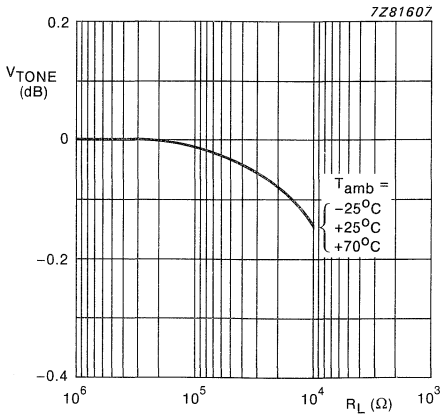


Fig. 31 Dual tone output voltage level as a function of output load resistance.

DEVELOPMENT DATA

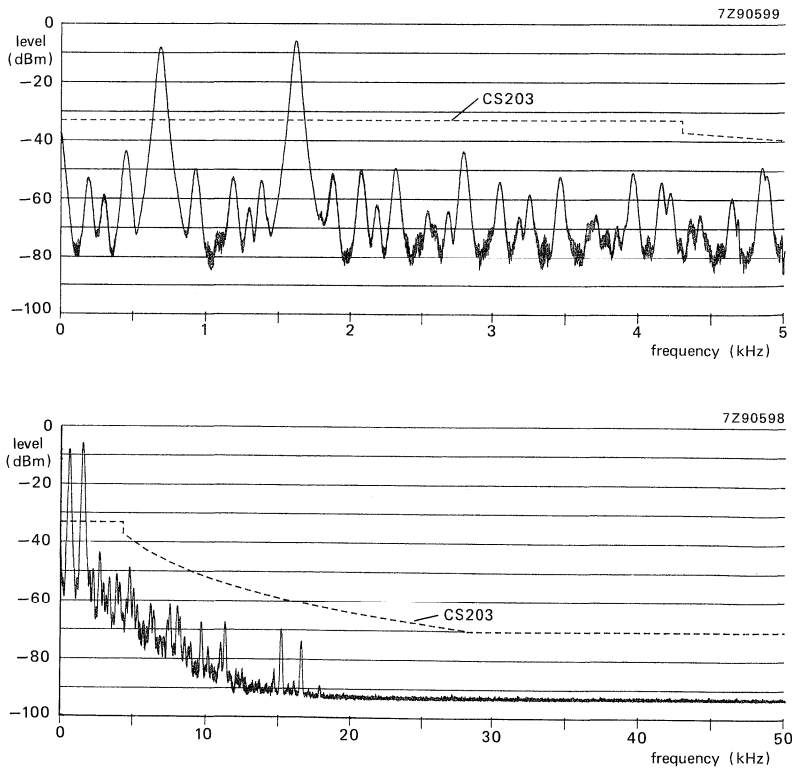


Fig. 32 Typical frequency spectrum of a dual tone signal after flat-band amplification of 6 dB.

APPLICATION INFORMATION

A block diagram of an electronic featurephone built around the PCD3347 is shown in Figure 31. It comprises the following dedicated telephony ICs:

- TEA1060/1061/1067/1068 transmission circuit for telephony
- PCF8576 or PCF8577 2 LCD drivers in LCD module MB7020160
- PCF8571 1 K RAMs with Serial I/O; the number of RAMs depends on the required amount of stored telephone numbers
- PCD3360 programmable multi-tone ringer

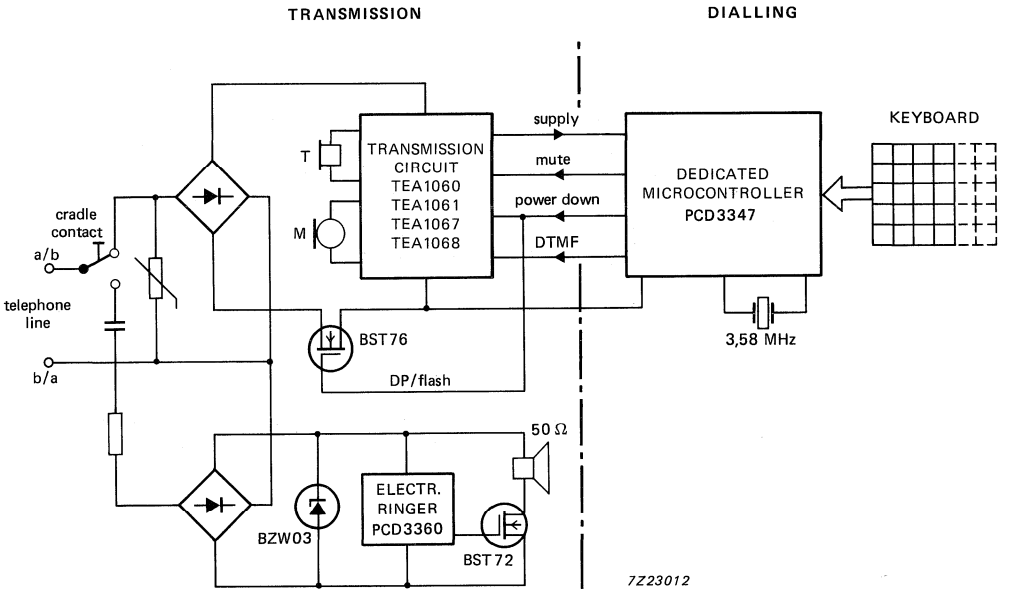
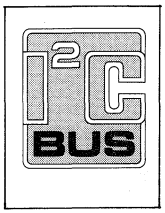


Fig. 33 Block diagram of electronic featurephone with common line interface.

Data sheet	
status	Product specification
date of issue	October 1990

PCD3348

CMOS microcontroller for telephone sets



FEATURES

- 8-bit CPU, ROM, RAM, I/O in a single 28-lead DIL or SO package
- 8 K ROM bytes
- 256 RAM bytes
- 20 quasi-bidirectional I/O port lines
- Two test inputs: one of which is also the external interrupt input (CE/$\overline{TO}$)
- Single-level vectored interrupts: external, timer/event counter, serial I/O
- Serial I/O which can be used in bus systems with more than one master (serial I/O data via an existing port line and clock via a dedicated line); tailored for I^2C-bus communications
- 8-bit programmable timer/event counter
- Over 80 instructions (based on MAB8048)
- All instructions 1 or 2 cycles
- Clock frequency 450 kHz to 10 MHz
- Single supply voltage from 1.8 V to 6 V
- Low standby voltage and current
- STOP and IDLE mode
- On-chip oscillator with output drive capability for peripherals (e.g. PCD3312C DTMF generator)
- Configuration of all I/O port lines individually selected by mask: pull-up, open drain or push-pull
- Power-on-reset circuit and low supply voltage detection
- Reset state of all ports individually selected by mask
- Operating temperature range: −25 to + 70 °C.

GENERAL DESCRIPTION

The PCD3348 is a single-chip 8-bit microcontroller fabricated in CMOS. It has special on-chip features for applications in telephone sets. The device is mask programmable, designed to provide telephone dialling facilities such as redial, repertory dial, emergency call, keyboard scan and control for liquid crystal display, pulse dial and/or DTMF dial via dedicated peripheral.

For detailed information see the PCD33XX family specification.

ORDERING INFORMATION

EXTENDED TYPE NUMBER	PACKAGE			
	PINS	PIN POSITION	MATERIAL	CODE
PCD3348P	28	DIL	plastic	SOT117
PCD3348T	28	mini-pack	plastic	SO28; SOT136A

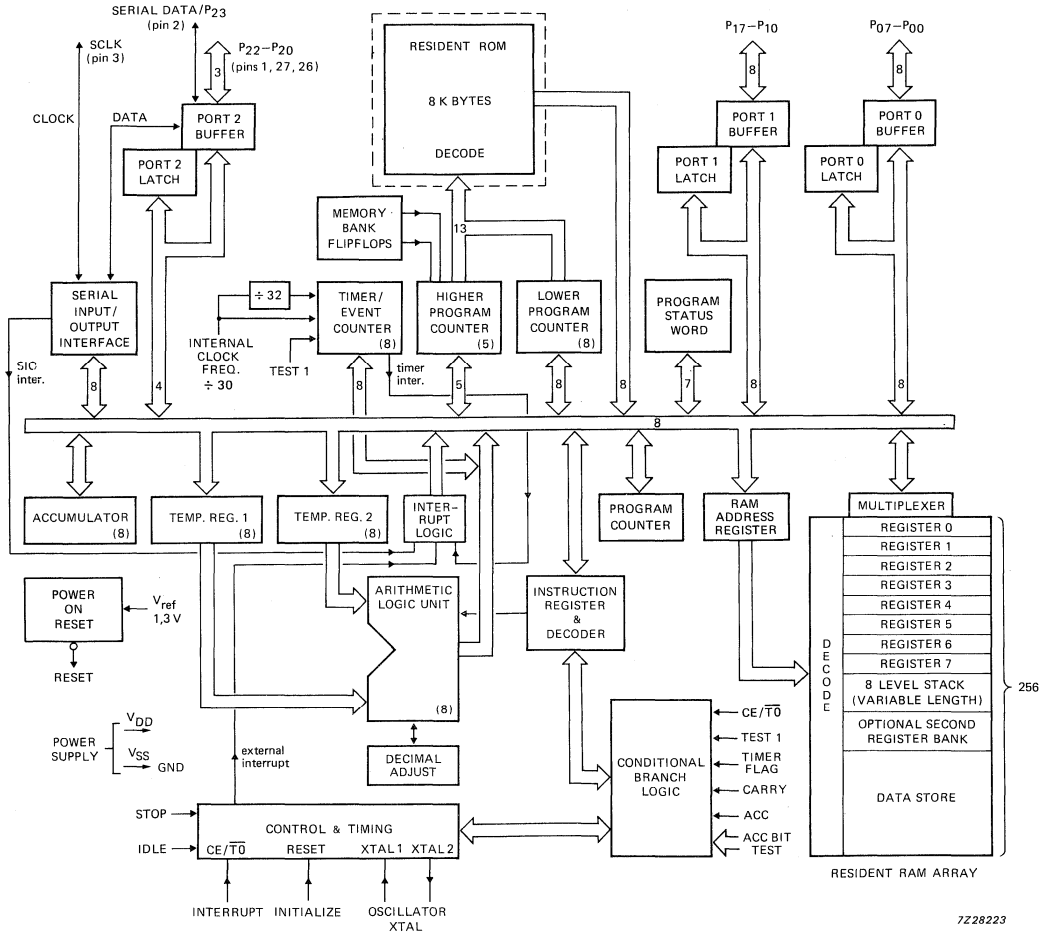
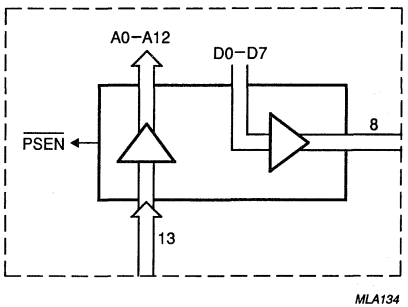


Fig.1 Block diagram.



(a)

Fig.1a Replacement of dotted part in Fig.1, for the PCD3301B 'Piggy-back' version.

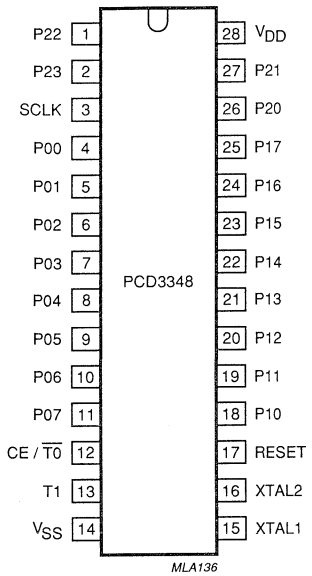


Fig.2 Pinning diagram: PCD3348 and bottom pinning PCD3301B.

PINNING

SYMBOL	PIN	DESCRIPTION
SCLK	3	**Clock:** bidirectional clock for serial I/O.
P00-P07	4-11	**Port 0:** 8-bit quasi-bidirectional I/O port.
CE/$\overline{TO}$	12	**Interrupt/Test 0:** external interrupt input (sensitive to positive-going edge) test input pin; when used as a test input directly tested by conditional branch instructions JTO and JNTO.
T1	13	**Test 1:** test input pin, directly tested by conditional branch instructions JT1 and JNT1. T1 also functions as an input to the 8-bit timer/event counter, using the STRT CNT instruction.
V_{SS}	14	**Ground:** circuit earth potential.
XTAL1	15	**Crystal input:** connection to timing component (crystal) which determines the frequency of the internal oscillator; also the input for an external clock source.
XTAL2	16	Connection to the other side of the timing component.
RESET	17	**Reset input:** used to initialize the processor (active HIGH), or output of power-on reset circuit.
P10-P17	18-25	**Port 1;** 8-bit quasi-bidirectional I/O port.
P20-P23	26, 27, 1, 2	**Port 2:** 4-bit quasi-bidirectional I/O port. P23 is the serial data input/output in serial I/O mode.
V_{DD}	28	**Power supply;** 1.8 V to 6 V.

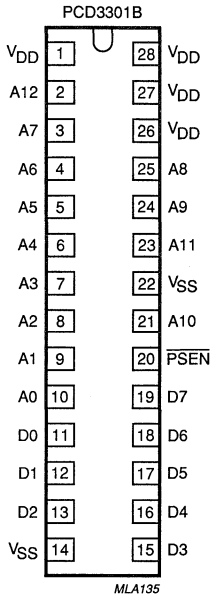


Fig.3 Pinning diagram: PCD3301B 'Piggy Back' version top pinning; to access a 2732 or 2764 EPROM.

PINNING

SYMBOL	PIN	DESCRIPTION
V_{SS}	14, 22	**Ground**
V_{DD}	1, 26-28	**Power supply**
A0-A12	10-3, 25, 24, 21, 23, 2	**Address outputs**
D0-D7	11-13, 15-19	**Data**
$\overline{PSEN}$	20	**Program Store Enable**

Notes

1. Access time for ROM/EPROMS to be below $7 \times 1/f_{XTAL}$.

FUNCTIONAL DESCRIPTION

'Piggy-Back' version PCD3301B

The PCD3301B is a special package that has standard pinning to the bottom which facilitates insertion as a mask-programmed device. An EPROM can be mounted on top in an additional socket. The total package height is greater than the standard DIL package. The RAM has 256 bytes and can also address 8 Kbytes of program memory.

Program memory PCD3348

The program memory consists of 8192 bytes (8-bit words), in a read-only memory (ROM). Each location is directly addressable by the Program Counter. The memory is mask-programmed at the factory. Figure 4 shows the program memory map.

Four program memory locations are of special importance:

- Location 0; contains the first instruction to be executed after the processor is initialized (RESET).
- Location 3; contains the first byte of an external interrupt service subroutine.
- Location 5; contains the first byte of a serial I/O interrupt service subroutine.
- Locations 7; contains the first byte of a timer/event counter interrupt service subroutine.

Program memory is arranged in banks of 2 Kbytes, which are selected by SEL MB instructions. The program memory is further divided into location 'pages', each of 256 bytes. This latter division applies only for conditional branches. Memory bank boundaries can be crossed only by using the unconditional branch instructions after the appropriate memory bank has been selected. A CALL instruction can transfer control to a subroutine on any 'page'; RET and RETR instructions can transfer control from a subroutine back to the main program.

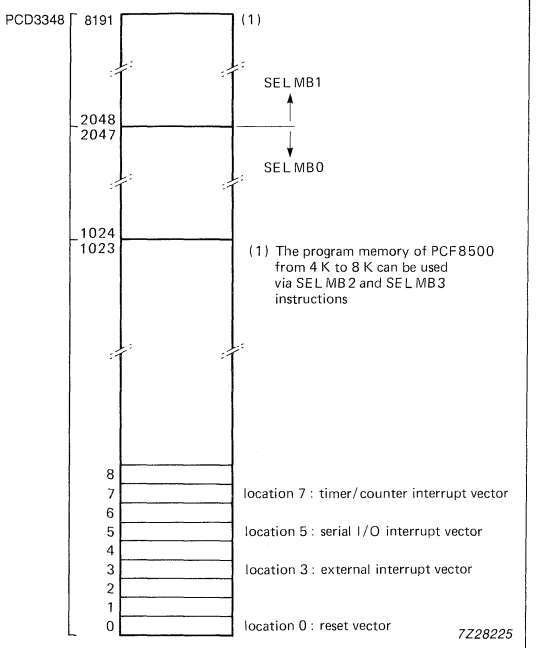


Fig.4 Program memory map.

Data memory PCD3348

Data memory consists of 256 bytes (8-bit words), random-access data memory (RAM). All locations are indirectly addressable using RAM pointer registers; up to 16 designated locations are directly addressable.

Memory also includes an 8-level Program Counter stack addressed by a 8-level Program Counter stack addressed by a 3-bit Stack Pointer. Figure 5 shows the data memory map.

WORKING REGISTERS

Locations 0 to 7 are designated as working registers, directly addressable by the direct register instructions. Ease of addressing, and a minimum requirement of instruction bytes to manipulate their contents, makes these locations suitable for storing frequently addressed intermediate results. This bank of registers can be selected by the SEL RB0 instruction.

Executing the select register bank instruction SEL RB1, designates locations 24 to 31 as working registers, instead of locations 0 to 7, and these are then directly addressable. This second bank of working registers may be used as an extension of the first or reserved for use during interrupt service subroutines saving the first bank for use in the main program. If the second bank is not used, locations 24 to 31 may serve as general purpose RAM.

The first locations of each bank contain the RAM pointer registers R0, R1, R0' and R1', which indirectly address all RAM locations.

All RAM locations make efficient program loop counters when used with the decrement register and test instruction DJNZ.

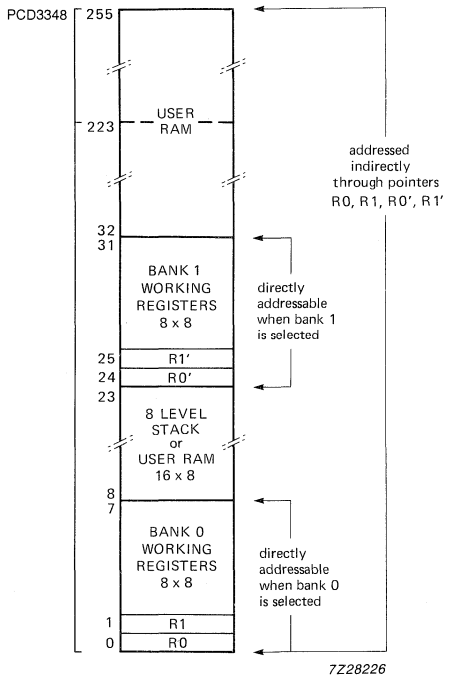


Fig.5 Data memory map.

PROGRAM COUNTER STACK

Locations 8 to 23 may be designated as an 8-level Program Counter stack (2 locations per level), or as general purpose RAM. The Program Counter stack (Fig.6) enables the processor to keep track of the return addresses and status generated by interrupts or CALL instructions by storing the contents of the Program Counter prior to servicing the subroutine.

A 3-bit Stack Pointer determines which of the eight register pairs of the Program Counter stack will be loaded with the next generated return address. The Stack Pointer, when initialized to 000 by RESET, points to RAM locations 8 and 9. On the first subroutine CALL or interrupt, the contents of the Program Counter and bits 4, 6 and 7 of the program status word (PSW) are transferred to locations 8 and 9. The Stack Pointer increments by one and points to locations 10 and 11 ready for another CALL. Because an address may be up to 13 bits long, two bytes must be used to store each address.

At the end of a subroutine, which is signalled by a return instruction (RET or RETR), the Stack Pointer decrements by one and the contents of the register pair on top of the stack are transferred to the Program Counter. The saved PSW bits are transferred to the PSW only by the RETR instruction.

If not all 8 levels of subroutine and interrupt nesting are used, the unused portion of the stack may be used as any other indirectly addressable RAM locations. Possible locations from 32 to 223 may be used for storage of program variables or data.

Nesting of subroutines within subroutines can continue up to 8 times without overflowing the stack. If overflow does occur the deepest address stored (locations 8 and 9) will be overwritten and lost since the Stack Pointer overflows from 111 to 000. It also underflows from 000 to 111.

The value of the saved contents of the Program Counter is different for an interrupt CALL compared to a normal CALL to subroutine. With an interrupt CALL, the Program Counter return address is saved; with a subroutine CALL, the saved Program Counter value is one less than the computer counter return address.

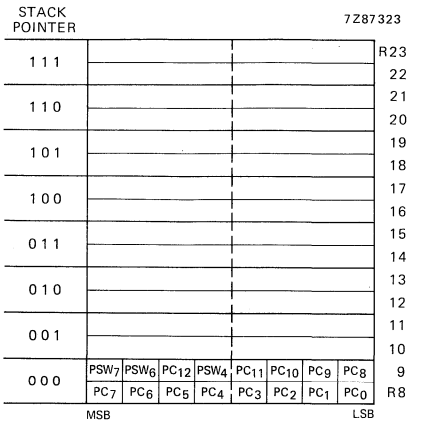


Fig.6 Program Counter stack.

IDLE and STOP modes

IDLE MODE

When the microcontroller enters the IDLE mode via the IDLE instruction (01H) the oscillator, timer/counter and serial I/O are kept running. The microcontroller exits from the IDLE mode by one of three interrupts if they are enabled or by activating a RESET. If the interrupt is not enabled the processor will remain in the IDLE mode. An active signal on the RESET pin restarts the microcontroller and a normal RESET sequence is executed (see Fig.7).

An active signal coming from an enabled interrupt causes the execution of the normal interrupt routine since normal interrupt scanning is still being carried out. A LOW-to-HIGH transition on the external interrupt pin (CE/$\overline{TO}$) reactivates the microcontroller. A HIGH level applied to CE/$\overline{TO}$ will reactive the microcontroller only in the STOP mode. Thus, if CE/$\overline{TO}$ was HIGH before the microcontroller entered the IDLE mode, it must go LOW before the microcontroller can be reactivated (see Fig.8).

Wake-up from the IDLE mode is ensured when CE/$\overline{TO}$ is LOW for 4 clock periods followed by a HIGH for 7 clock periods. After the initial forced CALL 003H operation (60 clock periods) the program continues with the external interrupt service routine.

STOP MODE

The microcontroller enters the STOP mode by the STOP instruction (22H). The oscillator is switched off. The internal status of the CPU, RAM contents and the state of I/O ports are not affected. The microcontroller can be brought-out of the STOP mode by an active signal at the external interrupt input by an external RESET signal. When one of these two signals is applied an internal delay of 1866 clock periods is provided to ensure that all internal clocks are operating correctly before restart (see Fig.9).

If the microcontroller exits from the STOP mode by activating RESET, a normal RESET sequence executed. If the microcontroller exits the STOP mode by pulling the external interrupt pin HIGH, an interrupt pin sequence is executed only if the external interrupt is enabled. In this event the microcontroller resumes the normal program sequence after returning from the interrupt routine, as in the normal mode. If the interrupt is not enabled, it continues the normal program sequence, executing the instruction following the STOP instruction.

The microcontroller is restarted by a HIGH level applied at the CE/$\overline{TO}$ pin, and not by a LOW-to-HIGH transition as in a normal interrupt mechanism. When the CE/$\overline{TO}$ level is active during the STOP instruction then no STOP is executed.A HIGH level on the external interrupt input of at least 1 µs will cause the microcontroller to exit the STOP mode.

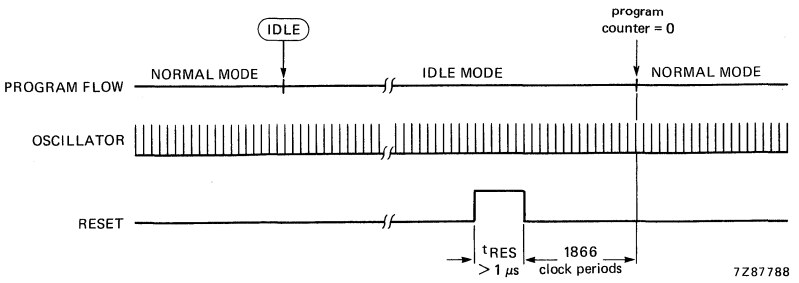


Fig.7 Exit from IDLE mode via RESET.

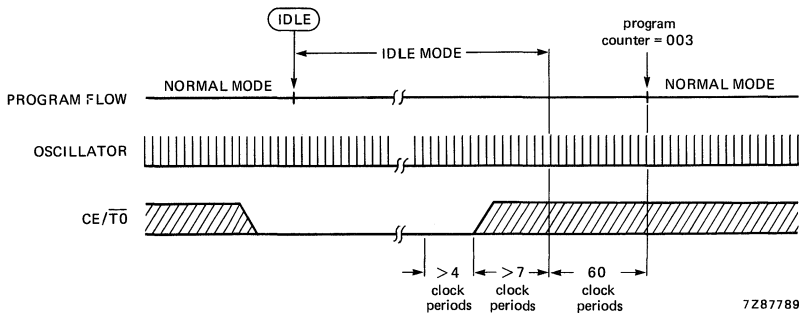


Fig.8 Exit from IDLE mode via an interrupt.

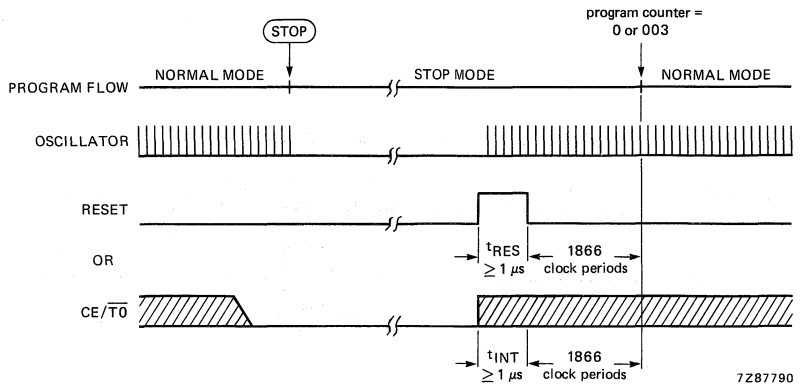


Fig.9 Entering and leaving the STOP mode.

I/O facilities

The PCD3348 family has 23 I/O lines arranged as:

Port 0 Parallel port of 8 lines (P00 to P07)
Port 1 Parallel port of 8 lines (P10 to P17)
Port 2 Parallel port of 4 lines (P20 to P23)
SCLK Serial I/O consisting of a data line shared with a parallel port line (P23) and a separate clock line SCLK
CE/$\overline{TO}$ External interrupt and test input. When used as a test input can be directly tested by conditional branch instructions JTO and JNTO
T1 Test input which can alter program sequences when tested by conditional jump instructions JT1 and JNT1. T1 also functions as an input to the 8-bit timer/event counter.

PARALLEL PORTS

All parallel ports can be used as outputs or inputs, their structure is quasi-bidirectional. Output data written to a port is latched and remains unchanged until rewritten. Input data is not latched and so must be present until read by an input instruction. Input lines are fully CMOS compatible, output lines can drive one LS-TTL or CMOS load.

Fig.11 shows the quasi-bidirectional I/O interface with push-pull output and switched pull-up current source. Each line is pulled up to V_{DD} via a constant current source (TR4), which is enabled via TR3 whenever one of the two output latches contains a logic 1. This current provides sufficient source current for a TTL HIGH level, yet can be pulled LOW by an external CMOS device, thus allowing the same pin to be used for both input and output. When a logic 1 is written to the line for the first time MQ = 1, SQ = 0), TR2 is switched on for the duration of the internal write pulse (one oscillator period), to provide a fast transition from logic 0 to logic 1. Subsequent writing of a logic 1 to the port lines will not switch TR2 on. This prevents unnecessary current through external components connected to the port lines of the same port which might be in the input mode and also connected to ground.

When a logic 0 is written to the line, TR3 switches off the current source. Current sinking capability is provided by TR1, which is now switched on. When used as an input, a logic 1 must first be written to the line, otherwise TR1 will remain low impedance.

In telephone applications this switched pull-up source may not be sufficient. Therefore the PCD3348 offers the possibility to select individually 19 of the 20 parallel port pins (not P23), by the following mask options:

Option 1 STANDARD PORT; quasi-bidirectional I/O with switched pull-up current source of 100 μA (typical) and P-channel booster transistor TR2 (1.5 mA). TR2 is only active during 1 clock cycle (0.28 μs at 3.58 MHz).
Option 2 OPEN DRAIN; quasi-bidirectional I/O with only an N-channel open drain output. Application as an output requires connection of an external pull-up resistor (Fig.12).
Option 3 PUSH-PULL OUTPUT; drive capability of the output will be 1.5 mA (typical) at V_{DD} = 3 V in both polarities. To avoid a large current flowing through the output transistors during the input mode, these push-pull pins must only be used as outputs (Fig.13).

Also, individual mask selection of the RESET state of these port pins can be achieved by appending the following options S and R to options 1, 2 or 3.

Option S SET; after RESET this pin will be initialized to HIGH.
Option R RESET; after RESET this pin will be initialized to LOW.

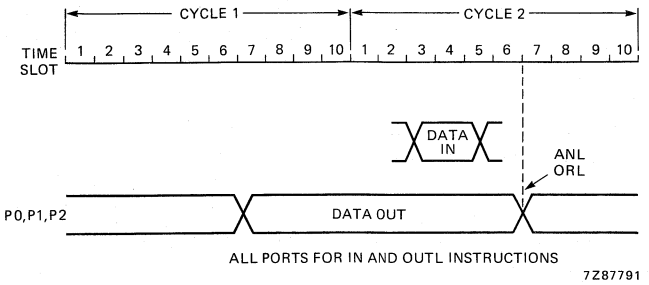


Fig.10 Timing diagram of all ports on IN and OUTL instructions; for ANL and ORL instructions, the ports change on the time slot 7 of cycle 2.

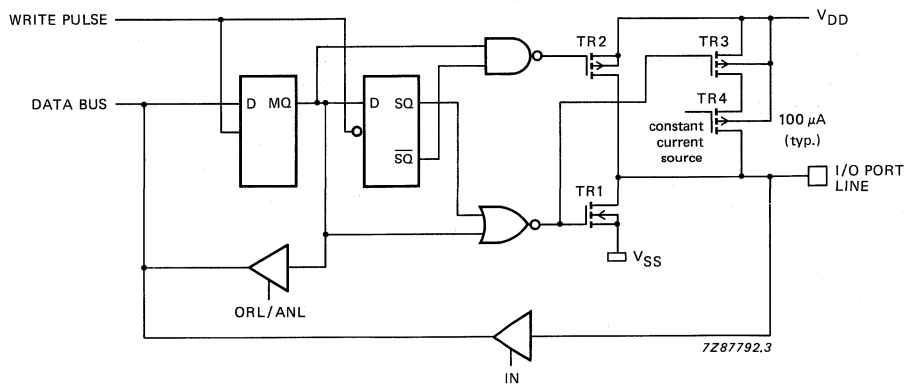


Fig.11 Standard output with switched pull-up current source.

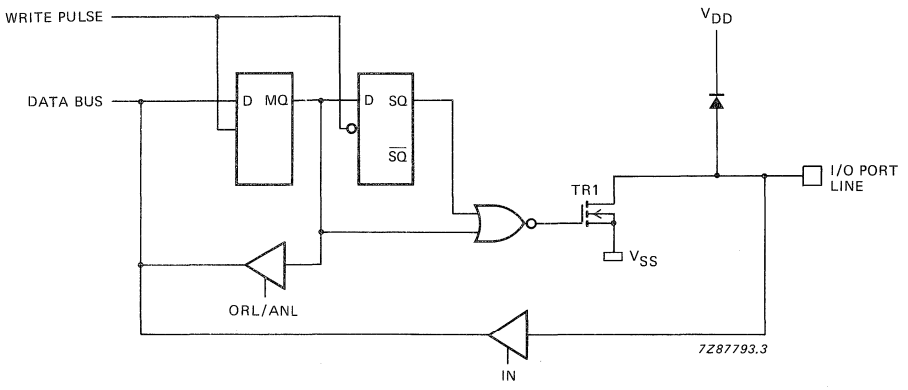


Fig.12 Open drain output.

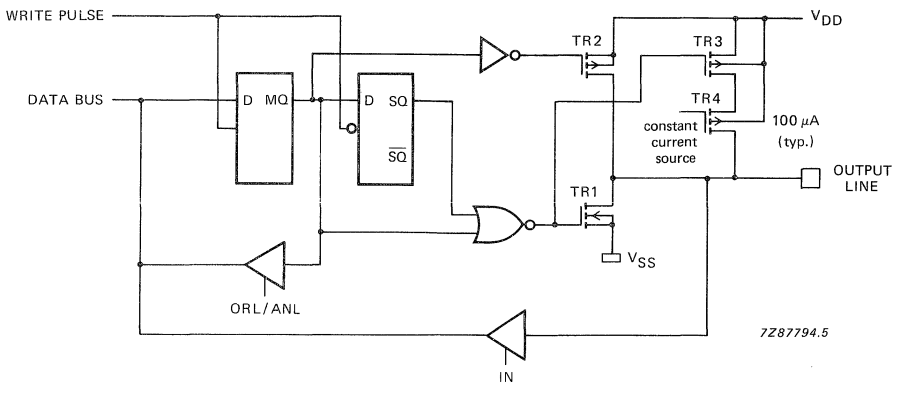


Fig.13 Push-pull output.

SERIAL I/O (SI0)

The PCD3348 has an on-chip serial I/O interface. This SI0 interface is a versatile feature in an intelligent telephone set, as shown in application diagram Fig.32. In this application the SI0 is used communicate with different peripherals, such as:

- DTMF generator (PCD3312C)
- LCD drivers (PCF8577)
- External RAM (PCD8571)
- Clock calendar (PCB8573)

No extra hardware is required for decoding, addressing and data processing.

Whereas a normal microcontroller must regularly monitor the serial data bus for the presence of data, the serial I/O interface detects, receives and converts the serial data stream into parallel format without interrupting the execution of the current program. An interrupt is sent to the PCD3343 only when a complete byte is received. It then reads the data byte in one instruction. Likewise, during transmission the serial I/O interface performs parallel to serial conversion and subsequent serial output of the data. The microcontroller in only interrupted in the execution of its programmed tasks when a complete byte has been transmitted.

The design of the PCD3348 serial I/O system allows any number of devices from the clips family to be connected via the two-line serial bus. The ability of any devices to communicate, without interrupting the operation of any other devices on the bus, is an outstanding attribute of the system. This is achieved by allocating a specific 7-bit address to each device and providing a system whereby a device reacts only to a message prefixed with its own address or the 'general CALL' address.
Address recognition is performed by the interface hardware so that operation of the microcontroller need only be interrupted when a valid address has been received. This saves significant processing time and memory space compared with a conventional microcontroller employing a software serial interface.
When the addressing facility is not required, for instance in a system with only two microcontrollers, direct data transfer without addressing can be performed. In multi-master systems, an automatically invoked arbitration procedure prevents two or more devices from continuing simultaneous transmission.

In NORMAL (running) and IDLE mode, the serial I/O logic remains active; its internal system clock will be switched off when there is no activity on the serial bus.

After execution of the STOP instruction, the oscillator of the PCD3348 is switched off. This means that the serial I/O logic will remain in the state it was at the occurrence of the STOP instruction. To avoid 'bus block' problems and to assure correct start-up of the bus after exit from the STOP mode, the user should disable the serial logic (ESO = 0) prior to the execution of the STOP instruction. This must be carried out only when the PCD3348 has finished a serial data transfer.

SERIAL I/O INTERFACE (tailored for I^2C-bus communications)
Fig.14 shows the serial I/O interface. The clock line of the serial bus has exclusive use of pin 3 (SCLK) while the data lines share pin 2 (serial data) with the I/O line P23 of Port 2. When the serial I/O is enabled, P23 is disabled as a parallel port line; (P23 and SCLK only open drain).

The microcontroller and interface communicate via the internal microcontroller bus and the Serial Interrupt Request line. Data and information controlling the operation of the interface are stored in four registers:

- Data shift register (S0)
- Serial I/O interface status word (S1)
- Serial clock control word (S2)
- Address register.

Data Shift Register (S0)

Register S0 converts serial data to parallel format and vice versa. A pending interrupt is generated only after a complete byte has been transmitted, or after a complete data byte, specific address or 'general CALL' address has been received. The most significant bit is transmitted first.

Serial I/O interface status word (S1)

Register S1 provides information concerning the state of the interface and stores information from the microcontroller. Bits 0 to 3 are duplicated: control bits in these positions can only be written by the microcontroller, while interface bits can only be read.

MST and **TRX**
These bits determine the operating mode of the serial I/O interface as shown in Table 1.

Table 1 Operating modes of the serial I/O interface.

MST	TRX	OPERATING MODE
0	0	Slave receiver
1	0	Master receiver
0	1	Slave transmitter
1	1	Master transmitter

BB (Bus Busy)

This flag indicates the status of the bus.

PIN (Pending Interrupt Not)

PIN = '0' indicates the presence of a pending interrupt, which will cause a Serial Interrupt Request when the serial interrupt mechanism is enabled.

ESO (Enable Serial Output)
The ESO flag enables/disables the serial I/O interface: ESO = '1' enables, ESO = ;0; disables. ESO can only be written by software.

BC0, **BC1** and **BC2**

Bits BC0, BC1 and BC2 indicate the number of bits received or transmitted in a data stream. These bits can only be written by software.

AL (Arbitration Lost)

The arbitration lost flag is set by hardware when the serial I/O interface, as master transmitter, looses a bus arbitration procedure.

AAS (Addressed As Slave)

This flag is set by hardware when the interface detects either its own specific address or the 'general CALL' address as the first byte of a transfer and the interface has been programmed to operate in the address recognition mode.

ADO (Address Zero)

This flag is set by hardware after detection of the 'general CALL' address when the interface is operating in the address recognition mode.

LRB (Last Received Bit)

This contains either the last data bit received, or a transmitting device in the acknowledgement mode, the acknowledgement signal from the receiving device.

Bits AL, AAS, AD0 and LRB can only be read by software.

Serial clock control word (S2)

Bits 0 to 4 of the Clock Control Register S2 are used to set the frequency of the serial clock signal. When a 3.58 MHz crystal is used, the frequency of the serial clock can be varied between 92 kHz and 580 Hz (see Table 2). An asymmetrical clock with a HIGH-to-LOW ratio of 3 : 1 can be generated using bit 5. The asymmetrical clock allows a microcontroller more time per clock period for sampling the data line, making the timing of this action less critical. Bit 6 can be used to activate the acknowledge mode of the serial I/O. S2 is a write only register.

Address Register

The Address Register contains the 7-bit address back-up latches and the bit (ALS) used to enable/disable the address recognition mode. The Address Register can be written to using the MOV S0, A and MOV S0, #data instructions, but only when ESO = '0'.

Serial I/O interrupt logic

An EN SI instruction enables and a DIS SI instruction disables the interrupt logic. When the logic is enabled, a pending interrupt results in a serial I/O interrupt to the processor, causing a CALL to location 5 in the ROM. When disabled, the presence of an interrupt is still indicated by PIN in S1, allowing the interrupt to be serviced. However, vectored interrupt will not occur.

Table 2 S10 clock pulse frequency control when using a 3.58 MHz crystal.

HEXADECIMAL S20-S24 CODE	DIVISOR	f_{SCLK} (kHz) (approx.)
0 (note 1)	-	-
1	39	92
2	45	80
3	51	70
4	63	57
5	75	48
6	87	41
7	99	36
8	123	29
9	147	24
A	171	21
B	195	18
C	243	15
D	291	12
E	339	11
F	387	9.2
10	483	7.4
11	579	6.2
12	675	5.3
13	771	4.6
14	963	3.7
15	1155	3.1
16	1347	2.7
17	1539	2.3
18	1923	1.9
19	2307	1.6
1A	2691	1.3
1B	3075	1.2
1C	3843	0.93
1D	4611	0.78
1E	5379	0.67
1F	6147	0.58

Note

1. This value is not allowed.

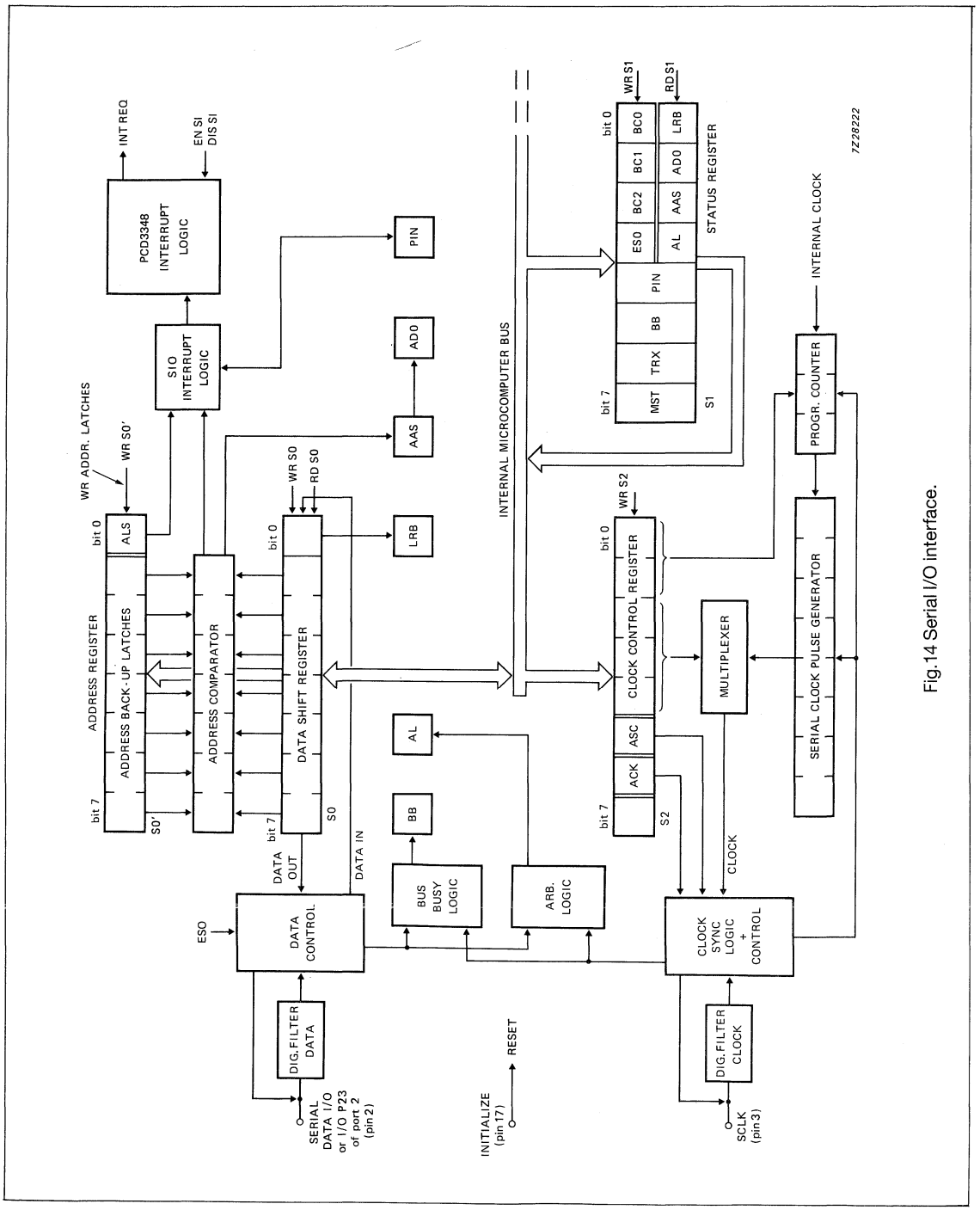


Fig.14 Serial I/O interface.

Interrupts (see Fig.15)

When the external interrupt is enabled, a LOW-to-HIGH transition on the CE/$\overline{TO}$ input initiates an external interrupt subroutine which causes a CALL to program memory location 3 following completion of the current instruction. The interrupt must remain enabled until the interrupt instruction is completed, otherwise the next instruction of the main program will be executed. Serial I/O interrupt, when enabled, causes a CALL to location 5, and a timer/event counter overflow forces a CALL to location 7 when the timer interrupt is enabled.

When an interrupt subroutine starts, the contents of the Program Counter bits 4, 6 and 7 of the PSW have been saved in the Program Counter stack. Accumulator contents have to be saved by software. Interrupt acknowledgement can be carried out by software via port pins. All interrupt subroutines must reside in memory bank 0.

Since the interrupt system is single level, once an interrupt is detected, all further interrupt requests are latched, but ignored, pending a RETR instruction to re-enable the interrupt input logic. After executing RETR, the program continues in the main part; this is independent of the occurrence of a second interrupt during the running of the first routine. If 2 or 3 interrupts occur simultaneously, their priority is:

1. External
2. Serial I/O
3. Timer/event counter

An external interrupt can be generated by using the timer/counter in the event counter mode. The counter is first preset to (FFH), then the EN TCNTI instruction is executed. A LOW-to-HIGH transition of the T1 input will then initiate an interrupt subroutine and cause a CALL to location 7.

On execution of a DIS 1 instruction, the PCD3348 always clears the digital filter/latch and the External Interrupt Flag. The Timer Flag (TF) is reset only when the JTF or JNTF instruction is executed or after RESET. The Timer Interrupt Flag is set when timer overflow occurs, only if the timer interrupt is enabled.

The microcontroller will exit the IDLE mode when any one of the following three interrupts is enabled:

- external
- serial I/O
- timer/event counter.

There is no internal pull-up or pull-down device connected to the external interrupt input (pin 12). If required pin 12 must be externally connected to a resistor ($R \leq 100\ k\Omega$). When the external interrupt is not used pin 12 must be connected to V_{SS}.

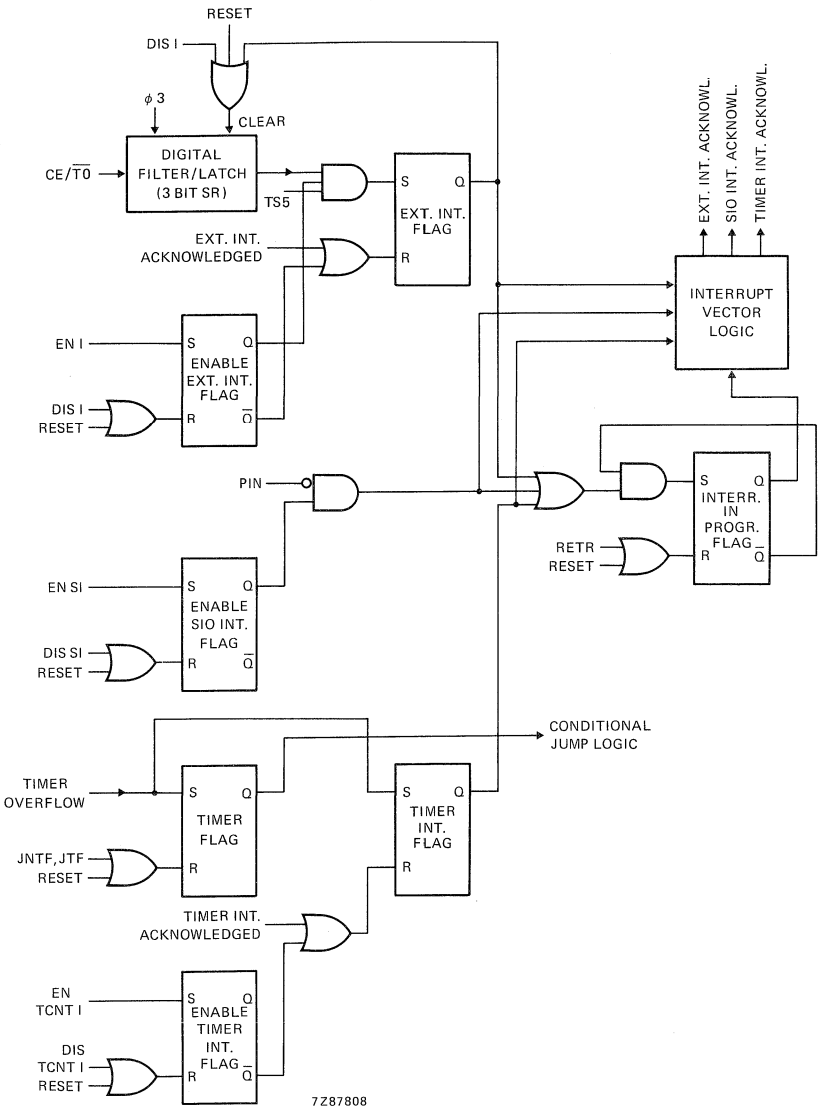


Fig.15 Interrupt logic.

Notes to Fig.15

1. CE/$\overline{TO}$ positive edge is always latched in the digital filter/latch.
2. Correct interrupt timing is ensured when CE/$\overline{TO}$ is LOW for < 4 CP followed by a HIGH for > 7 CP.
3. When the interrupt in progress flag is set, further interrupts are latched but ignored, until RETR is executed.
4. A DIS 1 instruction always clears a pending external interrupt.

Oscillator

The 3.58 MHz oscillator can be inhibited by the STOP instruction under software control. It is also inhibited when a low-voltage condition is present to prevent discharge of a weak back-up battery. Provided the supply voltage is within the operating range, the oscillator will be restarted after a STOP instruction by a HIGH level at either the CE/$\overline{TO}$ or RESET pin. The oscillator has the output drive capability for the DTMF generator (PCD3311C/3312C) via pin 16 (XTAL2). An external clock can be applied to pin 15 (XTAL1). A machine cycle consists of 10 time slots, each time slot being 3 oscillator periods.

In telephony applications the 3.58 MHz crystal provides a 8.4 µs machine cycle. The range of the clock frequency is from 450 kHz up to a maximum which is a function of the supply voltage (see Fig.23).

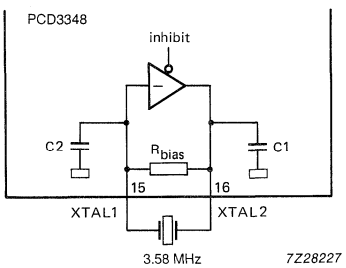


Fig.16 Oscillator with integrated elements.

Timer/event counter

An internal 8-bit up-counter is provided. This can count external events, modulo-32 machine cycles, or machine cycles directly. Table 3 gives the instructions that control the counter and prescaler, and the functions performed.

When used as a timer, the input to the counter is either the overflow or input of a 5-bit prescaler. When used as an event counter, LOW-to-HIGH transitions on pin 13 (T1) are counted. The maximum rate at which the counter may be incremented is once very machine cycle (182.6 kHz for a 8 µs machine cycle). When the counter overflows, the Timer Flag is set. The flag can be tested and reset using the JTF (jump if Timer Flag = 1) or JNTF instruction.

Overflow also generates an interrupt to the processor via setting of the Timer Interrupt Flag when the timer/event counter interrupt is enabled.

Table 3 Timer/event counter control.

FUNCTION	TIMER MODE MODULO-1, MODULO-32 (1)	COUNTER MODE
CLEAR	MOV T, A (A) = 0 or RESET	MOV T, A (A) = 0 or RESET
PRESET	MOV T, A	MOV T, A
START	STRT T	STRT CNT
STOP	STOP TCNT or RESET	STOP TCNT or RESET
TEST	JTF/JNTF	JTF/JNTF
READ (2)	MOV A, T	MOV A, T

Notes to Table 3

1. With prescaler select, PS = 0, the timer counts modulo-32 machines cycles, with PS = 1 it counts modulo-1 cycles (prescaler not used); prescaler cleared with STRT T, prescaler not readable.
2. READ does not disturb the counting process.

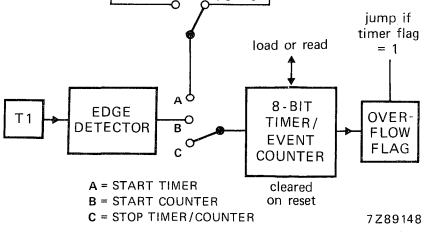


Fig.17 Timer/event counter.

Program Status Word

The Program Status Word (PSW) is an 8-bit word (1 byte) in the CPU which stores information about the current status of the microcontroller.

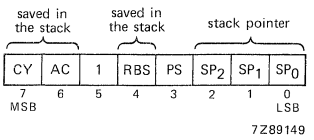


Fig.18 Program Status Word.

All bits can be read using the MOV A, PSW instruction. Bits 7 and 6 are set and cleared by CPU operation. Bit 4 can be changed by a SEL RB instruction, bit 3 by the MOV PSW, A instruction, and bits 0, 1 and 2 by the CALL, RET or RETR instructions and in the event of an interrupt. Bits 7, 6 and 4 are stored in the Program Counter stack during subroutine and interrupt calls. These bits are restored in the PSW with a RETR (return and restore) instruction which must be used at the end of an interrupt and can be used at the end of a normal subroutine. The RET instruction has no restore feature and cannot be used at the end of an interrupt.

Program Counter

A 13-bit Program Counter is used to facilitate 8 K of ROM being addressed. The arrangement of the bits is shown in Fig.19. During an interrupt subroutine PC_{11} and PC_{12} are forced to logic 0. All 13 bits are saved in the stack during CALL and interrupt routines.

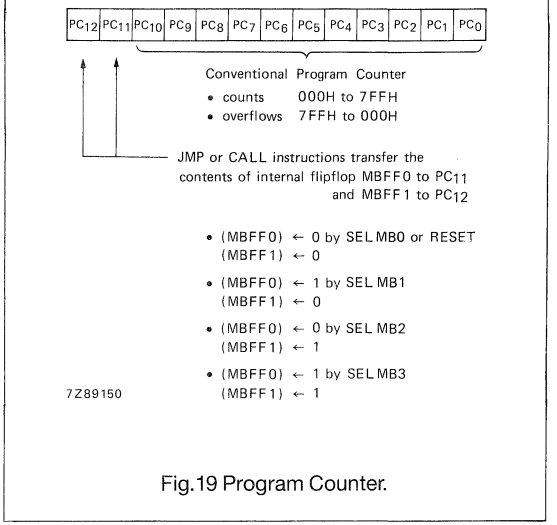


Fig.19 Program Counter.

Table 4 Program Status Word bits.

BIT	DESCRIPTION
0 to 2	Stack Pointer bits (SP_0), SP_1, SP_2)
3	prescaler select (PS); 0 = modulo-32; 1 = modulo-1 (no prescaling)
4	working register bank select (RBS); 0 = register bank 0; 1 = register bank 1
5	not used
6	auxiliary carry (AC); half-carry bit generated by an ADD instruction and used by the decimal adjust instruction DA A
7	carry (CY); the carry flag indicates that previous operation has resulted in an overflow of the accumulator.

Central Processing Unit

The PCD3348 has arithmetic, logical and branching capabilities. The DA A, SWAP A and XCHD instructions simplify BCD arithmetic and the handling of nibbles. The MOV A,@A instruction permits efficient table look-up from the current ROM page.

Conditional branch logic

The conditional branch logic within the processor enables several conditions, internal and external to the processor, to be tested by the user's program. Table 5 lists the conditional jump instructions used to change the program sequence. The DJNZ instruction decrements a designated register or data memory location and branches if the contents are not zero. This instruction is useful for looping control. The JMPP@A instruction allows multiway branches to destinations determined by the contents of the accumulator.

Test input T1 (pin 13)

The T1 input line can be used as:

- A test input for branch instructions JT1 and JNT1
- An external input to the event counter

When used as a test input:

- JT1 instruction tests for logic 1 level
- JNT1 instruction tests for logic 0 level

When used as an input to the event counter, T1 must be LOW for > 4 CP, followed by a HIGH for > 4 CP. The transition can be recognized with a repetition rate of once per 30 oscillator clock periods (1 machine cycle).

Reset

A positive-going signal on the RESET input/output:

- Sets the Program Counter to zero
- Selects location 0 of memory bank 0 and register bank 0
- Sets the Stack Pointer to zero (000); pointing to RAM address 8
- Disables the interrupts (external, timer and serial I/O)
- Stops the timer/event counter, then sets it to zero
- Sets the timer prescaler to modulo-32
- Resets the Timer Flag
- Sets all ports according to reset states
- Sets the serial I/O to slave receiver mode and disable the serial I/O
- Cancels IDLE and STOP mode

After the voltage is applied to RESET an internal delay of 1866 CP is introduced before the micro-controller commences operation.

Table 5 Conditional branches.

TEST	JUMP CONDITION	JUMP INSTRUCTION
Accumulator	all bits zero any bit non-zero	JZ JZN
Accumulator bit test	1	JB0 to JB7
Carry flag	1 0	JC JNC
Timer overflow flag	1 0	JTF JNTF
Test input T0	1 0	JNTO JTO (note 1)
Test input T1	1 0	JT1 JNT1
Register	non-zero	DJNZ

Note

1. Because of the inverted interrupt input CE/$\overline{TO}$ the conditional jump JTO is also inverted.

Power-on-reset and low-voltage detection

In telephony applications, correct operation of the PCD3348 during moments of slowly changing supply voltages and low-voltage conditions is essential. This achieved by the addition of an internal power-on-reset and low-voltage detection circuit. To allow an external RESET signal being fed into the PCD3348, the reset pin (pin 17) has been configured as an input/output.

While a reset condition exist in the detection circuit, pin 17 is pulled HIGH by TR1 controlled by the reset circuit. When the reset condition is not present a pull-down current source (TR2) will be activated. TR2 forces pin 17 LOW thus removing the RESET signal from the microcontroller. Since the level at pin 17 is recognized by the microcontroller, the reset time constant can be stretched by connecting an external capacitor between V_{DD} and pin 17 (see Fig.22). The signal at pin 17 can also be used as an output to reset other devices in the system.

The internal reset circuit monitors the PCD3348 supply voltage. If the voltage drops below the switching level (typ. 1.3 V), a reset is removed (pin 17 goes LOW), after a fixed delay (t_d), when the supply voltage rises above the switching level again. The delay ensures a complete reset even when the supply voltage quickly rises above switching level after initial switch-on. During a low-voltage condition the oscillator is inhibited to prevent complete discharge of a weak battery. The timing of the power-on-reset and low-voltage detection circuit is shown in Fig.21.

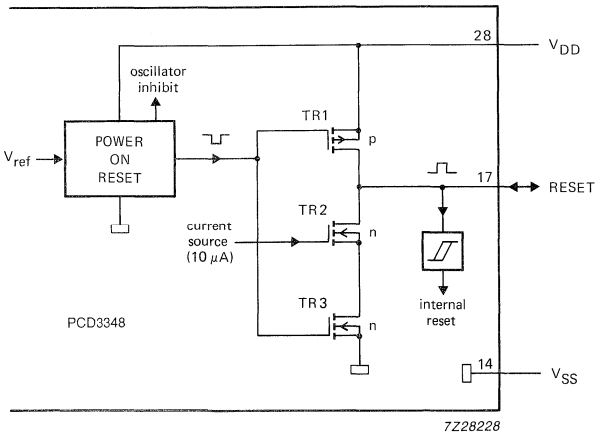
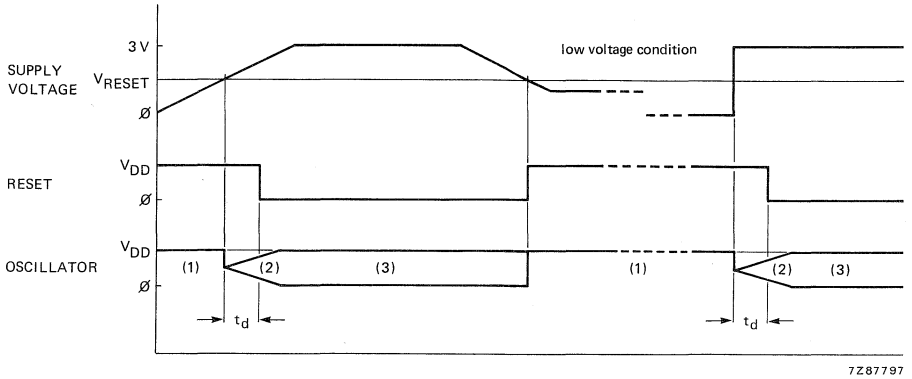


Fig.20 Power-on-reset configuration.



Notes to Fig.21.

(1) Oscillator inhibited
(2) Oscillator starting
(3) Oscillator running, but may be stopped with a STOP condition

Fig.21 Timing of power-on-reset and low-voltage detection.

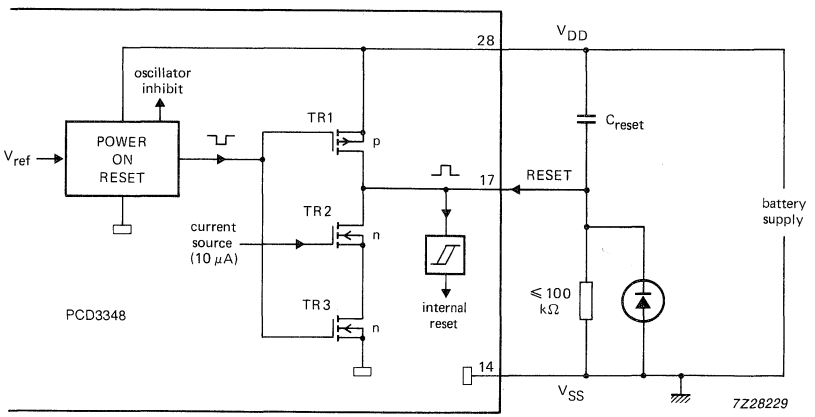


Fig.22 Stretched power-on-reset with external capacitor.

INSTRUCTION SET

The PCD3348 instruction set consists of over 80 one and two byte instructions and is based on the MAB8048 instruction set. New instructions include those for serial I/O operation and memory bank selection. Program code efficiency is high because all RAM locations and all ROM locations on a 256 byte page require only a single byte address.

Table 8 gives the instruction set of the PCD3348. Table 7 shows the instruction map and Table 6 details the symbols and definition descriptions that are used.

Table 6 Symbols and definition used in Table 8.

SYMBOL	DEFINITION DESCRIPTION
A	accumulator
ADDR	program memory address
Bb	bit designation (b = 0 to 7)
C	carry bit (bit CY)
CNT	event counter
D	mnemonic for 4-bit digit (nibble)
data	8-bit number or expression
I	interrupt
MBn	memory bank (n = 0 to 3)
MBFFn	memory bank flip-flop (n = 0 or 1)
P	mnemonic for 'in-page' operation
PC	Program Counter
Pp	port designation (p = 0, 1 or 2)
PSW	program status word
RB	register bank
RBS	register bank select
Rr	register designation (r = 0 to 7)
Sn	serial I/O register (n = 0, 1 or 2)
SP	Stack Pointer
T	timer
TF	Timer Flag
T0, T1	test 0 and 1 inputs
#	immediate data prefix
@	indirect address prefix
(X)	contents of X
((X))	contents of location addresses by X
←	is replaced by
↔	is exchanged with

Table 7 PCD3348 Instruction map.

first hexadecimal character of opcode ↓ — second hexadecimal character of opcode →

	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
0	NOP	IDLE		ADD A, #data	JMP page 0	EN I	JNTF addr	DEC A	IN A,Pp 0	1	2		MOV A,Sn 0	1		
1	INC @ Rr 0	1	JB0 addr	ADDC A,#data	CALL page 0	DIS I	JTF addr	INC A	INC Rr 0	1	2	3	4	5	6	7
2	XCH A, @Rr 0	1	STOP	MOV A, #data	JMP page 1	EN TCNTI	JNT0 addr	CLR A	XCH A,Rr 0	1	2	3	4	5	6	7
3	XCHD A, @Rr 0	1	JB1 addr		CALL page 1	DIS TCNTI	JT0 addr	CPL A	OUTL Pp,A 0	1	2		MOV Sn,A 0	1	2	
4	ORL A, @Rr 0	1	MOV A, T	ORL A, #data	JMP page 2	STRT CNT	JNT1 addr	SWAP A	ORL A,Rr 0	1	2	3	4	5	6	7
5	ANL A, @Rr 0	1	JB2 addr	ANL A, #data	CALL page 2	STRT T	JT1 addr	DA A	ANL A,Rr 0	1	2	3	4	5	6	7
6	ADD A, @Rr 0	1	MOV T, A		JMP page 3	STOP TCNT		RRC A	ADD A,Rr 0	1	2	3	4	5	6	7
7	ADDC A, @Rr 0	1	JB3 addr		CALL page 3			RR A	ADDC A,Rr 0	1	2	3	4	5	6	7
8				RET	JMP page 4	EN SI			ORL Pp,#data 0	1	2		MOV A,Dx	MOV Dx,A	ANL Dx,A	ORL Dx,A
9			JB4 addr	RETR	CALL page 4	DIS SI	JNZ addr	CLR C	ANL Pp,#data 0	1	2		MOV Sn,#data 0	1	2	
A	MOV @ Rr,A 0	1		MOVP A,@A	JMP page 5	SEL MB2		CPL C	MOV Rr,A 0	1	2	3	4	5	6	7
B	MOV @Rr, #data 0	1	JB5 addr	JMPP @A	CALL page 5	SEL MB3			MOV Rr,#data 0	1	2	3	4	5	6	7
C	DEC @Rr 0	1			JMP page 6	SEL RB0	JZ addr	MOV A,PSW	DEC Rr 0	1	2	3	4	5	6	7
D	XRL A, @Rr 0	1	JB6 addr	XRL A,#data	CALL page 6	SEL RB1		MOV PSW,A	XRL A,Rr 0	1	2	3	4	5	6	7
E	DJNZ @ Rr,addr 0	1			JMP page 7	SEL MB0	JNC addr	RL A	DJNZ Rr,addr 0	1	2	3	4	5	6	7
F	MOV A, @Rr 0	1	JB7 addr		CALL page 7	SEL MB1	JC addr	RLC A	MOV A,Rr 0	1	2	3	4	5	6	7

MBA281

Table 8 Instruction set.

MNEMONIC	OPCODE (HEX)	BYTES/ CYCLES	DESCRIPTION	FUNCTION		NOTES
ACCUMULATOR						
ADD A, Rr	6<8 + r>	1/1	Add register contents to A	(A) ← (A) + (Rr)	r = 0 to 7	1
ADD A, @Rr	6r	1/1	Add RAM data, addressed by Rr, to A	(A) ← (A) + ((Rr))	r = 0, 1	1
ADD A, #data	03 data	2/2	Add immediate data to A	(A) ← (A) + data		1
ADDC A, Rr	7<8 + r>	1/1	Add carry and register contents to A	(A) ← (A) + (Rr) + (C)	r = 0 to 7	1
ADDC A, @Rr	7r	1/1	Add carry and RAM data, addressed by Rr, to A	(A) ← (A) + ((Rr)) + (C)	r = 0, 1	1
ADDC A, #data	13 data	2/2	Add carry and immediate data to A	(A) ← (A) + data + (C)		1
ANL A, Rr	5<8 + r>	1/1	'AND' Rr with A	(A) ← (A) AND (Rr)	r = 0 to 7	
ANL A, @Rr	5r	1/1	'AND' RAM data, addressed by Rr, with A	(A) ← (A) AND ((Rr))	r = 0, 1	
ANL A, #data	53 data	2/2	'AND' immediate data with A	(A) ← (A) AND data		
ORL A, Rr	4<8 + r>	1/1	'OR' Rr with A	(A) ← (A) OR (Rr)	r = 0 to 7	
ORL A, @Rr	4r	1/1	'OR' RAM data, addressed by Rr, with A	(A) ← (A) OR ((Rr))	r = 0, 1	
ORL A, #data	43 data	2/2	'OR' immediate data with A	(A) ← (A) OR data		
XRL A, Rr	D<8 + r>	1/1	'XOR' Rr with A	(A) ← (A) XOR (Rr)	r = 0 to 7	
XRL A, @Rr	Dr	1/1	'XOR' RAM data, addressed by Rr, with A	(A) ← (A) XOR ((Rr))	r = 0, 1	
XRL A, #data	D3 data	2/2	'XOR' immediate data with A	(A) ← (A) XOR data		
INC A	17	1/1	Increment A by 1	(A) ← (A) + 1		
DEC A	07	1/1	Decrement A by 1	(A) ← (A) − 1		
CLR A	27	1/1	Clear A to zero	(A) ← 0		
CPL A	37	1/1	One's complement A	(A) ← NOT(A)		
RL A	E7	1/1	Rotate A left	$(A_{n+1}) \leftarrow (A_n)$ $(A_0) \leftarrow (A_7)$	n = 0 to 6	
RLC A	F7	1/1	Rotate A left through carry	$(A_{n+1}) \leftarrow A_n$ $(A_0) \leftarrow (C), (C) \leftarrow (A_7)$	n = 0 to 6	2
RR A	77	1/1	Rotate A right	$(A_n) \leftarrow (A_{n+1})$ $(A_7) \leftarrow (A_0)$	n = 0 to 6	
RRC A	67	1/1	Rotate A right through carry	$(A_n) \leftarrow (A_{n+1})$ $(A_7) \leftarrow (C), (C) \leftarrow (A_0)$	n = 0 to 6	2
DA A	57	1/1	Decimal adjust A	(A) ← (A) + 06H if AC = 1 or $(A_{0-3}) > 9$; (A) ← (A) + 60H if $(A_{4-7}) > 9$		2
SWAP A	47	1/1	Swap nibbles of A	$(A_{4-7}) \leftrightarrow (A_{0-3})$		2
DATA MOVES						
MOV A, Rr	F<8 + r>	1/1	Move register contents to A	(A) ← (Rr)	r = 0 to 7	
MOV A, @Rr	Fr	1/1	Move RAM data, addressed by Rr, to A	(A) ← ((Rr))	r = 0, 1	
MOV A, #data	23 data	2/2	Move immediate data to A	(A) ← data		
MOV Rr, A	A<8 + r>	1/1	Move accumulator contents to register	(Rr) ← (A)	r = 0 to 7	
MOV @Rr, A	Ar	1/1	Move accumulator contents to RAM location addressed by Rr	((Rr)) ← (A)	r = 0, 1	

MNEMONIC	OPCODE (HEX)	BYTES/ CYCLES	DESCRIPTION	FUNCTION		NOTES
MOV Rr, #data	B<8 + r> data	2/2	Move immediate data to Rr	(Rr) ← data	r = 0 to 7	
MOV @Rr, #data	Br data	2/2	Move immediate data to RAM location addressed by Rr	((R0)) ← data	r = 0, 1	
XCH A, Rr	2<8 + r>	1/1	Exchange accumulator contents with Rr	(A) ↔ (Rr)	r = 0 to 7	
XCH A, @Rr	2r	1/1	Exchange accumulator contents with RAM data addressed by Rr	(A) ↔ ((Rr))	r = 0, 1	
XCHD A, @Rr	3r	1/1	Exchange lower nibbles of A and RAM data addressed by Rr	$(A_{0-3}) \leftrightarrow ((Rr_{0-3}))$	r = 0, 1	
MOV A, PSW	C7	1/1	Move PSW contents to accumulator	(A) ← (PSW)		
MOV PSW, A	D7	1/1	Move accumulator bit 3 to PSW_3 (PS)	(PS) ← (A_3)		3
MOVP A, @A	A3	1/2	Move indirectly addressed data in current page to A	(PC_{0-7}) ← (A), (A) ← ((PC))		
CARRY FLAG						
CLR C	97	1/1	Clear carry bit	(C) ← 0		2
CPL C	A7	1/1	Complement carry bit	(C) ← NOT(C)		2
REGISTER						
INC Rr	1<8 + r>	1/1	Increment register by 1	(Rr) ← (Rr) + 1	r = 0 to 7	
INC @Rr	1r	1/1	Increment RAM data, addressed by Rr, by 1	((Rr)) ← ((Rr)) + 1	r = 0, 1	
DEC Rr	C<8 + r>	1/1	Decrement register by 1	(Rr) ← (Rr) – 1	r = 0 to 7	
DEC @Rr	Cr	1/1	Decrement RAM data, addressed by Rr, by 1	((Rr)) ← ((Rr)) – 1	r = 0, 1	
BRANCH						
JMP addr	<2n>4 addr	2/2	Unconditional jump within a 2 K bank	(PC_{8-10}) ← n (PC_{0-7}) ← addr (PC_{11-12}) ← (MBFF 0-1)	n = 0 to 7	
JMPP @A	B3	1/2	Indirect jump within a page	(PC_{0-7}) ← ((A))		
DJNZ Rr, addr	E<8 + r> addr	2/2	Decrement Rr by 1 and jump if not zero to addr	(Rr) ← (Rr) – 1; if (Rr) not zero, then (PC_{0-7}) ← addr	r = 0 to 7	
DJNZ @Rr, addr	Er	2/2	Decrement RAM data, addressed by Rr, by 1 and jump if not zero to addr	((Rr)) ← ((Rr)) – 1; if ((Rr)) not zero, then (PC_{0-7}) ← addr	r = 0, 1	
JBb addr	<2b + 1>2 addr	2/2	Jump to addr if Accumulator bit b = 1	If (A_b) = 1, then (PC_{0-7}) ← addr	b = 0 to 7	
JC addr	F6 addr	2/2	Jump to addr if C = 1	If (C) = 1, then (PC_{0-7}) ← addr		
JNC addr	E6 addr	2/2	Jump to addr if C = 0	If (C) = 0, then (PC_{0-7}) ← addr		
JZ addr	C6 addr	2/2	Jump to addr if A = 0	If (A) = 0, then (PC_{0-7}) ← addr		
JNZ addr	96 addr	2/2	Jump to addr if A is NOT zero	If (A) ≠ 0, then (PC_{0-7}) ← addr		
JT0 addr	36 addr	2/2	Jump to addr if T0 = 1	If T0 = 1, then (PC_{0-7}) ← addr		
JNT0 addr	26 addr	2/2	Jump to addr if T0 = 0	If T0 = 0: (PC_{0-7}) ← addr		

MNEMONIC	OPCODE (HEX)	BYTES/ CYCLES	DESCRIPTION	FUNCTION	NOTES
JT1 addr	56 addr	2/2	Jump to addr if T1 = 1	If T1 = 1: $(PC_{0-7}) \leftarrow$ addr	
JNT1 addr	46 addr	2/2	Jump to addr if T1 = 0	If T1 = 0: $(PC_{0-7}) \leftarrow$ addr	
JTF addr	16 addr	2/2	Jump to addr if Timer Flag = 1	If TF = 1: $(PC_{0-7}) \leftarrow$ addr	4
JNTF addr	06 addr	2/2	Jump to addr if Timer Flag = 0	If TF = 0: $(PC_{0-7}) \leftarrow$ addr	4
TIMER/EVENT COUNTER					
MOV A, T	42	1/1	Move timer/event counter contents to accumulator	(A) ← (T)	
MOV T, A	62	1/1	Move accumulator contents to timer/event counter	(T) ← (A)	
STRT CNT	45	1/1	Start event counter		
STRT T	55	1/1	Start timer		
STOP TCNT	65	1/1	Stop timer/event counter		
EN TCNTI	25	1/1	Enable timer/event counter interrupt		
DIS TCNTI	35	1/1	Disable timer/event counter interrupt		
CONTROL					
EN I	05	1/1	Enable external (chip enable) interrupt		
DIS I	15	1/1	Disable external (chip enable) interrupt		
SEL RB0	C5	1/1	Select register bank 0	(RBS) ← 0	5
SEL RB1	D5	1/1	Select register bank 1	(RBS) ← 1	5
SEL MB0	E5	1/1	Select program memory bank 0	(MBFF0) ← 0, (MBFF1) ← 0	10
SEL MB1	F5	1/1	Select program memory bank 1	(MBFF0) ← 1, (MBFF1) ← 0	10
SEL MB2	A5	1/1	Select program memory bank 2	(MBFF0) ← 0, (MBFF1) ← 1	10
SEL MB3	B5	1/1	Select program memory bank 3	(MBFF0) ← 1, (MBFF1) ← 1	10
STOP	22	1/1	Enter Stop mode		
IDLE	01	1/1	Enter Idle mode		
SUBROUTINE					
CALL addr	<2n + 1>4 addr	2/2	Jump to subroutine	$((SP)) \leftarrow (PC)$, $(PSW_{4,6,7})$ $(SP) \leftarrow (SP) + 1$ $(PC_{8-10}) \leftarrow n$ $(PC_{0-7}) \leftarrow$ addr $(PC_{11-12}) \leftarrow$ (MBFF0-1) n = 0 to 7	6
RET	83	1/2	Return from subroutine	$(SP) \leftarrow (SP) - 1$ $(PC) \leftarrow ((SP))$	6
RETR	93	1/2	Return from interrupt and restore bits 4, 6, 7 of PSW	$(SP) \leftarrow (SP) - 1$ $(PSW_{4,6,7}) +$ $(PC) \leftarrow ((SP))$	6

MNEMONIC	OPCODE (HEX)	BYTES/ CYCLES	DESCRIPTION	FUNCTION	NOTES
PARALLEL INPUT/OUTPUT					
IN A, Pp	08 09 0A	1/2	Input port p data to accumulator	(A) ← (P0) (A) ← (P1) (A) ← (P2)	 7
OUTL Pp, A	38 39 3A	1/2	Output accumulator data to port p	(P0) ← (A) (P1) ← (A) (P2) ← (A)	
ANL Pp, #data	98 data 99 data 9A data	2/2	AND port p with immediate data	(P0) ← (P0) AND data (P1) ← (P1) AND data (P2) ← (P2) AND data	
ORL Pp, #data	88 data 89 data 8A data	2/2	OR port p data with immediate data	(P0) ← (P0) OR data (P1) ← (P1) OR data (P2) ← (P2) OR data	
SERIAL INPUT/OUTPUT					
MOV A, S_n	0C 0D	1/2	Move serial I/O register contents to accumulator	(A) ← (S0) n = 0, 1 (A) ← (S1)	8
MOV S_n, A	3C 3D 3E	1/2	Move accumulator contents to serial I/O register	(S0) ← (A) n = 0, 1, 2 (S1) ← (A) (S2) ← (A)	9
MOV S_n, #data	9C data 9D data 9E data	2/2	Move immediate data to serial I/O register	(S0) ← data n = 0, 1, 2 (S1) ← data (S2) ← data	
EN SI	85	1/1	Enable serial I/O interrupt		
DIS SI	95	1/1	Disable serial I/O interrupt		
NOP	00	1/1	No operation	$(PC_{0-10}) \leftarrow (PC_{0-10}) + 1$	

Notes

1. PSW CY, AC affected.
2. PSW CY affected.
3. PSW PS affected.
4. Execution of a JTF or JNTF instruction resets the Timer Flag (TF).
5. PSW RBS affected.
6. PSW SP_0, SP_1, SP_2 affected.
7. (A) = 1111, P23, P22, P21, P20
8. (S1) has a different meaning for read and write operations. See section 4.11.4.
9. (S2) is a write only register. Reading S2 will give value FFH.
10. SEL MB instructions may not be used within interrupt routines.

LIMITING VALUES

In accordance with the Absolute Maximum System (IEC 134).

SYMBOL	PARAMETER	MIN.	MAX.	UNIT
V_{DD}	supply voltage	−0.8	+ 8	V
V_I	all input voltages	0.8	V_{DD} + 0.8	V
$\pm I_I$, $\pm I_O$	DC current into any input or output	-	10	mA
P_{tot}	total power dissipation	-	500	mW
P_O	power dissipation per output except P23, SCLK	-	50	mW
P_O	power dissipation per output P23, SCLK	-	180	mW
T_{stg}	storage temperature range	-65	+ 150	°C
T_{amb}	operating ambient temperature range	-25	+ 70	°C
Tj	operating junction temperature	-	+ 125	°C

THERMAL RESISTANCE

SYMBOL	PACKAGE	CONDITIONS	MAX.	UNIT
$R_{th\ j\text{-}a}$	SOT117	junction to ambient	120	K/W
$R_{th\ j\text{-}a}$	SOT136A	junction to ambient	150	K/W

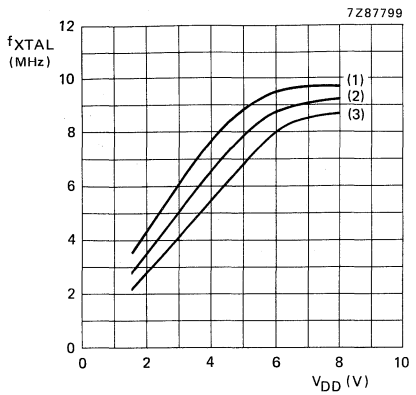
DC CHARACTERISTICS

V_{DD} = 2.75 to 6 V; V_{SS} = 0 V; T_{amb} = –25 to + 70 °C; all voltages with respect to V_{SS}; f = 3.58 MHz with R_S = 50; unless otherwise specified. See Figures 23 to 26.

SYMBOL	PARAMETER	CONDITIONS	MIN.	TYP.	MAX.	UNIT
V_{DD}	supply voltage operating		1.8	-	6	V
V_{DD}	STOP mode for RAM retention		1.0	-	6	V
I_{DD}	supply current operating	V_{DD} = 3 V	-	600	-	µA
I_{DD}	IDLE mode	V_{DD} = 3 V	-	300	-	µA
I_{DD}	STOP mode (see note 1)	V_{DD} = 1.8 V; T_{amb} = 25 °C	-	1.2	2.5	µA
I_{DD}	STOP mode (see note 1)	V_{DD} = 1.8 V; T_{amb} = 55 °C	-	-	5	µA
I_{DD}	STOP mode (see note 1)	V_{DD} = 1.8 V; T_{amb} = 70 °C	-	-	10	µA
RESET I/O						
V_{RESET}	switching level		-	1.3	-	V
I_{OL}	sink current	$V_{DD} > V_{RESET}$	-	7	-	µA
Inputs						
V_{IL}	input voltage LOW		0	-	0.3 V_{DD}	V
V_{IH}	input voltage HIGH		0.7 V_{DD}	-	V_{DD}	V
$\pm I_L$	input leakage current	$V_{SS} < V_I < V_{DD}$	-	-	1	µA
Outputs						
V_{OL}	output voltage LOW	$V_I = V_{SS}$ or V_{DD}; $\lvert I_O \rvert < 1$ µA	-	-	0.05	V
I_{OL}	output sink current LOW except P23/SDA,SCLK	V_{DD} = 3 V; V_O = 0.4 V	0.75	1.5	-	mA
I_{OL}	output sink current LOW P23/SDA, SCLK		1.5	-	-	mA
$-I_{OH}$	pull-up output source current HIGH	V_{DD} = 3 V; V_O = 0.9 V_{DD};	25	-	-	µA
$-I_{OH}$	pull-up output source current HIGH	V_{DD} = 3 V; V_O = V_{SS};	-	-	200	µA
$-I_{OH}$	push-pull output source current HIGH	V_{DD} = 3 V; V_O = V_{DD} –0.4 V	0.75	1.5	-	mA

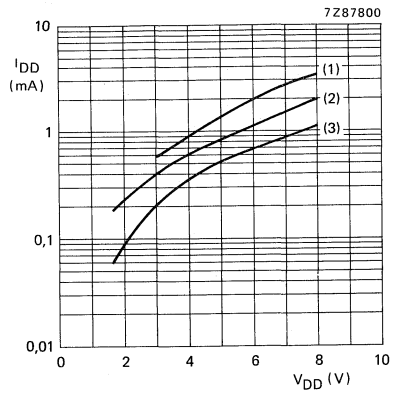
Note to the DC characteristics

1. Crystal connected between XTAL1 and XTAL2; SCL and SDA pulled to V_{DD} via 5.6 Ω resistor; CE and T1 as V_{SS}.



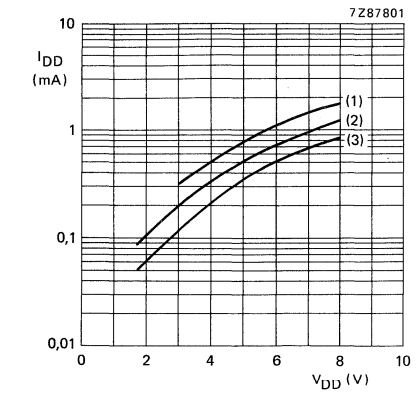
(1) T_{amb} = –25 °C
(2) T_{amb} = + 25 °C
(3) T_{amb} = + 70 °C

Fig.23 Maximum clock frequency (f_{XTAL}) as a function of the supply voltage (V_{DD}).



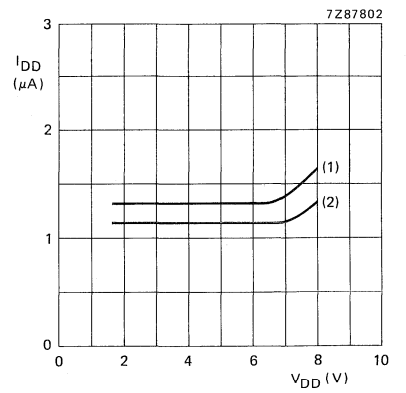
(1) clock frequency = 4 MHz
(2) clock frequency = 2 MHz
(3) clock frequency = 500 kHz

Fig.24 Typical supply current (I_{DD}) in operating mode as a function of the supply voltage (V_{DD}); T_{amb} = + 25 °C.



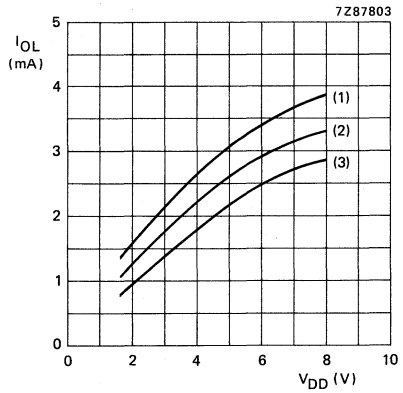
(1) clock frequency = 4 MHz
(2) clock frequency = 2 MHz
(3) clock frequency = 500 kHz

Fig.25 Typical supply current (I_{DD}) in IDLE mode as a function of the supply voltage (V_{DD}); T_{amb} = + 25 °C.



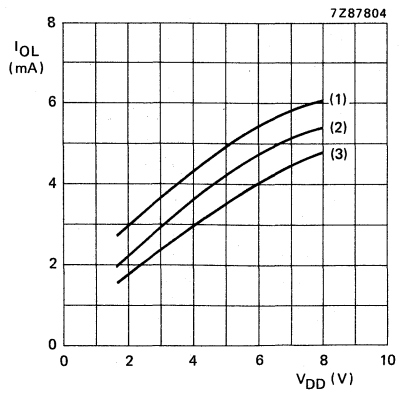
(1) T_{amb} = + 70 °C
(2) T_{amb} = + 25 °C

Fig.26 Typical supply current (I_{DD}) in STOP mode as a function of the supply voltage (V_{DD}).



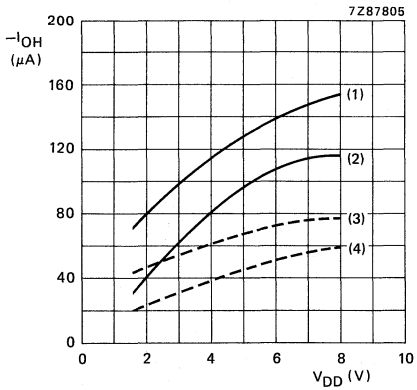
(1) T_{amb} = −25 °C
(2) T_{amb} = + 25 °C
(3) T_{amb} = + 70 °C

Fig.27 Output sink current LOW (I_{OL}), except outputs P23/SDA and SCLK, as a function of supply voltage (V_{DD}); V_O = 0.4 V.



(1) T_{amb} = −25 °C
(2) T_{amb} = + 25 °C
(3) T_{amb} = + 70 °C

Fig.28 Output current LOW (I_{OL}), outputs P23/SDA and SCLK, as a function of supply voltage (V_{DD}); V_O = 0.4 V.



(1) T_{amb} = + 25 °C; V_O = V_{SS}
(2) T_{amb} = + 25 °C; V_O = 0.9 V_{DD}
(3) T_{amb} = + 70 °C; V_O = V_{SS}
(4) T_{amb} = + 70 °C; V_O = 0.9 V_{DD}

Fig.29 Output source current HIGH ($-I_{OH}$) as a function of supply voltage (V_{DD})

AC CHARACTERISTICS

Maximum rise and fall times between 10 and 90% levels;
C_L = 50 pF; T_{amb} = + 70 °C.

SYMBOL	PARAMETER	SUPPLY VOLTAGE (VDD)			UNIT
		1.8	3.0	6.0	
t_f	fall time	200	100	70	ns
t_r	rise time	200	100	80	ns

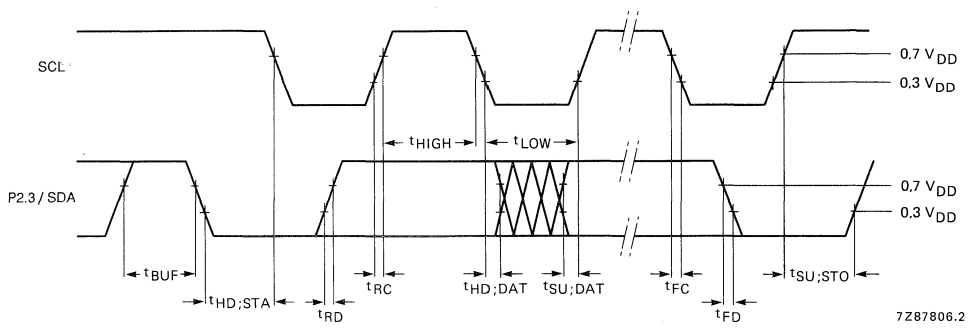


Fig.30 PCD3348 timing requirements for the P23 and SCLK input signals.

Table 9 Input timing shown in Fig.30.

SYMBOL	TIMING
t_{BUF}	≥ $14t_{XTAL}$
$t_{HD;STA}$	≥ $14t_{XTAL}$
t_{HIGH}	≥ $17t_{XTAL}$
t_{LOW}	≥ $17t_{XTAL}$
$t_{SY;STO}$	≥ $14t_{XTAL}$
$t_{HD;DAT}$	< 0
$t_{SU;DAT}$	≥ 250 ns
t_{RD}	≤ 1 µs
t_{RC}	≤ 1 µs
t_{FD}	≤ 1 µs
t_{FC}	≤ 0.3 µs

Notes to Table 9

1. t_{XTAL} = one period of the XTAL input frequency (f_{XTAL}).
   For f_{XTAL} = 3.58 MHz; t_{XTAL} = 280 ns.
2. These figures apply to all modes.

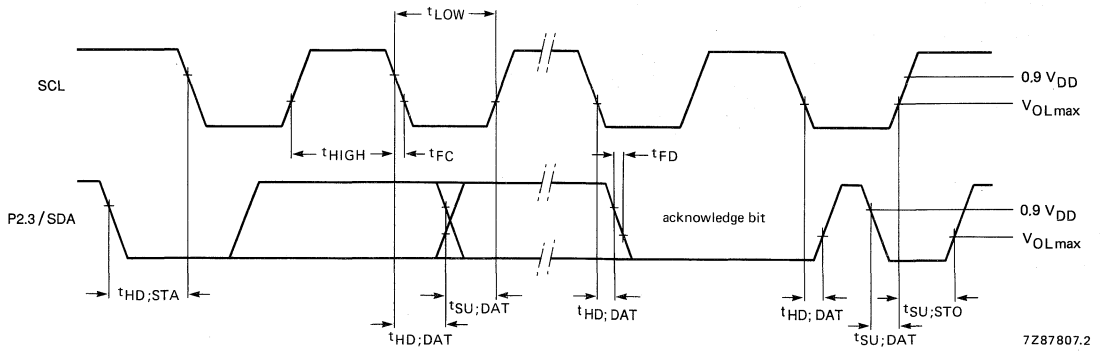


Fig.31 PCD3348 timing requirements for the P23 and SCLK output signals.

Table 10 Output timing shown in Fig.31.

SYMBOL	TIMING	
	NORMAL MODE (ASC in S2 = 0)	**LOW-SPEED MODE (ASC in S2 = 1)**
$t_{HD;STA}$	$1/2(DF + 9)t_{XTAL}$	$3/4(DF + 9)t_{XTAL}$
t_{HIGH}	$1/2(DF)t_{XTAL}$	$3/4\ (DF)t_{XTAL}$
t_{LOW}	$1/2(DF)t_{XTAL}$	$1/4(DF)t_{XTAL}$
$t_{SU;STO}$	$1/2(DF \circledR 3)t_{XTAL}$	$1/4(DF \circledR 3)t_{XTAL}$
$t_{HD;DAT}$ (slave transmitter) any DF	 $\geq 9t_{XTAL}$ $\leq 12t_{XTAL}$	 $\geq 9t_{XTAL}$ $\leq 12t_{XTAL}$
$t_{HD;DAT}$ for DF ≤ 51 for DF ≤ 99	$\geq 9t_{XTAL}$ $\leq 12t_{XTAL}$ - -	- - $\geq 9t_{XTAL}$ $\leq 12t_{XTAL}$
$t_{SU;DAT}$ (master transmitter) for DF > 51 for DF > 99 for DF ≤ 51 for DF < 99	 $\geq 15t_{XTAL}$ $\leq 24t_{XTAL}$ - $\geq 9t_{XTAL}$ -	 - - $\geq 15t_{XTAL}$ $\leq 24t_{XTAL}$ $\geq 9t_{XTAL}$
t_{AC}	$\geq 9t_{XTAL}$ $\leq 12t_{XTAL}$	$\geq 9t_{XTAL}$ $\leq 12t_{XTAL}$
t_{FD}, t_{FC}	≤ 100 ns at C_b = 400 pF	≤ 100 ns at C_b = 400 pF

Notes to Table 10

1. t_{XTAL} = one period of the XTAL input frequency (f_{XTAL}).
   For f_{XTAL} = 3.58 MHz; t_{XTAL} = 280 ns.
2. DF = divisor (see Table 2 Serial I/O section).
3. C_b = the maximum bus capacitance for each line.

APPLICATION INFORMATION

A block diagram of an electronic featurephone built around the PCD3348 is shown in Fig.32. It comprises the following dedicated telephony ICs:

TEA1060/1061	transmission circuit for telephony
PCD3312C	DTMF generator with Serial I/O
PCE2111 or	2 LCD drivers in LCD module
PCF8577	MB7020160
PCD8571	1 K RAM's with Serial I/O; the number of RAM's depend on the required amount of stored telephone numbers
PCD3360	programmable multi-tone ringer

A detailed application diagram of the PCD3348 with PCD3312C (DTMF), two PCD8571 (RAM) and two PCE2211 (LCD display drivers) is shown in Fig.33.

Row 5 of the keyboard contains the following special keys:

- P program and autodial
- FL flash or register recall
- R redial or extended redial
- AP access pause

Row 6 contains the different diode options.

Columns 5 and 6 contain the button keys M0 to M9; single name keys for repertory telephone numbers.

Additional information is available on request for the following:

- Serial I/O
- I^2C-bus specification
- Interrupt logic
- Instruction set descriptions
- Software routines for an intelligent telephone set.

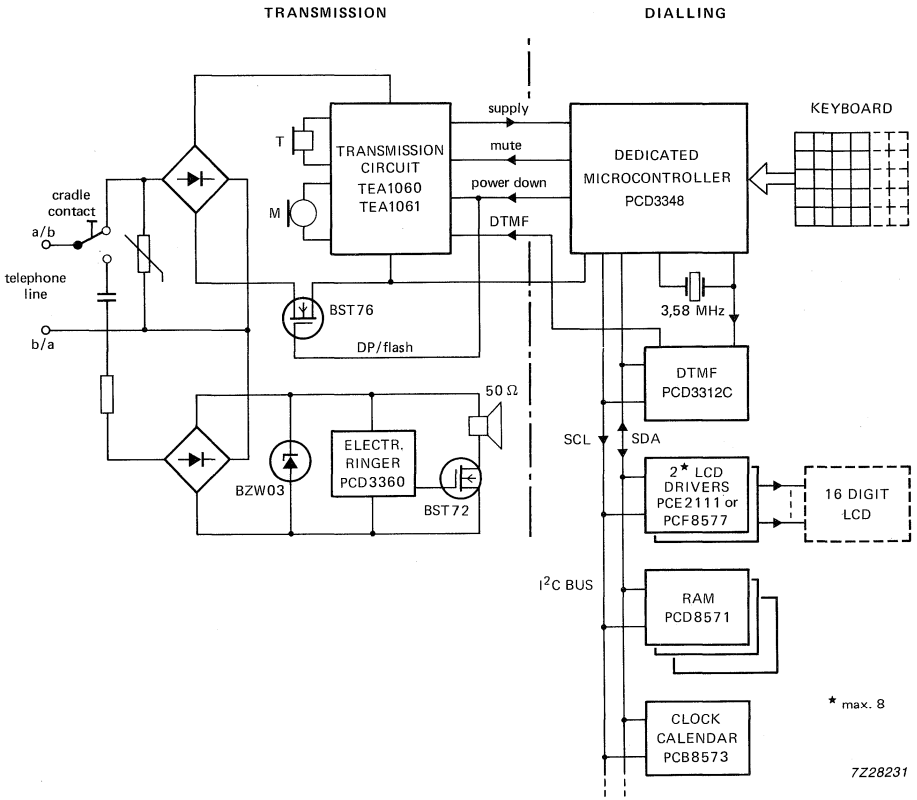


Fig.32 Block diagram of electronic featurephone with common line interface.

Purchase of Philips' I²C components conveys a license under the Philips' I²C patent to use the components in the I²C-system provided the system conforms to the I²C specifications defined by Philips.

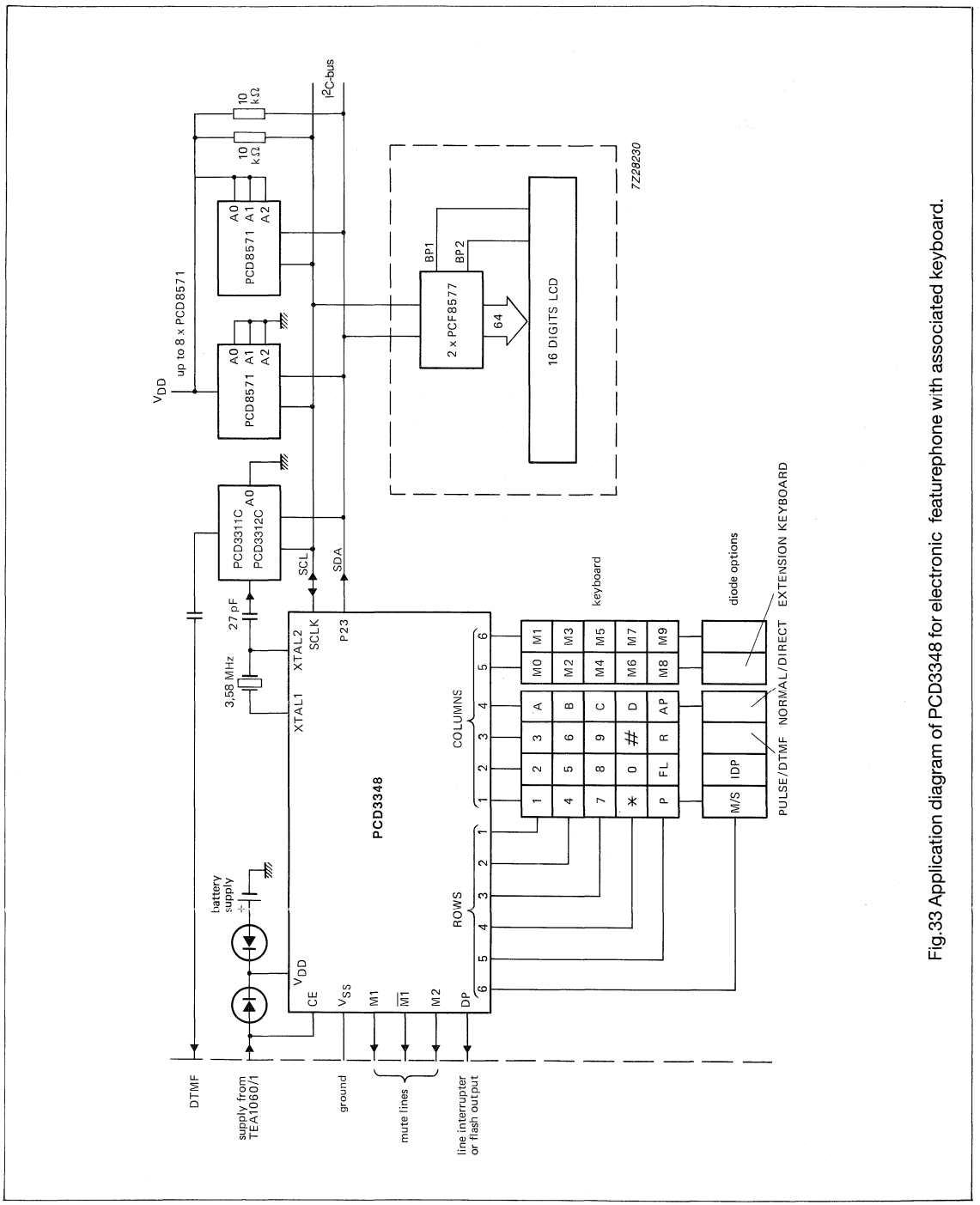


Fig.33 Application diagram of PCD3348 for electronic featurephone with associated keyboard.

CMOS MICROCONTROLLER WITH ON-CHIP DTMF GENERATOR

GENERAL DESCRIPTION

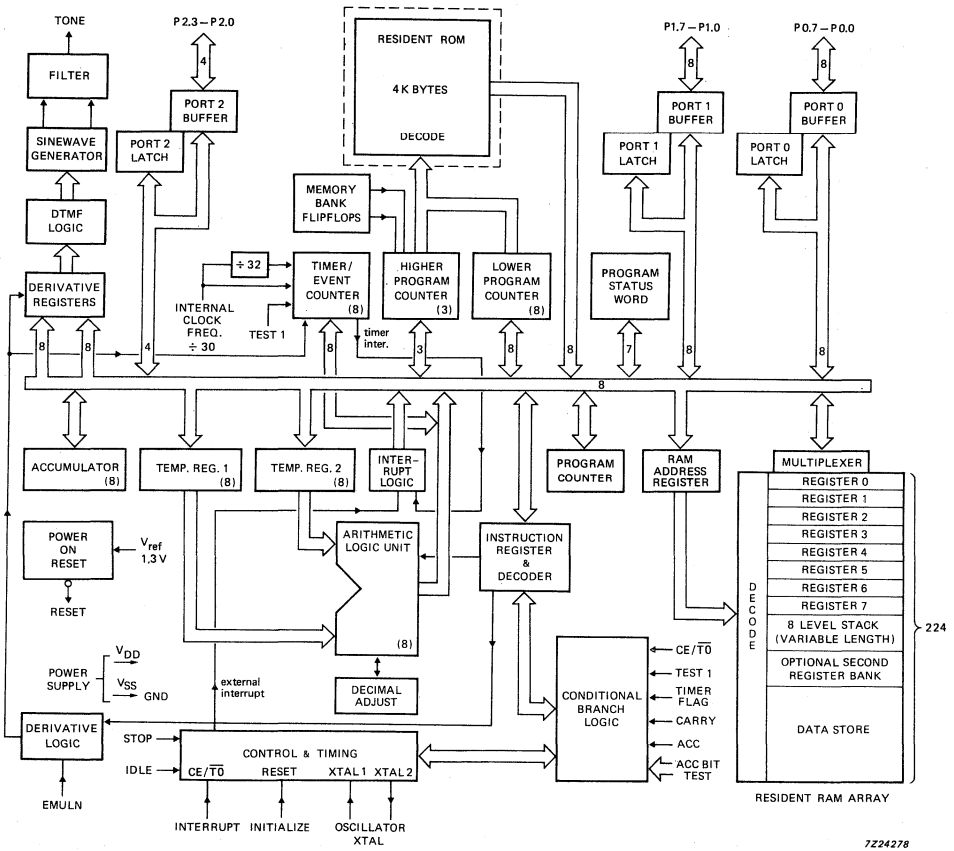
The PCD3349 is a single-chip 8-bit microcontroller fabricated in CMOS and is a member of the PCD33XX family. It has an on-chip dual tone multi-frequency (DTMF) generator and other features for application in telephone sets. For further detailed information, see PCD33XX family specifications.

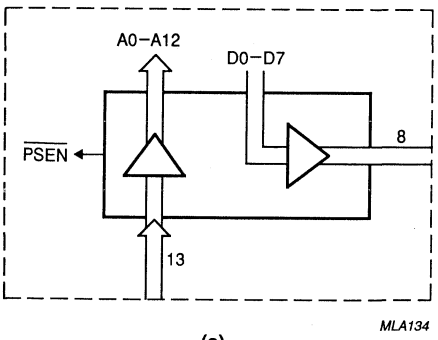
Features

- 8-bit CPU, ROM, RAM, I/O in a single 28-lead DIL or SO package
- 4096 ROM bytes
- 224 RAM bytes
- On-chip DTMF tone generator
- On-chip voltage reference for supply and temperature-independent tone output
- On-chip filtering for low output distortion (CEPT CS203 compatible)
- 20 quasi-bidirectional I/O port lines
- Two test inputs: one of which is also the external interrupt input (CE/$\overline{T0}$)
- Single-level vectored interrupts: external, timer/event counter
- 8-bit programmable timer/event counter
- Over 80 instructions (based on MAB8048)
- All instructions 1 or 2 cycles
- Clock frequency 3.58 MHz
- Single supply voltage from 2.5 V to 6 V
- Low standby voltage and current
- STOP and IDLE mode
- On-chip oscillator
- Configuration of all I/O port lines individually selected by mask: pull-up, open drain or push-pull
- Power-on-reset circuit and low supply voltage detection
- Reset state of all ports individually selected by mask
- Operating temperature range: −25 to + 70 °C

PACKAGE OUTLINES

PCD3349P: 28-lead DIL; plastic (SOT117).
PCD3349T: 28-lead mini-pack; plastic (SO28; SOT136A).





(a)

Fig. 1. PCD3349 block diagram: the function in the dotted outline is replaced as shown in (a) for the PCD3344B 'piggy-back' version.

DEVELOPMENT DATA

PINNING (for normal operation)

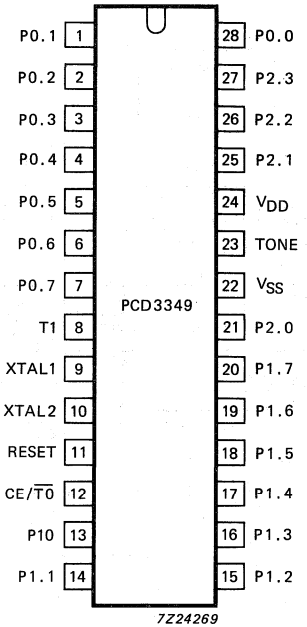


Fig. 2 Pinning diagram.

PIN DESIGNATION

28, 1-7	P0.0-P0.7	Port 0: 8-bit quasi-bidirectional I/O port.
8	T1	Test 1: test input, directly tested by conditional branch instructions JT1 and JNT1. T1 also functions as an input to the 8-bit timer/event counter using the STRT CNT instruction.
9	XTAL1	Crystal input: connection to the timing component (crystal) which determines the frequency of the internal oscillator; is also the input for an external clock source.
10	XTAL2	Connection to other side of timing component.
11	RESET	Reset input (active HIGH): used to initialize the processor or output of the power-on-reset circuit.
12	CE/$\overline{T0}$	Interrupt/Test 0: external interrupt input (sensitive to positive-going edge)/test input pin. When used as a test input is directly tested by conditional branch instructions JT0 and JNT0.
13-20	P1.0-P1.7	Port 1: 8-bit quasi-bidirectional I/O port.
21, 25-27	P2.0-P2.3	Port 2: 4-bit quasi-bidirectional I/O port.
22	V_{SS}	Ground: circuit earth potentional.
23	TONE	Tone output: single or dual tone frequency output with on-chip filtering for low output distortion (CEPT CS203 compatible). This generator is controlled via the internal processor bus.
24	V_{DD}	Power supply: 2.5 to 6 V.

FUNCTIONAL DESCRIPTION

Program memory PCD3349

The program memory comprises 4096 bytes (8-bit words) in a read-only memory (ROM). Each location is directly addressable by the program counter. The memory is mask-programmed at the factory. Figure 3 shows the program memory map.

Three program memory locations are of special importance:

- Location 0; contains the first instruction to be executed after the processor is initialized (RESET),
- Location 3; contains the first byte of an external interrupt service subroutine,
- Location 7; contains the first byte of a timer/event counter interrupt service subroutine.

The program memory is divided into location 'pages', each of 256 bytes. This division applies only for conditional branches. A CALL instruction can transfer control to a subroutine on any 'page'; RET and RETR instructions can transfer control from a subroutine back to the main program.

Data memory PCD3349

Data memory consists of 224 bytes (8-bit words) of random-access data memory (RAM). All locations are indirectly addressable using RAM pointer registers; up to 16 designated locations are directly addressable. Memory also includes an 8-level program counter stack addressed by a 3-bit stack pointer. Figure 4 shows the data memory map.

Working registers

Locations 0 to 7 are designated as working registers, directly addressable by the direct register instructions. Ease of addressing, and a minimum requirement of instruction bytes to manipulate their contents, makes these locations suitable for storing frequently-addressed intermediate results. This bank of registers can be selected by the SEL RB0 instruction.

Executing the select register bank instruction SEL RB1, designates locations 24 to 31 as working registers, instead of locations 0 to 7, and these are then directly addressable. This second bank of working registers may be used as an extension of the first or reserved for use during interrupt service subroutines saving the first bank for use in the main program. If the second bank is not used, locations 24 to 31 may serve as general purpose RAM.

The first locations of each bank contain the RAM pointer registers R0, R1, R0' and R1', which indirectly address all RAM locations.

All RAM locations make efficient program loop counters when used with the decrement register and test instruction DJNZ.

Program counter stack

Locations 8 to 23 may be designated as an 8-level program counter stack (2 locations per level), or as general purpose RAM. The program counter stack (Fig. 5) enables the processor to keep track of the return addresses and status generated by interrupts or CALL instructions by storing the contents of the program counter prior to servicing the subroutine. A 3-bit stack pointer determines which of the eight register pairs of the program counter stack will be loaded with next generated return address.

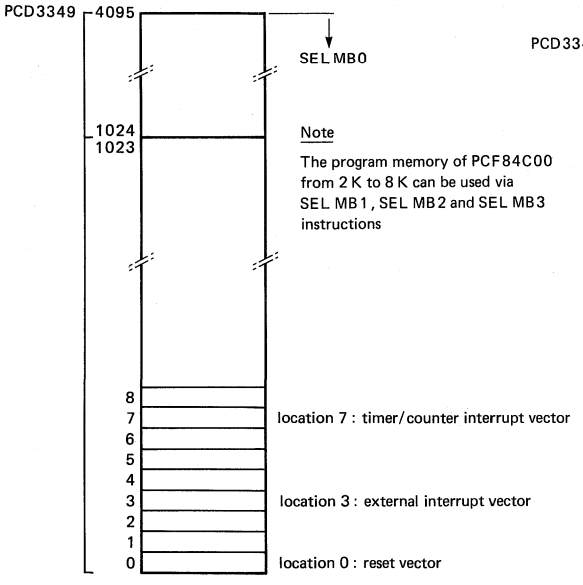


Fig. 3 Program memory map.

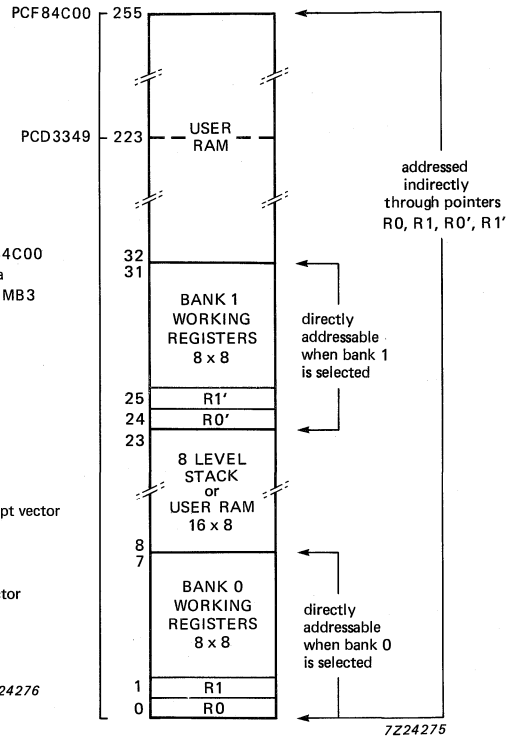


Fig. 4 Data memory map.

DEVELOPMENT DATA

The stack pointer, when initialized to 000 by RESET, points to RAM locations 8 and 9. On the first subroutine CALL or interrupt, the contents of the program counter and bits 4, 6 and 7 of the program status word (PSW) are transferred to locations 8 and 9. The stack pointer increments by one and points to locations 10 and 11 ready for another CALL. Because an address may be up to 13 bits long, two bytes must be used to store each address.

At the end of a subroutine, which is signalled by a return instruction (RET or RETR), the stack pointer decrements by one and the contents of the register pair on top of the stack are transferred to the program counter. The saved PSW bits are transferred to the PSW only by the RETR instruction.

If not all 8 levels of subroutine and interrupt nesting are used, the unused portion of the stack may be used as any other indirectly addressable RAM locations. Possible locations from 32 to 223 may be used for storage of program variables or data.

Nesting of subroutines within subroutines can continue up to 8 times without overflowing the stack. If overflow does occur the deepest address stored (locations 8 and 9) will be overwritten and lost since the stack pointer overflows from 111 to 000. It also underflows from 000 to 111.

The value of the saved contents of the program counter is different for an interrupt CALL compared to a normal CALL to subroutine. With an interrupt CALL, the program counter return address is saved; with a subroutine CALL, the saved program counter value is one less than the program counter return address.

FUNCTIONAL DESCRIPTION (continued)

Program counter stack (continued)

stack pointer									
1 1 1	----------								R23/22
1 1 0	----------								R21/20
1 0 1	----------								R19/18
1 0 0	----------								R17/16
0 1 1	----------								R15/14
0 1 0	----------								R13/12
0 0 1	----------								R11/10
0 0 0	PSW7	PSW6	PC12	PSW4	PC11	PC10	PC9	PC8	R9
	PC7	PC6	PC5	PC4	PC3	PC2	PC1	PC0	R8

Fig. 5 Program counter stack.

IDLE and STOP modes

IDLE mode

When the microcontroller enters the IDLE mode via the IDLE instruction (H'01') the oscillator and timer/counter are kept running. The microcontroller exits from the IDLE mode by one of two interrupts if they are enabled or by activating a RESET. If the interrupt is not enabled the processor will remain in the IDLE mode. An active signal on the RESET pin restarts the microcontroller and a normal RESET sequence is executed (see Fig. 6).

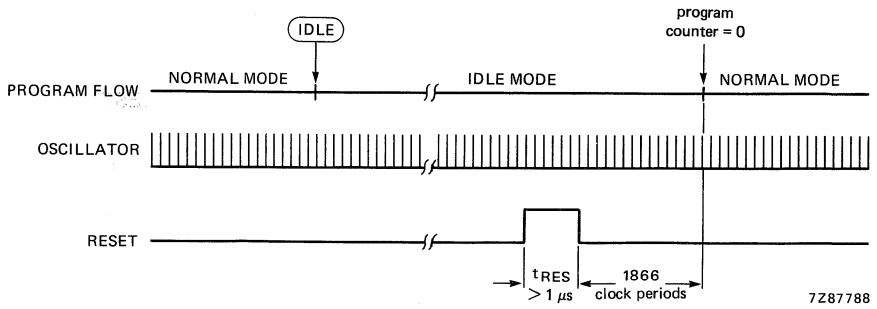


Fig. 6 Exit from IDLE mode via a RESET.

An active signal coming from an enabled interrupt causes the execution of the normal interrupt routine since normal interrupt scanning is still being carried out. A LOW-to-HIGH transition on the external interrupt pin (CE/$\overline{T0}$) reactivates the microcontroller. A HIGH level applied to CE/$\overline{T0}$ will reactivate the microcontroller only in the STOP mode. Thus, if CE/$\overline{T0}$ was HIGH before the microcontroller entered the IDLE mode, it must go LOW before the microcontroller can be reactivated (see Fig. 7).

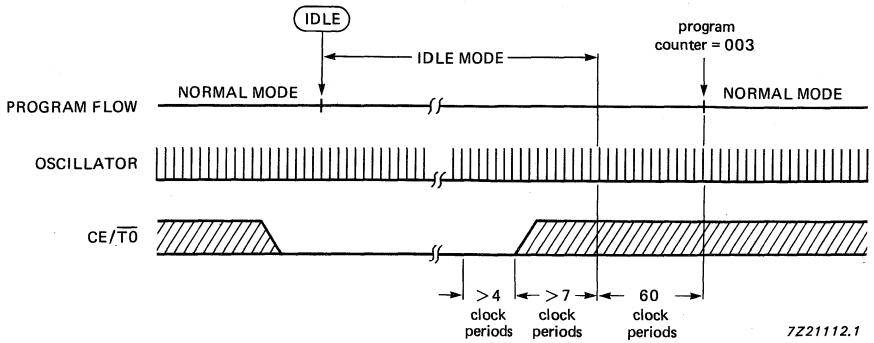


Fig. 7 Exit from IDLE mode via an interrupt.

Wake-up from the IDLE mode is ensured when CE/$\overline{T0}$ is LOW for 4 CP (clock periods) followed by a HIGH for 7 CP. After the initial forced CALL H'003' operation (60 CP) the program continues with the external interrupt service routine.

STOP mode

The microcontroller enters the STOP mode by the STOP instruction (H'22'). The oscillator is switched off. The internal status of the CPU, RAM contents and the state of I/O ports are not affected. The microcontroller can be brought-out of the STOP mode by an active signal at the external interrupt input or by an external RESET signal. When one of these two signals is applied an internal delay of 1866 CP is provided to ensure that all internal clocks are operating correctly before restart (see Fig. 8).

If the microcontroller exits from the STOP mode by activating RESET, a normal RESET sequence is executed.

If the microcontroller exits the STOP mode by pulling the external interrupt input pin HIGH, an interrupt sequence is executed only if the external interrupt is enabled. In this event the microcontroller resumes the normal program sequence after returning from the interrupt routine, as in the normal mode. If the interrupt is not enabled, it continues the normal program sequence, executing the instruction following the STOP instruction.

The microcontroller is restarted by a HIGH level applied at the CE/$\overline{T0}$ pin, and not by a LOW-to-HIGH transition as in a normal interrupt mechanism.

When the CE/$\overline{T0}$ level is active during the STOP instruction then no STOP is executed.

A HIGH level on the external interrupt input of at least 1 μs will cause the microcontroller to exit the STOP mode.

DEVELOPMENT DATA

FUNCTIONAL DESCRIPTION (continued)

STOP mode (continued)

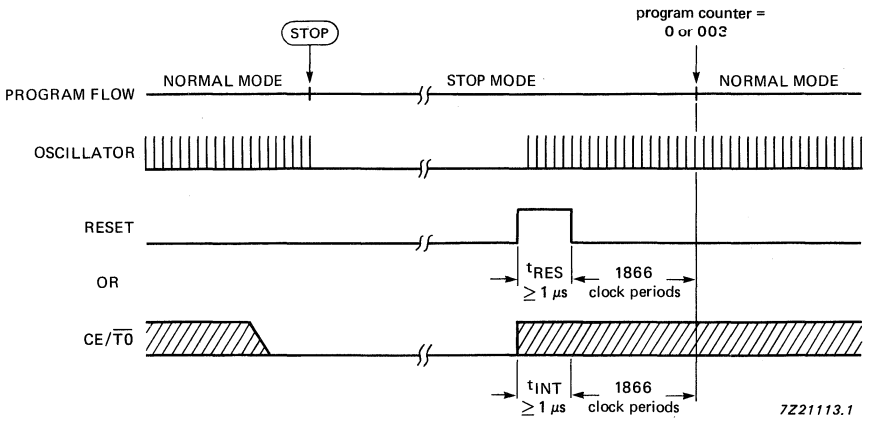


Fig. 8 Entering and exiting the STOP mode.

Tone output (DTMF mode)

Control of the sinewave generator

The on-chip sinewave oscillator is controlled by the 'derivative' registers Dx (x = H'0' to 'FF'). The instruction that controls the derivative registers is shown in Table 1.

Table 1 Derivative register control

mnemonic	opcode	description	function
MOV Dx,A	8D Dx	move accumulator contents to derivative register	(Dx) ← (A)

The instruction is 2 cycles/2 bytes. The second byte selects the derivative register to be addressed (H'0' to 'FF'). Register H'01' is for control of HIGH group frequencies, and register H'02' for control of LOW group frequencies. Thus data transport from accumulator to derivative register D01 is done by the 2-byte opcode 8D,01.

Generation of frequencies

The single and dual tones at the tone output are filtered by an on-chip switched-capacitor filter followed by an on-chip active RC low-pass filter. These ensure that the total harmonic distortion of the DTMF tones fulfil the CEPT CS 203 recommendations. An on-chip reference voltage provides output tone levels that are independent of the supply voltage.

The output frequency can be calculated as follows:

$$f_{out} = \frac{f_{XTAL}}{23\,(x + 2)}\ \text{Hz}$$

x = 60 to 255 and is the decimal value of the appropriate ROM-code (see Table 2)

Table 2 ROM-codes for DTMF applications

telephone keyboard symbol	contents of low register (hex)	contents of high register (hex)
0	A3	72
1	DD	7F
2	DD	72
3	DD	67
4	C8	7F
5	C8	72
6	C8	67
7	B5	7F
8	B5	72
9	B5	67
A	DD	5D
B	C8	5D
C	B5	5D
D	A3	5D
*	A3	7F
#	A3	67

DTMF generation is stopped by loading H'00' into both derivative registers.

I/O facilities

The PCD3349 family has 22 I/O lines arranged as:

- Port 0 parallel port of 8 lines (P0.0 to P0.7)
- Port 1 parallel port of 8 lines (P1.0 to P1.7)
- Port 2 parallel port of 4 lines (P2.0 to P2.3)
- CE/$\overline{T0}$ external interrupt and test input. When used as a test input it can be directly tested by conditional branch instructions JT0 and JNT0
- T1 test input which can alter program sequences when tested by conditional jump instructions JT1 and JNT1. T1 also functions as an input to the 8-bit timer/event counter.

DEVELOPMENT DATA

FUNCTIONAL DESCRIPTION (continued)

Parallel ports

All parallel ports can be used as outputs or inputs, their structure is quasi-bidirectional. Output data written to a port is latched and remains unchanged until rewritten. Input data is not latched and so must be present until read by an input instruction.

Input lines are fully CMOS compatible, output lines can drive one LS-TTL or CMOS load.

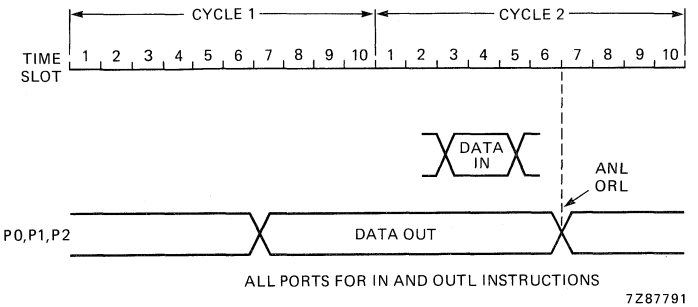


Fig. 9 Timing diagram of all ports on IN and OUTL instructions; for ANL and ORL instructions, the ports change on the time slot 7 of cycle 2.

Fig. 10 shows the quasi-bidirectional I/O interface with push-pull output and switched pull-up current source. Each line is pulled up to V_{DD} via a constant current source (TR4), which is enabled via TR3 whenever one of the two output latches contains a logic 1. This current source provides sufficient current for a TTL HIGH level, yet can be pulled LOW by an external CMOS device, thus allowing the same pin to be used for both input and output.

When a logic 1 is written to the line for the first time ($\overline{MQ}$ = 1, $\overline{SQ}$ = 0), TR2 is switched on for the duration of the internal write pulse (one oscillator period) to provide a fast transition from logic 0 to logic 1. Subsequent writing of a logic 1 to the port lines will not switch TR2 on. This prevents unnecessary current through external components connected to the port lines of the same port which might be in the input mode and also connected to ground.

When a logic 0 is written to the line, TR3 switches off the current source. Current sinking capability is provided by TR1, which is now switched on. When used as an input, a logic 1 must first be written to the line, otherwise TR1 will remain low impedance.

In telephone applications this switched pull-up source may not be sufficient. Therefore the PCD3349 offers the possibility to select individually the 20 parallel port pins by the following mask options:

Option 1 – STANDARD PORT; quasi-bidirectional I/O with switched pull-up current source of 100 μA (typ.) and P-channel booster transistor TR2 (1.5 mA). TR2 is active only during 1 clock cycle (0.28 μs at 3.58 MHz).

Option 2 – OPEN DRAIN; quasi-bidirectional I/O with only an N-channel open drain output. Application as an output requires connection of an external pull-up resistor (Fig. 11).

Option 3 – PUSH-PULL OUTPUT; drive capability of the output will be 1.5 mA (typ.) at V_{DD} = 3 V in both polarities. To avoid a large current flowing through the output transistors during the input mode, these push-pull pins must be used only as outputs (Fig. 12).

Also, individual mask selection of the RESET state of these port pins can be achieved by appending the following options S and R to options 1, 2 or 3.

Option S-SET; after RESET this pin will be initialized to HIGH.

Option R-RESET; after RESET this pin will be initialized to LOW.

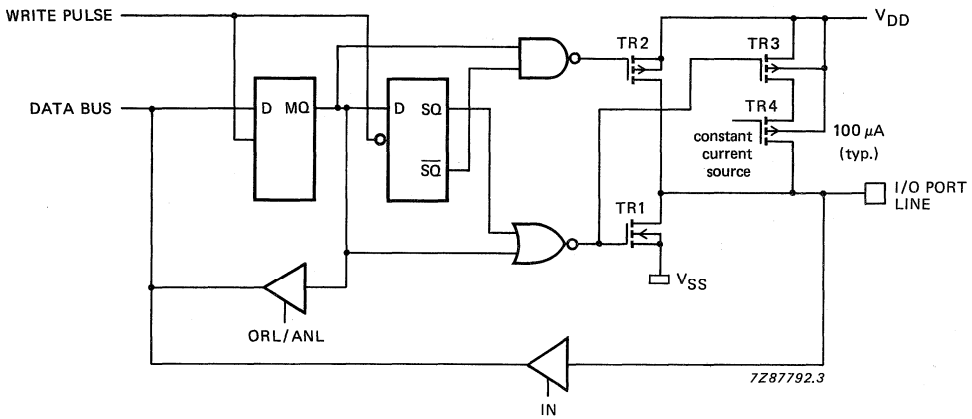


Fig. 10 Standard output with switched pull-up current source.

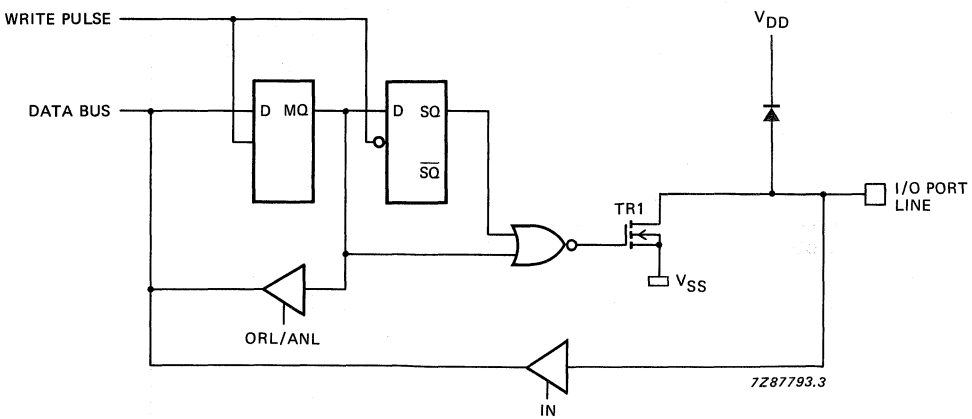


Fig. 11 Open drain output.

DEVELOPMENT DATA

FUNCTIONAL DESCRIPTION (continued)

Parallel ports (continued)

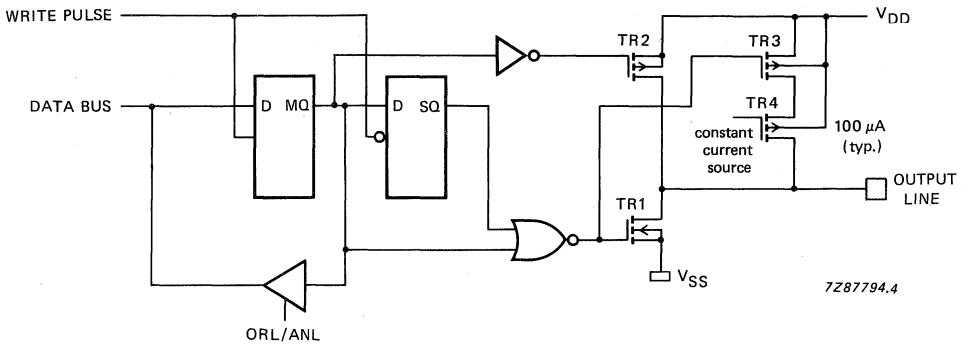


Fig. 12 Push-pull output.

Interrupts (see Fig. 13(a) and Fig. 13(b))

When an interrupt routine is entered, the contents of the program counter and bits 4, 6 and 7 of the PSW are saved in the program counter stack. The contents of the accumulator can only be saved by user software. Interrupt acknowledgement can be carried out by software via I/O ports. All interrupt routines must reside in memory bank 0; the SEL MB1, SEL MB2 and SEL MB3 instructions may not be used in an interrupt routine. An interrupt routine can only be terminated by the RETR (return and restore) instruction. During an interrupt routine, subroutine calls must be terminated by the RET instruction. Using the RETR instruction to terminate a subroutine called in an interrupt routine would terminate the interrupt routine prematurely and result in a wrong return address.

1. **External interrupt**

When the external interrupt is enabled, a LOW-to-HIGH transition on the CE/$\overline{TO}$ input initiates an external interrupt routine which forces a call to program memory location 3. The program counter points to the external interrupt vector address (003 H) between 2.6 and 3.6 machine cycles after the transition occurs. Interrupt latency depends on the instruction that is being executed when the transition occurs. External interrupts are latched in the External Interrupt Flag (EIF) even when they are not enabled. Execution of a DIS I instruction clears previously latched interrupts, the digital filter latch and the external interrupt flag.

2. **Timer/counter interrupt**

When the timer interrupt is enabled, a timer/counter overflow sets the Timer Interrupt Flag (TIF) and forces a CALL to location 7. The timer interrupts are only latched when they are enabled. The timer flag is set every time the timer/counter overflows and is not automatically reset when the timer/counter interrupt routine is called. It can only be cleared by the JTF and JNTF instructions or by a hardware RESET.

DEVELOPMENT DATA

3. Simultaneous interrupts

If simultaneous interrupts occur their priority is as follows:

external (highest);
timer/counter (lowest).

An interrupt routine can only be interrupted by a hardware RESET and cannot be interrupted by other interrupts (which will be latched if enabled). When the interrupt routine is terminated by the RETR instruction, at least one instruction of the main program will be executed before another interrupt routine is entered.

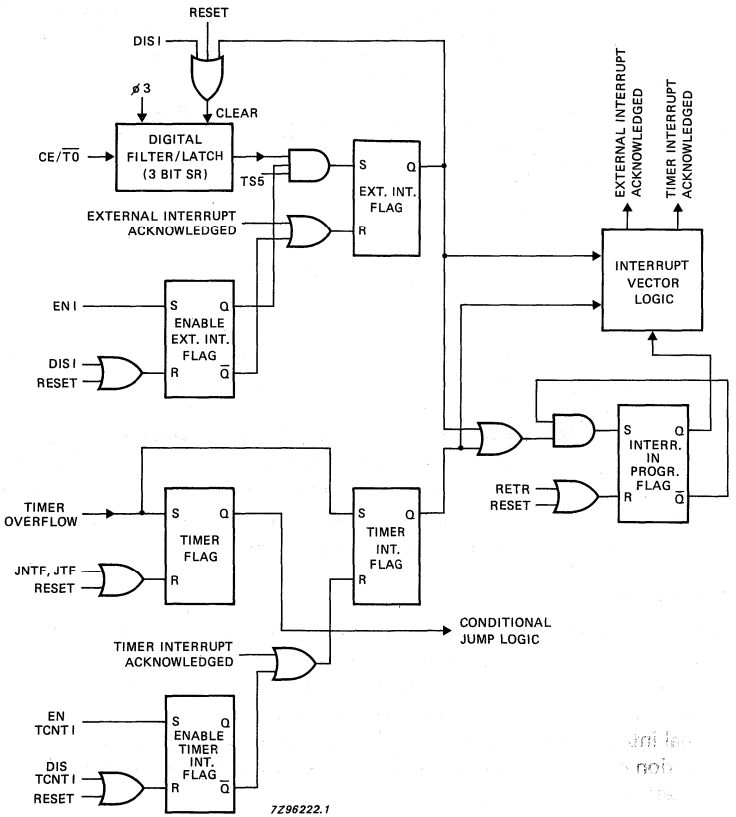


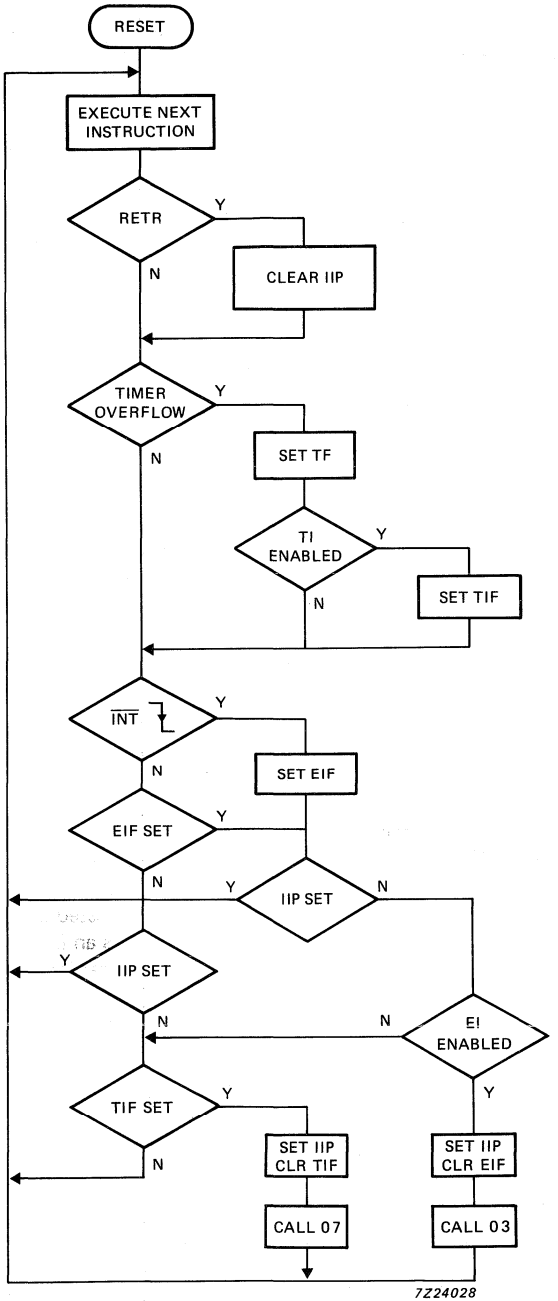
Fig. 13(a) Interrupt logic.

Notes to figure 13(a)

1. CE/$\overline{T0}$ positive edge is always latched in the digital filter/latch.
2. Correct interrupt timing is ensured when CE/$\overline{T0}$ is LOW for $>$ 4 CP followed by a HIGH for $>$ 7 CP.
3. When the interrupt in progress flag is set, further interrupts are latched but ignored, until RETR is executed.
4. A DIS I instruction always clears a pending external interrupt.
5. For all flip-flops, RESET overrules SET.

FUNCTIONAL DESCRIPTION (continued)

Interrupts (continued)



EI External Interrupt
TI Timer/counter Interrupt
EIF External Interrupt Flag
TF Timer Flag
IIP Interrupt In Progress flag

Fig. 13(b) Interrupt flowchart.

Oscillator (see Fig. 14)

The 3.58 MHz oscillator can be inhibited by the STOP instruction under software control. It is also inhibited when a low-voltage condition is present to prevent discharge of a weak back-up battery.

Provided the supply voltage is within the operating range, the oscillator will be restarted after a STOP instruction by a HIGH level at either the CE/$\overline{T0}$ or RESET pin.

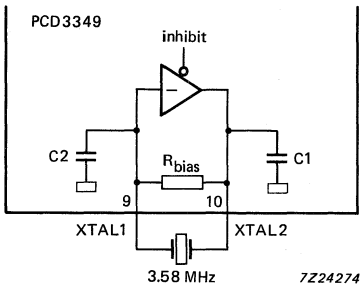


Fig. 14 Oscillator with integrated elements.

The oscillator has an output drive capability from pin 10 (XTAL2). An external clock can be applied to pin 9 (XTAL1). A machine cycle comprises 10 time slots, each time slot being 3 oscillator periods.

In telephony applications the 3.58 MHz crystal provides an 8.4 μs machine cycle. The range of the clock frequency is from 100 kHz up to a maximum which is a function of the supply voltage.

Timer/event counter (see Fig. 15)

An internal 8-bit up-counter is provided. This can count external events, modulo-32 machine cycles, or machine cycles directly. Table 3 gives the instructions that control the counter and the prescaler, and the functions performed.

When used as a timer, the input to the counter is either the overflow or input of a 5-bit prescaler. When used as an event counter, LOW-to-HIGH transitions on pin 8 (T1) are counted. The maximum rate at which the counter may be incremented is once every machine cycle (182.6 kHz for an 8.4 μs machine cycle). When the counter overflows, the timer flag is set. The flag can be tested and reset using the JTF (jump if timer flag = 1) or JNTF instruction. Overflow also generates an interrupt to the processor via setting of the Timer Interrupt Flag when the timer/event counter interrupt is enabled.

FUNCTIONAL DESCRIPTION (continued)

Timer/event counter (continued)

Table 3 Timer/event counter control

function	timer mode modulo-1, modulo-32*	counter mode
CLEAR	MOV T,A (A) = 0 or RESET	MOV T,A (A) = 0 or RESET
PRESET	MOV T,A	MOV T,A
START	STRT T	STRT CNT
STOP	STOP TCNT or RESET	STOP TCNT or RESET
TEST	JTF/JNTF	JTF/JNTF
READ**	MOV A,T	MOV A,T

* With prescaler select, PS = 0, the timer counts modulo-32 machine cycles, with PS = 1 it counts modulo-1 cycles (prescaler not used); prescaler cleared with STRT T, prescaler not readable.

** READ does not disturb the counting process.

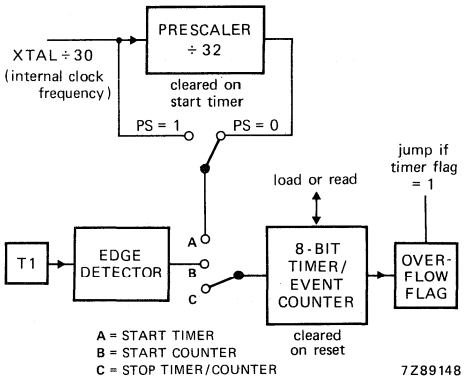


Fig. 15 Timer/event counter.

Program status word (see Fig. 16)

The program status word (PSW) is an 8-bit word (1 byte) in the CPU which stores information about the current status of the microcontroller.
The PSW bits are:

- Bits 0 to 2 — stack pointer bits (SP_0, SP_1, SP_2)
- Bit 3 — prescaler select (PS); 0 = modulo-32; 1 = modulo-1 (no prescaling)
- Bit 4 — working register bank select (RBS); 0 = register bank 0; 1 = register bank 1
- Bit 5 — not used (1)
- Bit 6 — auxiliary carry (AC); half-carry bit generated by an ADD instruction and used by the decimal adjust instruction DA A
- Bit 7 — carry (CY); the carry flag indicates that previous operation has resulted in an overflow of the accumulator.

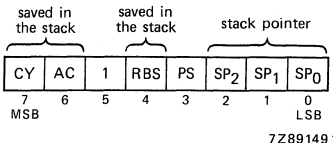


Fig. 16 Program status word.

All bits can be read using the MOV A, PSW instruction. Bits 7 and 6 are set and cleared by CPU operation. Bit 4 can be changed by a SEL RB instruction, bit 3 by the MOV PSW, A instruction, and bits 0, 1 and 2 by the CALL, RET or RETR instructions and in the event of an interrupt. Bits 7, 6 and 4 are stored in the program counter stack during subroutine and interrupt calls. These bits are restored in the PSW with a RETR (return and restore) instruction which must be used at the end of an interrupt and can be used at the end of a normal subroutine. The RET instruction has no restore feature and cannot be used at the end of an interrupt.

Program counter (see Fig. 17).

A 12-bit program counter is used to facilitate 8 K bytes of ROM being addressed. The arrangement of the bits is shown in figure 17. During an interrupt subroutine PC_{11} and PC_{12} are forced to logic 0. All 12 bits are saved in the stack during CALL and interrupt routines.

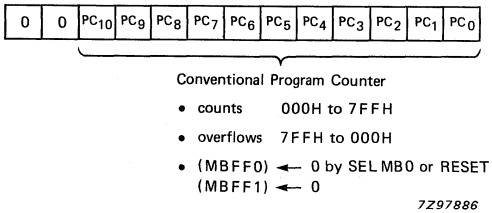


Fig. 17 Program counter.

Central processing unit

The PCD3349 has arithmetic, logical and branching capabilities. The DA A, SWAP A and XCHD instructions simplify BCD arithmetic and the handling of nibbles. The MOVP A,@A instruction permits efficient table look-up from the CURRENT ROM page.

FUNCTIONAL DESCRIPTION (continued)

Conditional branch logic

The conditional branch logic within the processor enables several conditions, internal and external to the processor, to be tested by the user's program.
Table 4 lists the conditional jump instructions used to change the program sequence. The DJNZ instruction decrements a designated register or data memory location and branches if the contents are not zero. This instruction is useful for looping control. The JMPP@A instruction allows multiway branches to destinations determined by the contents of the accumulator.

Table 4 Conditional branches

test	jump condition	jump instruction
accumulator	all bits zero	JZ
	any bit non-zero	JNZ
accumulator bit test	1	JB0 to JB7
carry flag	1	JC
	0	JNC
timer overflow flag	1	JTF
	0	JNTF
test input T0	1	JNT0
	0	JT0*
test input T1	1	JT1
	0	JNT1
register	non-zero	DJNZ

* Because of the inverted interrupt input CE/$\overline{T0}$ the conditional jump JT0 is also inverted.

Test input T1 (pin 8)

The T1 input line can be used as:

- A test input for branch instructions JT1 and JNT1
- An external input to the event counter

When used as a test input:

- JT1 instruction tests for logic 1 level
- JNT1 instruction tests for logic 0 level

When used as an input to the event counter, T1 must be LOW for > 4 CP, followed by a HIGH for > 4 CP. The transition can be recognized with a repetition rate of once per 30 oscillator clock periods (1 machine cycle).

There is no internal pull-up or pull-down resistor connected to the T1 input. If required it must be externally connected to a resistor ($R = \leqslant 100\ k\Omega$).
When T1 is not used pin 8 must be connected to V_{DD} or V_{SS}.

Reset (pin 11)

A positive-going signal on the RESET input/output:

- Sets the program counter to zero
- Selects location 0 of memory bank 0 and register bank 0
- Sets the stack pointer to zero (000); pointing to RAM address 8
- Disables the interrupts (external and timer)
- Stops the timer/event counter, then sets it to zero
- Sets the timer prescaler to modulo-32
- Resets the timer flag
- Sets all ports according to reset states
- Cancels IDLE and STOP mode

After the voltage is applied to RESET an internal delay of 1866 CP is introduced before the microcontroller commences operation.

Power-on reset

The internal power-on reset circuit monitors the supply voltage V_{DD}. As long as V_{DD} remains below the internal reference level V_{ref} (typically 1.3 V), the oscillator is inhibited and RESET (pin 11) has an undefined level. When V_{DD} rises above V_{ref}, the oscillator is released and RESET is pulled HIGH to V_{DD} by TR1 for a period t_D (typically 50 μs). Note that the start-up time of the oscillator is typically 10 ms because of the narrow bandwidth of the crystal.

Three modes of power-on reset are possible:

1. If V_{DD} has a fast rise time, i.e. V_{DD} reaches its minimum value before the RESET signal finishes (t_D), then no additional circuit is required (see Figs 18 and 19). Note that the first instruction is executed after the oscillator start-up time plus 1866 clock periods.
2. If V_{DD} has a slow rise time then the RESET signal should be stretched by an external CR circuit (see Figs 20 and 21). In the event of a short drop in V_{DD}, the diode path discharges the capacitor rapidly to ensure a reliable power-on reset. The RESET signal should reach at least 70% of the final value of V_{DD} to ensure a correct reset. Given that the RESET voltage and V_{DD} rise exponentially, the above requirement is satisfied when the time constant of the RESET pulse is > 8 times the time constant of V_{DD}. If V_{DD} rises linearly then a RESET time constant > 2 times the rise time of V_{DD} is required.

   When a reset is completed (RESET goes LOW) before the oscillator has started up, program execution begins after the oscillator start-up time plus 1866 clock periods (see Fig. 21). If the oscillator starts up prior to the completion of RESET then program execution begins 1866 clock periods after RESET goes LOW.
3. Fig. 22 shows an external reset applied during power-on. The external reset signal must remain HIGH until V_{DD} has reached its minimum operating value corresponding to the selected oscillator frequency. When a reset is completed (RESET goes LOW) before the oscillator has started up, program execution begins after the oscillator start-up time plus 1866 clock periods (see Fig. 23). If the oscillator starts up prior to the completion of RESET then program execution begins 1866 clock periods after RESET goes LOW.

FUNCTIONAL DESCRIPTION (continued)

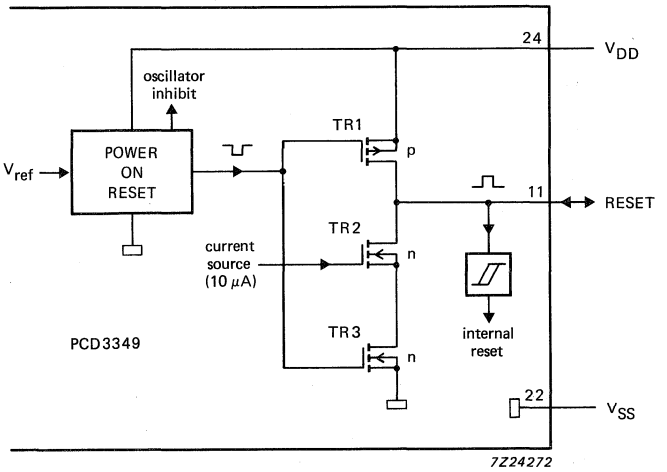


Fig. 18 Power-on reset configuration.

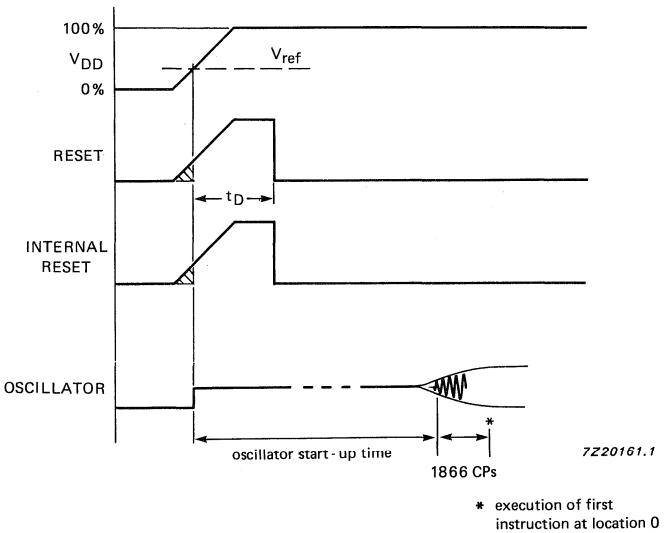


Fig. 19 Timing of power-on reset with fast rise time of V_{DD}.

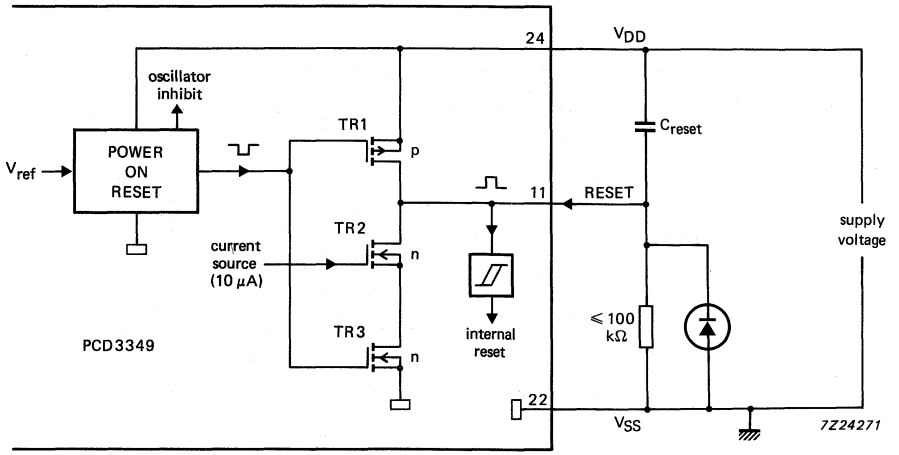


Fig. 20 Stretched power-on reset with external CR circuit.

DEVELOPMENT DATA

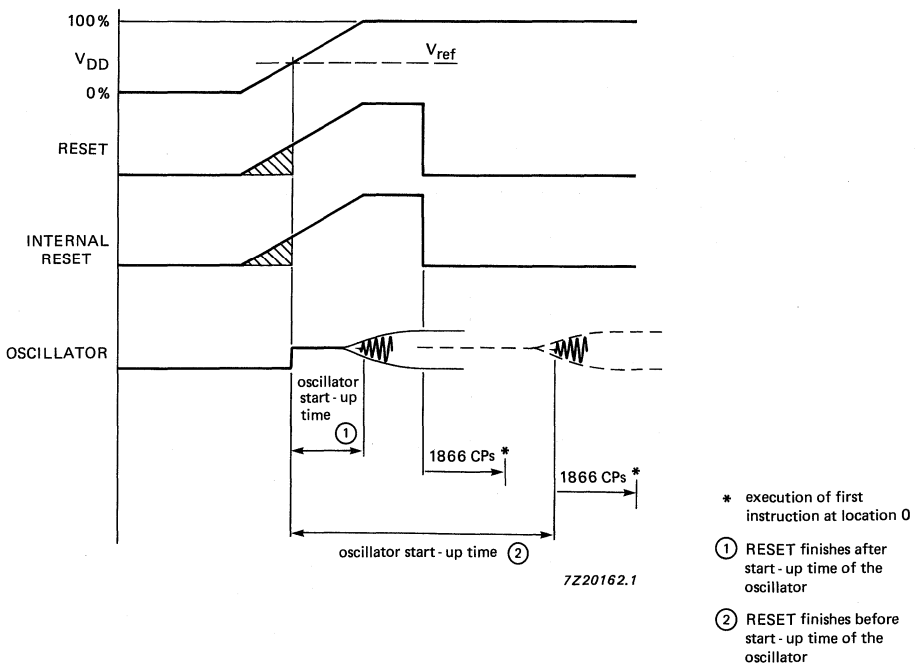


Fig. 21 Timing of power-on reset with a slowly rising V_{DD} and a stretched RESET pulse.

FUNCTIONAL DESCRIPTION (continued)

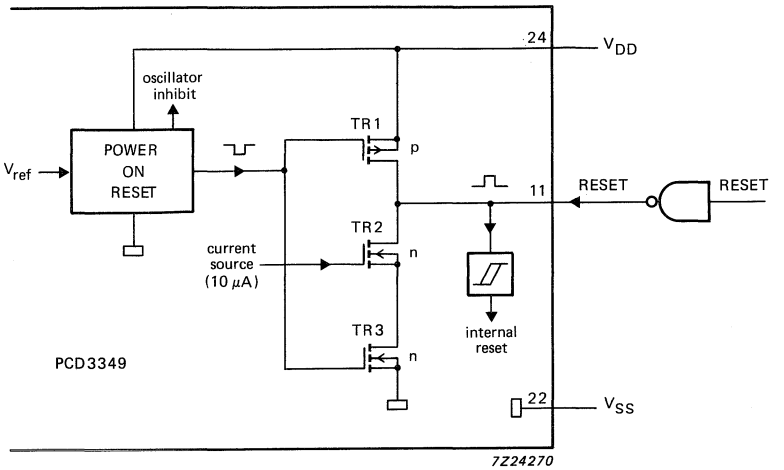


Fig. 22 External power-on reset configuration.

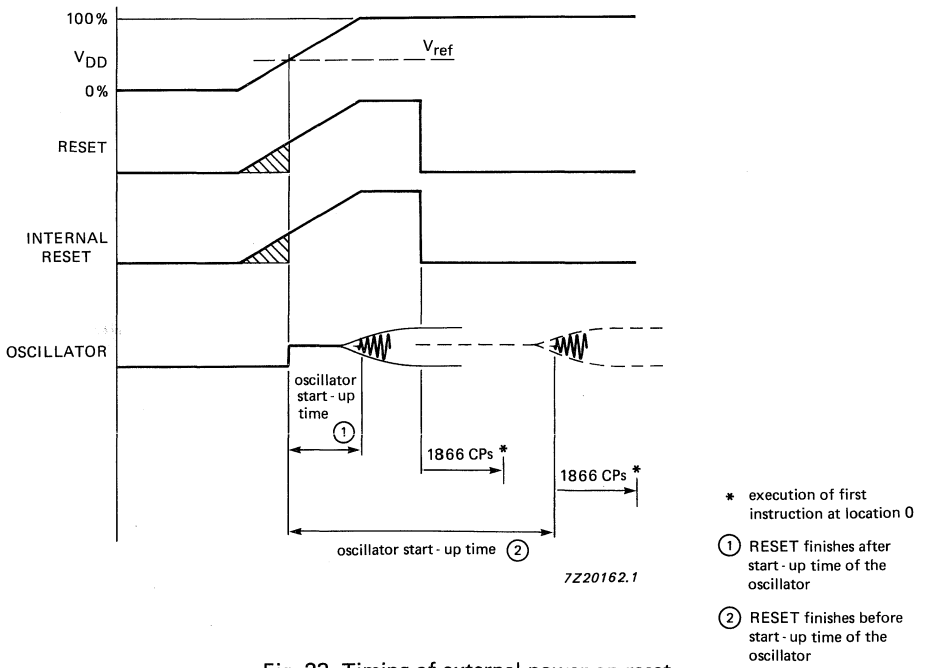


Fig. 23 Timing of external power-on reset.

INSTRUCTION SET

The PCD3349 instruction set consists of over 80 one and two byte instructions and is based on the MAB8048 instruction set. New instructions include those for memory bank selection and derivative control. Program code efficiency is high because all RAM locations and all ROM locations on a 256 byte page require only a single byte address.

Table 5 details the symbols and definition descriptions that are used in the instruction set of the PCD3349. Table 6 shows the instruction map and Table 7 gives the instruction set.

Table 5 Symbols and definitions used in Tables 6 and 7

symbol	definition description
A	accumulator
addr	program memory address
Bb	bit designation (b = 0 to 7)
RBS	register bank select
C	carry bit (bit CY)
CNT	event counter
D	mnemonic for 4-bit digit (nibble)
data	8-bit number or expression
I	interrupt
MB	memory bank
MBFF	memory bank flip-flop
P	mnemonic for 'in page' operation
PC	program counter
Pp	port designation (p = 0, 1 or 2)
PSW	program status word
RB	register bank
Rr	register designation (r = 0 to 7)
SP	stack pointer
T	timer
TF	timer flag
T0, T1	test 0 and 1 inputs
#	immediate data prefix
@	indirect address prefix
(X)	contents of X
((X))	contents of location addressed by X
Dx	derivative register (0 to H'FF')
←	is replaced by
↔	is exchanged with

DEVELOPMENT DATA

INSTRUCTION SET (continued)

Table 6 PCD3349 instruction map

first hexadecimal character of opcode ↓

second hexadecimal character of opcode →

	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
0	NOP	IDLE		ADD A, # data	JMP page 0	EN I	JNTF addr	DEC A	IN A,Pp 0	1	2					
1	INC @Rr 0	1	JB0 addr	ADDC A, # data	CALL page 0	DIS I	JTF addr	INC A	INC Rr 0	1	2	3	4	5	6	7
2	XCH A,@Rr 0	1	STOP	MOV A, # data	JMP page 1	EN TCNTI	JNT0 addr	CLR A	XCH A,Rr 0	1	2	3	4	5	6	7
3	XCHD A,@Rr 0	1	JB1 addr		CALL page 1	DIS TCNTI	JT0 addr	CPL A	OUTL Pp,A 0	1	2					
4	ORL A,@Rr 0	1	MOV A , T	ORL A, # data	JMP page 2	STRT CNT	JNT1 addr	SWAP A	ORL A,Rr 0	1	2	3	4	5	6	7
5	ANL A,@Rr 0	1	JB2 addr	ANL A, # data	CALL page 2	STRT T	JT1 addr	DA A	ANL A,Rr 0	1	2	3	4	5	6	7
6	ADD A,@Rr 0	1	MOV T , A		JMP page 3	STOP TCNT		RRC A	ADD A,Rr 0	1	2	3	4	5	6	7
7	ADDC A,@Rr 0	1	JB3 addr		CALL page 3			RR A	ADDC A,Rr 0	1	2	3	4	5	6	7
8				RET	JMP page 4				ORL Pp, # data 0	1	2			MOV Dx,A		
9			JB4 addr	RETR	CALL page 4		JNZ addr	CLR C	ANL Pp, # data 0	1	2					
A	MOV @Rr, A 0	1		MOVP A,@A	JMP page 5	SEL MB2		CPL C	MOV Rr,A 0	1	2	3	4	5	6	7
B	MOV @Rr, # data 0	1	JB5 addr	JMPP @A	CALL page 5	SEL MB3			MOV Rr, # data 0	1	2	3	4	5	6	7
C	DEC @Rr 0	1			JMP page 6	SEL RB0	JZ addr	MOV A, PSW	DEC Rr 0	1	2	3	4	5	6	7
D	XRL A,@Rr 0	1	JB6 addr	XRL A, # data	CALL page 6	SEL RB1		MOV PSW, A	XRL A,Rr 0	1	2	3	4	5	6	7
E	DJNZ @Rr,addr 0	1			JMP page 7	SEL MB0	JNC addr	RL A	DJNZ Rr, addr 0	1	2	3	4	5	6	7
F	MOV A,@Rr 0	1	JB7 addr		CALL page 7	SEL MB1	JC addr	RLC A	MOV A,Rr 0	1	2	3	4	5	6	7

Table 7 Instruction set

	mnemonic	opcode (hex.)	bytes/ cycles	description	function		notes
ACCUMULATOR	ADD A, Rr	6*	1/1	Add register contents to A	(A)←(A) + (Rr)	r = 0–7	1
	ADD A, @Rr	60 61	1/1	Add RAM data, addressed by Rr, to A	(A)←(A) + ((R0)) (A)←(A) + ((R1))		1
	ADD A, #data	03 data	2/2	Add immediate data to A	(A)←(A) + data		1
	ADDC A, Rr	7*	1/1	Add carry and register contents to A	(A)←(A) + (Rr) + (C)	r = 0–7	1
	ADDC A, @Rr	70 71	1/1	Add carry and RAM data, addressed by Rr, to A	(A)←(A) + ((R0)) + (C) (A)←(A) + ((R1)) + (C)		1
	ADDC A, #data	13 data	2/2	Add carry and immediate data to A	(A)←(A) + data + (C)		1
	ANL A, Rr	5*	1/1	'AND' Rr with A	(A)←(A) AND (Rr)	r = 0–7	
	ANL A, @Rr	50 51	1/1	'AND' RAM data, addressed by Rr, with A	(A)←(A) AND ((R0)) (A)←(A) AND ((R1))		
	ANL A, #data	53 data	2/2	'AND' immediate data with A	(A)←(A) AND data		
	ORL A, Rr	4*	1/1	'OR' Rr with A	(A)←(A) OR (Rr)	r = 0–7	
	ORL A, @Rr	40 41	1/1	'OR' RAM data, addressed by Rr, with A	(A)←(A) OR ((R0)) (A)←(A) OR ((R1))		
	ORL A, #data	43 data	2/2	'OR' immediate data with A	(A)←(A) OR data		
	XRL A, Rr	D*	1/1	'XOR' Rr with A	(A)←(A) XOR (Rr)	r = 0–7	
	XRL A, @Rr	D0 D1	1/1	'XOR' RAM, addressed by Rr, with A	(A)←(A) XOR ((R0)) (A)←(A) XOR ((R1))		
	XRL A, #data	D3 data	2/2	'XOR' immediate data with A	(A)←(A) XOR data		
	INC A	17	1/1	increment A by 1	(A)←(A) + 1		
	DEC A	07	1/1	decrement A by 1	(A)←(A) −1		
	CLR A	27	1/1	clear A to zero	(A)←0		
	CPL A	37	1/1	one's complement A	(A)←NOT(A)		
	RL A	E7	1/1	rotate A left	$(A_{n+1})\leftarrow(A_n)$ $(A_0)\leftarrow(A_7)$	n = 0–6	

INSTRUCTION SET (continued)

Table 7 (continued)

ACCUMULATOR (cont.)	RLC A	F7	1/1	rotate A left through carry	$(A_{n+1})\leftarrow A_n$ $(A_0)\leftarrow(C)$, $(C)\leftarrow(A_7)$	n = 0–6	2
	RR A	77	1/1	rotate A right	$(A_n)\leftarrow(A_{n+1})$ $(A_7)\leftarrow(A_0)$	n = 0–6	
	RRC A	67	1/1	rotate A right through carry	$(A_n)\leftarrow(A_{n+1})$ $(A_7)\leftarrow(C)$, $(C)\leftarrow(A_0)$	n = 0–6	2
	DA A	57	1/1	decimal adjust A			2
	SWAP A	47	1/1	swap nibbles of A	$(A_{4-7})\leftrightarrow(A_{0-3})$		
DATA MOVES	MOV A, Rr	F*	1/1	move register contents to A	(A)←(Rr)	r = 0–7	
	MOV A, @Rr	F0 F1	1/1	move RAM data, addressed by Rr, to A	(A)←((R0)) (A)←((R1))		
	MOV A, #data	23 data	2/2	move immediate data to A	(A)←data		
	MOV Rr, A	A*	1/1	move accumulator contents to register	(Rr)←(A)	r = 0–7	
	MOV @Rr, A	A0 A1	1/1	move accumulator contents to RAM location addressed by Rr	((R0))←(A) ((R1))←(A)		
	MOV Rr, #data	B* data	2/2	move immediate data to Rr	(Rr)←data	r = 0–7	
	MOV @Rr, #data	B0 data B1 data	2/2	move immediate data to RAM location addressed by Rr	((R0))←data ((R1))←data		
	XCH A, Rr	2*	1/1	exchange accumulator contents with Rr	(A)↔(Rr)	r = 0–7	
	XCH A, @Rr	20 21	1/1	exchange accumulator contents with RAM data addressed by Rr	(A)↔((R0)) (A)↔((R1))		
	XCHD A, @Rr	30 31	1/1	exchange lower nibbles of A and RAM data addressed by Rr	$(A_{0-3})\leftrightarrow((R0_{0-3}))$ $(A_{0-3})\leftrightarrow((R1_{0-3}))$		
	MOV A, PSW	C7	1/1	move PSW contents to accumulator	(A)←(PSW)		
	MOV PSW, A	D7	1/1	move accumulator bit 3 to PSW_3	$(PSW_3)\leftarrow(A_3)$		3
	MOVP A, @A	A3	1/2	move indirectly addressed data in current page to A	$(PC_{0-7})\leftarrow(A)$, $(A)\leftarrow((PC))$		
FLAGS	CLR C	97	1/1	clear carry bit	(C)←0		2
	CPL C	A7	1/1	complement carry bit	(C)←NOT(C)		2

DEVELOPMENT DATA

	mnemonic	opcode (hex.)	bytes/ cycles	description	function	notes
REGISTER	INC Rr	1*	1/1	increment register by 1	(Rr)←(Rr) + 1 r = 0–7	
	INC @Rr	10 11	1/1	increment RAM data, addressed by Rr, by 1	((R0))←((R0)) + 1 ((R1))←((R1)) + 1	
	DEC Rr	C*	1/1	decrement register by 1	(Rr)←(Rr) – 1 r = 0–7	
	DEC @Rr	C0 C1	1/1	decrement RAM data, addressed by Rr, by 1	((R0))←((R0)) – 1 ((R1))←((R1)) – 1	
BRANCH	JMP addr	● 4 address	2/2	unconditional jump within a 2 K bank	(PC_{8-10})←$addr_{8-10}$ (PC_{0-7})←$addr_{0-7}$ (PC_{11-12})←MBFF 0–1	
	JMPP @A	B3	1/2	indirect jump within a page	(PC_{0-7})←((A))	
	DJNZ Rr, addr	E* address	2/2	decrement Rr by 1 and jump if not zero to addr	(Rr)←(Rr) – 1 r = 0–7 if (Rr) not zero (PC_{0-7})←addr	
	DJNZ @Rr, addr	E0 address	2/2	decrement RAM data, addressed by Rr by 1 and jump if not zero to addr	((R0))←((R0)) – 1 if ((R0)) not zero (PC_{0-7})←addr	
		E1 address			((R1))←((R1)) – 1 if ((R1)) not zero (PC_{0-7})←addr	
	JBb addr	▲ 2 address	2/2	jump to addr if Acc. bit b = 1	if b = 1 : (PC_{0-7})←addr b = 0–7	
	JC addr	F6 address	2/2	jump to addr if C = 1	if C = 1 : (PC_{0-7})←addr	
	JNC addr	E6 address	2/2	jump to addr if C = 0	if C = 0 : (PC_{0-7})←addr	
	JZ addr	C6 address	2/2	jump to addr if A = 0	if A = 0 : (PC_{0-7})←addr	
	JNZ addr	96 address	2/2	jump to addr if A is NOT zero	if A≠0 : (PC_{0-7})←addr	
	JT0 addr	36 address	2/2	jump to addr if T0 = 1	if T0 = 1: (PC_{0-7})←addr	
	JNT0 addr	26 address	2/2	jump to addr if T0 = 0	if T0 = 0: (PC_{0-7})←addr	
	JT1 addr	56 address	2/2	jump to addr if T1 = 1	if T1 = 1: (PC_{0-7})←addr	
	JNT1 addr	46 address	2/2	jump to addr if T1 = 0	if T1 = 0: (PC_{0-7})←addr	
	JTF addr	16 address	2/2	jump to addr if Timer Flag = 1	if TF = 1: (PC_{0-7})←addr	4
	JNTF addr	06 address	2/2	jump to addr if Timer Flag = 0	if TF = 0: (PC_{0-7})←addr	

INSTRUCTION SET (continued)

Table 7 (continued)

TIMER/EVENT COUNTER	MOV A, T	42	1/1	move timer/event counter contents to accumulator	(A)←(T)	
	MOV T, A	62	1/1	move accumulator contents to timer/event counter	(T)←(A)	
	STRT CNT	45	1/1	start event counter		
	STRT T	55	1/1	start timer		
	STOP TCNT	65	1/1	stop timer/event counter		
	EN TCNTI	25	1/1	enable timer/event counter interrupt		
	DIS TCNTI	35	1/1	disable timer/event counter interrupt		
CONTROL	EN I	05	1/1	enable external interrupt		
	DIS I	15	1/1	disable external interrupt		
	SEL RB0	C5	1/1	select register bank 0	(RBS)←0	5
	SEL RB1	D5	1/1	select register bank 1	(RBS)←1	5
	SEL MB0	E5	1/1	select program memory bank 0	(MBFF0)←0, (MBFF1)←0	
	SEL MB1	F5	1/1	select program memory bank 1	(MBFF0)←1, (MBFF1)←0	
	SEL MB2	A5	1/1	select program memory bank 2	(MBFF0)←0, (MBFF1)←1	
	SEL MB3	B5	1/1	select program memory bank 3	(MBFF0)←1, (MBFF1)←1	
	STOP	22	1/1	enter STOP mode		
	IDLE	01	1/1	enter IDLE mode		
SUBROUTINE	CALL addr	▲ 4 address	2/2	jump to subroutine	((SP))←(PC), ($PSW_{4, 6, 7}$) (SP)←(SP) + 1 (PC_{8-10})←$addr_{8-10}$ (PC_{0-7})←$addr_{0-7}$ (PC_{11-12})←MBFF 0–1	6
	RET	83	1/2	return from subroutine	(SP)←(SP) −1 (PC)←((SP))	6
	RETR	93	1/2	return from interrupt and restore bits 4, 6, 7 of PSW	(SP)←(SP) −1 ($PSW_{4, 6, 7}$) + (PC)←((SP))	6

DEVELOPMENT DATA

	mnemonic	opcode (hex.)	bytes/ cycles	description	function	notes
PARALLEL INPUT/OUTPUT	IN A, Pp	08 09 0A	1/2	input port p data to accumulator	(A)←(P0) (A)←(P1) (A)←(P2)	7
	OUTL Pp, A	38 39 3A	1/2	output accumulator data to port p	(P0)←(A) (P1)←(A) (P2)←(A)	
	ANL Pp, #data	98 data 99 data 9A data	2/2	AND port p data with immediate data	(P0)←(P0) AND data (P1)←(P1) AND data (P2)←(P2) AND data	
	ORL Pp, #data	88 data 89 data 8A data	2/2	OR port p data with immediate data	(P0)←(P0) OR data (P1)←(P1) OR data (P2)←(P2) OR data	
DERIVATIVE INPUT/OUTPUT	MOV Dx, A	8D	2/2	move accumulator contents to derivative register	(Dx)←(A) x = 0 to 255	8
	NOP	00	1/1	no operation		

Notes to Table 7

1. PSW CY, AC affected
2. PSW CY affected
3. PSW PS affected
4. Execution of JTF and JNTF instructions resets the Timer Flag (TF).
5. PSW RBS affected
6. PSW SP_0, SP_1, SP_2 affected
7. (A) = 0000 P2.3, P2.2, P2.1, P2.0
8. Instructions for PCF84C00 only.

* : 8, 9, A, B, C, D, E, F
● : 0, 2, 4, 6, 8, A, C, E
▲ : 1, 3, 5, 7, 9, B, D, F

RATINGS

Limiting values in accordance with the Absolute Maximum System (IEC 134)

parameter	conditions	symbol	min.	max.	unit
Supply voltage (pin 24)		V_{DD}	−0.8	+ 8	V
Input voltage (any pin)		V_I	−0.8	V_{DD} + 0.8	V
DC current (any input or output)		$\pm I_I, \pm I_O$	–	10	mA
Total power dissipation	note 1	P_{tot}	–	500	mW
Power dissipation (per output)		P_O	–	50	mW
Storage temperature range		T_{stg}	−65	+ 150	°C
Operating ambient temperature range		T_{amb}	−25	+ 70	°C
Operating junction temperature		T_j	–	+ 125	°C

Note to the ratings

1. Derate according to thermal resistance.

THERMAL RESISTANCE

From junction to ambient

SOT117	$R_{th\ j\text{-}a}$	=	120	K/W
SOT136A	$R_{th\ j\text{-}a}$	=	150	K/W

DEVELOPMENT DATA

DC CHARACTERISTICS

V_{DD} = 2.5 to 6 V; V_{SS} = 0 V; T_{amb} = −25 to + 70 °C; all voltages with respect to V_{SS}; f = 3.58 MHz with R_S = 100 Ω; unless otherwise specified

parameter	conditions	symbol	min.	typ.	max.	unit
Supplies						
Supply voltage						
operating		V_{DD}	2.5	–	6	V
STOP mode for RAM data retention		V_{DD}	1.0	–	6	V
Supply current (Fig. 25)						
operating with tone generator on	V_{DD} = 3 V	I_{DD}	–	1.2	–	mA
operating without tone generator	V_{DD} = 3 V	I_{DD}	–	600	–	μA
IDLE mode (Fig. 26)						
with tone generator on	V_{DD} = 3 V	I_{DD}	–	900	–	μA
without tone generator	V_{DD} = 3 V	I_{DD}	–	300	–	μA
STOP mode (Fig. 27)	note 1; V_{DD} = 1.8 V					
	T_{amb} = 25 °C	I_{DD}	–	1.2	2.5	μA
	T_{amb} = 55 °C	I_{DD}	–	–	5	μA
	T_{amb} = 70 °C	I_{DD}	–	–	10	μA
RESET I/O						
Switching level		V_{RESET}	–	1.3	–	V
Sink current	$V_{DD} > V_{RESET}$	I_{OL}	–	7	–	μA
Inputs						
Input voltage LOW		V_{IL}	0	–	0.3 V_{DD}	V
Input voltage HIGH		V_{IH}	0.7 V_{DD}	–	V_{DD}	V
Input leakage current	$V_{SS} < V_I < V_{DD}$	$\pm I_{IL}$	–	–	1	μA
Outputs						
Output voltage LOW	V_I = V_{SS} or V_{DD}; $\lvert I_O \rvert < 1$ μA	V_{OL}	–	–	0.05	V
Output sink current LOW						
P1.2, P1.3, P1.4, P2.0, P2.1, P2.2, P2.3 (Fig. 28)	V_{DD} = 3 V; V_O = 0.4 V	I_{OL}	0.7	1.5	–	mA
for remaining ports (Fig. 29)	V_O = 0.4 V	I_{OL}	1.5	–	–	mA
Pull-up output source current HIGH (Fig. 30)	V_{DD} = 3 V;					
	V_O = 0.9 V_{DD}	$-I_{OH}$	10	–	–	μA
	V_O = V_{SS}	$-I_{OH}$	–	–	300	μA
Push-pull output source current HIGH	V_{DD} = 3 V;					
	V_O = V_{DD} −0.4 V	$-I_{OH}$	0.6	1.5	–	mA

TONE GENERATOR CHARACTERISTICS

V_{DD} = 2.5 to 6 V; V_{SS} = 0 V; T_{amb} = −25 to + 70 °C; all voltages with respect to V_{SS}; f = 3.58 MHz with R_S = 100 Ω; unless otherwise specified

parameter	conditions	symbol	min.	typ.	max.	unit
Tone output (Fig. 24)						
DTMF output voltage levels (RMS values)						
HIGH group		$V_{HG(rms)}$	158	192	205	mV
LOW group		$V_{LG(rms)}$	125	150	160	mV
Frequency deviation		$\Delta f/f$	−0.6	–	+ 0.6	%
DC voltage level		V_{dc}	–	½ V_{DD}	–	V
Output impedance		$\|Z_O\|$	–	0.1	0.5	kΩ
Pre-emphasis of group		ΔV_G	1.85	2.1	2.35	dB
Total harmonic distortion	note 2; T_{amb} = 25 °C	THD	–	−25	–	dB

Notes to the characteristics

1. Crystal connected between XTAL1 and XTAL2; CE and T1 at V_{SS}.
2. Related to the level of the LOW group frequency component (CEPT CS 203).

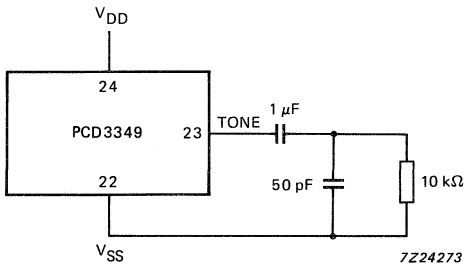


Fig. 24 Tone output test circuit.

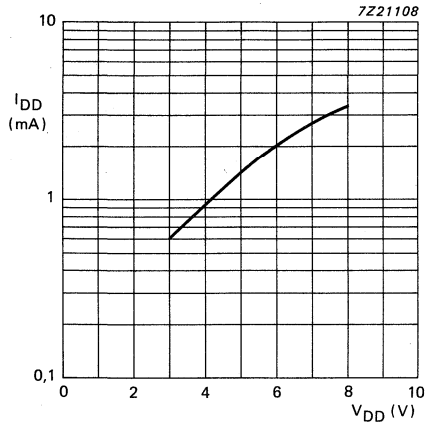


Fig. 25 Typical supply current (I_{DD}) in operating mode as a function of the supply voltage (V_{DD}); T_{amb} = 25 °C; clock frequency = 3.58 MHz; I_{DD} is increased by approximately 0.6 mA when the DTMF function is operating.

DEVELOPMENT DATA

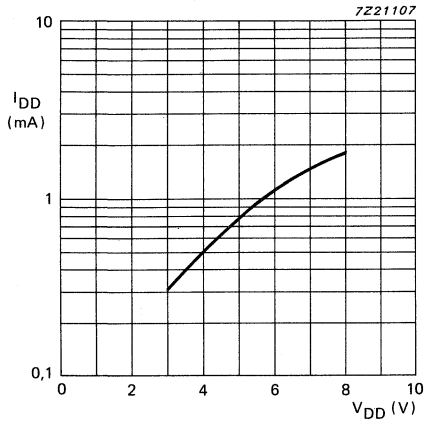
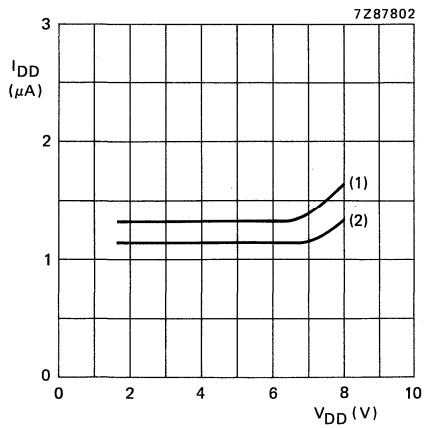
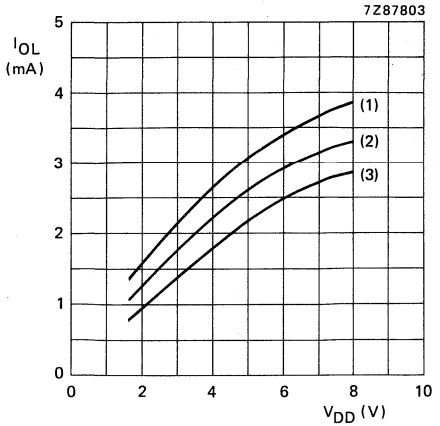


Fig. 26 Typical supply current (I_{DD}) in IDLE mode as a function of the supply voltage (V_{DD}); T_{amb} = 25 °C; clock frequency = 3.58 MHz.



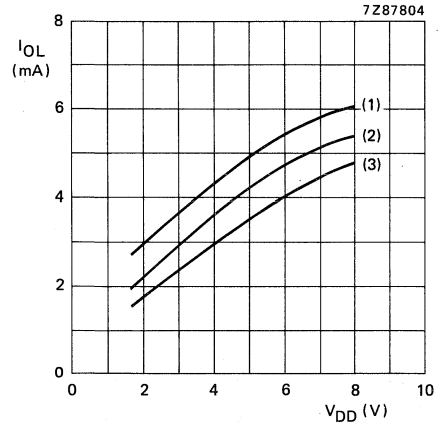
(1) T_{amb} = 25 °C
(2) T_{amb} = 70 °C

Fig. 27 Typical supply current (I_{DD}) in STOP mode as a function of the supply voltage (V_{DD}).



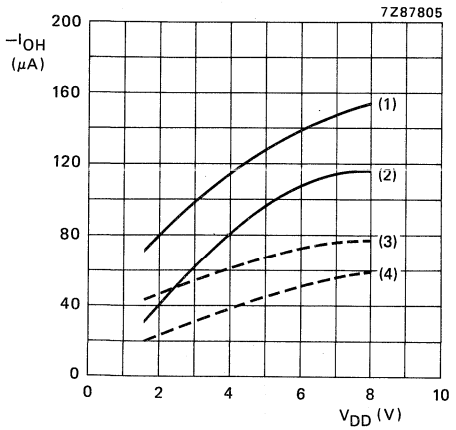
(1) $T_{amb} = -25\ ^{o}C$
(2) $T_{amb} = +25\ ^{o}C$
(3) $T_{amb} = +70\ ^{o}C$

Fig. 28 Output sink current LOW (I_{OL}), for P1.2, P1.3, P1.4, P2.0, P2.1, P2.2, P2.3 as a function of supply voltage (V_{DD}); $V_O = 0.4$ V.



(1) $T_{amb} = -25\ ^{o}C$
(2) $T_{amb} = +25\ ^{o}C$
(3) $T_{amb} = +70\ ^{o}C$

Fig. 29 Output sink current LOW (I_{OL}), for remaining ports as a function of supply voltage (V_{DD}); $V_O = 0.4$ V.



(1) $T_{amb} = 25\ ^{o}C$; $V_O = V_{SS}$
(2) $T_{amb} = 25\ ^{o}C$; $V_O = 0.9\ V_{DD}$
(3) $T_{amb} = 70\ ^{o}C$; $V_O = V_{SS}$
(4) $T_{amb} = 70\ ^{o}C$; $V_O = 0.9\ V_{DD}$

Fig. 30 Output source current HIGH ($-I_{OH}$) as a function of supply voltage (V_{DD}).

DEVELOPMENT DATA

APPLICATION INFORMATION

A block diagram of an electronic featurephone built around the PCD3349 is shown in Figure 31. It comprises the following dedicated telephony ICs:

- TEA1060/1061/1067/1068 transmission circuit for telephony
- PCF8576 or PCF8577 2 LCD drivers in LCD module MB7020160
- PCF8571 1 K RAMs with Serial I/O; the number of RAMs depends on the required amount of stored telephone numbers
- PCD3360 programmable multi-tone ringer

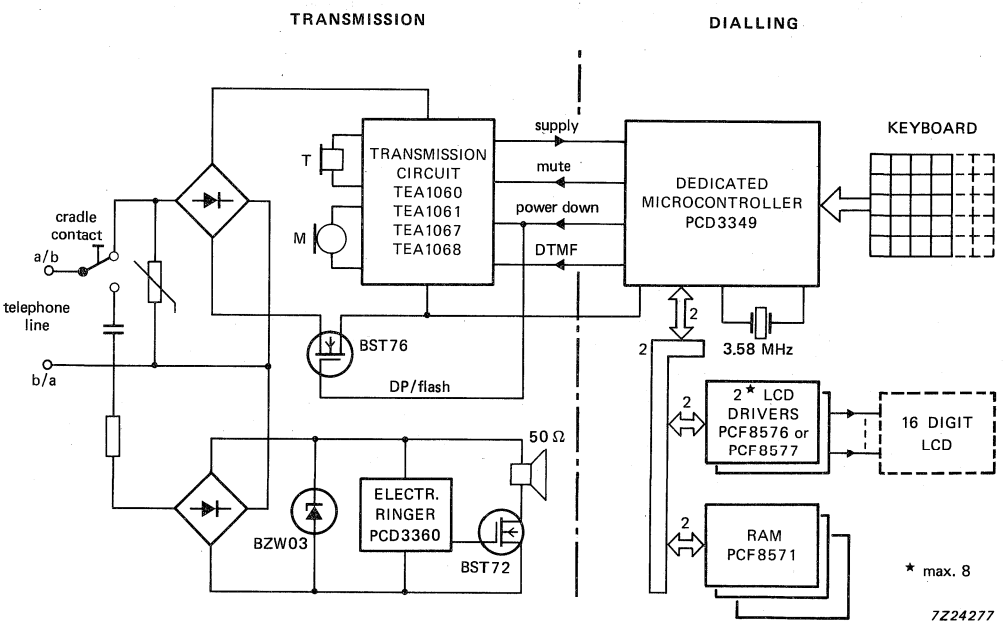


Fig. 31 Block diagram of electronic featurephone with common line interface.

Section 6 - The 8048 Based Instruction Set

THE 8048 BASED INSTRUCTION SET

Introduction

This chapter describes the instruction set of the 8048 based microcontrollers. The following terms are used throughout the chapter:

8048: refers to the MAB8048, MAB8049, MAB8050, and PCF80C49 microcontrollers.
84XX: refers to the MAB84XX family of microcontrollers.
84CXXX: refers to the PCF84CXXX family of microcontrollers.
33XX: refers to the PCD33XX family of microcontrollers.

The chapter is split into the following sections:

- Instructions common to all 8048 based microcontrollers; this section describes instructions that are common to all microcontrollers based on the 8048 instruction set. For full details on the instruction set of a particular microcontroller family, this section should be read in conjunction with the section that details the additional instructions which are specific to that microcontroller family.

- Additional 8048 instructions; this section describes the 8048 instructions which are not included in the common section; it should be used in conjunction with the common section.

- Additional 84XX instructions; this section describes the 84XX instructions which are not included in the common section; it should be used in conjunction with the common section.

- Additional 84CXXX instructions; this section describes the 84CXXX instructions which are not included in the common section; it should be used in conjunction with the common section.

- Additional 33XX instructions; this section describes the 33XX instructions which are not included in the common section; it should be used in conjunction with the common section.

Within the above sections, the instruction sets are described in alphabetical order. Some of the sections contain details on devices whose instruction set deviates from the family to which they belong. Any other differences are described in the data sheets.

Each instruction is introduced by its mnemonic followed by a descriptive title. The hexadecimal opcode is presented together with a byte pattern or binary equivalent. A description of the logical operation performed follows, and simple examples are given for most of the instructions.

Table 1 details the symbols and abbreviations used throughout the chapter. Table 2 gives a summary of **all** of the instructions. Note that a particular device will use only a subset of the instructions shown in Table 2.

Table 1 Symbols and abbreviations.

SYMBOL	DESCRIPTION
A	accumulator
AC	auxiliary carry
addr	program memory address
Bb	bit designation (b = 0 to 7)
BUS	BUS port
C	carry (bit CY)
CNT	event counter
D	mnemonic for a 4-bit digit (nibble)
Dx	mnemonic for a derivative register (x = 0 to 255)
direct	8-bit derivative register address
data	8-bit immediate data
F0, F1	Flag 0, Flag 1
H	hexadecimal data
I	interrupt
MBn	memory bank (n = 0 to 3)
MBFFn	memory bank flip-flop (n = 0 or 1)
P	mnemonic for 'in-page' operation
PC	program counter
Pp	port designation (p = 0, 1, or 2)
PS	timer prescaler select
PSW	program status word
RB	register bank
RBS	register bank select flag
@Rr	8-bit address register (r = 0, 1)
Rr	8-bit register (r = 0 to 7)
Sn	serial I/O register (n = 0, 1, or 2)
SP	stack pointer
T	timer
TCNT	timer/event counter
TF	timer flag
T0, T1	test 0 and test 1 inputs
#	immediate data prefix
@	indirect address prefix
(X)	contents of X
((X))	contents of location addressed by X
←	is replaced by
↔	is exchanged with
[x]	x represents a hex digit

Table 2 Instruction set summary.

MNEMONIC	OPCODE (HEX)	BYTES/ CYCLES	DESCRIPTION	FUNCTION	NOTES	
ACCUMULATOR						
ADD A,Rr	6[8+r]	1/1	Add register contents to A	(A) ← (A)+(Rr)	r = 0 to 7	1
ADD A,@Rr	6r	1/1	Add RAM data to A	(A) ← (A)+((Rr))	r = 0, 1	1
ADD A,#data	03 data	2/2	Add immediate data to A	(A) ← (A)+data		1
ADDC A,Rr	7[8+r]	1/1	Add carry and register contents to A	(A) ← (A)+(Rr)+(C)	r = 0 to 7	1
ADDC A,@Rr	7r	1/1	Add carry and RAM data to A	(A) ← (A)+((Rr))+(C)	r = 0, 1	1
ADDC A,#data	13 data	2/2	Add carry and immediate data to A	(A) ← (A)+data+(C)		1
ANL A,Rr	5[8+r]	1/1	'AND' Rr with A	(A) ← (A) AND (Rr)	r = 0 to 7	
ANL A,@Rr	5r	1/1	'AND' RAM data with A	(A) ← (A) AND ((Rr))	r = 0, 1	
ANL A,#data	53 data	2/2	'AND' immediate data with A	(A) ← (A) AND data		
ORL A,Rr	4[8+r]	1/1	'OR' Rr with A	(A) ← (A) OR (Rr)	r = 0 to 7	
ORL A,@Rr	4r	1/1	'OR' RAM data with A	(A) ← (A) OR ((Rr))	r = 0, 1	
ORL A,#data	43 data	2/2	'OR' immediate data with A	(A) ← (A) OR data		
XRL A,Rr	D[8+r]	1/1	'XOR' Rr with A	(A) ← (A) XOR (Rr)	r = 0 to 7	
XRL A,@Rr	Dr	1/1	'XOR' RAM data with A	(A) ← (A) XOR ((Rr))	r = 0, 1	
XRL A,#data	D3 data	2/2	'XOR' immediate data with A	(A) ← (A) XOR data		
INC A	17	1/1	Increment A by 1	(A) ← (A)+1		
DEC A	07	1/1	Decrement A by 1	(A) ← (A)–1		
CLR A	27	1/1	Clear A to zero	(A) ← 0		
CPL A	37	1/1	One's complement A	(A) ← NOT(A)		
RL A	E7	1/1	Rotate A left	$(A_{n+1}) \leftarrow (A_n)$ $(A_0) \leftarrow (A_7)$	n = 0 to 6	
RLC A	F7	1/1	Rotate A left through carry	$(A_{n+1}) \leftarrow A_n$ $(A_0) \leftarrow (C)$ $(C) \leftarrow (A_7)$	n = 0 to 6	2
RR A	77	1/1	Rotate A right	$(A_n) \leftarrow (A_{n+1})$ $(A_7) \leftarrow (A_0)$	n = 0 to 6	
RRC A	67	1/1	Rotate A right through carry	$(A_n) \leftarrow (A_{n+1})$ $(A_7) \leftarrow (C)$ $(C) \leftarrow (A_0)$	n = 0 to 6	2
DA A	57	1/1	Decimal adjust A	If AC = 1 or $(A_{0-3}) > 9$, then (A) ← (A)+06H If C = 1 or $(A_{4-7}) > 9$, then (A) ← (A)+60H		2
SWAP A	47	1/1	Swap nibbles of A	$(A_{4-7}) \leftrightarrow (A_{0-3})$		2

MNEMONIC	OPCODE (HEX)	BYTES/ CYCLES	DESCRIPTION	FUNCTION	NOTES	
DATA MOVES						
MOV A,Rr	F[8+r]	1/1	Move register contents to A	(A) ← (Rr)	r = 0 to 7	
MOV A,@Rr	Fr	1/1	Move RAM data to A	(A) ← ((Rr))	r = 0, 1	
MOV A,#data	23 data	2/2	Move immediate data to A	(A) ← data		
MOV Rr,A	A[8+r]	1/1	Move A contents to register	(Rr) ← (A)	r = 0 to 7	
MOV @Rr,A	Ar	1/1	Move A contents to RAM	((Rr)) ← (A)	r = 0, 1	
MOV Rr,#data	B[8+r] data	2/2	Move immediate data to Rr	(Rr) ← data	r = 0 to 7	
MOV @Rr,#data	Br data	2/2	Move immediate data to RAM	((Rr)) ← data	r = 0, 1	
XCH A,Rr	2[8+r]	1/1	Exchange A contents with Rr	(A) ↔ (Rr)	r = 0 to 7	
XCH A,@Rr	2r	1/1	Exchange A contents with RAM data	(A) ↔ ((Rr))	r = 0, 1	
XCHD A,@Rr	3r	1/1	Exchange lower nibbles of A and RAM data	$(A_{0-3}) \leftrightarrow ((Rr_{0-3}))$	r = 0, 1	
MOV A,PSW	C7	1/1	Move PSW contents to A	(A) ← (PSW)		
MOV PSW,A	D7	1/1	Move A contents to PSW	(PSW) ← (A)		a, 10
MOV PSW,A	D7	1/1	Move A bit 3 content to PSW bit 3	$(PS) \leftarrow (A_3)$		b, c, d, 3
MOVP A,@A	A3	1/2	Move indirectly addressed data in current page to A	$(PC_{0-7}) \leftarrow (A)$, (A) ← ((PC))		
MOVX A,@Rr	8r	1/2	Move external RAM data to A	(A) ← ((Rr))	r = 0, 1	a
MOVX @Rr,A	9r	1/2	Move A contents to external RAM location	((Rr)) ← (A)	r = 0, 1	a
MOVP3 A,@A	E3	1/2	Move page 3 data to A	$(PC_{0-7}) \leftarrow (A)$ $(PC_{8-11}) \leftarrow 1100$ (A) ← ((PC))		a
FLAGS						
CLR C	97	1/1	Clear carry bit	(C) ← 0		2
CPL C	A7	1/1	Complement carry bit	(C) ← NOT(C)		2
CLR F0	85	1/1	Clear flag 0	(F0) ← 0		a
CPL F0	95	1/1	Complement flag 0	(F0) ← NOT(F0)		a
CLR F1	A5	1/1	Clear flag 1	(F1) ← 0		a
CPL F1	B5	1/1	Complement flag 1	(F1) ← NOT(F1)		a
REGISTER						
INC Rr	1[8+r]	1/1	Increment register by 1	(Rr) ← (Rr)+1	r = 0 to 7	
INC @Rr	1r	1/1	Increment RAM data by 1	((Rr)) ← ((Rr))+1	r = 0, 1	
DEC Rr	C[8+r]	1/1	Decrement register by 1	(Rr) ← (Rr)–1	r = 0 to 7	
DEC @Rr	Cr	1/1	Decrement RAM data by 1	((Rr)) ← ((Rr))–1	r = 0, 1	b, c, d

MNEMONIC	OPCODE (HEX)	BYTES/ CYCLES	DESCRIPTION	FUNCTION	NOTES
BRANCH					
JMP addr	[2n]4 addr	2/2	Unconditional jump	$(PC_{8-10}) \leftarrow n$ $(PC_{0-7}) \leftarrow$ addr $(PC_{11}) \leftarrow$ (MBFF)	n = 0 to 7 a
JMP addr	[2n]4 addr	2/2	Unconditional jump	$(PC_{8-10}) \leftarrow n$ $(PC_{0-7}) \leftarrow$ addr $(PC_{11-12}) \leftarrow$ (MBFF0-1)	n = 0 to 7 b, c, d
JMPP @A	B3	1/2	Indirect jump within a page	$(PC_{0-7}) \leftarrow ((A))$	
DJNZ Rr,addr	E[8+r] addr	2/2	Decrement Rr by 1 and jump if not zero to addr	(Rr) ← (Rr)–1; if (Rr) not zero, then $(PC_{0-7}) \leftarrow$ addr	r = 0 to 7
DJNZ @Rr,addr	Er	2/2	Decrement RAM data by 1 and jump if not zero to addr	((Rr)) ← ((Rr))–1; if ((Rr)) not zero, then $(PC_{0-7}) \leftarrow$ addr	r = 0, 1 b, c, d
JBb addr	[2b+1]2 addr	2/2	Jump to addr if A bit b = 1	If $(A_b) = 1$, then $(PC_{0-7}) \leftarrow$ addr	b = 0 to 7
JC addr	F6 addr	2/2	Jump to addr if C = 1	If (C) = 1, then $(PC_{0-7}) \leftarrow$ addr	
JNC addr	E6 addr	2/2	Jump to addr if C = 0	If (C) = 0, then $(PC_{0-7}) \leftarrow$ addr	
JZ addr	C6 addr	2/2	Jump to addr if A = 0	If (A) = 0, then $(PC_{0-7}) \leftarrow$ addr	
JNZ addr	96 addr	2/2	Jump to addr if A is NOT zero	If (A) ≠ 0, then $(PC_{0-7}) \leftarrow$ addr	
JT0 addr	36 addr	2/2	Jump to addr if T0 = 1	If T0 = 1, then $(PC_{0-7}) \leftarrow$ addr	a, b, c
JT0 addr	36 addr	2/2	Jump to addr if T0 = 0	If T0 = 0, then $(PC_{0-7}) \leftarrow$ addr	d
JNT0 addr	26 addr	2/2	Jump to addr if T0 = 0	If T0 = 0, then $(PC_{0-7}) \leftarrow$ addr	a, b, c
JNT0 addr	26 addr	2/2	Jump to addr if T0 = 1	If T0 = 1, then $(PC_{0-7}) \leftarrow$ addr	d
JT1 addr	56 addr	2/2	Jump to addr if T1 = 1	If T1 = 1, then $(PC_{0-7}) \leftarrow$ addr	
JNT1 addr	46 addr	2/2	Jump to addr if T1 = 0	If T1 = 0, then $(PC_{0-7}) \leftarrow$ addr	
JTF addr	16 addr	2/2	Jump to addr if Timer Flag = 1	If TF = 1, then $(PC_{0-7}) \leftarrow$ addr	4
JNTF addr	06 addr	2/2	Jump to addr if Timer Flag = 0	If TF = 0, then $(PC_{0-7}) \leftarrow$ addr	b, c, d, 4
JF0 addr	B6 addr	2/2	Jump to addr if flag 0 = 1	If F0 = 1, then $(PC_{0-7}) \leftarrow$ addr	a
JF1 addr	76 addr	2/2	Jump to addr if flag 1 = 1	If F1 = 1, then $(PC_{0-7}) \leftarrow$ addr	a
JNI addr	86 addr	2/2	Jump to addr if interrupt input is LOW	If I = 0, then $(PC_{0-7}) \leftarrow$ addr	a

MNEMONIC	OPCODE (HEX)	BYTES/ CYCLES	DESCRIPTION	FUNCTION	NOTES
TIMER/EVENT COUNTER					
MOV A,T	42	1/1	Move timer/event counter contents to A	(A) ← (T)	
MOV T,A	62	1/1	Move A contents to timer/ event counter	(T) ← (A)	
STRT CNT	45	1/1	Start event counter		
STRT T	55	1/1	Start timer		
STOP TCNT	65	1/1	Stop timer/event counter		
EN TCNTI	25	1/1	Enable timer/event counter interrupt		
DIS TCNTI	35	1/1	Disable timer/event counter interrupt		
CONTROL					
ENT0 CLK	75	1/1	Enable clock output		a
EN I	05	1/1	Enable external interrupt		
DIS I	15	1/1	Disable external interrupt		
SEL RB0	C5	1/1	Select register bank 0	(RBS) ← 0	5
SEL RB1	D5	1/1	Select register bank 1	(RBS) ← 1	5
SEL MB0	E5	1/1	Select program memory bank 0	(MBFF) ← 0	a, 9
SEL MB0	E5	1/1	Select program memory bank 0	(MBFF0) ← 0, (MBFF1) ← 0	b, c, d, 9
SEL MB1	F5	1/1	Select program memory bank 1	(MBFF) ← 1	a, 9
SEL MB1	F5	1/1	Select program memory bank 1	(MBFF0) ← 1, (MBFF1) ← 0	b, c, d, 9
SEL MB2	A5	1/1	Select program memory bank 2	(MBFF0) ← 0, (MBFF1) ← 1	b, c, d, 9
SEL MB3	B5	1/1	Select program memory bank 3	(MBFF0) ← 1, (MBFF1) ← 1	b, c, d, 9
STOP	22	1/1	Enter Stop mode		c, d
IDLE	01	1/1	Enter Idle mode		c, d
IDLE	01	1/2	Enter Idle mode		e
SUBROUTINE					
CALL addr	[2n+1]4 addr	2/2	Jump to subroutine	((SP)) ← (PC), (PSW_{4-7}) (SP) ← (SP)+1 (PC_{8-10}) ← n (PC_{0-7}) ← addr (PC_{11}) ← (MBFF)	n = 0 to 7 a, 6
CALL addr	[2n+1]4 addr	2/2	Jump to subroutine	((SP)) ← (PC), ($PSW_{4,6,7}$) (SP) ← (SP)+1 (PC_{8-10}) ← n (PC_{0-7}) ← addr (PC_{11-12}) ← (MBFF0-1)	n = 0 to 7 b, c, d, 6
RET	83	1/2	Return from subroutine	(SP) ← (SP)–1 (PC) ← ((SP))	6
RETR	93	1/2	Return from interrupt and restore bits 4-7 of PSW	(SP) ← (SP)–1 (PSW_{4-7})+(PC) ← ((SP))	a, 6
RETR	93	1/2	Return from interrupt and restore bits 4, 6, 7 of PSW	(SP) ← (SP)–1 ($PSW_{4,6,7}$)+(PC) ← ((SP))	b, c, d, 6

MNEMONIC	OPCODE (HEX)	BYTES/ CYCLES	DESCRIPTION	FUNCTION	NOTES
INPUT/OUTPUT					
IN A,P0	08	1/2	Input port 0 data to A	(A) ← (P0)	b,c,d
IN A,P1	09	1/2	Input port 1 data to A	(A) ← (P1)	
IN A,P2	0A	1/2	Input port 2 data to A	(A) ← (P2)	7
OUTL P0,A	38	1/2	Output A data to port 0	(P0) ← (A)	b,c,d
OUTL P1,A	39	1/2	Output A data to port 1	(P1) ← (A)	
OUTL P2,A	3A	1/2	Output A data to port 2	(P2) ← (A)	
ANL P0, #data	98 data	2/2	AND port 0 with immediate data	(P0) ← (P0) AND data	b,c,d
ANL P1, #data	99 data	2/2	AND port 1 with immediate data	(P1) ← (P1) AND data	
ANL P2, #data	9A data	2/2	AND port 2 with immediate data	(P2) ← (P2) AND data	
ORL P0, #data	88 data	2/2	OR port 0 data with immediate data	(P0) ← (P0) OR data	b,c,d
ORL P1, #data	89 data	2/2	OR port 1 data with immediate data	(P1) ← (P1) OR data	
ORL P2, #data	8A data	2/2	OR port 2 data with immediate data	(P2) ← (P2) OR data	
INS A,BUS	08	1/2	Strobed input of BUS data to A	(A) ← (BUS)	a
OUTL BUS,A	02	1/2	Output A data to BUS	(BUS) ← (A)	a
ANL BUS, #data	98	2/2	AND BUS port with immediate data	(BUS) ← (BUS) AND data	a
ORL BUS, #data	88	2/2	OR BUS with immediate data	(BUS) ← (BUS) OR data	a
MOVD A,Pp	0C	1/2	Move port 4-7 data to A	$A_{(0-3)}$ ← (P4); $A_{(4-7)}$ ← 0	a
	0D			$A_{(0-3)}$ ← (P5); $A_{(4-7)}$ ← 0	a
	0E			$A_{(0-3)}$ ← (P6); $A_{(4-7)}$ ← 0	a
	0F			$A_{(0-3)}$ ← (P7); $A_{(4-7)}$ ← 0	a
MOVD Pp,A	3C	1/2	Move A to port 4-7	P4 ← $A_{(0-3)}$	a
	3D			P5 ← $A_{(0-3)}$	a
	3E			P6 ← $A_{(0-3)}$	a
	3F			P7 ← $A_{(0-3)}$	a
ANLD Pp,A	9C	1/2	AND port 4-7 data with A	(P4) ← (P4) AND $A_{(0-3)}$	a
	9D			(P5) ← (P5) AND $A_{(0-3)}$	a
	9E			(P6) ← (P6) AND $A_{(0-3)}$	a
	9F			(P7) ← (P7) AND $A_{(0-3)}$	a
ORLD Pp,A	8C	1/2	OR port 4-7 data with A	(P4) ← (P4) OR $A_{(0-3)}$	a
	8D			(P5) ← (P5) OR $A_{(0-3)}$	a
	8E			(P6) ← (P6) OR $A_{(0-3)}$	a
	8F			(P7) ← (P7) OR $A_{(0-3)}$	a

MNEMONIC	OPCODE (HEX)	BYTES/ CYCLES	DESCRIPTION	FUNCTION	NOTES
DERIVATIVE INPUT/OUTPUT					
MOV A,Dx	8C direct	2/2	Move derivative register contents to A	(A) ← (Dx)	x = 0 to 255 c, d, 8
MOV Dx,A	8D direct	2/2	Move A contents to derivative register	(Dx) ← (A)	x = 0 to 255 c, d, 8
ANL Dx,A	8E direct	2/2	AND derivative register with A	(Dx) ← (Dx) AND (A)	x = 0 to 255 c, d, 8
ORL Dx,A	8F direct	2/2	OR derivative register with A	(Dx) ← (Dx) OR (A)	x = 0 to 255 c, d, 8
SERIAL INPUT/OUTPUT					
MOV A,S_n	0C 0D	1/2	Move serial I/O register contents to A	(A) ← (S0) (A) ← (S1)	n = 0, 1 b, c, d
MOV S_n,A	3C 3D 3E	1/2	Move A contents to serial I/O register	(S0) ← (A) (S1) ← (A) (S2) ← (A)	n = 0, 1, 2 b, c, d
MOV S_n,#data	9C data 9D data 9E data	2/2	Move immediate data to serial I/O register	(S0) ← data (S1) ← data (S2) ← data	n = 0, 1, 2 b, c, d
EN SI	85	1/1	Enable serial I/O interrupt		b, c, d
DIS SI	95	1/1	Disable serial I/O interrupt		b, c, d
NOP	00	1/1	No operation	(PC_{0-10}) ← (PC_{0-10})+1	

Notes

a. 8048 only.
b. 84XX only.
c. 84CXXX only.
d. 33XX only.
e. 80C49 only.

1. PSW CY, AC affected.
2. PSW CY affected.
3. PSW PS affected.
4. Execution of a JTF or JNTF instruction resets the Timer Flag (TF).
5. PSW RBS affected.
6. PSW SP_0, SP_1, SP_2 affected.
7. (A) = 0000, P2.3, P2.2, P2.1, P2.0 for 84CXXX and 33XX; for 84XX, (A) = 1111, P2.3, P2.2, P2.1, P2.0.
8. For more information on the derivative input/output instructions of a particular microcontroller, consult the specific microcontroller data sheet.
9. SEL MB instructions may not be used within interrupt routines.
10. All PSW bits affected.

INSTRUCTIONS COMMON TO ALL 8048 BASED MICROCONTROLLERS

This section describes instructions that are common to all microcontrollers based on the 8048 instruction set. For full details on the instruction set of a particular microcontroller family, this section should be read in conjunction with the section that details the additional instructions which are specific to that microcontroller family.

ADD A,#data — Add immediate data to accumulator

Encoding | 0 0 0 0 | 0 0 1 1 | d_7 d_6 d_5 d_4 | d_3 d_2 d_1 d_0 | 03H

Description: This is a 2-cycle instruction. The specified data is added to the accumulator. The carry and half carry flags are affected.

Operation: (A) ← (A) + data

Example: ADD A,#ADDLE — ADD VALUE OF SYMBOL 'ADDLE' TO ACC

ADD A,Rr — Add register contents to accumulator

Encoding | 0 1 1 0 | 1 r r r | 68H-6FH

Description: The contents of working register 'r' are added to the accumulator. The carry and half carry flags are affected.

Operation: (A) ← (A) + (Rr) — r = 0-7

Example: ADD A,R6 — ADD REG 6 CONTENTS TO ACC

ADD A,@Rr — Add data memory contents to accumulator

Encoding | 0 1 1 0 | 0 0 0 r | 60H-61H

Description: The contents of the resident data memory location addressed by working register 'r' are added to the accumulator. The carry and half carry flags are affected.

Operation: (A) ← (A) + ((Rr)) — r = 0-1

Example:

```
MOV R0,#01FH    MOVE '1F' HEX TO REG 0
ADD A,@R0       ADD VALUE OF LOCATION 31 TO ACC
```

ADDC A,#data — Add carry and immediate data to accumulator

Encoding: 0 0 0 1 | 0 0 1 1 — d_7 d_6 d_5 d_4 | d_3 d_2 d_1 d_0 — 13H

Description: This is a 2-cycle instruction. The content of the carry bit is added to accumulator location 0 and the carry bit is cleared. Then the specified data is added to the accumulator. The carry and half carry flags are affected.

Operation: (A) ← (A) + (C) + data

Example:

ADDC A,#225	ADD CARRY AND '255' DEC TO ACC

ADDC A,Rr — Add carry and register contents to accumulator

Encoding: 0 1 1 1 | 1 r r r — 78H-7FH

Description: The content of the carry bit is added to accumulator location 0 and the carry bit is cleared. The contents of working register 'r' are then added to the accumulator. The carry and half carry flags are affected.

Operation: (A) ← (A) + (C) + (Rr) r = 0-7

Example:

ADDC A,R5	ADD CARRY AND REG 5 CONTENTS TO ACC

ADDC A,@Rr — Add carry and data memory contents to accumulator

Encoding: 0 1 1 1 | 0 0 0 r — 70H-71H

Description: The content of the carry bit is added to accumulator location 0 and the carry bit is cleared. Then the contents of the resident data memory location addressed by working register 'r' are added to the accumulator. The carry and half carry flags are affected.

Operation: (A) ← (A) + (C) + ((Rr)) r = 0-1

Example:

MOV R1,#40	MOVE '40' DEC TO REG 1
ADDC A,@R1	ADD CARRY AND LOCATION 40 CONTENTS TO ACC

ANL A,#data — Logical AND accumulator with immediate mask

Encoding: | 0 1 0 1 | 0 0 1 1 | | d_7 d_6 d_5 d_4 | d_3 d_2 d_1 d_0 | 53H

Description: This is a 2-cycle instruction. Data in the accumulator is logical ANDed with an immediately-specified mask.

Operation: (A) ← (A) AND data

Example:

```
ANL A,#0AFH      'AND' ACC CONTENTS WITH MASK 1010 1111
ANL A,#3 + X/Y   'AND' ACC CONTENTS WITH VALUE OF
                 EXPRESSION '3 + X/Y'
```

ANL A,Rr — Logical AND accumulator with register mask

Encoding: | 0 1 0 1 | 1 r r r | 58H-5FH

Description: Data in the accumulator is logically ANDed with the mask contained in working register 'r'.

Operation: (A) ← (A) AND (Rr) r = 0-7

Example:

```
AND A,R4         'AND' ACC CONTENTS WITH MASK IN REG 4
```

ANL A,@Rr — Logical AND accumulator with memory mask

Encoding: | 0 1 0 1 | 0 0 0 r | 50H-51H

Description: Data in the accumulator is logically ANDed with the mask contained in data memory location referenced by register 'r'.

Operation: (A) ← (A) AND ((Rr)) r = 0-1

Example:

```
MOV R0,#03FH     MOVE '3F' HEX TO REG 0
ANL A,@R0        'AND' ACC CONTENTS WITH MASK IN
                 LOCATIONS 63
```

ANL Pp,#data — Logical AND port with immediate mask

Encoding: | 1 0 0 1 | 1 0 p p | | d_7 d_6 d_5 d_4 | d_3 d_2 d_1 d_0 | 98H-9AH

Description: This is a 2-cycle instruction. Data on port 'p' is logically ANDed with an immediately-specified mask.

Operation: (Pp) ← (Pp) AND data p = 0-2; P0 is not used by 8048

Example:

```
ANL P1,#0F0H     'AND' PORT 1 CONTENTS WITH MASK 'F0' HEX
                 (CLEAR P1.0-P1.3)
```

CALL address — Subroutine call

Encoding

a_{10} a_9 a_8 1	0 1 0 0

a_7 a_6 a_5 a_4	a_3 a_2 a_1 a_0

Description

This is a 2-cycle instruction. The program counter and 3 or 4 PSW bits are saved in the stack. The stack pointer is updated (PSW bits 0-2). Program control is then passed to the location specified by 'address'. The status of PC bit 11 (and bit 12, if present) is determined by the most recent SEL MB instruction.

A CALL cannot commence from locations 2046-2047, 4094-4095 or 6142-6143. Execution continues at the instruction following the CALL upon return from the subroutine.

Operation

8048

$((SP)) \leftarrow (PC), (PSW_{4\text{-}7})$
$(SP) \leftarrow (SP) + 1$
$(PC_{8\text{-}10}) \leftarrow addr_{8\text{-}10}$
$(PC_{0\text{-}7}) \leftarrow addr_{0\text{-}7}$
$(PC_{11}) \leftarrow MBFF$

84XX, 84CXXX, 33XX

$((SP)) \leftarrow (PC), (PSW_{4,6,7})$
$(SP) \leftarrow (SP) + 1$
$(PC_{8\text{-}10}) \leftarrow addr_{8\text{-}10}$
$(PC_{0\text{-}7}) \leftarrow addr_{0\text{-}7}$
$(PC_{11\text{-}12}) \leftarrow MBFF_{0\text{-}1}$

Example

Add three groups of two numbers. Put subtotals in locations 50, 51 and total in location 52.

```
        MOV R0,#50      MOVE '50' DEC TO ADDRESS REG 0
        MOV A,R1        MOVE CONTENTS OF REG 1 TO ACC
        ADD A,R2        ADD REG 2 TO ACC
        CALL SUBTOT     CALL SUBROUTINE 'SUBTOT'
        ADDC A,R3       ADD REG 3 TO ACC
        ADDC A,R4       ADD REG 4 TO ACC
        CALL SUBTOT     CALL SUBROUTINE 'SUBTOT'
        ADDC A,R5       ADD REG 5 TO ACC
        ADDC A,R6       ADD REG 6 TO ACC
        CALL SUBTOT     CALL SUBROUTINE 'SUBTOT'

SUBTOT  MOV @R0,A       MOVE CONTENTS OF ACC TO LOCATION
                        ADDRESSED BY REG 0
        INC R0          INCREMENT REG 0
        RET             RETURN TO MAIN PROGRAM
```

CLR A — Clear accumulator

Encoding

0 0 1 0	0 1 1 1

27H

Description

The contents of the accumulator are cleared to zero.

Operation

$(A) \leftarrow 0$

CLR C — Clear carry bit

Encoding: 1 0 0 1 | 0 1 1 1 — 97H

Description: During normal program execution, the carry bit can be set to one by the ADD, ADDC, CPL C, RLC, RRC, and DAA instructions. This instruction resets the carry bit to zero.

Operation: (C) ← 0

CPL A — Complement accumulator

Encoding: 0 0 1 1 | 0 1 1 1 — 37H

Description: The contents of the accumulator are complemented. This is strictly a one's complement. Each zero is changed to one and vice-versa.

Operation: (A) ← NOT (A)

Example: Assume accumulator contains 1001 0101.

```
CPL A        ACC CONTENTS ARE COMPLEMENTED TO
             0110 1010
```

CPL C — Complement carry bit

Encoding: 1 0 1 0 | 0 1 1 1 — A7H

Description: The carry bit is complemented; a zero is changed to one, and vice-versa.

Operation: (C) ← NOT (C)

Example: Set C to one; current status is unknown.

```
CLR C        C IS CLEARED TO ZERO
CPL C        C IS SET TO ONE
```

DA A — Decimal adjust accumulator

Encoding

0 1 0 1	0 1 1 1	57H

Description

The 8-bit accumulator value is adjusted to form two 4-bit Binary Coded Decimal (BCD) digits following the binary addition of BCD numbers. The carry bit C is affected. If the contents of bits 0-3 are greater than nine, or if AC is one, the accumulator is incremented by six.

The four high-order bits are then checked. If bits 4-7 exceed nine, of if C is one, these bits are increased by six. If an overflow occurs, C is set to one.

Example

Assume accumulator contains 1001 1011.

```
DA A                          ACC ADJUSTED TO 0000 0001 WITH C SET

C AC 7 6 5 4 3 2 1 0
0 0  1 0 0 1 1 0 1 1
           0 1 1 0            ADD SIX TO BITS 0-7
0 0  1 0 1 0 0 0 0 1
     0 1 1 0                  ADD SIX TO BITS 4-7
1 0  0 0 0 0 0 0 0 1          OVERFLOW TO C
```

DEC A — Decrement accumulator

Encoding

0 0 0 0	0 1 1 1	07H

Description

The contents of the accumulator are decremented by one.

Operation

(A) ← (A) –1

Example

Decrement the contents of external data memory location 63.

```
MOV R0,#3FH       MOVE '3F' HEX TO REG 0
MOV A,@R0         MOVE CONTENTS OF LOCATION 63 TO ACC
DEC A             DECREMENT ACC
MOV @R0,A         MOVE CONTENTS OF ACC TO LOCATION 63 IN
                  EXTERNAL MEMORY
```

DEC Rr — Decrement register

Encoding | 1 1 0 0 | 1 r r r | C8H-CFH

Description: The contents of working register 'r' are decremented by one.

Operation: (Rr) ← (Rr) –1 r = 0-7

Example: DEC R1 DECREMENT CONTENTS OF REG 1

DIS I — Disable external interrupt

Encoding | 0 0 0 1 | 0 1 0 1 | 15H

Description: External interrupts are disabled. A LOW signal on the interrupt input has no effect.

DIS TCNTI — Disable timer/counter interrupt

Encoding | 0 0 1 1 | 0 1 0 1 | 35H

Description: Timer/counter interrupts are disabled. Any pending timer interrupt request is cleared. The interrupt sequence is not initiated by an overflow, but the timer flag is set and time accumulation continues.

DJNZ Rr,addr — Decrement register and test

Encoding

1 1 1 0	1 r r r

a_7 a_6 a_5 a_4	a_3 a_2 a_1 a_0

E8H-EFH

Description

This is a 2-cycle instruction. Register 'r' is decremented, then tested for zero. If the register contains all zeros, program control passes to the next instruction. If the register contents are not zero, program control jumps to the specified 'address'.

The address in this case must evaluate to 8-bits (the jump must be to a location within the current 256-location page).

Operation

(Rr) ← (Rr) –1 r = 0-7
If Rr not 0
then (PC_{0-7}) ← addr
else (PC) ← (PC) + 2

Note: A 12-bit address specification does not cause an error if the DJNZ instruction and the jump target address are on the same page. If the DJNZ instruction begins in location 255 of a page, it must jump to a target address on the following page.

Example

Increment values in data memory locations 50-54.

```
        MOV R0,#50          MOVE '50' DEC TO ADDRESS REG 0
        MOV R3,#5           MOVE '5' DEC TO COUNTER REG 3
INCRT   INC @R0             INCREMENT CONTENTS OF LOCATION
                            ADDRESSED BY REG 0
        INC R0              INCREMENT ADDRESS IN REG 0
        DJNZ R3,INCRT       INCREMENT REG 3 - JUMP TO 'INCRT' IF REG 3
                            NONZERO
        NEXT ---            'NEXT' ROUTINE EXECUTED IF R3 IS ZERO
```

EN I — Enable external interrupt

Encoding

0 0 0 0	0 1 0 1

05H

Description

External interrupts are enabled.

EN TCNTI — Enable timer/counter interrupt

Encoding

0 0 1 0	0 1 0 1

25H

Description

Timer/counter interrupts are enabled. An overflow of the timer/counter initiates the interrupt sequence.

IN A,Pp — Input port or data to accumulator

Encoding | 0 0 0 0 | 1 0 p p | 08H-0AH

Description: This is a 2-cycle instruction. Data present on Port 'p' is transferred (read) to the accumulator.

Operation: (A) ← (Pp) — p = 0-2; P0 is not used by 8048

Example:

```
IN A,P1     INPUT PORT 1 CONTENTS TO ACC
MOV R6,A    MOVE ACC CONTENTS TO REG 6
IN A,P2     INPUT PORT 2 CONTENTS TO ACC
MOV R7,A    MOVE ACC CONTENTS TO REG 7
```

INC A — Increment accumulator

Encoding | 0 0 0 1 | 0 1 1 1 | 17H

Description: Accumulator contents are incremented by one. Carry is not affected.

Operation: (A) ← (A) + 1

Example: Increment contents of the accumularor by 2.

```
INC A       INCREMENT ACC
INC A       INCREMENT ACC AGAIN
```

INC Rr — Increment register

Encoding | 0 0 0 1 | 1 r r r | 18H-1FH

Description: The contents of working register 'r' are incremented by one.

Operation: (Rr) ← (Rr) + 1 — r = 0-7

Example: INC R0 — INCREMENT ADDRESS REG 0

INC @Rr — Increment data memory location

Encoding: 0 0 0 1 | 0 0 0 r — 10H-11H

Description: The contents of the resident data memory location addressed by register 'r' are incremented by one.

Operation: ((Rr)) ← ((Rr)) + 1 — r = 0-1

Example:
MOV R1,#03FH — MOVE '3F' HEX TO REG 1
INC @R1 — INCREMENT LOCATION 63

JBb address — Jump if accumulator bit is set

Encoding: b_2 b_1 b_0 1 | 0 0 1 0 — a_7 a_6 a_5 a_4 | a_3 a_2 a_1 a_0 — 12H, 32H, 52H, 72H, 92H, B2H, D2H, F2H

Description: This is a 2-cycle instruction. Control passes to the specified address if accumulator bit 'b' is set to one.

Operation:
$(PC_{0-7}) \leftarrow$ addr — If Bb = 1
(PC) ← (PC) + 2 — If Bb = 0

Example:
JB4 NEXT — JUMP TO 'NEXT' ROUTINE IF ACC BIT 4 IS ONE

Note: If this instruction begins in location 255 of a page, the target address is located in the following page.

JC address — Jump if carry is set

Encoding: 1 1 1 1 | 0 1 1 0 — a_7 a_6 a_5 a_4 | a_3 a_2 a_1 a_0 — F6H

Description: This is a 2-cycle instruction. Control passes to the specified address if the carry bit is set to one.

Operation:
$(PC_{0-7}) \leftarrow$ addr — If C = 1
(PC) ← (PC) + 2 — If C = 0

Example:
JC OVFLOW — JUMP TO 'OVFLOW' ROUTINE IF CARRY IS ONE

Note: If this instruction begins in location 255 of a page, the target address is located in the following page.

JMP address — Direct jump within 2K block

Encoding

a_{10}	a_9	a_8	0	0	1	0	0

a_7	a_6	a_5	a_4	a_3	a_2	a_1	a_0

Description

This is a 2-cycle instruction. Bits 0-10 of the program counter are replaced with the directly-specified address. The setting of PC bit 11 (and bit 12, if present) is determined by the most recent SEL MB instruction.

Operation

8048	**84XX, 84CXXX, 33XX**
$(PC_{0-7}) \leftarrow$ addr 0-7	$(PC_{0-7}) \leftarrow$ addr 0-7
$(PC_{8-10}) \leftarrow$ addr 8-10	$(PC_{8-10}) \leftarrow$ addr 8-10
$(PC_{11}) \leftarrow$ MBFF	$(PC_{11-12}) \leftarrow MBFF_{0-1}$

Example

JMP SUBTOT	JUMP TO SUBROUTINE 'SUBTOT'
JMP $-6	JUMP TO INSTRUCTION SIX LOCATIONS BEFORE CURRENT LOCATION
JMP 2FH	JUMP TO ADDRESS '2F' HEX

JMPP @A — Indirect jump within page

Encoding

1	0	1	1	0	0	1	1

B3H

Description

This is a 2-cycle instruction. The contents of the program memory location pointed to by the accumulator are substituted for the 'page' portion of the program counter (PC bits 0-7)

Operation

$(PC_{0-7}) \leftarrow ((A))$

Example

Assume accumulator contains 0FH.

JMPP @A	JUMP TO ADDRESS STORED IN LOCATION 15 IN CURRENT PAGE

JNC address — Jump if carry is not set

Encoding

1	1	1	0	0	1	1	0

a_7	a_6	a_5	a_4	a_3	a_2	a_1	a_0

E6H

Description

This is a 2-cycle instruction. Control passes to the specified address if the carry bit is not set (equals zero).

Operation

$(PC_{0-7}) \leftarrow$ addr	If C = 0
$(PC) \leftarrow (PC) + 2$	If C = 1

Example

JNC NOVFLO	JUMP TO 'NOVFLO' ROUTINE IS CARRY IS ZERO

Note: If this instruction begins in location 255 of a page, the target address is located in the following page.

JNT0 address — Jump if Test 0 input is LOW

Encoding: 0 0 1 0 | 0 1 1 0 — a_7 a_6 a_5 a_4 | a_3 a_2 a_1 a_0 — 26H

Description: This is a 2-cycle instruction. Control passes to the specified address if the Test 0 input signal is LOW (HIGH for 33XX) when this instruction is executed. Otherwise, program control passes to the next instruction.

Operation:
$(PC_{0-7}) \leftarrow$ addr — If T0 = 0 (if T0 = 1 for 33XX)
$(PC) \leftarrow (PC) + 2$ — If T0 = 1 (if T0 = 0 for 33XX)

Example: JNT0 60 — JUMP TO LOCATION 60 DEC IF T0 IS LOW (HIGH for 33XX)

Note: If this instruction begins in location 255 of a page, the target address is located in the following page.

JNT1 address — Jump if Test 1 input is LOW

Encoding: 0 1 0 0 | 0 1 1 0 — a_7 a_6 a_5 a_4 | a_3 a_2 a_1 a_0 — 46H

Description: This is a 2-cycle instruction. Control passes to the specified address if the Test 1 signal is LOW.

Operation:
$(PC_{0-7}) \leftarrow$ addr — If T1 = 0
$(PC) \leftarrow (PC) + 2$ — If T1 = 1

Example: JNT1 110H — JUMP TO LOCATION '110' HEX IF T1 IS LOW

Note: If this instruction begins in location 255 of a page, the target address is located in the following page.

JNZ address — Jump if accumulator is not zero

Encoding

1 0 0 1	0 1 1 0	a_7 a_6 a_5 a_4	a_3 a_2 a_1 a_0

96H

Description

This is a 2-cycle instruction. Control passes to the specified address if the accumulator contents are non-zero when this instruction is executed. Otherwise, program control passes to the next instruction.

Operation

$(PC_{0-7}) \leftarrow$ addr	If $A \neq 0$
$(PC) \leftarrow (PC) + 2$	If $A = 0$

Example

JNZ ACCNOT	JUMP TO ROUTINE 'ACCNOT' IF ACC VALUE IS NON-ZERO

Note: If this instruction begins in location 255 of a page, the target address is located in the following page.

JT0 address — Jump if Test 0 input is HIGH

Encoding

0 0 1 1	0 1 1 0	a_7 a_6 a_5 a_4	a_3 a_2 a_1 a_0

36H

Description

This is a 2-cycle instruction. Control passes to the specified address, if the Test 0 input pin is HIGH (LOW for 33XX) when this instruction is executed. Otherwise, program control passes to the next instruction.

Operation

$(PC_{0-7}) \leftarrow$ addr	If T0 = 1 (if T0 = 0 for 33XX)
$(PC) \leftarrow (PC) + 2$	If T0 = 0 (if T0 = 1 for 33XX)

Example

JT0 COUNT	JUMP TO 'COUNT' ROUTINE IF T0 IS HIGH (LOW for 33XX)

Note: If this instruction begins in location 255 of a page, the target address is located in the following page.

JT1 address — Jump if Test 1 input is HIGH

Encoding: | 0 1 0 1 | 0 1 1 0 | a_7 a_6 a_5 a_4 | a_3 a_2 a_1 a_0 | 56H

Description: This is a 2-cycle instruction. Control passes to the specified address, if the Test 1 input pin is HIGH when this instruction is executed. Otherwise, program control passes to the next instruction.

Operation:
$(PC_{0-7}) \leftarrow$ addr — If T1 = 1
$(PC) \leftarrow (PC) + 2$ — If T1 = 0

Example: JT1 COUNT — JUMP TO 'COUNT' ROUTINE IF T1 IS HIGH

Note: If this instruction begins in location 255 of a page, the target address is located in the following page.

JTF address — Jump if timer flag is set

Encoding: | 0 0 0 1 | 0 1 1 0 | a_7 a_6 a_5 a_4 | a_3 a_2 a_1 a_0 | 16H

Description: This is a 2-cycle instruction. Control passes to the specified address if the timer flag is set to one, that is, the timer/counter has overflowed. Otherwise, program control passes to the next instruction. Testing the timer flag resets it to zero. (This overflow initiates an interrupt service sequence if the timer-overflow interrupt is enabled).

Operation:
$(PC_{0-7}) \leftarrow$ addr — If TF = 1
$(PC) \leftarrow (PC) + 2$ — If TF = 0

Example: JTF TIMER — JUMP TO TIMER ROUTINE IF THE TIMER HAS OVERFLOWED (TF=1)

Note: If this instruction begins in location 255 of a page, the target address is located in the following page.

JZ address — Jump if accumulator is zero

Encoding: | 1 1 0 0 | 0 1 1 0 | a_7 a_6 a_5 a_4 | a_3 a_2 a_1 a_0 | C6H

Description: This is a 2-cycle instruction. Control passes to the specified address if the accumulator contains all zeros when this instruction is executed. Otherwise, program control passes to the next instruction.

Operation:
$(PC_{0-7}) \leftarrow$ addr — If A = 0
$(PC) \leftarrow (PC) + 2$ — If A $\neq$ 0

Example: JZ 0A3H — JUMP TO LOCATION 'A3' HEX IF ACC VALUE IS ZERO

MOV A,#data — Move immediate data to accumulator

Encoding: 0 0 1 0 | 0 0 1 1 — d_7 d_6 d_5 d_4 | d_3 d_2 d_1 d_0 — 23H

Description: This is a 2-cycle instruction. The 8-bit value specified by 'data' is loaded in the accumulator.

Operation: (A) ← data

Example:

MOV A,#0A3H	MOVE 'A3' HEX TO ACC

MOV A,PSW — Move PSW contents to accumulator

Encoding: 1 1 0 0 | 0 1 1 1 — C7H

Description: The contents of the program status word are moved to the accumulator.

Operation: (A) ← (PSW)

Example: Jump to the 'RBISET' routine if PSW bank switch, bit 4, is set.

MOV A,PSW	MOVE PSW CONTENTS TO ACC
JB4 RBISET	JUMP TO 'RBISET' IF ACC BIT 4 = 1

MOV A,Rr — Move register contents to accumulator

Encoding: 1 1 1 1 | 1 r r r — F8H-FFH

Description: 8-bits of data are moved from working register 'r' into the accumulator.

Operation: (A) ← (Rr) r = 0-7

Example:

MOV A,R3	MOVE CONTENTS OF REG 3 TO ACC

MOV A,@Rr — Move data memory contents to accumulator

Encoding | 1 1 1 1 | 0 0 0 r | F0H-F1H

Description: The contents of the resident data memory location addressed by working register 'r' are moved to the accumulator. Register 'r' contents are unaffected.

Operation: (A) ← ((Rr)) r = 0-1

Example: Assume R1 contains 0011 0110

```
MOV A,@R1        MOVE CONTENTS OF DATA MEM LOCATION 54
                 TO ACC
```

MOV A,T — Move timer/counter contents to accumulator

Encoding | 0 1 0 0 | 0 0 1 0 | 42H

Description: The contents of the timer/event counter register are moved to the accumulator.

Operation: (A) ← (T)

Example: Jump to 'EXIT' if the timer has reached or exceeded '100'.

```
CLR C            CLEAR CARRY
MOV A,T          MOVE TIMER CONTENTS TO ACC
ADD A,#156       ADD 156 DEC TO ACC
JC EXIT          JUMP TO 'EXIT' ROUTINE IF CARRY IS SET
```

MOV PSW,A — Move accumulator contents to PSW

Encoding | 1 1 0 1 | 0 1 1 1 | D7H

8048

Description: The contents of the accumulator are moved into the program status word. All condition bits and the stack pointer are affected by this move.

Operation: (PSW) ← (A)

Example: Move up stack pointer by two memory locations, i.e. increment the pointer by one.

```
MOV A,PSW      MOVE PSW CONTENTS TO ACC
INC A          INCREMENT ACC BY ONE
MOV PSW,A      MOVE ACC CONTENTS TO PSW
```

84XX, 84CXXX, 33XX

Description: The content of accumulator bit 3 is moved into the prescaler switch.

Operation: (PS) ← (A_3)

Example: Set the timer to 'module 1' mode.

```
MOV A,#08H     SET ACC BIT 3 TO LOGIC ONE
MOV PSW,A      SET PS (= PSW3) TO LOGIC 1
```

MOV Rr,#data — Move immediate data to register

Encoding | 1 0 1 1 | 1 r r r | d_7 d_6 d_5 d_4 | d_3 d_2 d_1 d_0 | B8H-BFH

Description: This is a 2-cycle instruction. The 8-bit value specified by 'data' is moved to register 'r'.

Operation: (Rr) ← data r = 0-7

Example:

```
MOV R4,#HEXTEN      THE VALUE OF THE SYMBOL 'HEXTEN' IS
                    MOVED INTO REG 4
MOV R5,#P1*(R*R)    THE VALUE OF THE EXPRESSION 'P1*(R*R)' IS
                    MOVED INTO REG 5
MOV R3,#0ADH        'AD' HEX IS MOVED INTO REG 3
```

MOV @Rr,#data — Move immediate data to data memory

Encoding: 1 0 1 1 | 0 0 0 r | d_7 d_6 d_5 d_4 | d_3 d_2 d_1 d_0 — B0H-B1H

Description: This is a 2-cycle instruction. The 8-bit value specified by 'data' is moved to the resident data memory location addressed by register Rr.

Operation: ((Rr)) ← data — r = 0-1

Example: Move the hexadecimal value AC3F to locations 61-62.

```
MOV R0,#61          MOVE '61' DEC TO ADDR REG 0
MOV @R0,#0ACH       MOVE 'AC' HEX TO LOCATION 61
INC R0              INCREMENT REG 0 TO '62'
MOV @R0,#3FH        MOVE '3F' HEX TO LOCATION 62
```

MOV Rr,A — Move accumulator contents to register

Encoding: 1 0 1 0 | 1 r r r — A8H-AFH

Description: The contents of the accumulator are moved to working register 'r'.

Operation: (Rr) ← (A) — r = 0-7

Example:

```
MOV R0,A            MOVE CONTENTS OF ACC TO REG 0
```

MOV @Rr,A — Move accumulator contents to data memory

Encoding: 1 0 1 0 | 0 0 0 r — A0H-A1H

Description: The contents of the accumulator are moved to the resident data memory location whose address is specified by register Rr. Register Rr contents are unaffected.

Operation: ((Rr)) ← A — r = 0-1

Example: Assume R0 contains 0000 0111.

```
MOV @R0,A           MOVE CONTENTS OF ACC TO LOCATION 7
                    (REG 7)
```

MOV T,A — Move accumulator contents to timer/counter

Encoding | 0 1 1 0 | 0 0 1 0 | 62H

Description: The contents of the accumulator are moved to the timer/event counter register.

Operation: (T) ← (A)

Example: Initialize and start event counter.

```
CLR A       CLEAR ACC TO ZEROS
MOV T,A     MOVE ZEROS TO EVENT COUNTER
STRT CNT    START COUNTER
```

MOVP A,@A — Move current page data to accumulator

Encoding | 1 0 1 0 | 0 0 1 1 | A3H

Description: This is a 2-cycle instruction. The contents of the program memory location addressed by the accumulator are moved to the accumulator. Only bits 0-7 of the program counter are affected, limiting the program memory reference to the current page. The program counter is restored following this operation.

Operation:
$(PC_{0-7}) \leftarrow (A)$
$(A) \leftarrow ((PC))$

Note: This is a 1-byte, 2-cycle instruction. If it appears in location 255 of a program memory page, @A addresses a location in the following page.

Example:

```
MOV A,#128   MOVE '128' DEC TO ACC
MOVP A,@A    CONTENTS OF 129th LOCATION IN CURRENT
             PAGE ARE MOVED TO ACC
```

NOP — The NOP instruction

Encoding | 0 0 0 0 | 0 0 0 0 | 00H

Description: No operation is performed. Execution continues with the next instruction.

ORL A,#data — Logical OR accumulator with immediate mask

Encoding: 0 1 0 0 | 0 0 1 1 — d_7 d_6 d_5 d_4 | d_3 d_2 d_1 d_0 — 43H

Description: This is a 2-cycle instruction. Data in the accumulator is logically ORed with an immediately-specified mask.

Operation: (A) ← (A) OR data

Example: ORL A,#'X' — 'OR' ACC CONTENTS WITH MASK 0101 1000 (ASCII VALUE OF 'X').

ORL A,Rr — Logical OR accumulator with register mask

Encoding: 0 1 0 0 | 1 r r r — 48H-4FH

Description: Data in the accumulator is logically ORed with the mask contained in working register 'r'.

Operation: (A) ← (A) OR (Rr) — r = 0-7

Example: ORL A,R4 — 'OR' ACC CONTENTS WITH MASK IN REG 4.

ORL A,@Rr — Logical OR accumulator with memory mask

Encoding: 0 1 0 0 | 0 0 0 r — 40H-41H

Description: Data in the accumulator is logically ORed with the mask contained in the resident data memory location addressed by register Rr.

Operation: (A) ← (A) OR ((Rr)) — r = 0-1

Example:

```
MOV RO,#3FH     MOVE '3F' HEX TO REG 0
ORL A,@R0       'OR' ACC CONTENTS WITH MASK IN
                LOCATION 63.
```

ORL Pp,#data — Logical OR port with immediate mask

Encoding: 1 0 0 0 | 1 0 p p — d_7 d_6 d_5 d_4 | d_3 d_2 d_1 d_0 — 88H-8AH

Description: This is a 2-cycle instruction. Port 'p' data is logically ORed with an immediately-specified mask.

Operation: (Pp) ← (Pp) OR data — p = 0-2; P0 is not used by 8048

Example: ORL P1,#0FFH — 'OR' PORT 1 CONTENTS WITH MASK 'FF' HEX (SET PORT 1 TO ALL ONES)

OUTL Pp,A — Output accumulator data to port

Encoding: 0 0 1 1 | 1 0 p p — 38H-3AH

Description: This is a 2-cycle instruction. Data residing in the accumulator is transferred (written) to Port 'p' and latched.

Operation: (Pp) ← (A) — p = 0-2; P0 is not used by 8048

Example:

```
MOV A,R7      MOV REG 7 CONTENTS TO ACC
OUTL P1,A     OUTPUT ACC CONTENTS TO PORT 1
MOV A,R6      MOVE REG 6 CONTENTS TO ACC
OUTL P2,A     OUTPUT ACC CONTENTS TO PORT 2
```

RET — Return without PSW restore

Encoding: 1 0 0 0 | 0 0 1 1 — 83H

Description: This is a 2-cycle instruction. The stack pointer is decremented. The program counter is then restored from the stack. PSW bits are not restored.

Operation:
(SP) ← (SP)-1
(PC) ← ((SP))

RETR — Return with PSW restore

Encoding: 1 0 0 1 | 0 0 1 1 — 93H

Description: This is a 2-cycle instruction. The stack pointer is decremented. The program counter and PSW are then restored from the stack. Note that RETR should be used to return frominterrupt service routines, but should not be used within them as RETR signals the end of an interrupt routine.

Operation:

8048	**84XX, 84CXXX, 33XX**
(SP) ← (SP)-1	(SP) ← (SP)-1
(PC) ← ((SP))	(PC) ← ((SP))
(PSW_{4-7}) ← ((SP))	($PSW_{4,6,7}$) ← ((SP))

RL A — Rotate left without carry

Encoding: 1 1 1 0 | 0 1 1 1 — E7H

Description: The contents of the accumulator are rotated left one bit. Bit 7 is rotated into the bit 0 position.

Operation:

(An+1) ← (An) n = 0-6
(A0) ← (A7)

Example: Assume accumulator contains 1011 0001

RL A — NEW ACC CONTENTS ARE 0110 0011

RLC A — Rotate left through carry

Encoding: 1 1 1 1 | 0 1 1 1 — F7H

Description: The contents of the accumulator are rotated left one bit. Bit 7 replaces the carry bit; the carry bit is rotated into the bit 0 position.

Operation:

(An+1) ← (An) n = 0-6
(A0) ← (C)
(C) ← (A7)

Example: Assume accumulator contains a 'signed' number; isolate sign without changing value.

CLR C	CLEAR CARRY TO ZERO
RLC A	ROTATE ACC LEFT, SIGN BIT (7) IS PLACED IN CARRY
RR A	ROTATE ACC RIGHT - VALUE (BITS 0-6) IS RESTORED, CARRY UNCHANGED, BIT 7 IS ZERO.

RR A — Rotate right without carry

Encoding | 0 1 1 1 | 0 1 1 1 | 77H

Description: The contents of the accumulator are rotated right one bit. Bit 0 is rotated into the bit 7 position.

Operation:
(An) ← (An+1) n = 0-6
(A7) ← (A0)

Example: Assume accumulator contains 1011 0001.
RR A — NEW ACC CONTENTS ARE 1101 1000

RRC A — Rotate right trough carry

Encoding | 0 1 1 0 | 0 1 1 1 | 67H

Description: The contents of the accumulator are rotated right one bit. Bit 0 replaces the carry bit; the carry bit is rotated into the bit 7 position.

Operation:
(An) ← (An+1) n = 0-6
(A7) ← (C)
(C) ← (A0)

Example: Assume carry is not set and accumulator contains 1011 0001.
RRC A — CARRY IS SET AND ACC CONTAINS 0101 1000

SEL MB0 — Select memory bank 0

Encoding | 1 1 1 0 | 0 1 0 1 | E5H

Description: Only devices with more than 2 K bytes of program memory use this instruction. PC bit 11 (and PC bit 12, if present) is set to zero on the next JMP or CALL instruction. All references to program memory addresses fall within the range 0-2047.

Operation:

8048	**84XX, 84CXXX, 33XX**
(MBFF) ← (0)	$(MBFF_{1,0}) \leftarrow (0,0)$

SEL MB1 — Select memory bank 1

Encoding | 1 1 1 1 | 0 1 0 1 | F5H

Description: Only devices with more than 2 K bytes of program memory use this instruction. PC bit 11 is set to '1' (and PC bit 12, if present, is reset to '0') on the next JMP or CALL instruction. All references to program memory addresses fall within the range 2048-4095.

Operation

8048	**84XX, 84CXXX, 33XX**
(MBFF) ← (1)	$(MBFF_{1,0}) \leftarrow (0,1)$

SEL RB0 — Select register bank 0

Encoding | 1 1 0 0 | 0 1 0 1 | C5H

Description: PSW bit 4 is reset to zero. References to working registers 0-7 address data memory locations 0-7. This is the recommended setting for normal program execution.

Operation: (RBS) ← 0

SEL RB1 — Select register bank 1

Encoding | 1 1 0 1 | 0 1 0 1 | D5H

Description: PSW bit 4 is set to one. References to working registers 0-7 address data memory locations 24-31. This is the recommended setting for interrupt service routines, since locations 0-7 are left intact. The setting of PSW bit 4 in effect at the time of an interrupt is restored by the RETR instruction when the interrupt service routine is completed.

Operation: (RBS) ← 1

Example: Assume an external interrupt has occurred, control has passed to program memory location 3, and PSW bit 4 was zero before the interrupt.

```
LOC:3  JNI INIT          JUMP TO ROUTINE 'INIT' IF INTERRUPT INPUT IS
                         ZERO
INIT:  MOV R7,A          MOVE ACC CONTENTS TO LOCATION 7
       SEL RB1           SELECT REG BANK 1
       MOV R7,#0FAH      MOVE 'FA' HEX TO LOCATION 31
       •
       •
       •
       SEL RB0           SELECT REG BANK 0
       MOV A,R7          RESTORE ACC FROM LOCATION 7
       RETR              RETURN - RESTORE PC AND PSW
```

STOP TCNT Stop timer/event counter

Encoding | 0 1 1 0 | 0 1 0 1 | 65H

Description: This instruction is used to stop both time accumulation and event counting.

Example: Disable interrupt, but jump to interrupt routine after eight overflows and stop timer. Count overflows in register 7.

```
START:  DIS TCNTI       DISABLE TIMER INTERRUPT
        CLR A           CLEAR ACC TO ZERO
        MOV T,A         MOVE ZEROS TO TIMER
        MOV R7,A        MOVE ZEROS TO REG 7
        STRT T          START TIMER
MAIN:   JTF COUNT       JUMP TO ROUTINE 'COUNT' IF TF = 1 AND
                        CLEAR TIMER FLAG
        JMP MAIN        CLOSE LOOP
COUNT:  INC R7          INCREMENT COUNTER (REG 7)
        MOV A,R7        MOVE REG 7 CONTENTS TO ACC
        JB3 INT         JMP TO 'INT' ROUTINE IF ACC BIT 3 IS SET
                        (REG 7 = 8)
        JMP MAIN        OTHERWISE RETURN TO ROUTINE 'MAIN'
        •
        •
        •
INT:    STOP TCNT       STOP TIMER
        JMP 7H          JUMP TO LOCATION 7 (TIMER) INTERRUPT
                        ROUTINE
```

STRT CNT Start event counter

Encoding | 0 1 0 0 | 0 1 0 1 | 45H

Description: The Test 1 (T1) pin is configured as the event counter input and the counter is started. The event counter register is incremented with each LOW-to-HIGH (HIGH-to-LOW for 8048) transition on the T1 pin.

Example: Initialize and start event counter. Assume overflow is desired with first T1 transition.

```
EN TCNTI        ENABLE COUNTER INTERRUPT
MOV A,#0FFH     MOVE ALL ONES TO ACC
MOV T,A         MOVE ONES TO COUNTER
STRT CNT        ENABLE T1 AS COUNTER INPUT AND START
```

STRT T — Start timer

Encoding | 0 1 0 1 | 0 1 0 1 | 55H

Description: Time accumulation is initiated in the timer register. The register is incremented every 32 machine cycles. The prescaler which counts the 32 cycles is cleared but the timer register is not. For the 84XX, 84CXXX and 33XX, the prescaler may be bypassed. The 8048 prescaler may not be bypassed.

Example: Initialize and start timer.

```
CLR A          CLEAR ACC TO ZEROS
MOV T,A        MOVE ZEROS TO TIMER
EN TCNTI       ENABLE TIMER INTERRUPT
STRT T         START TIMER
```

SWAP A — Swap nibbles within accumulator

Encoding | 0 1 0 0 | 0 1 1 1 | 47H

Description: Bits 0-3 of the accumulator are swapped with bits 4-7 of the accumulator.

Operation: $(A_{4-7}) \leftrightarrow (A_{0-3})$

Example: Pack bits 0-3 of locations 50-51 into location 50.

```
MOV R0,#50     MOVE '50' DEC TO REG 0
MOV R1,#51     MOVE '51' DEC TO REG 1
XCHD A,@R0     EXCHANGE BITS 0-3 OF ACC AND LOCATION 50
SWAP A         SWAP BITS 0-3 AND 4-7 OF ACC
XCHD A,@R1     EXCHANGE BITS 0-3 OF ACC AND LOCATION 51
MOV @R0,A      MOVE CONTENTS OF ACC TO LOCATION 50
```

XCH A,Rr — Exchange accumulator register contents

Encoding | 0 0 1 0 | 1 r r r | 28H-2FH

Description: The contents of the accumulator and the contents of working register 'r' are exchanged.

Operation: (A) ↔ (Rr) r = 0-7

Example: Move PSW contents to Reg 6 without losing accumulator contents.

```
XCH A,R6       EXCHANGE CONTENTS OF REG 6 AND ACC
MOV A,PSW      MOVE PSW CONTENTS TO ACC
XCH A,R6       EXCHANGE CONTENTS OF REG 6 AND ACC
               AGAIN
```

XCH A,@Rr — Exchange accumulator and data memory contents

Encoding | 0 0 1 0 | 0 0 0 r | 20H-21H

Description: The contents of the accumulator and the contents of the resident data memory location addressed by register Rr are exchanged. Register Rr contents are unaffected.

Operation: (A) ↔ ((Rr)) r = 0-1

Example: Decrement contents of location 52

```
MOV R0,#52      MOVE 52 DEC TO ADDRESS REG 0
XCH A,@R0       EXCHANGE CONTENTS OF ACC AND
                LOCATION 52
DEC A           DECREMENT ACC CONTENTS
XCH A,@R0       EXCHANGE CONTENTS OF ACC AND
                LOCATION 52 AGAIN
```

XCHD A,@Rr — Exchange accumulator and data memory 4-bit data

Encoding | 0 0 1 1 | 0 0 0 r | 30H-31H

Description: This instruction exchanges bits 0-3 of the accumulator with bits 0-3 of the data memory location addressed by register Rr. Bits 4-7 of the accumulator, bits 4-7 of the data memory location, and the contents of register Rr are unaffected.

Operation: (A_{0-3}) ↔ ((Rr_{0-3})) r = 0-1

Example: Load bits 0-3 of location 32 into A0-3 and clear bits 0-3 of location 32.

```
MOV R0,#32      MOVE '32' DEC TO REG 0
CLR A           CLEAR ACC TO ZEROS
XCHD A,@R0      EXCHANGE BITS 0-3 OF ACC AND LOCATION 32
                (BITS 0-3 OF LOCATION 32 ARE ZEROED)
```

XRL A,#data — Logical XOR accumulator with immediate mask

Encoding | 1 1 0 1 | 0 0 1 1 | d_7 d_6 d_5 d_4 | d_3 d_2 d_1 d_0 | D3H

Description: This is a 2-cycle instruction. Data in the accumulator is EXCLUSIVE ORed with an immediately-specified mask.

Operation: (A) ← (A) OR data

Example:

```
XRL A,#HEXTWO   'XOR' CONTENTS OF ACC WITH MASK EQUAL
                VALUE OF SYMBOL 'HEXTWO'
```

XRL A,Rr — Logical X0R accumulator with register mask

Encoding | 1 1 0 1 | 1 r r r | D8H-DFH

Description: Data in the accumulator is EXCLUSIVE ORed with the mask contained in working register 'r'.

Operation: (A) ← (A) X0R (Rr) r = 0-7

Example:

```
XRL A,R4        'XOR' ACC CONTENTS WITH MASK IN REG 4
```

XRL A,@Rr — Logical X0R accumulator with memory mask

Encoding | 1 1 0 1 | 0 0 0 r | D0H-D1H

Description: Data in the accumulator is EXCLUSIVE ORed with the mask contained in the data memory location addressed by register Rr.

Operation: (A) ← (A) X0R ((Rr)) r = 0-1

Example:

```
MOV R1,#20H     MOVE '20' HEX TO REG 1
XRL A,@R1       'XOR' ACC CONTENTS WITH MASK IN
                LOCATION 32
```

ADDITIONAL 8048 INSTRUCTIONS

This section describes the 8048 instructions which are not included in the common section; it should be used in conjunction with the common section.

ANL BUS,#data — Logical AND BUS with immediate mask

Encoding | 1 0 0 1 | 1 0 0 0 | d_7 d_6 d_5 d_4 | d_3 d_2 d_1 d_0 | 98H

Description: This is a 2-cycle instruction. Data on the BUS port is logical ANDed with an immediately-specified mask. This instruction assumes prior execution of an 'OUTL BUS, A' instruction.

Operation: (BUS) ← (BUS) AND data

Example:

```
ANL BUS,#MASK        'AND' BUS CONTENTS WITH MASK EQUAL
                     VALUE OF SYMBOL 'MASK'
```

ANLD Pp,A — Logical AND port 4-7 with accumulator mask

Encoding | 1 0 0 1 | 1 1 p p | d_7 d_6 d_5 d_4 | d_3 d_2 d_1 d_0 | 9CH-9FH

Description: This is a 2-cycle instruction. Data on 8243 port 'p' is logically ANDed with the digit mask in accumulator bits 0-3.

Operation: (Pp) ← (Pp) AND (A0-3) p = 4-7

Note: The mapping of port 'p' to opcode bits 0-1 is as follows:

BIT 1	BIT 0	PORT
0	0	4
0	1	5
1	0	6
1	1	7

Example:

```
ANLD P4,A            'AND' PORT 4 CONTENTS WITH ACC BITS 0-3
```

CLR F0 — Clear flag 0

Encoding | 1 0 0 0 | 0 1 0 1 | 85H

Description: Flag 0 is cleared to zero.

Operation: (F0) ← 0

CLR F1 — Clear flag 1

Encoding | 1 0 1 0 | 0 1 0 1 | A5H

Description: Flag 1 is cleared to zero.

Operation: (F1) ← 0

CPL F0 — Complement flag 0

Encoding | 1 0 0 1 | 0 1 0 1 | 95H

Description: Flag 0 is complemented; a zero is changed to one and vice-versa.

Operation: (F0) ← NOT (F0)

CPL F1 — Complement flag 1

Encoding | 1 0 1 1 | 0 1 0 1 | B5H

Description: Flag 1 is complemented; a zero is changed to one and vice-versa.

Operation: (F1) ← NOT (F1)

ENT0 CLK — Enable clock output

Encoding | 0 1 1 1 | 0 1 0 1 | 75H

Description: The T0 test pin is enabled to act as the clock output. This function is disabled by a system reset.

Example:

```
ENT0 CLK        ENABLE T0 AS CLOCK OUTPUT
```

IDLE — Select idle operation (80C49 only)

Encoding | 0 0 0 0 | 0 0 0 1 | 01H

Description: This 2-cycle instruction places the microcontroller in a low-power mode. The oscillator, internal timer, external interrupt and counter pins continue to function and register and RAM status is maintained. To terminate the Idle mode, a system reset must be performed or interrupts must be enabled and an interrupt signal generated.

INS A,BUS — Strobed input of BUS data to accumulator

Encoding: 0 0 0 0 | 1 0 0 0 — 08H

Description: This is a 2-cycle instruction. Data present on the BUS port is transferred (read) to the accumulator when the RD signal goes LOW. This instruction may be used in internal program memory only.

Operation: (A) ← (BUS)

Example: INS A,BUS — INPUT BUS CONTENTS TO ACC

JF0 address — Jump if flag 0 is set

Encoding: 1 0 1 1 | 0 1 1 0 | a_7 a_6 a_5 a_4 | a_3 a_2 a_1 a_0 — B6H

Description: This is a 2-cycle instruction. Control passes to the specified address if flag 0 is set to one.

Operation: (PC_{0-7}) ← addr — If F0 = 1
(PC) ← (PC) + 2 — If F0 = 0

Example: JF0 TOTAL — JUMP TO 'TOTAL' ROUTINE IF F0 = 1

JF1 address — Jump if flag 1 is set

Encoding: 0 1 1 1 | 0 1 1 0 | a_7 a_6 a_5 a_4 | a_3 a_2 a_1 a_0 — 76H

Description: This is a 2-cycle instruction. Control passes to the specified address if flag 1 is set to one.

Operation: (PC_{0-7}) ← addr — If F1 = 1
(PC) ← (PC) + 2 — If F1 = 0

Example: JF1 FILBLUF — JUMP TO 'FILBLUF' ROUTINE IF F1 = 1

JNI address — Jump if interrupt input is LOW

Encoding

1 0 0 0	0 1 1 0	a_7 a_6 a_5 a_4	a_3 a_2 a_1 a_0

86H

Description

This is a 2-cycle instruction. Control passes to the specified address if the interrupt input is LOW (= 0), that is, an external interrupt has been signalled. (This signal initiates an interrupt service sequence if the external interrupt is enabled).

Operation

$(PC_{0-7}) \leftarrow$ addr	If I = 0
$(PC) \leftarrow (PC) + 2$	If I = 1

Example

JNI EXTINT	JUMP TO 'EXTINT' ROUTINE IF I = 0

MOVD A,Pp — Move port 4-7 data to accumulator

Encoding

0 0 0 0	1 1 p p

0CH-0FH

Description

This is a 2-cycle instruction. Data on 8243 port 'p' is moved (read) to accumulator bits 0-3. Accumulator bits 4-7 are reset to zero.

Operation

$(A_{0-3}) \leftarrow (Pp)$	p = 4-7
$(A_{4-7}) \leftarrow 0$	

Note: Bits 0-1 of the opcode are used to represent ports 4-7. If coding in binary rather than assembly language, the mapping of as follows:

Bit 1	Bit 0	Port
0	0	4
0	1	5
1	0	6
1	1	7

Example

MOVD A,P5	MOVE PORT 5 DATA TO ACC BITS 0-3, ZERO ACC BITS 4-7

MOVD Pp,A — Move accumulator data to port 4-7

Encoding | 0 0 1 1 | 1 1 p p | 3CH-3FH

Description: This is a 2-cycle instruction. Data in accumulator bits 0-3 is moved (written) to 8243 port 'p'. Accumulator bits 4-7 are unaffected. (See note above regarding port mapping).

Operation: $(Pp) \leftarrow (A_{0-3})$ p = 4-7

Example: Move data in accumulator to ports 4 and 5.

```
MOVD P4,A     MOVE ACC BITS 0-3 TO PORT 4
SWAP A        EXCHANGE ACC BITS 0-3 AND 4-7
MOVD P5,A     MOVE ACC BITS 0-3 TO PORT 5
```

MOVP3 A,@A — Move page 3 data to accumulator

Encoding | 1 1 1 0 | 0 0 1 1 | E3H

Description: This is a 2-cycle instruction. The contents of the program memory location (within page 3) addressed by the accumulator are moved to the accumulator. The program counter is restored following this operation.

Operation:
$(PC_{0-7}) \leftarrow A$
$(PC_{11,10,9,8}) \leftarrow 0011$
$(A) \leftarrow ((PC))$

Example: Look up ASCII equivalent of hexadecimal code in table contained at the beginning of page 3. Note that ASCII characters are designated by a 7-bit code; the eighth bit is always reset.

```
MOV A,#0B8H   MOVE 'B8' HEX TO ACC (1011 1000)
ANL A,#7FH    LOGICAL AND ACC TO MASK BIT 7
MOVP3 A,@A    MOVE CONTENTS OF LOCATION '38' HEX IN
              PAGE 3 TO ACC (ASCII '8')
```

MOVX A,@Rr — Move external data memory contents to accumulator

Encoding | 1 0 0 0 | 0 0 0 r | 80H-81H

Description: This is a 2-cycle instruction. The contents of the external data memory location addressed by register 'r' are moved to the accumulator. Register 'r' contents are unaffected.

Operation: $(A) \leftarrow ((Rr))$ r = 0-1

Example:
```
Assume R1 contains 0111 0110
MOVX A,@R1    MOVE CONTENTS OF LOCATION 76H TO ACC
```

MOVX @Rr,A — Move accumulator contents to external data memory

Encoding | 1 0 0 1 | 0 0 0 r | 90H-91H

Description: This is a 2-cycle instruction. The contents of the accumulator are moved to the external data memory location addressed by register 'r'. Register 'r' contents are unaffected.

Operation: ((Rr)) ← (A) r = 0-1

Example:
Assume R0 contains 1100 0111
MOVX @R0,A — MOVE CONTENTS OF ACC TO LOCATION C7H IN EXPANDED DATA MEMORY

ORL BUS,#data — Logical OR BUS with immediate mask

Encoding | 1 0 0 0 | 1 0 0 0 | d_7 d_6 d_5 d_4 | d_3 d_2 d_1 d_0 | 88H

Description: This is a 2-cycle instruction. Data on the BUS port is logically ORed with an immediately-specified mask. This instruction assumes prior execution of an 'OUTL BUS,A' instruction.

Operation: (BUS) ← (BUS) OR data

Example: ORL BUS,#HEXMSK — 'OR' BUS CONTENTS WITH MASK EQUAL VALUE OF SYMBOL 'HEXMSK'

ORLD Pp,A — Logical OR port 4-7 with immediate mask

Encoding | 1 0 0 0 | 1 1 p p | 8CH-8FH

Description: This is a 2-cycle instruction. Data on 8243 port 'p' is logically ORed with the mask contained in accumulator bits 0-3.

Operation: (Pp) ← (Pp) AND (A_{0-3}) p = 4-7

Example: ORLD P6,A — 'OR' PORT 6 CONTENTS WITH ACC BITS 0-3

OUTL BUS,A — Output accumulator data to bus

Encoding

0 0 0 0	0 0 1 0

02H

Description

This is a 2-cycle instruction. Data residing in the accumulator is transferred (written) to the BUS port and latched. The latched data remains valid until altered by another OUTL instruction. Any other instruction requiring use of the BUS port (except INS) overwrites the contents of the BUS latch. This includes expanded memory operations (such as the MOVX instruction). Logical operations on BUS data (AND, OR) assume the OUTL BUS,A instruction has been previously executed.

Operation

(BUS) ← (A)

Example

```
OUTL BUS,A          OUTPUT ACC CONTENTS TO BUS
```

ADDITIONAL 84XX INSTRUCTIONS

This section describes the 84XX instructions which are not included in the common section; it should be used in conjunction with the common section.

DEC @Rr — Decrement data memory location

Encoding | 1 1 0 0 | 0 0 0 r | C0H-C1H

Description: The contents of the data memory location addressed by register Rr are decremented by one.

Operation: ((Rr)) ← ((Rr)) –1 r = 0-1

Example:

```
MOV R1,#4FH     MOVE '4F' HEX TO REG 1
DEC @R1         DECREMENT LOCATION 4F
```

DIS SI — Disable serial input/output interrupt

Encoding | 1 0 0 1 | 0 1 0 1 | 95H

Description: Serial input/output interrupts are disabled. An interrupt request from the SI0 has no effect. If the microcontroller enters the IDLE mode when the SI0 interrupt is disabled, it can't be restarted by this interrupt.

Operation: (SIFF) ← 0

DJNZ @Rr,addr — Decrement data memory location and test

Encoding

1 1 1 0	0 0 0 r	a_7 a_6 a_5 a_4	a_3 a_2 a_1 a_0	E0H-E1H

Description

This is a 2-cycle instruction. The memory location addressed by register Rr is decremented, then tested for zero. If the memory location contains all zeros, program control passes to the next instruction. If the register contents are not zero, control jumps to the specified 'address'.

The address in this case must evaluate to 8 bits (the jump must be to a location within the current 256-location page).

Operation

((Rr)) ← ((Rr)) −1 r = 0-1
If ((Rr)) ≠ 0
then (PC_{0-7}) ← addr
else (PC) ← (PC) + 2

Example

Increment values in data memory locations 81-86.

```
        MOV R0,#32      MOVE '32' DEC TO ADDRESS REG 0
        MOV @R0,#6      MOVE '6' DEC TO COUNTER REG 32
        MOV R1,#81      MOVE '81' DEC TO ADDRESS REG 1
INCRT   INC @R1         INCREMENT CONTENTS OF MEMORY
                        LOCATION ADDRESSED BY REG 1
        INC R1          INCREMENT ADDRESS IN REG 1
        DJNZ @R0,INCRT  DECREMENT LOC. 32 - JUMP TO 'INCRT' IF
                        LOCATION 32 NONZERO
        NEXT ---        'NEXT' ROUTINE EXECUTED IF LOCATION 32
                        IS ZERO
```

EN SI — Enable serial input/output interrupt

Encoding

1 0 0 0	0 1 0 1	85H

Description

Serial input/output interrupts are enabled. An interrupt request from the serial I/O initiates the interrupt sequence, in normal mode or in Idle mode.

Operation

(SIFF) ← 1

JNTF address — Jump if timer flag is not set

Encoding: 0 0 0 0 | 0 1 1 0 — a_7 a_6 a_5 a_4 | a_3 a_2 a_1 a_0 — 06H

Description: This is a 2-cycle instruction. Control passes to the specified address if the timer flag is not set, that is, if the timer/counter has not overflowed. Otherwise, program control passes to the next instruction. Testing the timer flag resets it to zero.

Operation:
$(PC_{0-7}) \leftarrow$ addr — If TF = 0
$(PC) \leftarrow (PC) + 2$ — If TF = 1

Example: JNTF NOTIM — JUMP TO 'NOTIM' ROUTINE IF THE TIMER HAS NOT OVERFLOWED

Note: if this instruction begins in location 255 of a page, the target address is located in the following page.

MOV A,Sn — Move serial I/O register contents to accumulator

Encoding: 0 0 0 0 | 1 1 0 n — 0CH, 0DH

Description: This is a 2-cycle instruction. The contents of serial I/O register 'n' are moved to the accumulator.

Operation: $(A) \leftarrow (Sn)$ — n = 0-1

Example: MOV A,S0 — MOVE CONTENTS OF SERIAL I/O DATA REGISTER TO ACC

MOV Sn,A — Move accumulator contents to serial I/O register

Encoding: 0 0 1 1 | 1 1 n n — 3CH-3EH

Description: This is a 2-cycle instruction. The contents of the accumulator are moved to serial I/O register 'n'.

Operation: $(Sn) \leftarrow A$ — n = 0-2

Example: MOV S0,A — MOVE CONTENTS OF ACC TO SIO DATA REGISTER

MOV Sn,#data — Move immediate data to serial I/O register

Encoding	1 0 0 1 | 1 1 n n	d_7 d_6 d_5 d_4 | d_3 d_2 d_1 d_0	9CH-9EH
Description	This is a 2-cycle instruction. The byte specified by 'data' is moved to the serial I/O register 'n'.		
Operation	(Sn) ← data	n = 0-2	
Example	MOV S2,#28H	LOAD SIO CLOCK REGISTER	

SEL MB2 — Select memory bank 2

Encoding	1 0 1 0 | 0 1 0 1	A5H
Description	Only devices with more than 4 K bytes of program memory use this instruction. PC bit 11 is set to zero and PC bit 12 is set to 1 on the next JMP or CALL instruction. All references to program memory addresses fall within the range 4092-6143.	
Operation	(MBFF1,0) ← (1,0)	

SEL MB3 — Select memory bank 3

Encoding	1 0 1 1 | 0 1 0 1	B5H
Description	Only devices with more than 6 K bytes of program memory use this instruction. PC bits 11 and 12 are set to 1 on the next JMP or CALL instruction. All references to program memory addresses fall within the range 6144-8191.	
Operation	(MBFF1,0) ← 1,1	

ADDITIONAL 84CXXX INSTRUCTIONS

This section describes the 84CXXX instructions which are not included in the common section; it should be used in conjunction with the common section.

ANL Dx,A — Logical AND derivative register with accumulator

Encoding	1 0 0 0 | 1 1 1 0	a_7 a_6 a_5 a_4 | a_3 a_2 a_1 a_0	8EH
Description	This is a 2-byte, 2-cycle instruction. Data in the derivative register addressed by the second byte is logically ANDed with the accumulator contents.		
Operation	(Dx) ← (Dx) AND (A)		
Example	ANL D7,A	'AND' CONTENTS OF DERIVATIVE REGISTER D7 WITH ACC	

DEC @Rr — Decrement data memory location

Encoding	1 1 0 0 | 0 0 0 r		C0H-C1H
Description	The contents of the data memory location addressed by working register 'r' are decremented by one.		
Operation	((Rr)) ←((Rr)) –1	r = 0-1	
Example	MOV R1,#4FH DEC @R1	MOVE '4F' HEX TO REG 1 DECREMENT LOCATION 4F	

DIS SI — Disable serial input/output interrupt

Encoding	1 0 0 1 | 0 1 0 1	95H
Description	Serial input/output interrupts are disabled. An interrupt request from the SI0 has no effect. If the microcontroller enters the IDLE mode when the SI0 interrupt is disabled, it can't be restarted by this interrupt.	
Operation	(SIFF) ← 0	

DJNZ @Rr,addr — Decrement data memory location and test

Encoding

1 1 1 0	0 0 0 r	a_7 a_6 a_5 a_4	a_3 a_2 a_1 a_0	E0H-E1H

Description

This is a 2-cycle instruction. The memory location addressed by the working register 'r' is decremented, then tested for zero. If the memory location contains all zeros, program control passes to the next instruction. If the register contents are not zero, control jumps to the specified 'address'.

The address in this case must evaluate to 8 bits (the jump must be to a location within the current 256-location page).

Operation

((Rr)) ← ((Rr)) –1 r = 0-1
If ((Rr)) ≠ 0
then (PC_{0-7}) ← addr
else (PC) ← (PC) + 2

Example

Increment values in data memory locations 81-86.

```
        MOV R0,#32      MOVE '32' DEC TO ADDRESS REG 0
        MOV @R0,#6      MOVE '6' DEC TO COUNTER REG 32
        MOV R1,#81      MOVE '81' DEC TO ADDRESS REG 1
INCRT   INC @R1         INCREMENT CONTENTS OF MEMORY
                        LOCATION ADDRESSED BY REG 1
        INC R1          INCREMENT ADDRESS IN REG 1
        DJNZ @R0,INCRT  DECREMENT LOC. 32 - JUMP TO 'INCRT' IF
                        LOCATION 32 NONZERO
        NEXT ---        'NEXT' ROUTINE EXECUTED IF LOCATION 32 IS
                        ZERO
```

EN SI — Enable serial input/output interrupt

Encoding

1 0 0 0	0 1 0 1	85H

Description

Serial input/output interrupts are enabled. An interrupt request from the serial I/O initiates the interrupt sequence, in normal mode or in Idle mode.

Operation

(SIFF) ← 1

IDLE — Select IDLE operation

Encoding: 0 0 0 0 | 0 0 0 1 — 01H

Description: This instruction places the microcontroller in the Idle mode. The oscillator, timer/counter and serial I/O continue to function. The Idle mode is terminated:
- when an enabled interrupt is activated, or
- by a RESET.

An active signal on the RESET pin always restarts the microcontroller and a normal RESET sequence is executed.

An active signal from an interrupt source (if enabled) causes the execution of the normal interrupt routine since normal interrupt scanning is still being carried out. A HIGH-to-LOW transition on the external interrupt pin reactivates the microcontroller.

JNTF address — Jump if timer flag is not set

Encoding: 0 0 0 0 | 0 1 1 0 — a_7 a_6 a_5 a_4 | a_3 a_2 a_1 a_0 — 06H

Description: This is a 2-cycle instruction. Control passes to the specified address if the timer flag is not set, that is, if the timer/counter has not overflowed. Otherwise, program control passes to the next instruction. Testing the timer flag resets it to zero.

Operation:

$(PC_{0-7}) \leftarrow$ addr	If TF = 0
$(PC) \leftarrow (PC) + 2$	If TF = 1

Example:

JNTF NOTIM	JUMP TO 'NOTIM' ROUTINE IF THE TIMER HAS NOT OVERFLOWED

Note: If this instruction begins in location 255 of a page, the target address is located in the following page.

MOV A,Dx — Move derivative register contents to accumulator

Encoding: 1 0 0 0 | 1 1 0 0 — a_7 a_6 a_5 a_4 | a_3 a_2 a_1 a_0 — 8CH

Description: This is a 2-cycle instruction. The contents of the derivative register (x = 0 to 255) addressed by the second byte are moved to the accumulator.

Operation:

$(A) \leftarrow (Dx)$	x = 0 to 255

Example:

MOV A,D5	MOVE THE CONTENTS OF DERIVATIVE REGISTER 5 TO THE ACCUMULATOR

MOV A,Sn — Move serial I/O register contents to accumulator

Encoding | 0 0 0 0 | 1 1 0 n | 0CH, 0DH

Description: This is a 2-cycle instruction. The contents of serial I/O register 'n' are moved to the accumulator.

Operation: (A) ← (Sn) — n = 0-1

Example: MOV A,S0 — MOVE CONTENTS OF SERIAL I/O DATA REGISTER TO ACC

MOV Dx,A — Move accumulator contents to derivative register

Encoding | 1 0 0 0 | 1 1 0 1 | a_7 a_6 a_5 a_4 | a_3 a_2 a_1 a_0 | 8DH

Description: This is a 2-cycle instruction. The contents of the accumulator are moved to the derivative register addressed by the second byte.

Operation: (Dx) ← (A) — x = 0 to 255

Example: MOV D2,A — MOVE CONTENTS OF ACC TO DERIVATIVE REGISTER 2

MOV Sn,A — Move accumulator contents to serial I/O register

Encoding | 0 0 1 1 | 1 1 n n | 3CH-3EH

Description: This is a 2-cycle instruction. The contents of the accumulator are moved to serial I/O register 'n'.

Operation: (Sn) ← A — n = 0-2

Example: MOV S0,A — MOVE CONTENTS OF ACC TO SIO DATA REGISTER

MOV Sn,#data — Move immediate data to serial I/O register

Encoding	1 0 0 1 | 1 1 n n	d_7 d_6 d_5 d_4 | d_3 d_2 d_1 d_0	9CH-9EH

Description: This is a 2-cycle instruction. The byte specified by 'data' is moved to the serial I/O register 'n'.

Operation: (Sn) ← data — n = 0-2

Example: MOV S2,#28H — LOAD SIO CLOCK REGISTER

ORL Dx,A — Logical OR derivative register contents with accumulator

Encoding	1 0 0 0 | 1 1 1 1	a_7 a_6 a_5 a_4 | a_3 a_2 a_1 a_0	8FH

Description: This is a 2-cycle instruction. The contents of the accumulator are logically ORed with the contents of the derivative register addressed by the second byte, and the result placed in the same derivative register.

Operation: (Dx) ← (Dx) OR (A) — x = 0-255

SEL MB2 — Select memory bank 2

Encoding	1 0 1 0 | 0 1 0 1	A5H

Description: Only devices with more than 4 K bytes of program memory use this instruction. PC bit 11 is set to zero and PC bit 12 is set to 1 on the next JMP or CALL instruction. All references to program memory addresses fall within the range 4092-6143.

Operation: (MBFF1,0) ← (1,0)

SEL MB3 — Select memory bank 3

Encoding	1 0 1 1 | 0 1 0 1	B5H

Description: Only devices with more than 6 K bytes of program memory use this instruction. PC bits 11 and 12 are set to 1 on the next JMP or CALL instruction. All references to program memory addresses fall within the range 6144-8191.

Operation: (MBFF1,0) ← 1,1

STOP Stop processor

Encoding

0 0 1 0	0 0 1 0

22H

Description

This instruction places the microcontroller in the Stop mode. The oscillator is switched off; the internal status of the CPU, RAM contents and the state of I/O ports are not affected. The Stop mode may be terminated either by an active signal at the external interrupt or by an external RESET signal. When the Stop mode is terminated, an internal delay is provided to ensure that, before restarting, all internal clocks are working correctly.

If the Stop mode is terminated with a RESET, a normal RESET sequence is executed.

If the Stop mode is terminated by pulling the external interrupt pin LOW, an interrupt sequence is only executed if the external interrupt has been enabled and the microcontroller resumes the normal program sequence after returning from the interrupt routine, as in the normal mode. If the external interrupt is not enabled, the microcontroller continues the normal program sequence, executing the instruction following the Stop instruction in the main program.

Note: The microcontroller is restarted by a LOW level applied at the $\overline{\text{INT}}$/TO pin, and not by a HIGH-to-LOW transition (the normal interrupt mechanism). If the $\overline{\text{INT}}$/TO pin is LOW during the STOP instruction then the STOP instruction will be ignored.

ADDITIONAL 33XX INSTRUCTIONS

This section describes the 33XX instructions which are not included in the common section; it should be used in conjunction with the common section.

ANL Dx,A — Logical AND derivative register with accumulator

Encoding	1 0 0 0 | 1 1 1 0	a_7 a_6 a_5 a_4 | a_3 a_2 a_1 a_0	8EH
Description	This is a 2-byte, 2-cycle instruction. Data in the derivative register addressed by the second byte is logically ANDed with the accumulator contents.		
Operation	(Dx) ← (Dx) AND (A)		
Example	ANL D7,A	'AND' CONTENTS OF DERIVATIVE REGISTER D7 WITH ACC	

DEC @Rr — Decrement data memory location

Encoding	1 1 0 0 | 0 0 0 r		C0H-C1H
Description	The contents of the data memory location addressed by working register 'r' are decremented by one.		
Operation	((Rr)) ←((Rr)) –1	r = 0-1	
Example	MOV R1,#4FH DEC @R1	MOVE '4F' HEX TO REG 1 DECREMENT LOCATION 4F	

DIS SI — Disable serial input/output interrupt

Encoding	1 0 0 1 | 0 1 0 1		95H
Description	Serial input/output interrupts are disabled. An interrupt request from the SI0 has no effect. If the microcontroller enters the IDLE mode when the SI0 interrupt is disabled, it can't be restarted by this interrupt.		
Operation	(SIFF) ← 0		

DJNZ @Rr,addr — Decrement data memory location and test

Encoding

1 1 1 0	0 0 0 r	a_7 a_6 a_5 a_4	a_3 a_2 a_1 a_0	E0H-E1H

Description

This is a 2-cycle instruction. The memory location addressed by the working register 'r' is decremented, then tested for zero. If the memory location contains all zeros, program control passes to the next instruction. If the register contents are not zero, control jumps to the specified 'address'.

The address in this case must evaluate to 8 bits (the jump must be to a location within the current 256-location page).

Operation

((Rr)) ← ((Rr)) –1 r = 0-1
If ((Rr)) ≠ 0
then (PC_{0-7}) ← addr
else (PC) ← (PC) + 2

Example

Increment values in data memory locations 81-86.

```
        MOV R0,#32          MOVE '32' DEC TO ADDRESS REG 0
        MOV @R0,#6          MOVE '6' DEC TO COUNTER REG 32
        MOV R1,#81          MOVE '81' DEC TO ADDRESS REG 1
INCRT   INC @R1             INCREMENT CONTENTS OF MEMORY
                            LOCATION ADDRESSED BY REG 1
        INC R1              INCREMENT ADDRESS IN REG 1
        DJNZ @R0,INCRT      DECREMENT LOC. 32 - JUMP TO 'INCRT' IF
                            LOCATION 32 NONZERO
        NEXT ---            'NEXT' ROUTINE EXECUTED IF LOCATION 32 IS
                            ZERO
```

EN SI — Enable serial input/output interrupt

Encoding

1 0 0 0	0 1 0 1	85H

Description

Serial input/output interrupts are enabled. An interrupt request from the serial I/O initiates the interrupt sequence, in normal mode or in Idle mode.

Operation

(SIFF) ← 1

IDLE — Select IDLE operation

Encoding | 0 0 0 0 | 0 0 0 1 | 01H

Description

This instruction places the microcontroller in the Idle mode. The oscillator, timer/counter and serial I/O continue to function. The Idle mode is terminated:
- when an enabled interrupt is activated, or
- by a RESET.

An active signal on the RESET pin always restarts the microcontroller and a normal RESET sequence is executed.

An active signal from an interrupt source (if enabled) causes the execution of the normal interrupt routine since normal interrupt scanning is still being carried out. A LOW-to-HIGH transition on the external interrupt pin (CE pin) reactivates the microcontroller.

JNTF address — Jump if timer flag is not set

Encoding | 0 0 0 0 | 0 1 1 0 | a_7 a_6 a_5 a_4 | a_3 a_2 a_1 a_0 | 06H

Description

This is a 2-cycle instruction. Control passes to the specified address if the timer flag is not set, that is, if the timer/counter has not overflowed. Otherwise, program control passes to the next instruction. Testing the timer flag resets it to zero.

Operation		
	$(PC_{0-7}) \leftarrow$ addr	If TF = 0
	(PC) $\leftarrow$ (PC) + 2	If TF = 1

Example		
	JNTF NOTIM	JUMP TO 'NOTIM' ROUTINE IF THE TIMER HAS NOT OVERFLOWED

Note: If this instruction begins in location 255 of a page, the target address is located in the following page.

MOV A,Dx — Move derivative register contents to accumulator

Encoding | 1 0 0 0 | 1 1 0 0 | a_7 a_6 a_5 a_4 | a_3 a_2 a_1 a_0 | 8CH

Description

This is a 2-cycle instruction. The contents of the derivative register (x = 0 to 255) addressed by the second byte are moved to the accumulator.

Operation		
	(A) $\leftarrow$ (Dx)	x = 0 to 225

Example		
	MOV A,D5	MOVE THE CONTENTS OF DERIVATIVE REGISTER 5 TO THE ACCUMULATOR

MOV A,Sn — Move serial I/O register contents to accumulator

Encoding	0 0 0 0 | 1 1 0 n	0CH, 0DH
Description	This is a 2-cycle instruction. The contents of serial I/O register 'n' are moved to the accumulator.	
Operation	(A) ← (Sn)	n = 0-1
Example	MOV A,S0	MOVE CONTENTS OF SERIAL I/O DATA REGISTER TO ACC

MOV Dx,A — Move accumulator contents to derivative register

Encoding	1 0 0 0 | 1 1 0 1	a_7 a_6 a_5 a_4 | a_3 a_2 a_1 a_0	8DH
Description	This is a 2-cycle instruction. The contents of the accumulator are moved to the derivative register addressed by the second byte.		
Operation	(Dx) ← (A)	x = 0 to 255	
Example	MOV D2,A	MOVE CONTENTS OF ACC TO DERIVATIVE REGISTER 2	

MOV Sn,A — Move accumulator contents to serial I/O register

Encoding	0 0 1 1 | 1 1 n n	3CH-3EH
Description	This is a 2-cycle instruction. The contents of the accumulator are moved to serial I/O register 'n'.	
Operation	(Sn) ← A	n = 0-2
Example	MOV S0,A	MOVE CONTENTS OF ACC TO SIO DATA REGISTER

MOV Sn,#data — Move immediate data to serial I/O register

Encoding	1 0 0 1 | 1 1 n n	d_7 d_6 d_5 d_4 | d_3 d_2 d_1 d_0	9CH-9EH
Description	This is a 2-cycle instruction. The byte specified by 'data' is moved to the serial I/O register 'n'.		
Operation	(Sn) ← data	n = 0-2	
Example	MOV S2,#28H	LOAD SIO CLOCK REGISTER	

ORL Dx,A — Logical OR derivative register contents with accumulator

Encoding	1 0 0 0 | 1 1 1 1	a_7 a_6 a_5 a_4 | a_3 a_2 a_1 a_0	8FH
Description	This is a 2-cycle instruction. The contents of the accumulator are logically ORed with the contents of the derivative register addressed by the second byte, and the result placed in the same derivative register.		
Operation	(Dx) ← (Dx) OR (A)	x = 0-255	

SEL MB2 — Select memory bank 2

Encoding	1 0 1 0 | 0 1 0 1	A5H
Description	Only devices with more than 4 K bytes of program memory use this instruction. PC bit 11 is set to zero and PC bit 12 is set to 1 on the next JMP or CALL instruction. All references to program memory addresses fall within the range 4092-6143.	
Operation	(MBFF1,0) ← (1,0)	

SEL MB3 — Select memory bank 3

Encoding	1 0 1 1 | 0 1 0 1	B5H
Description	Only devices with more than 6 K bytes of program memory use this instruction. PC bits 11 and 12 are set to 1 on the next JMP or CALL instruction. All references to program memory addresses fall within the range 6144-8191.	
Operation	(MBFF1,0) ← 1,1	

STOP — Stop processor

Encoding

0 0 1 0	0 0 1 0

22H

Description

This instruction places the microcontroller in the Stop mode. The oscillator is switched off; the internal status of the CPU, RAM contents and the state of I/O ports are not affected. The Stop mode may be terminated either by an active signal at the external interrupt or by an external RESET signal. When the Stop mode is terminated, an internal delay is provided to ensure that before restarting, all internal clocks are working correctly.

If the Stop mode is terminated via a RESET, a normal RESET sequence is executed.

If the Stop mode is terminated by pulling the CE pin HIGH, an interrupt sequence is only executed if the external interrupt has been enabled and the microcontroller resumes the normal program sequence after returning from the interrupt routine, as in the normal mode. If the external interrupt is not enabled, the microcontroller continues the normal program sequence, executing the instruction following the Stop instruction in the main program.

Note: The microcontroller is restarted by a HIGH level applied at the CE/$\overline{TO}$ pin, and not by a LOW-to-HIGH transition (the normal interrupt mechanism). If the CE/$\overline{TO}$ pin is HIGH during the STOP instruction then the STOP instruction will be ignored.

Section 7 - 84XX/84CXX/33XX Serial I/O and Software Examples

CONTENTS – 84XX/84CXX/33XX Serial I/O

1.0 INTRODUCTION

The increasing demand for microcontrollers in domestic and industrial applications has led to the development of the MAB84X1 family. More widespread use of distributed intelligence in today's controller sytems, has demanded the introduction of simpler, more efficient data exchange. To the MAB84X1 family this has meant the addition of a serial communication interface (SIO). The MAB8422/42 has not the full SIO hardware on-chip, but can easily handle the I^2C-bus with software.

Normally, serial data transfer imposes a heavy processing load on microcontrollers, with the serial data bus having to be regularly monitored for information. To overcome this problem the Serial I/O (SIO) interface was introduced. The MAB84X1 (SIO) interface can detect, receive and convert the serial data stream to parallel format without interrupting current program execution.

The use of serial data communication between devices reduces the number of necessary connections, leading to simpler circuit layouts and economies in both connector size and circuit board area. A hardware serial I/O allows the interconnection of any number of devices by the two-wire serial bus.

To demonstrate how serial communication works, consider two microcontrollers connected via an I^2C (Inter-IC) bus to a serial memory and a display (Fig. 1). In a typical system such as this, the device which controls the transfer of messages is called a 'master', the device(s) controlled by the master is named a 'slave'. So, in our example, the master would be either of the two microcontrollers, with the serial memory and display acting as slaves. The master device generates the timing signals for data transmission. Only one master can control the I^2C-bus at any one time. The controlling device can act as either a master transmitter or master receiver. The slave, under control of the master, can also transmit or receive.

An important feature of the I^2C-bus is its true multi-master capabilities, which means that more than one master can try to control the bus at the same time without risking data clash. During transmission, an arbitration procedure decides priority. Therefore there is no central master in the bus structure and any master transceiver, if it loses the arbitration, can be addressed as a slave by the elected master.

To explain what is meant by slave transmitters and receivers, master receivers and transmitters a definition is given below.

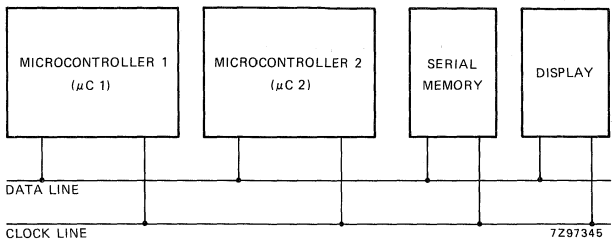
Transmitter	- The device which sends data to the bus
Receiver	- The device which receives data from the bus
Master	- The device which initiates a transfer, generates clock signals and terminates a transfer
Slave	- The device addressed by a master
Multi-master	- More than one master can attempt to control the bus at the same time without corrupting the message

Arbitration - Procedure to ensure that if more than one master simultaneously tries to control the bus, only one is allowed to do so and there is no data clash.

If either of the microcontrollers is sending information to the serial memory then the memory is a 'slave receiver' and the microcontroller is the 'master transmitter'.

If a microcontroller is receiving information from the serial memory then the memory is a 'slave transmitter' and the microcontroller is the 'master receiver'.

Of course, if after arbitration the losing microcontroller is addressed by the winner, then the loser is considered a 'slave transmitter' or 'slave receiver' and the winner, a 'master transmitter' or 'master receiver'.



	uC 1	uC 2	Memory	Display
Master Tx	0	X	/	/
Slave Rx	X	0	0	0
			X	X
Master Rx	0	X	/	/
Slave Tx	X	0	0 X	/
Slave Rx			0	0
			X	X

X = First instance

0 = Second instance

NOTE: Display can never become a transmitter

Fig. 1 Example of typical serial I/O connection.

2.0 SERIAL INPUT/OUTPUT

The ability of any two devices to communicate, without interruption to any other devices tied to the bus, is an outstanding attribute of the serial I/O system. Communication is achieved by allocating a specific 7-bit address to each device and providing a system whereby a device reacts only to messages prefixed with its own address or the 'general call' address (00 H). Address recognition is performed by the interface hardware so that operation of the microcontroller need only be interrupted when a valid address nas been received. This results in a significant saving of processing time and memory space compared with a conventional microcontroller employing a software serial interface. The addressing facilitiy is not required, for instance in a system with only two microcontrollers, direct data transfer without addressing can be performed i.e. the receiving device reacts after each received byte from the serial bus (see free data format Chapter 6).

3.0 SERIAL BUS STRUCTURE

The serial data (SDA) and serial clock (SCL) lines are both bidirectional. Each is connected to a positive supply voltage via a pull-up resistor (see Fig. 2). When the bus is free, both lines are HIGH. The output stages of peripheral IC's connected to the bus must have an open drain or open collector to perform the wired-AND function. The number of ICs that can be connected to the bus, and its length, are solely limited by the maximum bus capacitance of 400 pF.

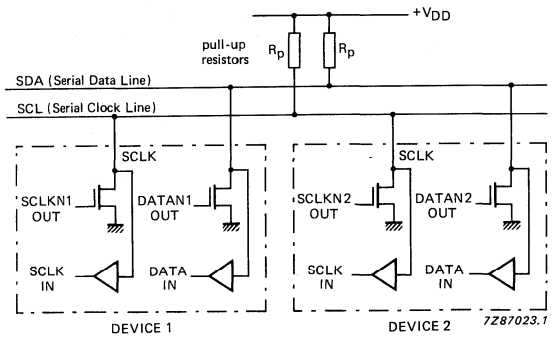


Fig. 2 Connection of two microcontrollers to the serial bus.

R_p = 1 to 5 ko (depending upon current required)

Each microcontroller can be software programmed to function as a 'transmitter' or a 'receiver' operating in either a 'master' or a 'slave' mode.

4.0 SERIAL DATA TRANSFER

Fig. 3 shows the serial data transfer sequence for the SIO interface. The following conditions can be distinguished:

F (free): The bus is free; data line SDA and clock line SCL are both HIGH

S (start): A data transfer commences with a 'start' condition during which the level of data line SDA changes from HIGH to LOW, and clock line SCL remains HIGH. The bus is now busy.

C (change): During the LOW perion of the clock, the data bit to be transmitted is applied to data line SDA. The level on the SDA line may therefore change during this period

D (data): One data bit is transmitted during the HIGH period of the clock. As SDA line is sampled on an SCL HIGH pulse, the level (bit state) on line SDA must remain stable during this period to prevent it being interpreted as a 'start' or 'stop' condition

P (stop): A data transfer concludes with a 'stop' condition during which the level on data line SDA changes from LOW to HIGH and clock line SCL remains HIGH. The bus is now free again.

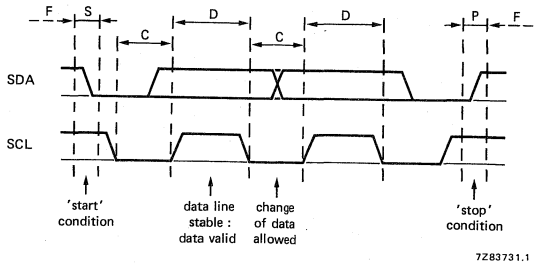


Fig. 3 Sequence of bit transfer on the serial bus.

Every byte transferred on the SDA line must contain 8-bits. The number of bytes that can be transmitted per transfer is unrestricted. Each byte must be followed by an acknowledge bit. If a receiving device cannot receive another complete byte of data until it has performed some other function, for example serviced an internal interrupt, it may hold the clock line SCL LOW to force the transmitter into a wait state. Transfer then continues when the receiver is ready for another byte and releases clock line SCL.

5.0 SERIAL DATA FORMATS

The data transfer format is shown in figure 4. After the start condition, a 7-bit slave address is transmitted which is follwed by a data direction bit (R/W); a '0' indicates a write action, a '1' indicates a read. A data transfer is always terminated by a stop condition generated by the master. However, if a master still wishes to communicate on the bus, it can generate another start condition, and address a further slave without first generating a stop condition. Various combinations of read/write formats are then possible within such a transfer. Figures 5(a), (b), (c) and (d) illustrate the four basic types of transfer formats used:

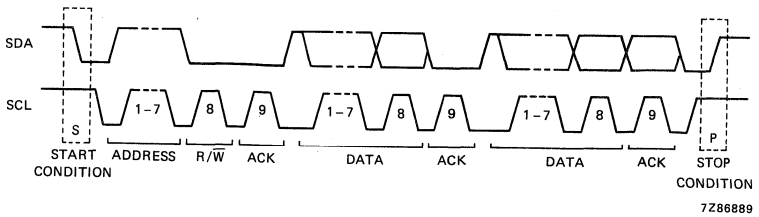


Fig. 4 A complete data transfer.

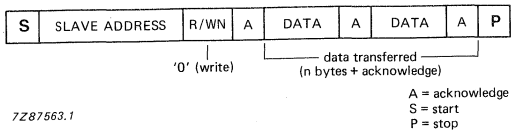


Fig. 5(a) Master transmitter transmits to slave, direction unchanged.

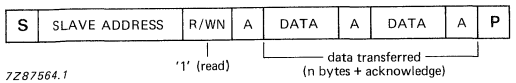


Fig. 5(b) Master reads slave immediatly after first byte.

At the moment of the first acknowledge, the master transmitter becomes a master receiver and the slave receiver becomes a slave transmitter. This acknowledge is still generated by the slave. The stop condition is generated by the master.

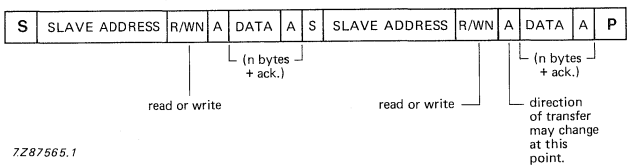


Fig. 5(c) Combined formats.

During a change of direction within a transfer, the start condition and slave address are both repeated, but with the R/W bit reversed (See ALS='0').

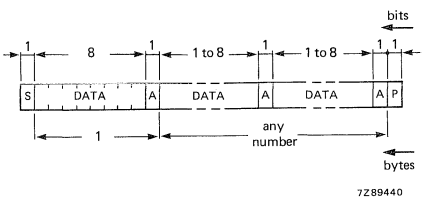


Fig. 5(d) Free data format.

In the free data format, the direction of transmission remains constant throughout the data transfer (See ALS ='1').

In the first three formats (Figs. 5(a), (b) and (c)), byte SLA is the address of the target slave to be selected by the master. The least significant bit of the address byte indicates the direction of transmission of the following byte(s).

If R/W is 0 the master will write (transmit) data to the selected slave; if it is set to 1 the master will read data from the slave. In the last case the direction will change after transmission of the first byte (Fig. 5(c)).

6.0 OPERATING MODES OF THE SERIAL I/O INTERFACE

The serial I/O functions in four basic operating modes to service all facilities of the I^2C (Inter IC) bus, P/C bus or point to point connections:

- 'master transmitter'
- 'master receiver'
- 'slave receiver'
- 'slave transmitter'

6.1 Master transmitter

In this mode, data assembled in one of the previously described data formats is shifted-out on the serial data line synchronous with the clock pulses on SCL. The clock pulses are inhibited and SCL held LOW when the intervention of the processor is required (see address format (a) or (c)).

6.2 Master receiver

This mode can only be entered from the 'master transmitter' mode. Using the address formats ((b) or (c)), the 'master receiver' mode is entered after the transmission of address byte SLA with the R/W bit set to 1. Serial data bits received on bus line SDA by a 'master receiver', are shifted-in synchronized with the self-generated clock pulses on SCL. Again clock pulses are inhibited and SCL held LOW when the intervention of the processor is required after reception of a byte. At the end of a transfer, the 'master receiver' generates the 'stop' condition P.

6.3 Slave receiver

Serial data bits received on bus line SDA by a 'slave receiver', are shifted-in synchronized with the clock pulses at SCL generated by the 'master' device. A 'slave receiver' does not generate clock pulses. The 'slave receiver' holds clock line SCL LOW whilst intervention of the processor is required following the reception of a byte. 'Start' and 'stop' conditions are recognized and interpreted accordingly.

Note; When a slave receiver does not acknowledge its address, for example if it is not able to receive because it is performing some real-time function, the data line (SDA) is left HIGH by the slave. The master can then generate a stop condition to abort transfer.

If a slave receiver acknowledges its slave address, but later in the transfer cannot receive further data, the master must again abort the transfer. This is indicated by the slave not acknowledgeing a data byte by leaving the data line HIGH. The master then generates a stop condition.

6.4 Slave transmitter

The 'slave transmitter' mode can only be entered from the 'slave receiver' mode. Using either of the address formats ((b) or (c)), the 'slave transmitter' mode is entered when the address received in byte SLA matches its own slave address and the R/W bit is a 1. A 'slave transmitter' shifts out the serial data on data line SLA, in synchrony with the clock pulses generated on clock line SCL by the 'master' device. A 'slave transmitter' does not generate clock pulses. The 'slave transmitter' holds clock line SCL LOW if invervention of the processor is required after transmission of a byte. 'Start' and 'stop' conditions are recognized and interpreted accordingly.

7.0 SERIAL I/O INTERFACE

A block diagram of the SIO interface is shown in figure 6. The clock line of the serial bus has exclusive use of pin 3, while the data line shares pin 2 with I/O signal P23 of port 2. Consequently, only three I/O lines are available for Port 2 when the SIO interface is enabled. When the SIO interface is enabled, P23 is disabled as a parallel port line (P23 as SDA open drain only).

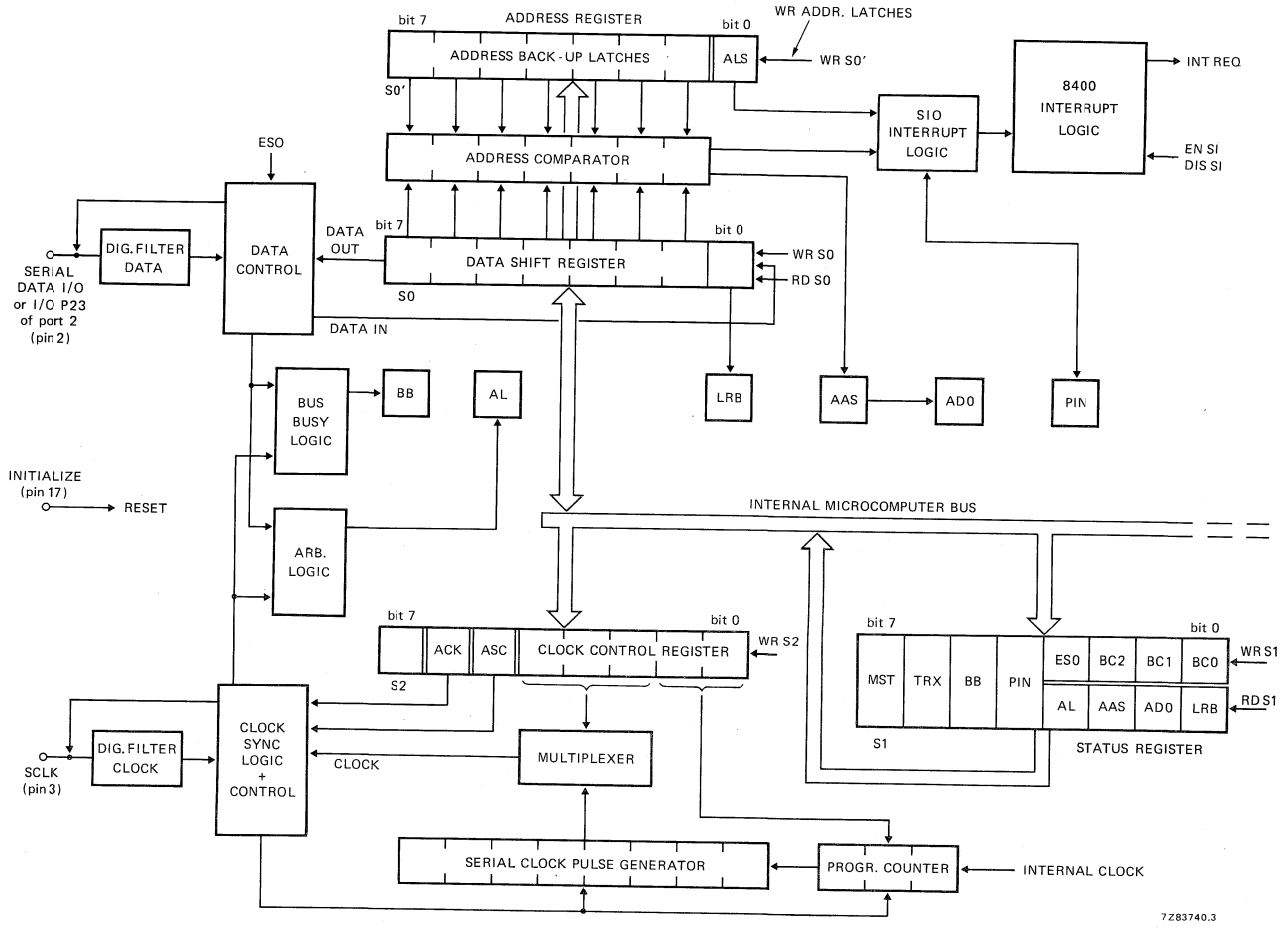


Fig. 6 Block diagram of the MAB84X1 SIO interface.

Communication between the microcontroller and interface takes place via the internal bus of the microcontroller and the Serial Interrupt Request line. Four registers are used to store data and information controlling the operation of the interface.

- data shift register S0
- address register S0'
- status register S1
- clock control register S2.

Eight instructions control operation on these four registers enabling data to be written into or read via the internal microcontroller bus. All bits with the exception of PIN (pending interrupt not) in the status register are cleared to 0 by a RESET. PIN is set to 1 by a RESET. The address comparator shown in figure 5, can instruct the interrupt logic block (if enabled) to generate an SIO interrupt request to the microcontroller.

Communications with the four registers and the function of the address comparator will now be described in more detail.

7.1 Data shift register SO

SO is the data shift register used to perform the conversion between serial and parallel data format. Data to be transmitted is loaded in parallel into SO and shifted out serially, most significant bit first. Data received on the serial bus is shifted into SO, most significant bit first. After transmission orreception of a complete data byte, the specific address or the general address, a pending interrupt is generated to the microcontroller.

To address this 8-bit data shift register, the ESO (enable serial output) bit in status register S1 must be be set to 1. The write instruction MOV SO,A or MOV SO,# data, causes data shift register SO to be parallel-loaded via the internal bus with the data for transmission. During the transmission, the contents of the register are shifted out, most significant bit first, on to data line SDA.

During reception, the serial data is shifted into the data shift register SO. The last received data bit present on data line SDA is shifted into the least-significant bit position in the data shift register. While in the acknowledgement mode, the acknowledgement bit is not shifted into register SO but into the LRB (last received bit) position of S1. The read instruction MOV A,SO causes the contents of register SO to be parallel-loaded into the accumulator via the internal bus.

7.2 Address register SO

The address register contains the 7-bit address back-up latches and the ALS flag, as shown in Fig. 7. The address latches hold the address allocated to the slave device, while the ALS (Always Selected) flag is used to enable/disable the address recognition mode.

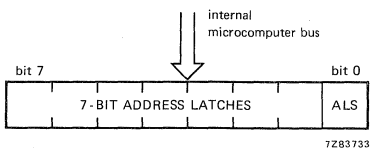


Fig. 7 Address register in the SIO interface.

ALS = 1, The address recognition has been disabled. The SIO interface will respond to the free data formats (Fig 5 (d)), i.e. it reacts after each received byte from the serial bus. This mode may be used when only two devices or memory peripherals share the bus.

ALS = 0 The address recognition has been enabled. The SIO interface will respond to the addressing format i.e. it reacts only to messages containing its own slave address or general address (00 H). In this mode, the least significant bit of the first byte received performs the function of read/write command (write = 0).

The address register can only be loaded from the internal bus of the microcontroller when the SIO interface is disabled (ESO = 0). Under this condition, a MOV SO,A instruction will move the contents of the accumulator to the address register instead of SO, as would normally be the case.

7.3 Status register S1

The status word in status register S1 is shown in figure 8, it contains all information regarding the status of the SIO interface. Read instruction MOV A,S1 causes the status word to be read via the internal bus. To control the SIO interface, information is written into the status register with the write instruction MOV S1,A or MOV S1,# data. Two latches at each of the four least significant bit positions in the status register, enable two separate status bits to be stored at each position. The four least-significant bits ESO, BC2, BC1, and BCO are write only control bits. The four low order bits AL, AAS and LRB are read only status bits. The four high order bits MST, TRX, BB and PIN which can be both written and read. The function of each bit in the status register will now be described.

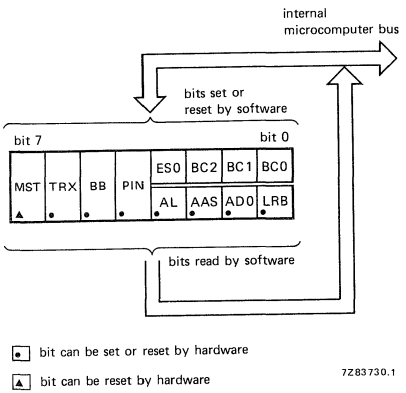


Fig. 8 Bit allocation of the status word in register S1.

MST: 'Master' bit. Setting MST = 1 places the SIO in the 'master' mode and generates the pulses on clock line SCL for timing the transmission or reception of the serial data. Setting MST = 0 places the SIO in the 'slave' mode and the clock pulses are received fram the 'master' device on clock line SCL. The MST bit is reset to 0 by the hardware if an arbitration of this master device is lost. When a data transfer has been completed, a 'stop' condition is generated by the 'master' device and the hardware resets the MST bit of this master device to 0.

TRX: 'Transmitter' bit. Setting TRX = 1 places the SIO interface in the 'transmitter' mode and the data in register SO is shifted out on data line SDA in synchrony with the pulses on clock line SCL. If TRX = 0, the SIO interface is in the 'receiver' mode and the data on bus line SDA is shifted into data register SO in synchrony with the pulses on clock line SCL. The TRX bit is reset to 0 by hardware if an arbitration is lost. When a data transfer has been completed, a 'stop' condition is generated by the 'master' device and the hardware resets the TRX bit of this master device to 0. In either of the addressing formats (Fig. 5(a),(b) or (c)), The TRX bit is set by the hardware in accordance with the read/write direction bit R/W. The operating modes set by the MST and TRX bits of the status word are summarized in Table 1.

Table 1 Operating modes set by bits MST and TRX

MST	TRX	operating code
0	0	'slave receiver'
1	0	'master receiver'
0	1	'slave transmitter'
1	1	'master transmitter'

BB: 'bus busy' bit. This bit indicates the state of the serial bus. If it is 0, the bus is free. If it is 1, the bus is occupied. On reception of a 'start' condition, the bus busy logic sets BB to 1. BB is reset to 0 one LOW period of the internal serial clock after reception of a 'stop' condition. In the 'master' mode, BB is controlled by software. To start a transmission with a 'start' condition, bits MST, TRX and BB of the status word are set to 1. To finish a transmission with a 'stop' condition, bit BB is reset to 0 and bits MST and TRX are set to 1.

PIN: 'pending interrupt not' bit. Setting this bit to 0 initiates a serial I/O interrupt only if the EN SI (enable serial interrupt) instruction has been previously executed. As long as PIN is 0, the clock pulses are inhibited and clock line SCL is held LOW. The PIN bit is reset to zero only when:

- A complete byte has been transmitted (even if abitration has been lost).
- The address comparator has recognized its own slave address, or the general address of all 8 zeros has been received (ALS flag = 0).
- Another byte has been received following a previous recognition
- A byte has been received in the free data format shown in Fig. 5 (d).

The PIN bit is reset to 1 by:

- Read instruction MOV A,S0 or write instruction MOV S0,A or MOV S0,# data.
- Loading a 1 into the PIN bit position in status register S1 as a result of write instruction MOV S1,A or MOV S1,# data.

ESO: 'Enable serial output' bit. With ESO = 1, the SIO interface is enabled and can receive or transmit on serial data line SDA, in synchrony with the pulses on clock line SCL. When the ESO bit = 0, the SIO interface is disabled, pin 2 reverts to its P23 (I/O line of port 2) function and pin 3 (SCLK) is in the high impedance state. Whilst ESO is 0, a 7-bit slave address, plus ALS bit, can be loaded into address register SO' with write instruction MOV S0,A or MOV S0,# data. As long as ESO remains 0, PIN is held HIGH, AL is held LOW by hardware and the contents of data shift register S0 are not affected by a write instruction to S0.

BC2, BC1 and BC0: 'bit count'. As shown in Table 2, the binary-coded number preset in bit position BC2, BC1 and BC0 is the number or bits (excluding the acknowledge bit) of the next byte which are yet to be received or transmitted.

Table 2 Binary numbers in bit-count locations BC2, BC1 and BC0

BC2	BC1	BC0	bits/byte without ACK	bits/byte with ACK
0	0	1	1	2
0	1	0	2	3
0	1	1	3	4
1	0	0	4	5
1	0	1	5	6
1	1	0	6	7
1	1	1	7	8
0	0	0	8	9

A 'start' condition resets the bit count to 000, so the first byte to be received or transmitted will always consist of eight bits excluding the acknowledge bit. After each complete byte has been received or transmitted, the bit count has reduced to 000.

AL: 'Arbitration lost' bit. When in the 'master transmitter' mode, this bit is set to 1 when the arbitration logic senses the SIO interface has lost an arbitration. It is also set to 1 if an attempt is made to occupy the bus while BB is set to 1. After a byte has been transmitted, PIN is reset to 0 and the status word in register S1 indicates in which mode the SIO has been addressed. AL is reset to 0 by read instruction MOV A,S0 or any of the write instructions;

```
MOV S0,A
MOV S0,#data
MOV S1,A
MOV S1,#data
```

AAS: 'Addressed as slave' bit. AAS is set to 1 when the address comparator has recognized its own slave address or an address of all (8) zeros. AAS is also set to 1 if the first byte has been received in the free data format (ALS =1). The AAS bit is reset to 0 by the read instruction MOV A,S0 or write instruction MOV S0,A or MOV S0,# data. The device remains selected until the STOP condition.

ADO: 'Address zero' bit. This bit is set to 1 if the address comparator detects the address of all (8) zeros. The ADO bit is reset to 0 when a 'start' or 'stop' condition is detected.

LRB: 'Last received bit' If a byte is transmitted with an acknowledge (ACK) bit, the LRB bit position in status register S1 of the 'transmitter' contains the acknowledgement of the 'receiver'. When LRB is 0, reception of the transmission has been acknowledged. If a transmission does not include an acknowledgement bit, it is the last bit of the transmitted or received byte that will be in the LRB position.

7.4 Clock control register S2

This is an 8-bit write-only register with bit positions as shown in figure 9. The seven bits S20 to S26 can be parallel-loaded with the write instruction MOV S2,A or MOV S2,# data. The eighth bit position (S27) is not used. Bits S20 to S24 control the frequency of the clock pulses for the SIO interface. Bit S25 (ASC) determines whether the mark-space ratio of the clock pulses is 1:1 or 3:1. Bit S26 (ACK) determines whether or not the device is operating in the acknowledgement mode.

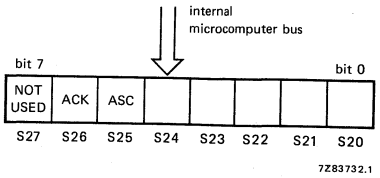


Fig. 9 Bit allocation in clock control register S2

S20 to S24: clock frequency control bits. These bits form a binary-coded frequency control number (0 to 31) which determines the factor by which the microcontroller clock frequency will be devided to obtain the required frequency of pulses on clock line SCL. Table 3 lists the divisors together with hexadecimal equivalents of the frequency control numbers, and shows the resulting frequency of the clock pulses on SCL if the microcontroller clock is controlled by a 4,43 or 6 MHz crystal.

ASC: 'Asymmetrical clock' bit at S25. If ASC = 1 the clock pulses on SCL have a mark-space ratio of 3:1 and the clock frequency control number may must be HEX'18', HEX'19', HEX'1A' or HEX'1B' which gives a clock-pulse frequency range of 1,4 kHz to 2,3 kHz. This is the low-speed mode of the SIO interface. For example, if the clock-frequency control number HEX'19' is in bit positions S20 to S24, the frequency of the pulses on clock line SCL will be about 1,9 kHz (1/f = 520 us for a clock input frequency of 4,4 MHz). Since ASC is 1, the mark/space ratio will be 3:1 so that the HIGH time of the clock pulses is 390 us and the LOW time is 130 us.

If ASC is cleared to 0, the pulses on clock line SCL have a mark-space ratio of 1:1 and any of the clock frequency control numbers, except 0, may be loaded into bit positions S20 to S24. The clock-pulse frequency may therefore be selected within the range 700 Hz to 114 kHz. This is the high-speed mode of the SIO interface.

Table 3 Clock pulse frequency control when using a 4,43 or 6 MHz crystal

HEX S20-S24 code	4,43 MHz crystal		6 MHz crystal
	divisor	approx. f_{clock}(kHz)	approx. f_{clock}(kHz)
0	NOT ALLOWED		NOT ALLOWED
1	39	114	154
2	45	98	133
3	51	87	118
4	63	70	95
5	75	59	80
6	87	51	70
7	99	45	61
8	123	36	49
9	147	30	41
A	171	26	35
B	195	23	31
C	243	18	25
D	291	15	21
E	339	13	18
F	387	11	16
10	483	9,2	12
11	579	7,7	10,4
12	675	6,6	8,9
13	771	5,8	7,8
14	963	4,6	6,2
15	1155	3,8	5,2
16	1347	3,3	4,5
17	1539	2,9	3,9
18*	1923	2,3	3,1
19*	2307	1,9	2,6
1A*	2691	1,7	2,2
1B*	3075	1,4	2,0
1C	3843	1,2	1,6
1D	4611	1,0	1,3
1E	5379	0,8	1,1
1F	6147	0,7	0,98

* Only values that may be used in the low-speed mode (ASC=1)

ACK: 'Acknowledge' bit at S26. For operation in the acknowledgement mode, the ACK bit in register S2 must be set to 1. The acknowledgement procedure is illustrated in figure 10. The operation of devices in each of the four SIO interface modes will now be described.

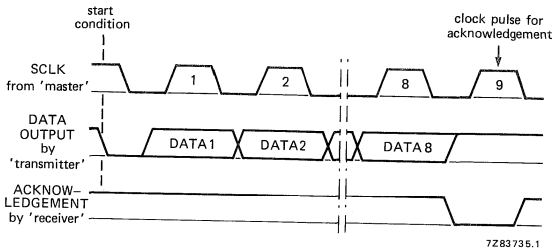


Fig. 10 Acknowledgement between 'receiver' and 'transmitter'.

If the ACK bit in register S2 of a 'master transmitter' is 1, the device generates an extra clock pulse on SCL at the end of each transmitted byte. During this clock pulse, the SERIAL DATA I/O (pin 2) becomes an input and, if a 'receiver' generates an acknowledgement (LOW level on data line SDA), it resets the LRB flag to 0. If acknowledgement is not received, the level on data line SDA remains HIGH setting the LRB flag to 1.

A 'slave transmitter' reacts to an acknowledgement in the same manner as a 'master transmitter' except it receives the extra clock pulse from the 'master' device instead of generating it.

If the ACK bit in clock control register S2 of a 'master receiver' is set to 1, the device generates an extra clock pulse after each received byte and applies it to clock line SCL. During this clock pulse, the SERIAL DATA I/O (pin 2) becomes an output where a LOW level is generated on data line SDA to acknowledge receipt of the transmission.

A 'slave receiver' generates an acknowledgement in the same manner as a 'master receiver' except that it receives the extra clock pulse from the 'master' device instead of generating it.

It should be noted that, if a device changes from a 'receiver' to a 'transmitter' after the R/$\overline{W}$ bit, an acknowledgement will be generated for the first byte by that device. If a device is changed from a 'transmitter' to a 'receiver' after the R/$\overline{W}$ bit, the acknowledgement for the first byte will be received by that device.

If the ACK bit position in clock control register S2 is loaded with a 0, The SIO interface is no longer in the acknowledgement mode. A 'master' device does not then generate an extra clock pulse after each byte and a 'receiver' does not generate an acknowledgement LOW level on bus line SDA.

8.0 SERIAL I/O OPERATIONS

8.1 Arbitration procedure

If two or more master transmitters start a transmission on the same bus almost simultaneously, arbitration procedure will be invoked. The arbitration procedure uses the data presented on the serial data line by the competing transmitters, hence it does not delay transfer of information. When a transmitter senses that a HIGH signal it has presented on the line has been overruled by a LOW signal, it switches to the slave receiver mode, sets the AL flag and generates a pending interrupt at the end of a byte. Figure 11 shows the arbitration procedure between two devices. As soon as the master generating DATA 1 generates an internal HIGH level whilst the data on the SDA line is LOW, it switches its output off so that the master generating DATA 2 wins the arbitration and its data stream continues on the SDA line. Should two or more devices send identical first bytes, arbitration will continue on the subsequent bytes. Since arbitrations are decided solely on the basis of addresses and data transmitted by competing masters, there is no central master, or any order of priority on the bus.

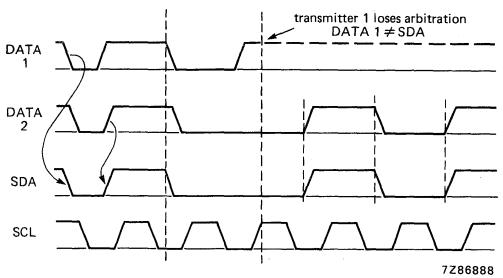


Fig. 11 Arbitration procedure between two master transmitters.

8.2 Clock synchronization

All masters generate their own clock on the SCL line during transfer of data. Data is only valid during the clock HIGH period. A defined clock is therefore needed to allow the bit-by-bit arbitration procedure to take place. Clock synchronization is performed using the wired-AND configuration of the bus. As shown in figure 12, a HIGH to LOW transition on the SCL line causes all competing devices to count off their LOW periods and the SCL line is held low until the device with the longest LOW period finishes its count. Devices with shorter LOW periods remain in a HIGH wait state during this time. When all devices concerned have counted off their LOW periods the SCL line is released and goes HIGH. There is then no difference between the state of the device clocks and the SCL line, and all devices start counting their HIGH periods. The first device to complete its HIGH period drags the SCL line LOW. In this way a synchronized SCL clock is generated, the LOW period of which is determined by the device with the longest LOW period and the HIGH period determined by the device with the shortest clock HIGH period.

If a device pulls down the clock line for a longer time, the result will be that all clock generators enter the WAIT state. In this way a slave can slow down a fast master and the slow device can create enough time to store a received byte, or to prepare a byte to be transmitted. An illustration of clock synchronization is given in figure 12.

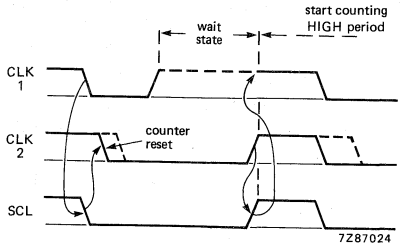


Fig. 12 Synchronization of two SIO clock generators.

9.0 PROGRAMMING

9.1 Machine state following RESET

A reset to pin 17 clears all SIO interface registers (S0, S0', S1 and S2) except for the PIN (pending interrupt not) bit in status register S1 which is set to 1. Because the ESO (enable serial output) bit in status register S1 is 0, the SIO interface is disabled, pin 2 functions as P23 of port 2 (HIGH after RESET) and pin 3 (SCLK) is in the high impedance state. An initialization procedure must therefore be carried out before the SIO interface can be used to transfer serial data.

9.2 Initialization procedure

A flow chart of the initialization procedure is given in Fig. 14. Each step of the procedure will now be explained in more detail.

9.2.1 Clock control register S2

The first step of the initialization procedure is to set the bit pattern in Register S2. Bit position S20 to S24 are loaded, as indicated in Table 3, to set the frequency of the pulses on clock line SCL. Bit position ASC and ACK are loaded as previously described to determine whether the SIO interface operates with a symmetrical or asymetrical clock, and whether or not it operates in the acknowledgement mode. After S2 has been loaded, the clock pulse generator requires one LOW period of its programmed clock cycle-time to stabilize. Since a data transfer must not be started during this stabilization period, the BB flag is set to indicate such an occurence. The BB flag is reset when the clock generator has stabilized.

9.2.2 Address register SO

The next step is to load address register SO'. To address this register, the SIO interface must remain disabled by leaving the ESO (enable serial output) bit in status register S1 at 0. Address register SO' is loaded with the 7-bit slave address plus the ALS bit. If the ALS bit is 1, the SIO will service free data format so the slave address in SO' is immaterial.

9.2.3 Status register S1

The final step of the initialization procedure is to change the ESO bit in status register S1 to 1. All the other bits in register S1 remain as they were following the RESET, i.e. all bits 0 except PIN which is 1. The status register therefore contains HEX'18' and the SIO interface is in the 'slave receiver' mode and ready to receive serial data.

9.2.4 Enable/disable serial I/O interrupt

The instruction EN SI must be performed to enable the serial interrupt logic. If the serial interrupt logic is not enabled, the SIO interface can be serviced by polling PIN.

9.2.5 Example of an initialization procedure

After a RESET and a jump to label INSI, initialization of the SIO interface can be achieved as follows.

```
INSI MOV S2,# H'43'     ; WITH ACK, FSCLK   87 kHz at 4,43 MHz
     MOV SO,# H'84'     ; LOAD SLAVE ADD HEX '42'(bit7-1) IN SO' ALS=0(bit 0)
     MOV S1,# H'18'     ; ENABLE SIO, SELECT SLAVE RECEIVER MODE
     EN SI              ; ENABLE SERIAL I/O INTERRUPT
```

The SIO interface is now in the 'slave receiver' mode. If address HEX'42' is received, the device will generate an acknowledge bit and cause a serial I/O interrupt call. Furthermore software responses depend on the contents of status register S1 as shown in Table 4.

Table 4 Status word bit patterns

Serial I/O status word S1

MST	TRX	BB	PIN	AL	AAS	ADO	LRB	Mode	Data received or transmitted	Software response
0	0	1	0	0	1	1/0	0	SLV/REC	Own slave/all 0´s address received with R/W = 0	A dummy read of S0
0	0	1	0	0	0	1/0	0	SLV/REC	Data received in S0	00101XXX to S1 and/or read S0
0	1	1	0	0	1	0	0	SLV/TRX	Own slave address received with R/W = 1	Write 01101XXX to S1 and/or load S0
0	1	1	0	0	0	0	1/0	SLV/TRX	Transmitted out S0 LRB = 1/0 -> NO ACK/ACK returned	Write 01101XXX to S1 and/or load S0, or write 00111000 to S1
1	1	1	0	0	0	0	1/0	MST/TRX	Transmitted out S0. LRB = 1/0 -> NO ACK/ACK returned	Write 11101XXX to S1 and/or load S0, or * write 11011000 to S1
1	0	1	0	0	0	0	1/0	MST/REC	Slave address with R/W = 1 transmitted LRB = 1/0 -> NO ACK/ACK returned	Write 10111XXX to S1 or a dummy read of S0
1	0	1	0	0	0	0	0	MST/REC	Data received in S0	Write 10101XXX to S1 and/or read S0, or * write 11011000 to S1
0	0	1	0	1	0	0	X	MST/TRX lost arbit.	Slave address or data transmitted and arbitration lost	A dummy read of S0
0	0	1	0	1	1	1/0	0	MST/TRX lost arbit. now SLV/REC	During transmission of slave address arbitration lost and own slave/all 0´s address received with R/W = 0	A dummy read of S0
0	1	1	0	1	1	0	0	MST/TRX lost arbit. now SLV/TRX	During transmission of slave address arbitration lost and own slave address received with R/W = 1	Write 01101XXX to S1 and/or load S0
0	0	0	1	0	0	0	X		The bus is free, a transmission may be started	
X	X	1	X	X	X	X	X		The bus is busy, a transmission may not be started	
X	X	X	1	X	X	X	X		No serial I/O pending i.e. no complete byte received or transmitted (polling PIN)	

Note:

X = 1 or 0. Underlined bits do not contribute to the status information

XXX = BC2, BC1, BC0 according to table 2.

* Generation of a STOP condition.

9.3 Generation of a 'start condition' and the first byte.

After completion of the initialization procedure, serial data can be transmitted by selecting the 'master transmitter' modes as shown by the flow chart in figure 15. If the device is connected in a multi-transmitter bus system, the state of the BB flag must be tested to verify whether the serial bus is free. If the bus is free (BB = 0), data register S0 is loaded with the first byte for transmission. Bit position MST, TRX, and BB in status register S1 must each be set to 1 so as to begin transmission with a 'start' condition. When the first byte has been transmitted, the PIN flag is reset to 0. A serial interrupt call is initiated if the instruction EN SI has been performed previously. The status word in register S1 determines the software responses that will ensue as listed in Table 4. An example of a program which generates the 'start' condition and transmits the contents of working register Rr is shown below:

```
MSTX MOV A,S1          ; LOAD STATUS WORD TO ACCU
     JB5 MSTX          ; TEST IF BUS IS FREE
     MOV A,Rr          ; LOAD BYTE TO BE TRANSMITTED FROM Rr TO ACCU
     MOV S0,A          ; LOAD BYTE TO BE TRANSMITTED FROM ACCU TO S0
     MOV S1,#H'F8'     ; SELECT MASTER TRANSMITTER MODE AND
                       ; START TRANSMISSION
```

After checking that the serial bus is free, but before the transmission starts, another device may engage the bus. If this occurs, the SIO interface will not start its transmission. To prevent received data from being corrupted, The SIO hardware inhibits data from being written by software into data register S0 and status register S1. If an instruction then attempts to write into register S1, the AL (arbitration lost) flag is set to indicate that the attempt to transmit has failed. When the AL flag is set, a serial interrupt request is always generated at the end of the serial byte.

9.4 Software responses after transmission or reception of a byte

After transmission or reception of a byte, the PIN flag is reset to 0. If a serial I/O interrupt call to program memory location 5 (serial interrupt enabled) is made or if the PIN flag is polled (serial interrupt disabled), a software response has to be initiated to service the SIO interface. As long as the PIN flag is 0, the transfer of serial data is halted because clock bus line SCL is held LOW. Reading out or writing information to data shift register S0 sets the PIN flag to 1 and allows the transfer of serial data to continue.

The status word in register S1 contains all the information to determine the required software response. It also indicates the operating mode of the SIO interface, and gives information about the transmitted or received serial byte. Table 4 lists all possible bit combinations for the status word. It must be remembered that the four least-significant bits of the status word that can be read are not the same as those that can be written (see Fig. 8). The least-significant bit (LRB) of the status word during a read operation only indicates an acknowledgement if the SIO interface is programmed to operate in the acknowledgement mode, i.e. the ACK bit in clock control register S2 is set to 1. If the ACK bit is 0, the LRB bit position in the status register contains the last bit of the byte that has been transmitted or received.

The following is an example of a software response by a 'master transmitter'. The next 8-bit byte to be transmitted is in working register Rq.

```
     ORG H'05'          ; SERIAL INTERRUPT VECTOR
     JMP SIR            ; JUMP TO SERIAL INTERFACE ROUTINE
*
SIR  SEL RB1            ; SELECT REGISTER BANK 1
     MOV Rp,A           ; SAVE ACCU CONTENTS IN Rp
     MOV A,S1           ; LOAD SERIAL STATUS IN ACCU
     JB7 TXTS           ; TEST MST BIT
     JMP SLVR           ; JUMP TO SLAVE ROUTINE IF MST=0
TXTS JB6 TNBY           ; TEST TRX BIT
     JMP MRRC           ; JUMP TO MST/REC ROUTINE IF TRX=0
TNBY MOV A,Rq           ; LOAD NEXT BYTE TO BE TRANSMITTED
*                       ; FROM Rq TO ACCU
     MOV SO,A           ; LOAD IT TO SO=START TRANSMISSION
     MOV A,Rp           ; RESTORE CONTENTS OF ACCU
     RETR               ; RETURN TO MAIN PROGRAM
```

Note that bit counter bits BC0, BC1 and BC2 in the status register need not be preset; they all remain 0 because 8 data bits have to be transmitted (see Table 2). The state of the LRB flag can be tested to check whether the transmitted data has been acknowledged. The result will be interpreted by the programmer.

9.5 Generation of the 'stop ' condition

A data transfer ends with a 'stop' condition generated by the 'master' device. To generate the 'stop' condition, the program in section 9.4 is followed until TXTS. The number of bytes to be transmitted is stored in a RAM memory location or working register. Each time a byte is transmitted this register is decremented. After checking that the last byte has been transferred the program continues with:

```
  MOV S1,#H'D8'         ; RESET BB TO 0, MST, TRX, PIN TO 1
*                       ; THIS GENERATES THE STOP CONDITION
  MOV A,Rp              ; RESTORE CONTENTS OF ACCU
  RETR                  ; RETURN TO MAIN PROGRAM
```

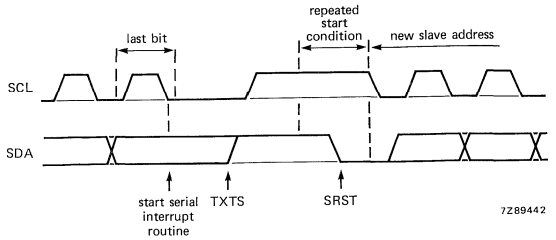


Fig. 13 Repeated 'start' condition on the serial bus.

Figure 13 shows the generation of a repeated 'start' condition followed by a new slave address. The addressing format with repeated 'start' condition is shown in figure 5(c). The serial bus remains busy at the end of the preceding data transfer because a 'stop' condition is not generated. The repeated 'start' condition is generated by a program which follows the program in section 9.4 until TXTS. After a check to verify that a repeated 'start' condition must be transmitted, the program continues with:

```
     MOV S1,#H'18'       ; RESET MST, SLV, BB TO 0
*                        ; SET PIN TO 1
*                        ; NOW BOTH SDA AND SCL
*                        ; BECOME HIGH (NO STOP CONDITION), PROVIDED SCL IS
*                        ; NOT PULLED DOWN BY ANOTHER DEVICE
MSTX MOV A,S1            ; LOAD STATUS WORD TO ACCU **
     JB5 MSTX            ; TEST BB. AS LONG AS THE INTERNAL SERIAL
*                        ; CLOCK IS LOW, BB = 1 **
     CPL A               ; COMPLEMENT ACCU
     JBO MSTX            ; TEST LRB=TEST IF SCL IS HIGH *** LRB IS SET TO 1
*                        ; AT THE POSITIVE GOING EDGE OF SCL (SDA=1)
     MOV A,Rq            ; LOAD NEW SLAVE ADD. TO ACCU
     MOV SO,A            ; LOAD NEW SLAVE ADD. TO SO
SRTS MOV S1,#H'F8'       ; SELECT MASTER TRANSMITTER MODE
*                        ; START TRANSMISSION
     MOV A,Rp            ; RESTORE CONTENTS OF ACCU
     RETR                ; RETURN TO MAIN PROGRAM
```

** This instruction may be omitted if it is certain that the LOW period of the serial clock finishes before label SRTS is reached.

*** As long as a slow slave pulls down the SCL line, the master is not able to generate a repeated 'start condition.

9.7 Generation of the 'stop' condition by a 'Master Receiver'

If a 'master receiver' wants to conclude a data transfer, it must instruct the 'slave transmitter' to free data bus line SDA so that it can generate the 'stop' condition. The 'master receiver' therefore generates a negative acknowledge (data line SDA held HIGH) after reception of the current data byte. The 'slave transmitter' recognizes the negative acknowledge by testing the LRB flag and must then be changed to the 'slave receiver' mode by software. The generation of the 'stop' condition consists of the following three stages:

1. An 8-bit byte is received without generating an acknowledgement bit because bit position ACK in clock control register S2 is loaded with a 0.

2. The 'master receiver' gives a negative acknowledgement by generating an extra clock pulse, i.e. it generates a single HIGH-state bit on data line SDA.

3. The 'stop' condition is generated as a 'master transmitter'.

The 'master receiver' program for the last received byte is the same as that given in section 9.4 until TNBY. After a check has verified that the last byte is to be transferred, the program continues with:

```
     MOV S2,#H'03'       ; NO ACK, FSCL = 87 kHz
     MOV A,SO            ; READ SO AND START
*                        ; WITH LAST BYTE TRANSFER
     MOV @RO,A           ; STORE RECEIVED DATA
     MOV A,Rp            ; RESTORE CONTENTS OF ACCU
     RETR                ; RETURN TO MAIN PROGRAM
```

During the next serial interrupt routine, the program in section 9.4 is again followed until TNBY. After a check has verified that the last byte has been transferred, a negative acknowledge and a 'stop' condition must be transmitted. The program continues with:

```
     MOV S1,#H'A9'       ; LOAD BIT COUNTER FOR
*                        ; ONE BIT TRANSFER
     MOV A,SO            ; READ SO AND START
*                        ; ONE BIT TRANSFER =
*                        ; NEGATIVE ACKNOWLEDGE
     MOV @RO,A           ; STORE RECEIVED DATA
TSPI MOV A,S1            ; LOAD STATUS WORD TO ACCU **
     JB4 TSPI            ; TEST END OF ONE BIT TRANSFER **
     MOV S2,#H'43'       ; RESTORE ACK BIT IN S2
     MOV S1,#H'D8'       ; GENERATE STOP CONDITION
     MOV A,Rp            ; RESTORE ACCU
     RETR                ; RETURN TO MAIN PROGRAM
```

** These instructions can be omitted if it is certain that the negative acknowledge has been transferred before instruction MOV S2,# H'43' is given.

9.8 Flow charts for the 'master' and 'slave' functions of the SIO interface

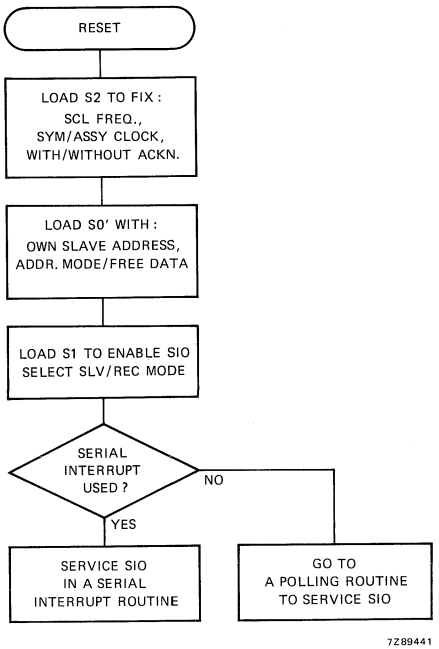


Fig. 14 Initialization flow chart. A prior RESET signal has cleared all bits of the SIO registers except PIN = 1 in status register S1.

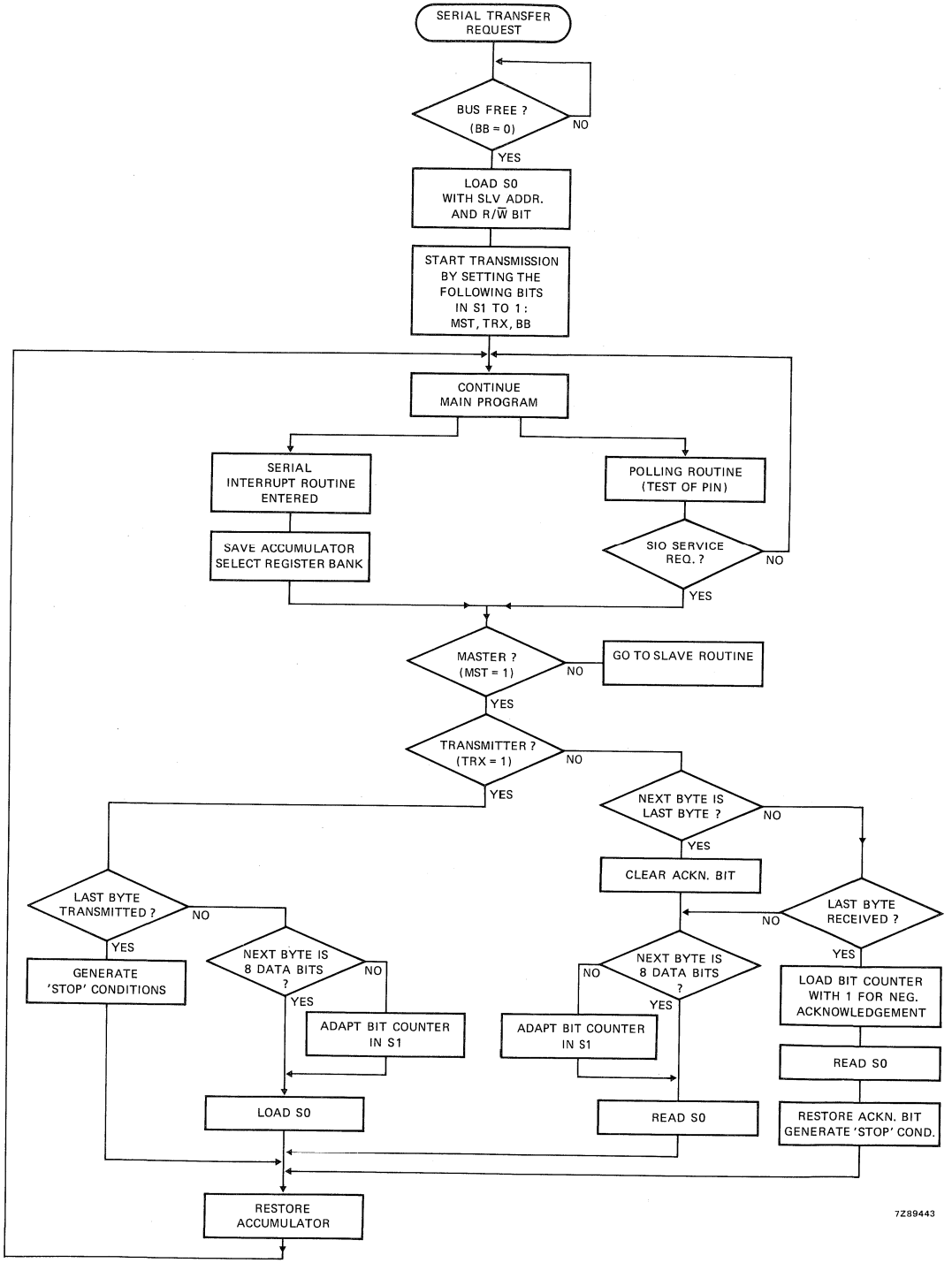


Fig. 15 Flow chart of the 'master' mode after initialization.

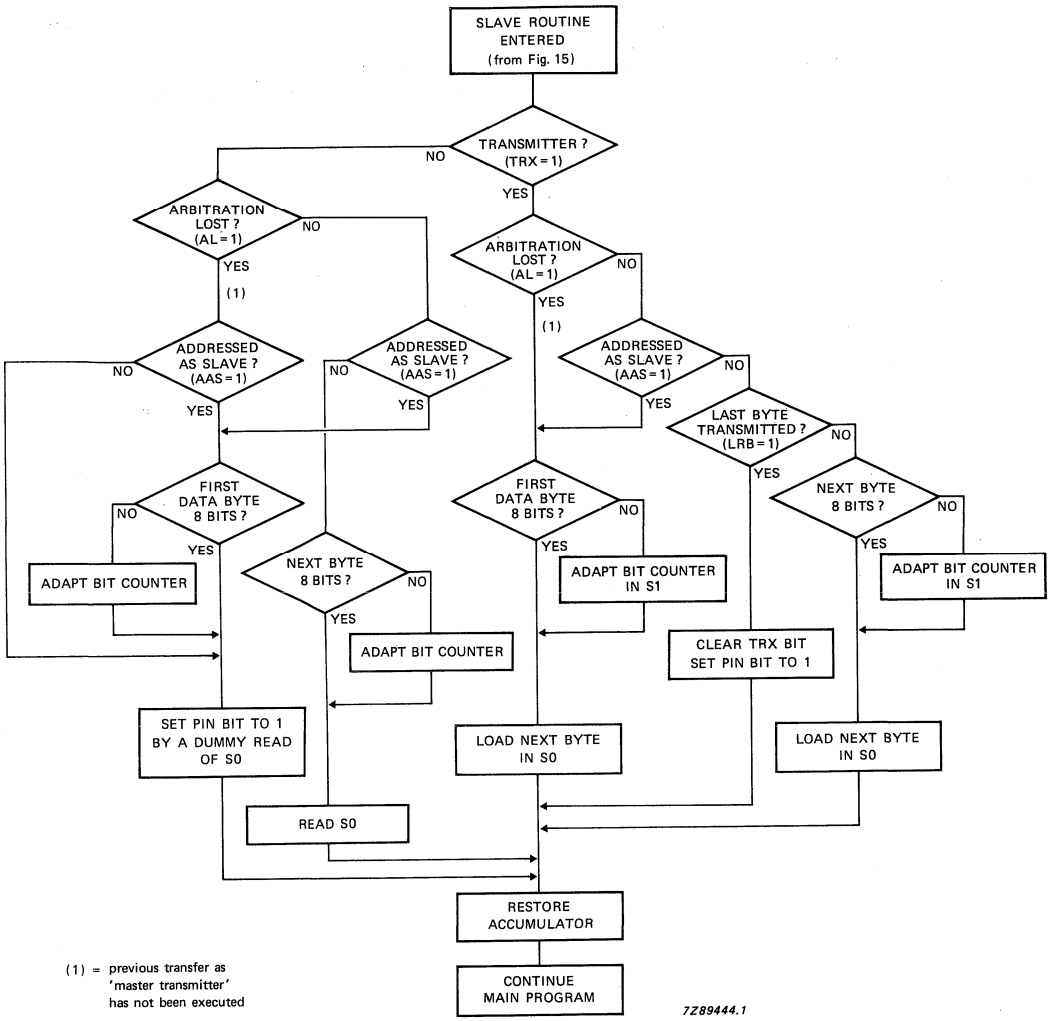


Fig. 16 Flow chart of the 'slave' mode after initialization.

9.9 Solutions to bus-error software problems

The following are solutions to an undefined bus-error state, due to noise or interference, that the SIO interface may suffer while operating in the I^2C mode.

1. Problem:

   In multi-master operation it is feared that the SIO interface has been disturbed during transmission of the acknowledge bit.

   Solution:

   This can be solved by rewriting register S2:

```
MOV S2, #H'02'      ; Clear ACK bit
MOV S2, #H'42'      ; Set ACK bit and frequency
```

2. Problem:

   To recover succesfully from a bus-error state, register S1 must also be reinitialized. When writing to S1, not all the bits are set as expected.

   Solution:

   Write to S1 twice:

```
MOV S1, #ES0 + PIN
MOV S1, #ES0 + PIN
```

While operating in the 'Master transmitter' mode and the end of a byte is reached, the following subroutine illustrates a possible bus-error recovery program incorporating the above instructions.

```
MERROR   MOV R1,A           ; Save accumulator
         MOV A,S1           ; Get status
         JB2 ADDSLV         ; Jump if addressed as slave
         JB7 I2CSST         ; Jump if MST = 1
         JB3 MERR2          ; Jump if AL = 1
         MOV S2,#H'02'      ; Release bus at error condition
                            ; AAS + MST + AL = 0
         MOV S2,#H'42'      ;
         MOV S1,#ES0+PIN    ;
         MOV S1,#ES0+PIN    ;
         JMP MERR1          ;
MERR2    MOV A,S0           ; Release bus method if AL = 1
         JMP MERR1          ;
I2CSST   MOV A,Rq           ; Send next data byte
         MOV S0,A           ;
 MERR1   MOV A,R1           ; Restore accumulator
         RET                ; Leave
ADDSLV   CALL SLAVE         ; Call slave routine
         MOV A,R1           ; Restore accumulator
         RET                ; Leave
```

Software examples

CONTENTS – SOFTWARE EXAMPLES

1.0 INTRODUCTION

Software Examples links the other sections within this manual. However, it is not the only section in which software examples can be found. Flow charts and examples for the MAB84X1 family can also be found in the Serial I/O section.

All routines in this section are given as examples only- the construction of these routines in section 3.3 gives an indication of how the 84X1 and 8048 operate using the I^2C bus at system level. Each example is taken from a Philips Microcomputer Development System (PMDS II) using 84X1 cross assembler (Rel: 3.0). (This does not apply to section 4 where the 8048 cross assembler was used).

2.0 GENERAL SOFTWARE EXAMPLES (APPLICABLE FOR 8048 FAMILY AND 84X1 FAMILY)

2.1 Double precision unsigned addition subroutine

Add two unsigned 16 bit quantities from data memory locations (OPRNDA, OPRNDA + 1) and store result in data memory locations RSLT, RSLT + 1.

R0 and R1 are used as pointers, R2 for temporary storage.

```
                                  9 OPRNDB EQU     H'20'          DEFINE OPERANDS: OPERAND B HIGH: LOC 20
                                 10 *                                              OPERAND B LOW:  LOC 21
                                 11 OPRNDA EQU     OPRNDB+2                       OPERAND A HIGH: LOC 22
                                 12 *                                             OPERAND A LOW:  LOC 23
                                 13 RSLT   EQU     OPRNDB+4       DEFINE RESULT:  RESULT HIGH:    LOC 24
                                 14 *                                             RESULT LOW:     LOC 25
                                 15 *
000000 B823             1       16 DADD   MOV     R0,#OPRNDA+1   R0 POINTS TO OPERAND A LOW
000002 B921             2       17        MOV     R1,#OPRNDB+1   R1 POINTS TO OPERAND B LOW
000004 F0               3       18        MOV     A,@R0          GET OPERAND A LOW
000005 61               4       19        ADD     A,@R1          ADD OPERAND B LOW
000006 AA               5       20        MOV     R2,A           SAVE RESULT TEMPORARILY
000007 C8               6       21        DEC     R0             R0 POINTS TO OPERAND A HIGH
000008 C9               7       22        DEC     R1             R1 POINTS TO OPERAND B HIGH
000009 F0               8       23        MOV     A,@R0          GET OPERAND A HIGH
00000A 71               9       24        ADDC    A,@R1          ADD OPERAND B HIGH AND CARRY
00000B B824            10       25        MOV     R0,#RSLT       R0 POINTS TO RESULT HIGH
00000D A0              11       26        MOV     @R0,A          STORE RESULT HIGH
00000E 18              12       27        INC     R0             R0 POINTS TO RESULT LOW
00000F FA              13       28        MOV     A,R2           GET RESULT LOW
000010 A0              14       29        MOV     @R0,A          STORE RESULT LOW
000011 83              15       30        RET
000012                          31        END
```

2.2 Double precision unsigned subtraction subroutine

Subtract an unsigned 16 bit subtrahend in data memory locations OPRNDB, OPRNDB + 1 from an unsigned 16 bit minuend in data memory locations OPRNDA, OPRNDA + 1 and store result in data memory locations RSLT, RSL + 1.

R0 and R1 are used as pointers and R2 is used for temporary storage.

```
                                 9 *
                                10 OPRNDB EQU     H'20'          DEFINE OPERANDS: SUBTRAHEND HIGH: LOC 20
                                11 *                                              SUBTRAHEND LOW:  LOC 21
                                12 OPRNDA EQU     OPRNDB+2                        MINUEND HIGH:    LOC 22
                                13 *                                              MINUEND LOW:     LOC 23
                                14 RSLT   EQU     OPRNDB+4       DEFINE RESULT:   RESULT HIGH:     LOC 24
                                15 *                                              RESULT LOW:      LOC 25
                                16 *
000000 B823           1        17 DSUB   MOV     R0,#OPRNDA+1   R0 POINTS TO MINUEND LOW
000002 B921           2        18        MOV     R1,#OPRNDB+1   R1 POINTS TO SUBTRAHEND LOW
000004 F1             3        19        MOV     A,@R1          GET SUBTRAHEND LOW
000005 37             4        20        CPL     A              MAKE 2'S COMPLEMENT
000006 0301           5        21        ADD     A,#1
000008 AA             6        22        MOV     R2,A           SAVE 2'S COMPLEMENTED SUBTRAHEND LOW
000009 C9             7        23        DEC     R1             R1 POINTS TO SUBTRAHEND HIGH
00000A F1             8        24        MOV     A,@R1          GET SUBTRAHEND HIGH
00000B 37             9        25        CPL     A              COMPLEMENT BUT INCREMENT ONLY IF CARRY
00000C 1300          10        26        ADDC    A,#0           FROM LOW ORDER
00000E 2A            11        27        XCH     A,R2           SAVE 2'S COMPLEMENTED SUBTRAHEND HIGH
00000F 60            12        28        ADD     A,@R0          GET LOW AND ADD MINUEND LOW
000010 B925          13        29        MOV     R1,#RSLT+1     R1 POINTS TO RESULT LOW
000012 A1            14        30        MOV     @R1,A          STORE DIFFERENCE LOW
000013 C9            15        31        DEC     R1             R1 POINTS TO RESULT HIGH
000014 C8            16        32        DEC     R0             R0 POINTS TO MINUEND HIGH
000015 FA            17        33        MOV     A,R2           GET 2'S COMPLEMENTED SUBTRAHEND HIGH
000016 70            18        34        ADDC    A,@R0          ADD MINUEND HIGH WITH CARRY
000017 A1            19        35        MOV     @R1,A          STORE DIFFERENCE HIGH
000018 83            20        36        RET
000019                         37        END
```

2.3 Double precision product unsigned multiplication

Multiply two unsigned 8 bit quantities from data memory locations OPRNDA, OPRNDB and store the 16 bit result in data memory locations RSLT, RSL + 1.

R0, R1 and R2 are used as pointers and temporary storage registers.

R3 is used as a counter.

```
                                 9 *
                                10 OPRNDA EQU     H'20'          DEFINE OPERAND LOCATIONS
                                11 OPRNDB EQU     OPRNDA+1
                                12 RSLT   EQU     OPRNDA+2       DEFINE RESULT LOCATIONS
                                13 *
000000 B820           1        14 DMUL   MOV     R0,#OPRNDA     R0 POINTS TO MULTIPLIER
000002 B921           2        15        MOV     R1,#OPRNDB     R1 POINTS TO MULTIPLICAND
000004 BB09           3        16        MOV     R3,#9          INITIALIZE LOOP COUNTER
000006 F0             4        17        MOV     A,@R0          GET MULTIPLIER
000007 A8             5        18        MOV     R0,A           SAVE TEMPORAILY IN R0
000008 F1             6        19        MOV     A,@R1          GET MULTIPLICAND
000009 A9             7        20        MOV     R1,A           SAVE TEMPORARILY IN R1
00000A 27             8        21        CLR     A              CLEAR PRODUCT
00000B 97             9        22        CLR     C              CLEAR CARRY
00000C 67            10        23 DMULLP RRC     A              RIGHT SHIFT PRODUCT
00000D 28            11        24        XCH     A,R0           R0 <-- PRODUCT; A <-- MULTIPLIER
00000E 67            12        25        RRC     A              SHIFT MULTIPLIER LSB INTO CARRY
00000F 28            13        26        XCH     A,R0           A <-- PRODUCT; R0 <-- MULTIPLIER
000010 E613          14        27        JNC     ZROBIT         JUMP IF MULTIPLIER LSB WAS 0
000012 69            15        28        ADD     A,R1           ELSE ADD MULTIPLICAND TO PRODUCT
000013 EB0C          16        29 ZROBIT DJNZ    R3,DMULLP      LOOP UNTIL ALL MULTIPLIER BITS TESTED
000015 B922          17        30        MOV     R1,#RSLT       R1 POINTS TO RESULT HIGH
000017 A1            18        31        MOV     @R1,A          STORE RESULT HIGH
000018 19            19        32        INC     R1             R1 POINTS TO RESULT LOW
000019 F8            20        33        MOV     A,R0           GET RESULT LOW
00001A A1            21        34        MOV     @R1,A          STORE
00001B 83            22        35        RET                    RETURN
00001C                         36        END
```

2.4 Branch on input value

This routine inputs a value from an input device on PORT 1 (e.g. keyboard)looks it up in a table and branches to an address specified in another table.

R6 and R7 are used for temporary storage.

```
000000 09              1      9 LOOKUP IN      A,P1                INPUT VALUE FROM PORT 1
000001 AF              2     10        MOV     R7,A                SAVE TEMPORARILY IN R7
000002 BE0F            3     11        MOV     R6,#VALTBL-1        INITIALIZE TABLE LOOK UP POINTER
000004 1E              4     12 LOOKLP INC     R6                  R6 POINTS TO NEXT VALUE IN TABLE
000005 FE              5     13        MOV     A,R6                MOVE POINTER TO ACCU
000006 A3              6     14        MOVP    A,@A                GET A VALUE FROM TABLE
000007 C617            7     15        JZ      NOTFND              JUMP TO ERROR ROUTINE IF END OF LIST
000009 DF              8     16        XRL     A,R7                COMPARE TABLE VALUE WITH INPUT VALUE
00000A 9604            9     17        JNZ     LOOKLP              LOOP IF NO MATCH
00000C FE             10     18        MOV     A,R6                MOVE LOOKUP TABLE POINTER TO ACCU
00000D 0304           11     19        ADD     A,#BRCHTB-VALTBL    ADD OFFSET ONTO BRANCH TABLE
00000F B3             12     20        JMPP    @A                  BRANCH TO ON PAGE ROUTINE
                             21 *
000010 41                    22 VALTBL DATA    'A'                 TABLE OF LOOKUP VALUES
000011 31                    23        DATA    '1'
000012 0F                    24        DATA    H'0F'
000013 00                    25        DATA    0
                             26 *
000014 18                    27 BRCHTB DATA    .LOW.ALETTR         TABLE OF ROUTINE ADDRESSES CORRESPONDING
000015 19                    28        DATA    .LOW.ONENUM         TO VALUES ABOVE
000016 1A                    29        DATA    .LOW.CRCHAR
                             30 *
000017 00             13     31 NOTFND NOP                         ERROR HANDLING ROUTINE STARTS HERE
                             32 *
000018 00             14     33 ALETTR NOP                         CHARACTER 'A' HANDLING ROUTINE STARTS HERE
                             34 *
000019 00             15     35 ONENUM NOP                         CHARACTER '1' HANDLING ROUTINE STARTS HERE
                             36 *
00001A 00             16     37 CRCHAR NOP                         CARRIAGE RETURN HANDLING ROUTINE STARTS HERE
00001B                       38        END
```

2.5 Arithmetic shift left and right subroutines

These subroutines arithmetically shift the 16 bit number formed by the contents of R7 (high-order byte) and the contents of the accumulator (low-order byte).

R6 specifies the number of places to shift.

```
                             8 *
000000 97              1     9 ASLSUB CLR     C                   CLEAR CARRY SO SHIFT 0'S FROM THE RIGHT
000001 F7              2    10        RLC     A                   ROTATE LOW BYTE LEFT
000002 2F              3    11        XCH     A,R7                GET HIGH BYTE
000003 F7              4    12        RLC     A                   ROTATE LEFT WITH MSB OF LOW BYTE
000004 2F              5    13        XCH     A,R7                EXCHANGE BACK
000005 EE00            6    14        DJNZ    R6,ASLSUB           LOOP UNTILL DONE
000007 83              7    15        RET
                            16 *
000008 97              8    17 ASRSUB CLR     C                   CLEAR CARRY SO SHIFT 0'S FROM THE LEFT
000009 2F              9    18        XCH     A,R7                GET HIGH ORDER IN ACCU
00000A 67             10    19        RRC     A                   SHIFT RIGHT
00000B 2F             11    20        XCH     A,R7                GET LOW ORDER, SAVE HIGH ORDER
00000C 67             12    21        RRC     A                   SHIFT LOW ORDER RIGHT WITH MSB OF HIGH
00000D EE08           13    22        DJNZ    R6,ASRSUB             ORDER, LOOP UNTIL DONE
00000F 83             14    23        RET
                            24 *
000010                      25        END
```

2.6 BCD to binary conversion subroutine

This subroutine is entered with a two digit BCD number in the accumulator.

The number is converted to binary and the result is returned in the accumulator.

R6 and R7 are used as temporary storage registers.

```
                          8 *
000000 AF              1  9 BCDBIN MOV     R7,A            SAVE BCD NUMBER TEMPORARILY
000001 53F0            2 10        ANL     A,#H'F0'        MASK OUT LOWER DIGIT
000003 47              3 11        SWAP    A               MOVE HIGH DIGIT DOWN
000004 AE              4 12        MOV     R6,A            SAVE TEMPORARILY
000005 E7              5 13        RL      A               MULTIPLY HIGH DIGIT BY 10 DECIMAL
000006 E7              6 14        RL      A               10X=2(4X+X)
000007 6E              7 15        ADD     A,R6
000008 E7              8 16        RL      A
000009 2F              9 17        XCH     A,R7            SWAP RESULT WITH ORIGINAL NUMBER
00000A 530F           10 18        ANL     A,#H'0F'        GET LOW DIGIT
00000C 6F             11 19        ADD     A,R7
00000D 83             12 20        RET
                         21 *
00000E                   22        END
```

3.0 SERIAL I/O SOFTWARE EXAMPLES: APPLICABLE TO MAB84X1 FAMILY

The serial I/O equate list given below, is used throughout the programs in section 3.0 to 3.2 inclusive.

```
3 * SERIAL I/O EQUATE LIST
4 ************************
5 *
6 ***** REGISTER S0' *****
7 ALS     EQU     H'01'           ALWAYS SELECTED BIT
8 SLAB    EQU     H'FE'           SLAVE ADDRESS BITS
9 *
10 ***** REGISTER S0 *****
11 RW     EQU     H'01'           READ/WRITE NOT CONTROL BIT
12 *
13 ***** REGISTER S1 *****
14 * READ AND WRITE BITS
15 MST    EQU     H'80'           MASTER BIT
16 TRX    EQU     H'40'           TRANSMITTER BIT
17 BB     EQU     H'20'           BUS BUSY BIT
18 PIN    EQU     H'10'           PENDING INTERRUPT NOT BIT
19 *
20 * WRITE BITS
21 ESO    EQU     H'08'           ENABLE SERIAL I/O BIT
22 BC2    EQU     H'04'   %)      SIO BITCOUNTER BIT 2
23 BC1    EQU     H'02'   %)      SIO BITCOUNTER BIT 1
24 BC0    EQU     H'01'   %)      SIO BITCOUNTER BIT 0
25 *
26 STRTC  EQU     MST+TRX+BB+PIN+ESO  %)  START CONDITION BITS
27 STOPC  EQU     MST+TRX+PIN+ESO     %)  STOP CONDITION BITS
28 *
29 * READ BITS
30 AL     EQU     H'08'           ARBITRATION LOST BIT
31 AAS    EQU     H'04'           ADDRESSED AS SLAVE BIT
32 AD0    EQU     H'02'           ADDRESS ZERO BIT
33 LRB    EQU     H'01'           LAST RECEIVED BIT
34 *
35 MTTST  EQU     MST+TRX+BB  %)  MASTER TRANSMITTER TEST BITS
36 MRTST  EQU     MST+BB      %)  MASTER RECEIVER TEST BITS
37 *
38 ***** REGISTER S2 *****
39 ASC    EQU     H'20'           ASYNCHRONE CLOCK BIT
40 ACK    EQU     H'40'           WITH ACKNOWLEDGE BIT
41 SCLFB  EQU     H'1F'           SCL CLOCK FREQUENCY BITS
42 *
```

3.1 Single-master routines

These routines can only be used if no other masters are connected to the I^2C-bus. All these routines are based on polling of the PIN bit in register S1.

3.1.1 Master transmitter routine (MAB84X1 - SAA1300)

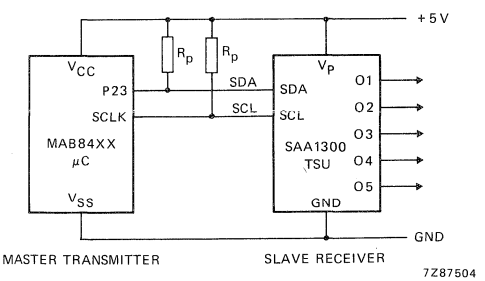


Fig. 3.1 Block diagram of MAB84X1 - SAA1300 configuration.

Initialization:

Normally the serial I/O (SIO) of the MAB84X1 family has to be initialized as follows:

- program the SCL frequency in register S2
- select the "with acknowledge" mode in register S2
- load its own slave address in register S0'
- enable the SIO logic with the ESO bit in register S1
- select the slave receiver mode in register S1
- enable the SIO interrupt

The slave address is not programmed here because there are no other masters connected to the I^2C-bus. So this MAB84X1 microcomputer cannot be addressed as a slave.

Also, the SIO interrupt is not enabled because the routines are based on polling of the PIN bit in status register S1.

```
                              52 IICFR  EQU     H'04'           Fscl = 95 kHz @Fxtal = 6,0 MHz
                              53 *
                              54 * POWER-ON RESET:
                              55 *
                              56        PAGE    256
                              57 ROM0   ASECT   ROM
000000                        58        ORG     H'000'
                              59 *
000000 2400          1        60 RESET  JMP     INIT            JUMP TO INITIALIZE ROUTINE


000002                        77        ORG     H'100'
                              78 *
000100 9E44          2        79 INIT   MOV     S2,#IICFR+ACK   LOAD SCL FREQUENCY WORD
                              80 *                              SET WITH ACKNOWLEDGE MODE
000102 9D18          3        81        MOV     S1,#PIN+ESO     ENABLE SIO
                              82 *                              SET SLAVE RECEIVER MODE
000104 B8FF          4        83        MOV     R0,#H'FF'       LOAD DATA TO BE WRITTEN TO TSU
                              84 *
```

Main program:

The main program transmits data to TSU (SAA1300) by calling the subroutine WTSU0 or WTSU1.

If the transmission is not successful, the data transfer is repeated, otherwise the data is incremented and the sequence starts again.

The format of the data transfer to TSU is given in Fig. 3.2:

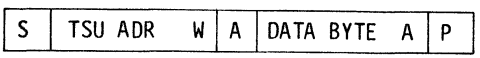


Fig. 3.2 Data format of transfer to TSU.

S	is the START condition
TSU ADR	is the slave address of the TSU
W	is the read/write bit in write state (= 0)
A	is the acknowledge bit; this has to be 0
P	is the STOP condition.

In all these software examples each slave address is defined as an eight bit word with a R/W bit in the WRITE state (= 0).

```
                          102 * The slave address of TSU is 0100 0XX
                          103 TSUAD  EQU     H'40'
                          104 *
                          105 *
000106 18            5    106 MAIN   INC     R0              INCR. DATA TO BE TRANSMITTED
                          107 *
000107 3422          6    108 MAIN1  CALL    WTSU1           TRANSMIT DATA BYTE IN R0 TO TSU
                          109 *                          CALL SUBROUTINE WTSU0 OR WTSU1
000109 9607          7    110        JNZ     MAIN1           REPEAT IF TRANSMISSION IS NOT SUCCESSFUL
00010B 2406          8    111        JMP     MAIN            CONTINUE
                          112 *
```

Master writes one data byte to TSU (SAA1300) subroutine 1:

To start a transmission, the slave address together with the read/write bit (R/W) - in write state - has to be loaded into the SIO data shift register S0. The transmission really starts when the start condition word (STRTC) is loaded in the SIO status register S1.

The start condition word STRTC is programmed in the SIO equate list as MST + TRX + BB + PIN + ESO. The combination of status bits is always defined with the "+" statement. This can be done while in the SIO equate list; each SIO status bit is defined individually.

The MAB84X1 family microcomputer generates the START condition, slave address, R/W bit and an acknowledge related clock pulse. During this last clock pulse it inputs the acknowledge bit which will be represented by the LRB bit in the status register S1.

If the SIO logic has finished this part of the transmission, this is indicated by the SIO status bit PIN, which becomes 0. Now the SIO status can be tested by comparing (XRL instr.) it with the mask MTTST (= MST + TRX + BB).

If equal (Accumulator = 0), the data transfer continues by loading the data byte - to be transmitted and stored in R0 - into the SIO data register S0. If PIN becomes zero again, the SIO status is tested and the data transfer to TSU is terminated by a STOP condition.

The subroutine returns with the accumulator contents equal to zero if the data transfer to TSU has been successfull.

If the slave address is not acknowledged, the transmission is terminated immediately with a STOP condition and the Accumulator contents equal to H'01'.

If the data byte is not acknowledged, the transmission is also terminated by a STOP condition with the accumulator contents equal to H'01'.

entry: R0 contains data to be transmitted
exit : A = 0 if transmission is successful

```
00010D 9C40          9   143 WTSU0   MOV     S0,#TSUAD     LOAD TSU SLAVE ADDRESS AND WRITE BIT
00010F 9DF8         10   144         MOV     S1,#STRTC     OUTPUT START COND, SLAVE ADDRESS AND WRITE BIT
                         145 *
000111 0D           11   146 WTSU01 MOV     A,S1          FETCH SIO STATUS
000112 9211         12   147         JB4     WTSU01        WAIT FOR PIN = 0
000114 D3E0         13   148         XRL     A,#MTTST      TEST MST/TRX SIO BUS STATUS
000116 961F         14   149         JNZ     WTSU03        JUMP IF NO ACKN. RECEIVED OR IF ERROR
000118 F8           15   150         MOV     A,R0          FETCH DATA TO BE TRANSMITTED
000119 3C           16   151         MOV     S0,A          TRANSMIT DATA BYTE'
                         152 *
00011A 0D           17   153 WTSU02 MOV     A,S1          FETCH SIO STATUS
00011B 921A         18   154         JB4     WTSU02        WAIT FOR PIN = 0
00011D D3E0         19   155         XRL     A,#MTTST      TEST MST/TRX SIO BUS STATUS
                         156 *
00011F 9DD8         20   157 WTSU03 MOV     S1,#STOPC     OUTPUT STOP CONDITION
000121 83           21   158         RET
                         159 *
```

Disturbances of the bus signals:

When using the previous subroutine, the bus can become blocked by disturbances from an external source (e.g. flash-overs in a TV system).

As a safeguard against this, first of all, a digital filter is designed in at the SDA and SCL inputs. Spikes have to be longer than 2 xtal clock cycles Txtal before they will be seen by the internal SIO logic. However to prevent a "bus blocked" situation some software precautions also have to be taken.

The influences of disturbance on the data line are described below. Similar results occur with disturbance on the clock line and are solved in a similar manner to that described below.

Disturbance of a data bit in master transmitter mode can create:

- An arbitration lost situation if the data bit which is output high is completely pulled-down.

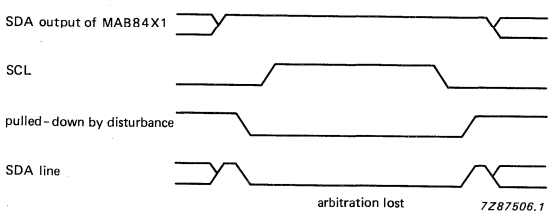


Fig. 3.3. Arbitration lost

If the master loses arbitration, it switches over to the slave receiver mode and inputs the data word by generating clock pulses until the bit counter becomes zero. PIN is then set to zero and the bus status indicated in S1 is 0010 1XXX.

The bit AAS is set if the master's own address is received. When a general call address is received, the bits AAS and ADO are set.

Since there are no other masters in the system, an arbitration lost situation can only occur due to a disturbance in the data bit. In the above subroutine, the microcomputer tries to generate a STOP condition and exits with A ≠0.

- A new start condition if the last part of a data byte which was output high is pulled low.

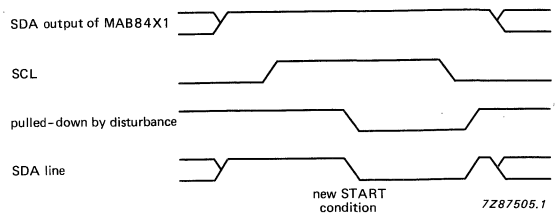


Fig. 3.4. New start condition

Such a new START condition resets the bit counter and starts again to transmit the current contents of S0 as a slave address. Because the contents of S0 is already shifted some bits to the left, this output slave address is not correct.

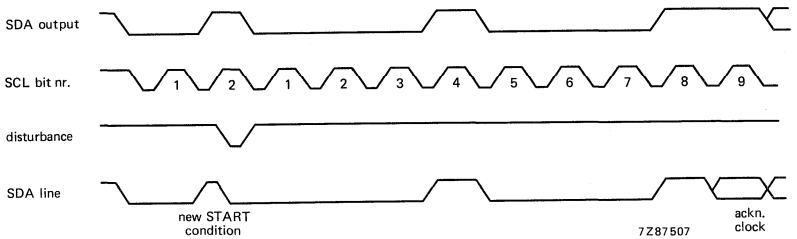


Fig. 3.5.

In the above timing diagram the master wants to transmit the slave address H'44'. However, a new START condition is generated due to a disturbance of the second data bit. At that moment the contents of S0 has become H'11' because S0 has already been shifted to the left twice. Now this H'11' code is transmitted as a new slave address. If PIN becomes zero the bus status will be 1010 000X. After comparison with the MTTST mask, a STOP condition is generated and the subroutine returns with A $\neq$ 0.

- Generation of a STOP condition if the first part of a data bit which is output high is pulled-down.

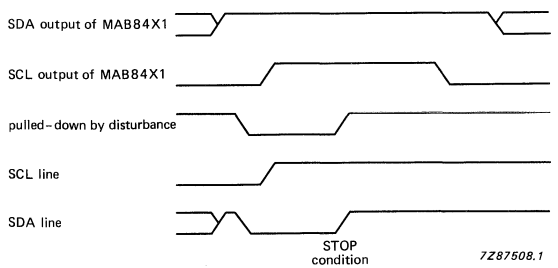


Fig. 3.6.

A STOP condition stops the data transmission and the bus status becomes 0001 100X. The bit counter also stops and stays at its current value. Now the status bit PIN will never become zero.

In the above subroutine the bus is blocked, which is unacceptable.

- Generation of a START/STOP condition, if a small part in the middle of a data bit which is output high is pulled-down.

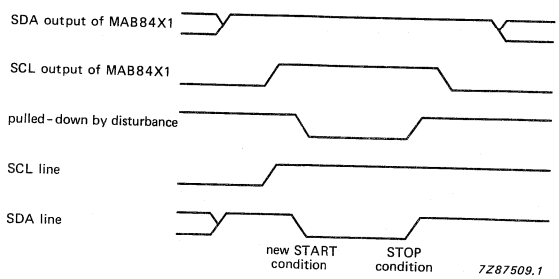


Fig. 3.7.

The new START condition resets the bit counter and the STOP condition stops the data transfer.

To solve a "bus blocked" problem a PIN = 0 time out counter can be added (see subroutine WTSU1 and MTTB1).

Master writes one data byte to TSU (SAA1300) subroutine 2:

This subroutine also returns with A ≠ 0 if the transmission is not successful due to a not received acknowledge or a disturbance of a data bit.

In the first case, the subroutine generates a STOP condition to release the bus. In the second case, the bus is released by returning to the SLAVE mode.

By testing the contents of the accumulator, the main program can repeat the transmission until it succeeds, if necessary.

Because the status of the bus is not always known at the beginning of the subroutine, it is reset with the MOV S1,#PIN+ESO instruction.

*In the second case the bus is released by returning to the slave mode.

If a certain transmission is followed by another transmission, the time between the MOV S1,#STOPC instruction and the MOV S1,#PIN+ESO has to be longer than the time the SIO logic needs to execute the STOP condition. This time is about one SCL clock pulse.

To prevent errors in the case of a slow SCL clock, a test on BB = 0 can be done for a certain maximum time which is longer than one SCL clock cycle before the execution of the MOV S1,#PIN+ESO instruction.

Since the test of PIN = 0 has to be performed twice, a subroutine is used.

entry: R0 contains data to be transmitted,
exit : A = 0 if transmission is successful.

```
000122 9D18      22   295 WTSU1   MOV     S1,#PIN+ESO       RESET BUS STATUS
000124 9C40      23   296         MOV     S0,#TSUAD         LOAD TSU SLAVE ADDRESS AND WRITE BIT
000126 9DF8      24   297         MOV     S1,#STRTC         OUTPUT START COND, SLAVE ADDRESS AND WRITE BIT
000128 345A      25   298         CALL    MTTBS             TEST BUS STATUS
                      299 *                                 CALL SUBROUTINE MTTB1, MTTB2 OR MTTBS
00012A 9632      26   300         JNZ     ERROR             JUMP IF NO ACK. RECEIVED OR ERROR
00012C F8        27   301         MOV     A,R0              FETCH DATA TO BE TRANSMITTED
00012D 3C        28   302         MOV     S0,A              TRANSMIT DATA BYTE
00012E 345A      29   303         CALL    MTTBS             TEST BUS STATUS
                      304 *                                 CALL SUBROUTINE MTTB1, MTTB2 OR MTTBS
000130 C644      30   305         JZ      STOP              JUMP IF ACK. RECEIVED & NO ERRORA
                      306 *
                      307 * NO ACKNOWLEDGE RECEIVED OR ERROR EXIT ROUTINE
                      308 *
000132 D301      31   309 ERROR   XRL     A,#LRB            INVERT  ACK
000134 C642      32   310         JZ      ERROR1            IF NO ACK. OUTPUT STOP COND.
000136 9E04      33   311         MOV     S2,#IICFR
000138 9E44      34   312         MOV     S2,#IICFR+ACK
00013A 9D18      35   313         MOV     S1,#PIN+ESO       RESET BUS STATUS TWICE IF ERROR FEARED
00013C 9D18      36   314         MOV     S1,#PIN+ESO       RETURN TO SLAVE MODE
00013E 0C        37   315         MOV     A,S0              DUMMY READ
00013F 2302      38   316         MOV     A,#2              TO RETURN WITH A ≠ 0
000141 83        39   317         RET
000142 2301      40   318 ERROR1  MOV     A,#1              TO RETURN WITH A = 1
000144 9DD8      41   319 STOP    MOV     S1,#STOPC         OUTPUT STOP CONDITION
000146 83        42   320         RET
```

Master transmitter test bus status subroutine 1:

In this subroutine a time-out counter is inserted to prevent a bus block in the case where PIN doesn't become 0 because of a disturbance of a data bit.

The contents of register R2 are modified.

```
                        324 *  in case PIN will never become 0 because of a disturbance of a data bit.
                        325 * The contents of register R2 are modified.
                        326 *
000147 BA00       43    327 MTTB1  MOV     R2,#00          LOAD PIN=0 TIME-OUT COUNTER
                        328 *
000149 0D         44    329 MTTB11 MOV     A,S1            FETCH BUS STATUS
00014A 924F       45    330        JB4     MTTB12          JUMP IF PIN = 1
00014C D3E0       46    331        XRL     A,#MTTST        TEST MST/TRX SIO STATUS
00014E 83         47    332        RET
                        333 *
00014F EA49       48    334 MTTB12 DJNZ    R2,MTTB11       DECR. AND TEST TIME-OUT COUNTER
000151 83         49    335        RET
                        336 *
```

Master transmitter test bus status subroutine 2:

If the SCL clock frequency is fixed, the test on PIN = 0 can be replaced by a fixed delay. Therefore subroutine MTTB2 can be used instead of subroutine MTTB1.

NOTE: this can only be done if none of the slaves are able to stretch the SCL clock.

The delay has to be longer than 9 SCL clock pulses. This subroutine saves some bytes in comparison with subroutine MTTB1.

The contents of register R2 are modified.

```
                        345 *
                        346 * The contents of register R2 are modified.
                        347 *
000152 BA0A       50    348 MTTB2  MOV     R2,#10          LOAD DELAY COUNTER
000154 EA54       51    349        DJNZ    R2,$            DECR. DELAY COUNTER
000156 0D         52    350        MOV     A,S1            FETCH BUS STATUS
000157 D3E0       53    351        XRL     A,#MTTST        TEST MST/TRX BUS STATUS
000159 83         54    352        RET
                        353 *
```

Master transmitter test bus status subroutine 3:

If slaves are able to stretch the SCL clock, the PIN = 0 time out counter can be replaced by a test on the MST bit. In this case the subroutine MTTBS can be used instead of subroutines MTTB1 or MTTB2.

As already mentioned, a STOP condition due to a disturbance of a data bit resets the MST bit. In this case, the subroutine MTTBS will be terminated with A ≠ 0 and the routine ERROR will try to generate a STOP condition if a NOT ACK. situation occurs. The routine ERROR performs an escape sequence and frees the bus if another bus error occurs.

```
                        367 * register contents are not modified.
                        368 * MASTER TRANSMITTER TEST BUS STATUS SUBROUTINE:
                        369 *
00015A 0D         55    370 MTTBS  MOV     A,S1            FETCH BUS STATUS
00015B F25E       56    371        JB7     MTTBS1          JUMP IF MST = 1
00015D 83         57    372        RET
                        373 *
00015E 925A       58    374 MTTBS1 JB4     MTTBS           JUMP IF PIN = 1
000160 D3E0       59    375        XRL     A,#MTTST        TEST MST/TRX BUS STATUS
000162 83         60    376        RET
                        377 *
000163                  378        END
```

3.1.2 Master receiver routine (MAB84X1 - SAB3028)

These routines can only be used if no other masters are connected to the I^2C-bus.

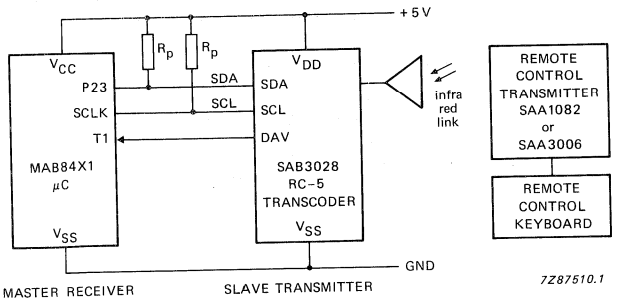


Fig. 3.8 Block diagram of MAB84X1 - SAB3028 configuration.

Initialization:

```
                                11 * SYMBOL DEFINITION:
                                12 *
                                13 IICFR   EQU     H'04'          Fscl =  95 kHz @Fxtal = 6,00 MHz
                                14 *
                                15 MRDTR1  EQU     H'20'          MASTER RECEIVER DATA BYTE 1 REGISTER
                                16 MRDTR2  EQU     MRDTR1+1                                 2     ''
                                17 MRDTR3  EQU     MRDTR2+1                                 3     ''
                                18 MRDTR4  EQU     MRDTR3+1                                 4     ''
                                19 *
                                20 * POWER-ON RESET:
                                21 *
                                22         PAGE    256
                                23 ROM0    ASECT   ROM
                                24         ORG     H'000'
000000                          25 *
000000 2400             1       26 RESET   JMP     INIT           JUMP TO INITIALIZE ROUTINE
                                27 *
                                28 * INITIALISATION:
                                29 *
000002                          30         ORG     H'100'
                                31 *
000100 9E44             2       32 INIT    MOV     S2,#IICFR+ACK  LOAD SCL FREQUENCY WORD
                                33 *                              SET WITH ACKNOWLEDGE MODE
000102 9D18             3       34         MOV     S1,#PIN+ESO    ENABLE SIO
                                35 *                              SET SLAVE RECEIVER MODE
```

Main program:

The main program polls the input T1, which is connected to the data valid output (DAV) of the RC-5 transcoder SAB3028. This output goes low if the RC-5 transcoder has received a valid remote control message. If it is low, the main program calls subroutine RRCT, which reads the data out of the RC-5 transcoder via the I^2C-bus.
The subroutine RRCT returns with Accu = 0 if the data transfer is successful.

The received data is stored in the registers MRDTR1-4. If the transfer is successful DAV is reset to 1.

If the data transfer is not successful ($A \neq 0$), the transmission is repeated.

The format of the data transfer from the RC-5 transcoder SAB3028 is shown in Fig. 3.9:

S	RCT ADR. R A	DATA 1 A	DATA 2 A	DATA 3 A	DATA 4 NA	P

Fig. 3.9 RC-5 transcoder data transfer

S	is the START condition
RCT ADR	is the slave address of RC-5 transcoder
R	is the read/write bit in read state (= 1)
A	is the acknowledge bit; this has to be 0
NA	is the not acknowledge bit, this has to be 1
P	is the STOP condition

```
                          59 * The slave address of RC-5 transcoder is 0100 110.
                          60 RCTAD  EQU     H'4C'
                          61 *
000104 5604         4     62 MAIN   JT1     MAIN            WAIT FOR T1 = DAV = 0
                          63 *
000106 340C         5     64 MAIN1  CALL    RRCT            READ DATA OUT OF RC-5 TRANSCODER
000108 9606         6     65        JNZ     MAIN1           REPEAT IF TRANSMISSION IS NOT SUCCESSFUL
00010A 2404         7     66        JMP     MAIN            CONTINUE
```

Master reads data out of RC-5 transcoder (SAB3028) subroutine:

This subroutine starts to output the slave address and R/W bit (in read state = 1) to the RC-5 transcoder. If this address is not acknowledged, the subroutine is terminated with a STOP condition and with the accumulator contents equal to H'01'. Otherwise the direction of the data transfer changes and the data is clocked out of the RC-5 transcoder. After each of the first three data bytes the microcomputer gives an acknowledge (a LOW level).

To tell the slave that no more data bytes are requested, the master has to give a not-acknowledge (a HIGH level) after the last received data byte.

At that moment the slave knows that the transfer will be terminated and has to allow the SDA line to go HIGH. Now the master is able to generate a STOP condition.

To generate a not-acknowledge the transmission of the last data word is split up in the reception of 8 bits of data and the generation of a "not-acknowledge" bit. After the reception of the third data byte the acknowledge mode bit in S2 is set in the "without acknowledge" state. Then the computer receives 8 bits of data. If PIN becomes '0' again, the bit-counter is set to 1 and the computer inputs the last bit. In input mode, the SDA output is HIGH, therefore the slave will receive an acknowledge bit which is a 1.

Before generating a STOP condition, the acknowledge mode bit in S2 has to be reset to the "acknowledge" state.

If during the transmission of the slave address a disturbance on the data line causes an error (for example an arbitration lost, start or stop condition or a NO ACK is received), the subroutine jumps to label ERROR whereupon the program tests the ACK bit. Upon the result of this test, the SIO will either output a STOP condition (no acknowledge) or invokes a re-initialize routine to free the SIO interface.

The same happens if, during the reception of data, a disturbance occurs.

exit: A = 0 if the transmission is successful, with the received data stored in registers MRDTR1-4.
A ≠ 0 if the transmission is not successful.

The contents of register R1 is modified.

```
00010C 9D18          8   103 RRCT   MOV     S1,#PIN+ESO       RESET SIO STATUS
00010E 9C4D          9   104         MOV     S0,#RCTAD+RW      LOAD SLAVE ADDRESS AND READ BIT
000110 9DF8         10   105         MOV     S1,#STRTC         OUTPUT START COND, SLAVE ADDRESS AND READ BIT
000112 343D         11   106         CALL    MRTBS             TEST BUS STATUS
000114 9646         12   107         JNZ     ERROR             JUMP IF NO ACKN. RECEIVED OR ERROR
000116 0C           13   108         MOV     A,S0              START RECEPTION OF DATA
000117 B920         14   109         MOV     R1,#MRDTR1        SET INDEX TO DATA REGISTER IN WHICH FIRST
                         110 *                                  RECEIVED DATA BYTE HAS TO BE STORED
000119 343D         15   111         CALL    MRTBS             TEST BUS STATUS
00011B 9646         16   112         JNZ     ERROR             JUMP IF ERROR
00011D 343A         17   113         CALL    STORDT            STORE FIRST DATA BYTE AND TEST BUS STATUS
00011F 9646         18   114         JNZ     ERROR             JUMP IF ERROR
000121 343A         19   115         CALL    STORDT            STORE SECOND DATA BYTE AND TEST BUS STATUS
000123 9646         20   116         JNZ     ERROR             JUMP IF ERROR
000125 9E04         21   117         MOV     S2,#IICFR         SET WITHOUT ACKNOWLEDGE MODE
000127 343A         22   118         CALL    STORDT            STORE THIRD DATA BYTE AND TEST BUS STATUS
000129 53FE         23   119         ANL     A,#.NOT.LRB       CLEAR LRB
00012B 9633         24   120         JNZ     RRCT1             JUMP IF ERROR
00012D 9DA9         25   121         MOV     S1,#MST+BB+ESO+BCO SET BIT COUNTER TO ONE
00012F 343A         26   122         CALL    STORDT            STORE FOURTH DATA BYTE AND TEST BUS STATUS
                         123 *                                  GENERATE NOT ACKNOWLEDGE
000131 D301         27   124         XRL     A,#LRB            INVERT NOT ACKNOWLEDGE BIT
                         125 *
000133 9E44         28   126 RRCT1  MOV     S2,#IICFR+ACK     SET WITH ACKNOWLEDGE MODE
000135 9646         29   127         JNZ     ERROR             ESCAPE & RESET BUS H/W
                         128 *
000137 9DD8         30   129         MOV     S1,#STOPC         OUTPUT STOP CONDITION
000139 83           31   130         RET
                         131 *
                         132 * STORE DATA AND TEST MASTER RECEIVER BUS STATUS SUBROUTINE
                         133 *
00013A 0C           32   134 STORDT MOV     A,S0              FETCH DATA OUT OF SIO DATA REGISTER
00013B A1           33   135         MOV     @R1,A             STORE DATA IN MST/REC DATA REGISTER
00013C 19           34   136         INC     R1                INCR. INDEX OF DATA REGISTER
                         137 *
                         138 * MASTER RECEIVER TEST BUS STATUS SUBROUTINE
                         139 *
00013D 0D           35   140 MRTBS  MOV     A,S1              FETCH BUS STATUS
00013E F241         36   141         JB7     MRTBS1            JUMP IF MST = 1
000140 83           37   142         RET
                         143 *
000141 923D         38   144 MRTBS1 JB4     MRTBS             JUMP IF PIN = 1
000143 D3A0         39   145         XRL     A,#MRTST          TEST MST/REC BUS STATUS
000145 83           40   146         RET
                         147 *
                         148 * NO ACKNOWLEDGE RECEIVED OR ERROR EXIT ROUTINE
                         149 *
000146 D301         41   150 ERROR  XRL     A,#LRB            INVERT  ACK
000148 C656         42   151         JZ      ERROR1            IF NO ACK. OUTPUT STOP COND.
00014A 9E04         43   152         MOV     S2,#IICFR
00014C 9E44         44   153         MOV     S2,#IICFR+ACK
00014E 9D18         45   154         MOV     S1,#PIN+ESO       RESET BUS STATUS TWICE IF ERROR FEARED
000150 9D18         46   155         MOV     S1,#PIN+ESO       RETURN TO SLAVE MODE
000152 0C           47   156         MOV     A,S0              DUMMY READ
000153 2302         48   157         MOV     A,#2              TO RETURN WITH A ≠ 0
000155 83           49   158         RET
000156 2301         50   159 ERROR1 MOV     A,#1              TO RETURN WITH A = 1
000158 9DD8         51   160         MOV     S1,#STOPC
00015A 83           52   161         RET
                         162 *
00015B                   163         END
```

3.1.3 Master transmitter/receiver routine including general call reset (MAB84X1 - SAB3035/36/37)

These routines can only be used if no other masters are connected to the I^2C-bus.

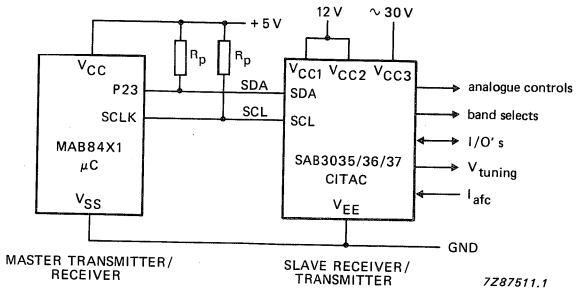


Fig. 3.10 Block diagram MAB84X1 - SAB3035/36/37 CITAC configuration.

Initialization:

```
                                    13 * SYMBOL DEFINITION:
                                    14 *
                                    15 IICFR  EQU     H'04'          Fscl =  95 kHz @Fxtal = 6,00 MHz
                                    16 *
                                    17 * POWER-ON RESET:
                                    18 *
                                    19        PAGE    256
000000                              20 ROM0   ASECT   ROM
                                    21        ORG     H'000'
                                    22 *
000000 2400          1              23 RESET  JMP     INIT           JUMP TO INITIALIZE ROUTINE
                                    24 *
                                    25 * INITIALISATION:
                                    26 *
000002                              27        ORG     H'100'
                                    28 *
000100 9E44          2              29 INIT   MOV     S2,#IICFR+ACK  LOAD SCL FREQUENCY WORD
                                    30 *                            SET WITH ACKNOWLEDGE MODE
000102 9D18          3              31        MOV     S1,#PIN+ESO    ENABLE SIO
                                    32 *                            SET SLAVE RECEIVER MODE
000104 BB00          4              33        MOV     R3,#00         SET DATA BYTE 2; DATA OF ANALOGUE REGISTER 4
```

Main program:

The main program calls the three following I^2C-bus transmission subroutines:

- subroutine GCCIT which resets the CITAC (SAB3035/36/37). The format of this transmission shown in Fig. 3.11:

S	GC ADR	A	RESET CODE	A	P

Fig. 3.11 Data format of transfer to reset CITAC.

- subroutine WCIT which writes two bytes of data to the CITAC. The format is shown in Fig. 3.12:

S	CIT. ADR.	W	A	DATA 1	A	DATA 2	A	P

Fig. 3.12 Data format of data write to CITAC.

- subroutine RCIT which reads two data bytes out of the CITAC. The format is shown in Fig. 3.13:

S	CIT. ADR.	R	A	DATA 1	A	DATA 2	NA	P

Fig. 3.13 Data format of data read from CITAC.

S	is the START condition
CIT ADR	is the slave address of CITAC
GC ADR	is the general call address (H'00')
RESET CODE	is defined as H'06'
W	is the read/write bit in write state (= 0)
R	is the read/write bit in read state (= 1)
A	is the acknowledge bit; this has to be 0
NA	is the not acknowledge bit; this has to be 1
P	is the STOP condition

If one of the transmissions does not succeed (A ≠ 0), it is repeated.

```
                          64 CITAD  EQU     H'C0'
                          65 *
                          66 * The general call address is:
                          67 GCAD   EQU     H'00'
                          68 *
                          69 *
000106 3417          5    70 MAIN   CALL    GCCIT       GENERAL CALL RESET TO CITAC
000108 9606          6    71        JNZ     MAIN        REPEAT IF TRANSMISSION IS NOT SUCCESSFUL
00010A BA24          7    72        MOV     R2,#H'24'   SET DATA BYTE 1; SUBADDRESS OF ANALOGUE 4
00010C 18            8    73        INC     R3          INCR. DATA BYTE 2 TO BE WRITTEN TO CITAC
                          74 *
00010D 342A          9    75 MAIN1  CALL    WCIT        WRITE DATA OF R2 AND R3 TO CITAC
00010F 960D         10    76        JNZ     MAIN1       REPEAT IF TRANSMISSION IS NOT SUCCESSFUL
                          77 *
000111 3443         11    78 MAIN2  CALL    RCIT        READ DATA OUT OF CITAC AND STORE IN R4 AND R5
000113 9611         12    79        JNZ     MAIN2       REPEAT IF TRANSMISSION IS NOT SUCCESSFUL
000115 2406         13    80        JMP     MAIN        CONTINUE
                          81 *
```

Masters writes general call reset code to CITAC (SAB3035/36/37) subroutine:

exit: A = 0 if transmission is successful
A ≠ 0 if transmission is not successful;
register contents are not modified.

```
000117 9D18         14    89 GCCIT  MOV     S1,#PIN+ESO RESET BUS STATUS
000119 9C00         15    90        MOV     S0,#GCAD    LOAD GENERAL CALL ADDRESS
00011B 9DF8         16    91        MOV     S1,#STRTC   OUTPUT START COND. AND GENERAL CALL ADDRESS
00011D 346B         17    92        CALL    MTTBS       TEST BUS STATUS
00011F 967D         18    93        JNZ     ERROR       JUMP IF NO ACKN. RECEIVED OR ERROR
000121 9C06         19    94        MOV     S0,#H'06'   TRANSMIT RESET CODE
000123 346B         20    95        CALL    MTTBS       TEST BUS STATUS
000125 967D         21    96        JNZ     ERROR       JUMP IF NO ACK. RECEIVED OR ERROR
000127 9DD8         22    97        MOV     S1,#STOPC   OUTPUT STOP CONDITION
000129 83           23    98        RET
```

Master writes two data bytes to CITAC (SAB3035/36/37) subroutine:

```
entry: R2 contains first data byte to be transmitted
       R3 contains second data byte
exit: A = 0 if transmission is successful
      A ≠ 0 if transmission is not successful;
      register contents are not modified.
```

```
00012A 9D18          24    108 WCIT   MOV     S1,#PIN+ESO      RESET BUS STATUS
00012C 9CC0          25    109        MOV     S0,#CITAD        LOAD CITAC SLAVE ADDRESS AND WRITE BIT

00012E 9DF8          26    110        MOV     S1,#STRTC        OUTPUT START COND, SLAVE ADDRESS AND WRITE BIT
000130 346B          27    111        CALL    MTTBS            TEST BUS STATUS
000132 967D          28    112        JNZ     ERROR            JUMP IF NO ACKN. RECEIVED OR ERROR
000134 FA            29    113        MOV     A,R2             FETCH FIRST DATA BYTE
000135 3C            30    114        MOV     S0,A             TRANSMIT DATA BYTE 1
000136 346B          31    115        CALL    MTTBS            TEST BUS STATUS
000138 967D          32    116        JNZ     ERROR            JUMP IF NO ACKN. RECEIVED OR ERROR
00013A FB            33    117        MOV     A,R3             FETCH SECOND DATA BYTE
00013B 3C            34    118        MOV     S0,A             TRANSMIT DATA BYTE 2
00013C 346B          35    119        CALL    MTTBS            TEST BUS STATUS
00013E 967D          36    120        JNZ     ERROR            JUMP IF NO ACK.RECEIVED OR ERROR
000140 9DD8          37    121        MOV     S1,#STOPC        OUTPUT STOP CONDITION
000142 83            38    122        RET
```

Master reads two data bytes out of CITAC (SAB3035/36/37) subroutine:

```
exit: A = 0 if transmission is successful and received data is stored in
          registers R4 and R5.
      A ≠ 0 if transmission is not successful.
Register R4 and R5 contents are modified.
```

```
000143 9D18          39    132 RCIT   MOV     S1,#PIN+ESO      RESET SIO STATUS
000145 9CC1          40    133        MOV     S0,#CITAD+RW     LOAD SLAVE ADDRESS AND READ BIT
000147 9DF8          41    134        MOV     S1,#STRTC        OUTPUT START COND, SLAVE ADDRESS AND READ BIT
000149 3474          42    135        CALL    MRTBS            TEST BUS STATUS
00014B 967D          43    136        JNZ     ERROR            JUMP IF NO ACKN. RECEIVED OR ERROR
00014D 0C            44    137        MOV     A,S0             START RECEPTION OF DATA
00014E 3474          45    138        CALL    MRTBS            TEST BUS STATUS
000150 967D          46    139        JNZ     ERROR            JUMP IF NO ACKN. RECEIVED OR ERROR
000152 9E04          47    140        MOV     S2,#IICFR        SET WITHOUT ACKNOWLEDGE MODE
000154 0C            48    141        MOV     A,S0             FETCH FIRST RECEIVED DATA BYTE
000155 AC            49    142        MOV     R4,A             SAVE
000156 3474          50    143        CALL    MRTBS            TEST BUS STATUS
000158 53FE          51    144        ANL     A,#.NOT.LRB      CLEAR LRB
00015A 9664          52    145        JNZ     RCIT1            JUMP IF ERROR
00015C 9DA9          53    146        MOV     S1,#MST+BB+ESO+BC0 SET BIT COUNTER TO ONE
00015E 0C            54    147        MOV     A,S0             FETCH SECOND DATA BYTE
00015F AD            55    148        MOV     R5,A             SAVE DATA BYTE
000160 3474          56    149        CALL    MRTBS            TEST BUS STATUS
                           150 *                               GENERATE NOT ACKNOWLEDGE
000162 D301          57    151        XRL     A,#LRB           INVERT NOT ACKNOWLEDGE BIT
                           152 *
000164 9E44          58    153 RCIT1  MOV     S2,#IICFR+ACK    SET WITH ACKNOWLEDGE MODE
000166 967D          59    154        JNZ     ERROR            JUMP IF ERROR
                           155 *
000168 9DD8          60    156        MOV     S1,#STOPC        OUTPUT STOP CONDITION
00016A 83            61    157        RET
```

Master transmitter test bus status subroutine:

```
                           159 *
                           160 *
00016B 0D            62    161 MTTBS  MOV     A,S1             FETCH BUS STATUS
00016C F26F          63    162        JB7     MTTBS1           JUMP IF MST = 1
00016E 83            64    163        RET
                           164 *
00016F 926B          65    165 MTTBS1 JB4     MTTBS            JUMP IF PIN = 1
000171 D3E0          66    166        XRL     A,#MTTST         TEST MST/TRX BUS STATUS
000173 83            67    167        RET
                           168 *
```

Master receiver test bus status subroutine:

```
                          *
                          *
000174 0D           68    MRTBS   MOV    A,S1            FETCH BUS STATUS
000175 F278         69            JB7    MRTBS1          JUMP IF MST = 1
000177 83           70            RET
                          *
000178 9274         71    MRTBS1  JB4    MRTBS           JUMP IF PIN = 1
00017A D3A0         72            XRL    A,#MRTST        TEST MST/REC BUS STATUS
00017C 83           73            RET
                          *
                          * NO ACKNOWLEDGE RECEIVED OR ERROR EXIT ROUTINE
                          *
00017D D301         74    ERROR   XRL    A,#LRB          INVERT  ACK
00017F C68D         75            JZ     ERROR1          IF NO ACK. OUTPUT STOP COND.
000181 9E04         76            MOV    S2,#IICFR
000183 9E44         77            MOV    S2,#IICFR+ACK
000185 9D18         78            MOV    S1,#PIN+ESO     RESET BUS STATUS TWICE IF ERROR FEARED
000187 9D18         79            MOV    S1,#PIN+ESO     RETURN TO SLAVE MODE
000189 0C           80            MOV    A,S0            DUMMY READ
00018A 2302         81            MOV    A,#2            TO RETURN WITH A ≠ 0
00018C 83           82            RET
00018D 2301         83    ERROR1  MOV    A,#1            TO RETURN WITH A = 1
00018F 9DD8         84            MOV    S1,#STOPC       OUTPUT STOP CONDITION
000191 83           85            RET
                          *
000192                            END
```

3.1.4 Master transmitter/receiver routine with repeated start (MAB84X1 - PCD8571)

These routines can only be used if no other masters are connected to the I^2C-bus.

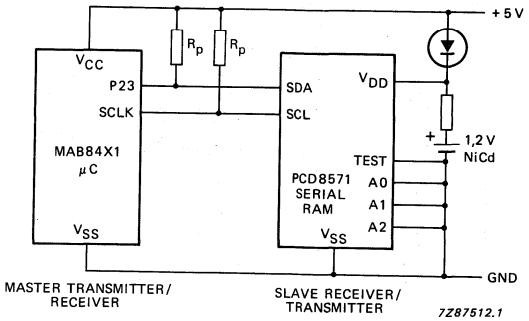


Fig. 3.14 Block diagram MAB84X1 - PCD8571 serial RAM configuration.

Initialization:

```
                                  13 * SYMBOL DEFINITION:
                                  14 *
                                  15 IICFR  EQU     H'04'          Fscl =  95 kHz @Fxtal = 6,00 MHz
                                  16 *
                                  17 * POWER-ON RESET:
                                  18 *
                                  19        PAGE    256
                                  20 ROM0   ASECT   ROM
000000                            21        ORG     H'000'
                                  22 *
000000 2400           1           23 RESET  JMP     INIT           JUMP TO INITIALIZE ROUTINE
                                  24 *
                                  25 * INITIALISATION:
                                  26 *
000002                            27        ORG     H'100'
                                  28 *
000100 9E44           2           29 INIT   MOV     S2,#IICFR+ACK  LOAD SCL FREQUENCY WORD
                                  30 *                             SET WITH ACKNOWLEDGE MODE
000102 9D18           3           31        MOV     S1,#PIN+ESO    ENABLE SIO
                                  32 *                             SET SLAVE RECEIVER MODE
000104 23FF           4           33        MOV     A,#H'FF'
000106 AD             5           34        MOV     R5,A           SET POINTER VALUE
000107 A9             6           35        MOV     R1,A           SET DATA BYTE 1 TO BE WRITTEN
000108 AA             7           36        MOV     R2,A                  ''    2     ''
                                  37 *
```

Main program:

The main program calls the two following I^2C-bus transmission subroutines:

- subroutine WMEM which writes the pointer value and two data bytes to the PCD8571 CMOS memory.
  The format of this transmission is shown in Fig. 3.15:

S	MEM. ADR. W	A	POINT	A	DATA 1	A	DATA 2	A	P

Fig. 3.15 Format to write pointer value and two data bytes to the PCD8571.

- subroutine RMEM which writes the pointer value to the PCD8571 memory and reads two data bytes out of it.
  The format of this transmission is shown in Fig. 3.16:

S	MEM. ADR. W A	POINT A	S MEM. ADR. R A	DATA 1 A	DATA 2 NA	P

Fig. 3.16 Format to write pointer value and read two data bytes.

S	is the START condition
MEM ADR	is the slave address of the CMOS memory PCD8571
POINT	is the pointer address (internal RAM location address)
W	is the read/write bit in write state (= 0)
R	is the read/write bit in read state (= 1)
A	is the acknowledge bit; this has to be 0
NA	is the not acknowledge bit, this has to be 1
P	is the STOP condition

If one of the transmissions does not succeed (A ≠ 0), it is repeated.

```
                          65 * The slave address of the CMOS memory is 1010 XXX.
                          66 MEMAD  EQU     H'A0'
                          67 *
000109 1D            8    68 MAIN   INC     R5              INCR. POINTER VALUE
00010A 2301          9    69        MOV     A,#1            INCR. DATA TO BE WRITTEN
00010C 6A           10    70        ADD     A,R2             R1,R2 + 1 --> R1,R2
00010D AA           11    71        MOV     R2,A
00010E 27           12    72        CLR     A
00010F 79           13    73        ADDC    A,R1
000110 A9           14    74        MOV     R1,A
                          75 *
000111 341B         15    76 MAIN1  CALL    WMEM            WRITE DATA TO CMOS MEMORY
000113 9611         16    77        JNZ     MAIN1           REPEAT IF TRANSMISSION NOT SUCCESSFUL
                          78 *
000115 3443         17    79 MAIN2  CALL    RMEM            READ DATA OUT OF CMOS MEMORY
000117 9615         18    80        JNZ     MAIN2           REPEAT IF TRANSMISSION IS NOT SUCCESSFUL
000119 2409         19    81        JMP     MAIN            CONTINUE
```

Master writes two data bytes to CMOS memory (PCD8571) subroutine:

To write data to the memory, the memory needs to know at which address (location) this data has to be stored. The CMOS memory PCD8571 is designed in such a way that the first data word after the slave address and write bit is always the internal address or pointer.

If more data bytes are written into the memory, during one transmission, this pointer is automatically incremented.

entry: R5 contains the pointer value to be transmitted
R1 contains the first data byte which has to be written in the memory
R2 contains the second data byte which has to be written in the memory

exit: A = 0 if transmission is successful
A ≠ 0 if transmission is not successful;
register contents are not modified.

```
                          97 *
00011B 9D18       20      98 WMEM   MOV     S1,#PIN+ESO     RESET BUS STATUS
00011D 9CA0       21      99        MOV     S0,#MEMAD       LOAD MEMORY SLAVE ADDRESS AND WRITE BIT
00011F 9DF8       22     100        MOV     S1,#STRTC       OUTPUT START COND, SLAVE ADDRESS AND WRITE BIT
000121 343A       23     101        CALL    MTTBS           TEST BUS STATUS
000123 968F       24     102        JNZ     ERROR           JUMP IF NO ACKN. RECEIVED OR ERROR
000125 FD         25     103        MOV     A,R5            FETCH POINTER VALUE
000126 3C         26     104        MOV     S0,A            TRANSMIT POINTER VALUE
000127 343A       27     105        CALL    MTTBS           TEST BUS STATUS
000129 968F       28     106        JNZ     ERROR           JUMP IF NO ACKN. RECEIVED OR ERROR
00012B F9         29     107        MOV     A,R1            FETCH FIRST DATA BYTE
00012C 3C         30     108        MOV     S0,A            TRANSMIT DATA BYTE 1
00012D 343A       31     109        CALL    MTTBS           TEST BUS STATUS

00012F 968F       32     110        JNZ     ERROR           JUMP IF NO ACKN. RECEIVED OR ERROR
000131 FA         33     111        MOV     A,R2            FETCH SECOND DATA BYTE
000132 3C         34     112        MOV     S0,A            TRANSMIT DATA BYTE 2
000133 343A       35     113        CALL    MTTBS           TEST BUS STATUS
000135 968F       36     114        JNZ     ERROR           JUMP IF NO ACKN. RECEIVED OR ERROR
                         115 *
000137 9DD8       37     116 ERROR  MOV     S1,#STOPC       OUTPUT STOP CONDITION
000139 83         38     117        RET
                         118 *
```

Master transmitter test bus status subroutine:

```
00013A 0D         39     121 MTTBS  MOV     A,S1            FETCH BUS STATUS
00013B F23E       40     122        JB7     MTTBS1          JUMP IF MST = 1
00013D 83         41     123        RET
                         124 *
00013E 923A       42     125 MTTBS1 JB4     MTTBS           JUMP IF PIN = 1
000140 D3E0       43     126        XRL     A,#MTTST        TEST MST/TRX BUS STATUS
000142 83         44     127        RET
```

Master reads two data bytes out of CMOS memory (PCD8571) subroutine:

When the master wants to read data out of the memory, it first has to write the pointer into the memory. Since changing of the data direction is only possible after the R/W bit of the slave address, the master has to repeat the START condition, the slave address and the R/W bit (in read state) to change the direction within one transmission.

In the subroutine given below, the MAB84Xi family releases the bus with the MOV S1,#PIN + ESO instruction. Now SDA goes high followed by SCL, if all the slaves release the SCL line and no STOP condition is generated.

If SCL goes high the information on the SDA line is latched into the LRB flag of the serial I/O status register S1. It is possible that slaves can keep the SCL line low, therefore the MAB84X1 tests if the SCL line is released by all slaves connected to the bus. This is done by testing the status bits BB and LRB (see Fig. 3.17).

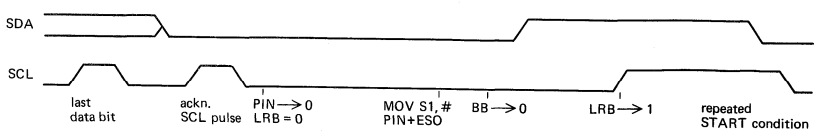


Fig. 3.17 Testing status bits BB and LRB to see whether the SCL line has been released.

As Fig. 3.17 shows, if BB has become 0 and LRB has become 1, a new start condition can be generated.

entry: R5 contains pointer value to be written to the memory.
exit: A = 0 if the transmission is successful and the received data is stored in registers R3 and R4.
A ≠ 0 if the transmission is not successful.
Register R0, R3 and R4 contents are modified.

```
000143 9D18          45   162 RMEM    MOV     S1,#PIN+ESO     RESET SIO STATUS
000145 9CA0          46   163         MOV     S0,#MEMAD       LOAD SLAVE ADDRESS AND WRITE BIT
000147 9DF8          47   164         MOV     S1,#STRTC       OUTPUT START COND, SLAVE ADDRESS AND WRITE BIT
000149 343A          48   165         CALL    MTTBS           TEST BUS STATUS

00014B 968F          49   166         JNZ     ERROR           JUMP IF NO ACKN. RECEIVED OR ERROR
00014D FD            50   167         MOV     A,R5            FETCH POINTER VALUE
00014E 3C            51   168         MOV     S0,A            TRANSMIT POINTER VALUE
00014F 343A          52   169         CALL    MTTBS           TEST BUS STATUS
000151 968F          53   170         JNZ     ERROR           JUMP IF NO ACKN. RECEIVED OR ERROR
000153 9D18          54   171         MOV     S1,#PIN+ESO     RELEASE SDA AND SCL LINE
                          172 *                                NO STOP CONDITION
000155 B80A          55   173         MOV     R0,#10          LOAD REPEATED START COUNTER
                          174 *
000157 0D            56   175 RPSTR1  MOV     A,S1            FETCH BUS STATUS
000158 B25C          57   176         JB5     RPSTR2          JUMP IF BB = 1
00015A 1260          58   177         JB0     RPSTR3          JUMP IF BB = 0 AND LRB = 1
                          178 *                                SCL HAS BECOME HIGH
00015C E857          59   179 RPSTR2  DJNZ    R0,RPSTR1       DECR. AND TEST REP. START COUNTER
00015E 248F          60   180         JMP     ERROR           JUMP IF ERROR
                          181 *
000160 9CA1          61   182 RPSTR3  MOV     S0,#MEMAD+RW    LOAD SLAVE ADDRESS AND READ BIT
000162 9DF8          62   183         MOV     S1,#STRTC       OUTPUT START COND, SLAVE ADDRESS AND READ BIT
000164 3486          63   184         CALL    MRTBS           TEST BUS STATUS
000166 968F          64   185         JNZ     ERROR           JUMP IF NO ACKN. RECEIVED OR ERROR
000168 0C            65   186         MOV     A,S0            START RECEPTION OF DATA
000169 3486          66   187         CALL    MRTBS           TEST BUS STATUS
00016B 968F          67   188         JNZ     ERROR           JUMP IF ERROR
00016D 9E04          68   189         MOV     S2,#IICFR       SET WITHOUT ACKNOWLEDGE MODE
00016F 0C            69   190         MOV     A,S0            FETCH FIRST RECEIVED DATA BYTE
000170 AB            70   191         MOV     R3,A            SAVE
000171 3486          71   192         CALL    MRTBS           TEST BUS STATUS
000173 53FE          72   193         ANL     A,#.NOT.LRB     CLEAR LRB
000175 967F          73   194         JNZ     RMEM1           JUMP IF ERROR
000177 9DA9          74   195         MOV     S1,#MST+BB+ESO+BCO SET BIT COUNTER TO ONE
000179 0C            75   196         MOV     A,S0            FETCH SECOND DATA BYTE
00017A AC            76   197         MOV     R4,A            SAVE DATA BYTE
00017B 3486          77   198         CALL    MRTBS           TEST BUS STATUS
                          199 *                                GENERATE NOT ACKNOWLEDGE
00017D D301          78   200         XRL     A,#LRB          INVERT NOT ACKNOWLEDGE BIT
                          201 *
00017F 9E44          79   202 RMEM1   MOV     S2,#IICFR+ACK   SET WITH ACKNOWLEDGE MODE
000181 968F          80   203         JNZ     ERROR           RESET BUS H/W & ESCAPE
000183 9DD8          81   204         MOV     S1,#STOPC       OUTPUT STOP CONDITION
000185 83            82   205         RET
                          206 *
```

Master receiver test bus status subroutine:

```
                          208 *
000186 0D            83   209 MRTBS   MOV     A,S1            FETCH BUS STATUS
000187 F28A          84   210         JB7     MRTBS1          JUMP IF MST = 1
000189 83            85   211         RET
                          212 *
00018A 9286          86   213 MRTBS1  JB4     MRTBS           JUMP IF PIN = 1
00018C D3A0          87   214         XRL     A,#MRTST        TEST MST/REC BUS STATUS
00018E 83            88   215         RET
                          216 *
                          217 * NO ACKNOWLEDGE RECEIVED OR ERROR EXIT ROUTINE
                          218 *
00018F D301          89   219 ERROR   XRL     A,#LRB          INVERT  ACK
000191 C69F          90   220         JZ      ERROR1          IF NOT.ACK OUTPUT STOP COND.
000193 9E04          91   221         MOV     S2,#IICFR+.NOT.ACK

000195 9E44          92   222         MOV     S2,#IICFR+ACK
000197 9D18          93   223         MOV     S1,#PIN+ESO     RESET BUS STATUS TWICE IF ERROR FEARED
000199 9D18          94   224         MOV     S1,#PIN+ESO     RETURN TO SLAVE MODE
00019B 0C            95   225         MOV     A,S0            DUMMY READ
00019C 2302          96   226         MOV     A,#2            TO RETURN WITH A ≠ 0
00019E 83            97   227         RET
00019F 2301          98   228 ERROR1  MOV     A,#1            TO RETURN WITH A = 1
0001A1 9DD8          99   229         MOV     S1,#STOPC       OUTPUT STOP CONDITION
0001A3 83           100   230         RET
                          231 *
0001A4                    232         END
```

3.2 Slave routines

All these slave routines are I/O interrupt driven

3.2.1 Slave receiver routine (MAB84X1 - master)

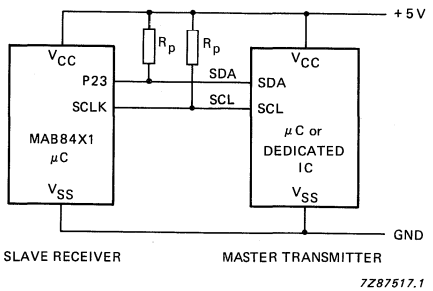


Fig. 3.18 Block diagram MAB84XX - master configuration.

Symbol definition:

```
 8 *
 9 * SYMBOL DEFINITION:
10 IICFR   EQU     H'02'   Fscl = 98,4 kHz @Fcrystal = 4.43 MHz
11 NRSR    EQU     4       number of data bytes which have to be received in
12 *                        SLV/REC mode
13 OWNAD   EQU     H'50'   Own slave address
14 *
```

Data register definition:

```
15 * DATA REGISTER DEFINITION:
16 *
17 * Register Bank 1
18 R11     EQU     H'19'   SIO interrupt data index register
19 R13     EQU     H'1B'   SIO interrupt data byte counter register
20 R14     EQU     H'1C'   SIO interrupt IIC-BUS status register
21 R17     EQU     H'1F'   accu save register during interrupt service subroutines
22 *
23 SRDTR1  EQU     H'3C'           slave receiver data byte 1 register
24 SRDTR2  EQU     SRDTR1+1                  ''              2    ''
25 SRDTR3  EQU     SRDTR2+1                  ''              3    ''
26 SRDTR4  EQU     SRDTR3+1                  ''              4    ''
```

Status flag definition of status register R14:
SRDAVF EQU H'10' SLV/REC data valid flag.

Flag SRDAVF is set in the serial I/O interrupt service subroutine, if the microcomputer has received its own slave address and "NRSR" data bytes.

The flag is cleared in the main program when the received data has been handled.

Initialization:

```
                                37 ROM0   ASECT   ROM
000000                          38        ORG     H'000'
                                39 *
000000 2400           1         40 RESET  JMP     INIT
                                41 *
                                42 * SERIAL I/O INTERRUPT SERVICE SUBROUTINE
                                43 *
000002                          44        ORG     H'005'
                                45 *
000005 4400           2         46 SIOINT JMP     SIOIN1
                                47 *
                                48 * POWER-ON RESET INITIALISATION ROUTINE
                                49 *
000007                          50        ORG     H'100'
                                51 *
000100 B83F           3         52 INIT   MOV     R0,#H'3F'
000102 27             4         53        CLR     A
                                54 *
000103 A0             5         55 INIT1  MOV     @R0,A           CLEAR ALL DATA REGISTERS
000104 E803           6         56        DJNZ    R0,INIT1
                                57 *
                                58 * INITIALIZE SIO
                                59 *
000106 9E42           7         60        MOV     S2,#IICFR+ACK   INITIALISE IIC-BUS FREQUENCY;
                                61 *                               SET WITH ACKNOWLEDGE MODE
000108 9C50           8         62        MOV     S0,#OWNAD       SET OWN SLAVE ADDRESS
00010A 9D18           9         63        MOV     S1,#PIN+ESO     INITALIZE SIO STATUS
00010C 85            10         64        EN      SI
                                65 *
```

Main program:

With this software, data with the format in Fig. 3.19 can be received.

S	OWN ADR W A	DATA 1 A	DATA 2 A	DATA 3 A	DATA 4 A	P

Fig. 3.19 Data format of transfer to slave receiver.

S	is the START condition
OWN ADR	is the device's own slave address
W	is the read/write bit in the write state (= 0)
A	is the acknowledge bit; this has to be zero
P	is the STOP condition

The main program tests the slave receiver data valid "SRDAVF" flag which is set in the serial I/O interrupt service subroutine, if its own slave address, followed by "NRSR" data bytes, is received.

If this flag is set the main program handles the received data stored in register SRDTR1-4, and resets the flag.

If the microcontroller receives its own slave address while the "SRDAVF" flag is set, it releases the bus again and generates a not acknowledge after the first following data byte.

When more than "NRSR" data bytes are received, these data bytes are ignored.

Since the microcontroller cannot function as master in this program, the MST and AL bits are not tested in the interrupt service subroutine.

At the reception of a general call message (ADO = 1), the bus is released every time by reading the register S0. The received data itself is ignored.

This interrupt service subroutine does not serve a slave transmitter. If it is addressed as a slave transmitter, it outputs code FFH.

```
00010D B81C          11      98 MAIN    MOV     R0,#R14
                             99 *
00010F F0            12     100 MAIN1   MOV     A,@R0           FETCH STATUS
000110 37            13     101         CPL     A
000111 920F          14     102         JB4     MAIN1           JUMP IF SRDAVF IS NOT SET
000113 B83C          15     103         MOV     R0,#SRDTR1
000115 B904          16     104         MOV     R1,#4           TRANSFER DATA FROM SRDTR TO R4-R7
000117 BA04          17     105         MOV     R2,#4
                            106 *
000119 F0            18     107 MAIN2   MOV     A,@R0           TRANSFER DATA
00011A A1            19     108         MOV     @R1,A
00011B 19            20     109         INC     R1

00011C 18            21     110         INC     R0
00011D EA19          22     111         DJNZ    R2,MAIN2
00011F B81C          23     112         MOV     R0,#R14
000121 23EF          24     113         MOV     A,#.NOT.SRDAVF
000123 95            25     114         DIS     SI
000124 50            26     115         ANL     A,@R0           RESET SRDAVF
000125 A0            27     116         MOV     @R0,A
000126 85            28     117         EN      SI
000127 240F          29     118         JMP     MAIN1
                            119 *
                            120 * SERIAL I/O INTERRUPT SERVICE SUBROUTINE:
                            121 *
000129                      122         ORG     H'200'
                            123 *
                            124 *
000200 D5            30     125 SIOIN1  SEL     RB1             SELECT REGISTER BANK 1
000201 AF            31     126         MOV     R7,A            SAVE ACCU
000202 0D            32     127         MOV     A,S1            FETCH BUS STATUS
000203 D221          33     128         JB6     STERR           JUMP IF TRANSMITTER
000205 321E          34     129         JB1     SRENA           JUMP IF GENERAL CALL
000207 5217          35     130         JB2     SRADR           JUMP IF ADDRESSED AS SLAVE
                            131 *
000209 F8            36     132 SRDAT   MOV     A,R3            FETCH DATA BYTE COUNTER
00020A C61E          37     133         JZ      SRENA           JUMP IF MORE THAN "NRSR" DATA BYTES RECEIVED
00020C EB12          38     134         DJNZ    R3,SRDAT1       JUMP IF LESS THAN "NRSR" DATA BYTES RECEIVED
00020E FC            39     135         MOV     A,R4
00020F 4310          40     136         ORL     A,#SRDAVF       SET SRDAVF
000211 AC            41     137         MOV     R4,A
                            138 *
000212 0C            42     139 SRDAT1  MOV     A,S0            FETCH RECEIVED DATA BYTE
000213 A1            43     140         MOV     @R1,A           SAVE DATA BYTE IN "SRDTR" REGISTERS
000214 19            44     141         INC     R1              INCR. DATA BYTE INDEX
000215 FF            45     142         MOV     A,R7            RESTORE ACCU
000216 93            46     143         RETR                    RETURN TO MAIN PROGRAM
                            144 *
000217 FC            47     145 SRADR   MOV     A,R4
000218 9223          48     146         JB4     SRNAC           JUMP IF SRDAVF IS ALREADY SET
00021A B93C          49     147         MOV     R1,#SRDTR1      SET DATA BYTE INDEX
00021C BB04          50     148         MOV     R3,#NRSR        SET DATA BYTE COUNTER
                            149 *
00021E 0C            51     150 SRENA   MOV     A,S0            RELEASE BUS
00021F FF            52     151         MOV     A,R7
000220 93            53     152         RETR
                            153 *
000221 1229          54     154 STERR   JB0     STERR1          JUMP IF NO ACKN RECEIVED
                            155 *
000223 9D68          55     156 SRNAC   MOV     S1,#TRX+BB+ESO  GENERATE NOT ACKNOWLEDGE ON NEXT BYTE
000225 9CFF          56     157         MOV     S0,#H'FF'       OUTPUT H'FF'
000227 FF            57     158         MOV     A,R7
000228 93            58     159         RETR
                            160 *
000229 9D38          59     161 STERR1  MOV     S1,#PIN+ESO+BB  SET IN SLV/REC MODE
00022B FF            60     162         MOV     A,R7
00022C 93            61     163         RETR
                            164 *
00022D                      165         END
```

3.2.2 Slave transmitter routine (MAB84XX - master)

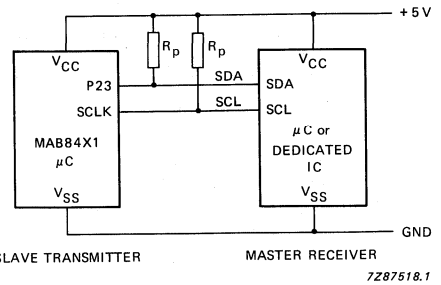


Fig. 3.20 Block diagram MAB84XX - master configuration.

Symbol definition:

```
10 IICFR  EQU     H'02'   Fscl = 98.4 kHz @Fcrystal = 4.43 MHz
11 OWNAD  EQU     H'50'   Own slave address
12 NRST   EQU     4       number of data bytes to be transmitted in SLV/TRM mode
13 *
```

Data register definition:

```
16 * Register Bank 1
17 R11    EQU     H'19'   SIO interrupt data index register
18 R17    EQU     H'1F'   accu save register during interrupt service subroutines
```

slave transmitter data registers:

```
20 * slave transmitter data registers:
21 STDTR1 EQU     H'38'           slave transmitter data byte 1 register
22 STDTR2 EQU     STDTR1+1                    ,,              2   ,,
23 * to
24 STDTRN EQU     STDTR1+NRST-1               ,,              N   ,,
```

Initialization:

```
000000                    31        ORG     H'000'
                          32 *
000000 2400          1    33 RESET  JMP     INIT
                          34 *
                          35 * SERIAL I/O INTERRUPT SERVICE SUBROUTINE:
                          36 *
000002                    37        ORG     H'005'
                          38 *
000005 4400          2    39 SIOINT JMP     SIOIN1
                          40 *
                          41 * POWER-ON RESET INITIALISATION ROUTINE:
                          42 *
000007                    43        ORG     H'100'
                          44 *
000100 B83F          3    45 INIT   MOV     R0,#H'3F'
000102 27            4    46        CLR     A
                          47 *
000103 A0            5    48 INIT1  MOV     @R0,A           CLEAR ALL DATA REGISTERS
000104 E803          6    49        DJNZ    R0,INIT1
                          50 *
                          51 * INITIALIZE SIO ROUTINE:
                          52 *
000106 9E42          7    53        MOV     S2,#IICFR+ACK   INITIALISE IIC-BUS FREQUENCY;
                          54 *                               SET WITH ACKNOWLEDGE MODE
000108 9C50          8    55        MOV     S0,#OWNAD       SET OWN SLAVE ADDRESS
00010A 9D18          9    56        MOV     S1,#PIN+ESO     INITALIZE SIO STATUS
00010C 85           10    57        EN      SI
```

Main program:

With this software, data can be transmitted according the format in Fig. 3.21:

S	OWN ADR. R A	DATA 1 A	DATA 2 A	DATA 3 A	DATA 4 NA	P

Fig. 3.21 Format of a slave transmitter routine

S	is the START condition
OWN ADR	is the device's own slave address
R	is the read/write bit in read state (= 1)
A	is the acknowledge bit; this has to be '0'
NA	is the not acknowledge bit; this has to be '1'
P	is the STOP condition

The main program increments the contents of the slave transmitter data registers STDTR1-STDTRN when the bus is free.

The serial I/O interrupt service subroutine outputs the contents of the slave transmitter data registers if the microcomputer is addressed as slave transmitter.

If a "not acknowledge" is received, it switches to the slave receiver mode to enable the master to generate a STOP condition.

If the microcomputer is addressed as slave receiver (own address or general call address) it releases the bus and ignores the received data.

In this program the microcomputer cannot function as a master; therefore the MST and AL bits are not tested in the interrupt service subroutine.

```
                        83 *
00010D 0D           11  84 MAIN   MOV     A,S1
00010E B20D         12  85        JB5     MAIN            WAIT FOR BB = 0
000110 95           13  86        DIS     SI              DISABLE INTERRUPT
000111 B93B         14  87        MOV     R1,#STDTRN
000113 BA04         15  88        MOV     R2,#NRST
000115 97           16  89        CLR     C
000116 A7           17  90        CPL     C
                        91 *
000117 F1           18  92 MAIN1  MOV     A,@R1           INCR. CONTENTS OF REGISTERS STDTR1-N
000118 1300         19  93        ADDC    A,#0
00011A A1           20  94        MOV     @R1,A
00011B C9           21  95        DEC     R1
00011C EA17         22  96        DJNZ    R2,MAIN1
00011E 85           23  97        EN      SI
00011F EA1F         24  98        DJNZ    R2,$
000121 240D         25  99        JMP     MAIN
                       100 *
                       101 *
                       102 * SERIAL I/O INTERRUPT SERVICE SUBROUTINE:
                       103 *
000123                 104        ORG     H'200'
                       105 *
000200 D5           26  106 SIOIN1 SEL     RB1             SELECT REGISTER BANK 1
000201 AF           27  107       MOV     R7,A            SAVE ACCU
000202 0D           28  108       MOV     A,S1            FETCH BUS STATUS
000203 D208         29  109       JB6     SLVTRM          JUMP IF TRANSMITTER

000205 0C           30  110       MOV     A,S0            RELEASE SCL
000206 FF           31  111       MOV     A,R7
000207 93           32  112       RETR
                       113 *
000208 5211         33  114 SLVTRM JB2     STADR           JUMP IF ADDRESSED AS SLAVE
00020A 37           34  115       CPL     A
00020B 1213         35  116       JB0     STDAT           JUMP IF ACKN RECEIVED
00020D 9D38         36  117       MOV     S1,#PIN+ESO+BB  SET IN SLV/REC MODE
00020F FF           37  118       MOV     A,R7
000210 93           38  119       RETR
                       120 *
000211 B938         39  121 STADR  MOV     R1,#STDTR1      SET DATA BYTE INDEX
                       122 *
000213 F1           40  123 STDAT  MOV     A,@R1           FETCH DATA BYTE
000214 3C           41  124       MOV     S0,A            TRM DATA BYTE
000215 19           42  125       INC     R1              INCR. DATA BYTE INDEX
000216 FF           43  126       MOV     A,R7
000217 93           44  127       RETR
                       128 *
000218                 129        END
```

3.2.3 Slave transmitter/receiver routine including general call reception (MAB84X1 - master)

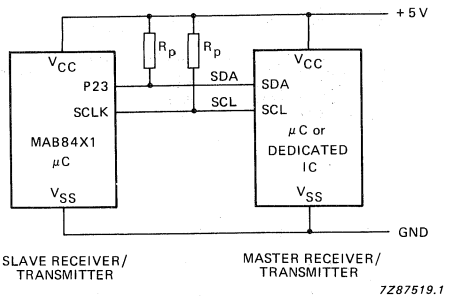


Fig. 3.22 Block diagram MAB84X1 - master configuration.

Symbol definition:

```
IICFR    EQU  H'02'     Fscl = 98,4 kHz @ fcrystal = 4,43 MHz.
NRSR     EQU  4         Number of data bytes which have to be received in
                        SLV/REC mode.
NRGC     EQU  4         Number of data bytes which have to be received in
                        GENERAL CALL mode.
NRST     EQU  4         Number of data bytes to be transmitted in SLV/TRM
                        mode.
OWNAD    EQU  H'50'     Own slave address.
```

Data register definition:

```
Register Bank 1

R11      EQU  H'19'     SIO interrupt data index register
R13      EQU  H'1B'     SIO interrupt data byte counter register
R14      EQU  H'1C'     SIO interrupt I²C-bus status register

Status flag definition of status register R14:

SRDAVF   EQU  H'10'     SLV/REC data valid flag
```

The flag SRDAVF is set in the serial I/O interrupt service subroutine, if the computer has received its own address and "NRSR" data bytes. It is cleared in the main program when the received data has been handled.

```
GCDAVF   EQU  H'08'     GENERAL CALL data valid flag
```

The flag GCDAVF is set in the serial I/O interrupt service subroutine, if the computer has received the general call address and "NRGC" data bytes. It is cleared in the main program when the received data has been handled.

```
R17      EQU  H'1F'     Accumulator-save register during interrupt service
                        subroutines.
```

Slave transmitter data registers:

```
                                36 STDTR1 EQU     H'38'          slave transmitter data byte 1 register
                                37 STDTR2 EQU     STDTR1+1                    ''              2   ''
                                38 * to
                                39 STDTRN EQU     STDTR1+NRST-1               ''              N   ''
                                40 *
                                41 * slave receiver data registers:
                                42 SRDTR1 EQU     H'3C'          slave receiver data byte 1 register
                                43 SRDTR2 EQU     SRDTR1+1                    ''              2   ''
                                44 SRDTR3 EQU     SRDTR2+1                    ''              3   ''
                                45 SRDTR4 EQU     SRDTR3+1                    ''              4   ''
```

Initialization:

```
                                51 *
000000                          52        ORG     H'000'
                                53 *
000000 2400           1         54 RESET  JMP     INIT
                                55 *
                                56 * SERIAL I/O INTERRUPT SERVICE SUBROUTINE:
                                57 *
000002                          58        ORG     H'005'
                                59 *
000005 4400           2         60 SIOINT JMP     SIOIN1
                                61 *
                                62 * POWER-ON RESET INITIALISATION ROUTINE:
                                63 *
000007                          64        ORG     H'100'
                                65 *
000100 B83F           3         66 INIT   MOV     R0,#H'3F'
000102 27             4         67        CLR     A
                                68 *
000103 A0             5         69 INIT1  MOV     @R0,A          CLEAR ALL DATA REGISTERS
000104 E803           6         70        DJNZ    R0,INIT1
                                71 *
                                72 * INITIALIZE SIO ROUTINE:
                                73 *
000106 9E42           7         74        MOV     S2,#IICFR+ACK  INITIALISE IIC-BUS FREQUENCY;
                                75 *                              SET WITH ACKNOWLEDGE MODE
000108 9C50           8         76        MOV     S0,#OWNAD      SET OWN SLAVE ADDRESS
00010A 9D18           9         77        MOV     S1,#PIN+ESO    INITALIZE SIO STATUS
00010C 85            10         78        EN      SI
                                79 *
```

Main program:

This program is a combination of the routines given is Sections 3.2.1/ 3.1.2 extended by dealing with the reception of the GENERAL CALL DATA.

The microcomputer can function as slave according to the following formats:

- Slave receiver general call address

S	GC ADR W A	DATA 1 A	DATA 2 A	DATA 3 A	DATA 4 A	P

Fig. 3.23 Slave receiver general call format.

- Slave receiver

S	OWN ADR W A	DATA 1 A	DATA 2 A	DATA 3 A	DATA 4 A	P

Fig. 3.24 Slave receiver format.

- Slave transmitter

S	OWN ADR R A	DATA 1 A	DATA 2 A	DATA 3 A	DATA 4 NA	P

Fig. 3.25 Slave transmitter format.

S	is the START condition
OWN ADR	is the device's own slave address
GC ADR	is general call address
W	is read/write bit in write state (=0)
R	is the read/write bit in read state (= 1)
A	is the acknowledge bit; this has to be 0
NA	is the not acknowledge bit; this has to be 1
P	is the STOP condition

If the microcomputer is addressed with the GENERAL CALL ADDRESS it receives the data bytes and sets the GENERAL CALL DATA valid flag "GCDAVF".

If the "SRDAVF" flag is already set and the microcomputer is addressed with the GENERAL CALL address, this flag is cleared and data is overwritten.

The flag "GCDAVF" is tested in the main program. If set, it is cleared when the received data has been handled.

```
000100 B81C          11    109 MAIN    MOV     R0,#R14
00010F F0            12    110         MOV     A,@R0           FETCH STATUS
000110 922A          13    111         JB4     MAIN1           JUMP IF SRDAVF IS SET
000112 722A          14    112         JB3     MAIN1           JUMP IF GCDAVF IS SET
000114 0D            15    113         MOV     A,S1
000115 B20D          16    114         JB5     MAIN            JUMP IF BB = 1
000117 95            17    115         DIS     SI              DISABLE INTERRUPT
000118 B93B          18    116         MOV     R1,#STDTRN
00011A BA04          19    117         MOV     R2,#NRST
00011C 97            20    118         CLR     C
00011D A7            21    119         CPL     C
                           120 *
00011E F1            22    121 MAIN0   MOV     A,@R1           INCR. CONTENTS OF REGISTERS STDTR1-N
00011F 1300          23    122         ADDC    A,#0
000121 A1            24    123         MOV     @R1,A
000122 C9            25    124         DEC     R1
000123 EA1E          26    125         DJNZ    R2,MAIN0
000125 85            27    126         EN      SI
000126 EA26          28    127         DJNZ    R2,$            DELAY
000128 240D          29    128         JMP     MAIN
                           129 *
00012A B83C          30    130 MAIN1   MOV     R0,#SRDTR1
00012C B904          31    131         MOV     R1,#4           TRANSFER DATA FROM SRDTR TO R4-R7
00012E BA04          32    132         MOV     R2,#4
                           133 *
000130 F0            33    134 MAIN2   MOV     A,@R0           TRANSFER DATA
000131 A1            34    135         MOV     @R1,A
000132 19            35    136         INC     R1
000133 18            36    137         INC     R0
000134 EA30          37    138         DJNZ    R2,MAIN2
000136 B81C          38    139         MOV     R0,#R14
000138 23E7          39    140         MOV     A,#.NOT.(SRDAVF+GCDAVF)
00013A 95            40    141         DIS     SI
00013B 50            41    142         ANL     A,@R0           RESET SRDAVF OR GCDAVF
00013C A0            42    143         MOV     @R0,A
00013D 85            43    144         EN      SI
00013E 240D          44    145         JMP     MAIN
                           146 *
                           147 *
000140                     148         ORG     H'200'
                           149 *
                           150 * SERIAL I/O INTERRUPT SERVICE SUBROUTINE:
                           151 *
000200 D5            45    152 SIOIN1  SEL     RB1             SELECT REGISTER BANK 1
000201 AF            46    153         MOV     R7,A            SAVE ACCU
000202 0D            47    154         MOV     A,S1            FETCH SIO STATUS
000203 D237          48    155         JB6     SLVTR           JUMP IF TRX = 1
000205 521D          49    156         JB2     SRADR           JUMP IF ADDRESSED AS SLAVE RECEIVER
                           157 *
                           158 * SLV/REC; INPUT DATA SUBROUTINE:
                           159 *
000207 67            50    160 SRDAT   RRC     A
000208 67            51    161         RRC     A               ADD IN CARRY
000209 FB            52    162         MOV     A,R3
00020A C634          53    163         JZ      SRENA           JUMP IF MORE THAN NRSR OR NRGC DATA
                           164 *                                 BYTES RECEIVED
00020C EB16          54    165         DJNZ    R3,SRDAT2       DECR. AND TEST DATA BYTE COUNTER
```

```
00020E 2310          55    166          MOV     A,#SRDAVF       SET SRDAVF
000210 E614          56    167          JNC     SRDAT1          JUMP IF ADO = 0
000212 2308          57    168          MOV     A,#GCDAVF       SET GCDAVF
                           169 *
000214 4C            58    170 SRDAT1   ORL     A,R4            SET SRDAVF OR GCDAVF FLAG
000215 AC            59    171          MOV     R4,A
                           172 *
000216 0C            60    173 SRDAT2   MOV     A,S0            FETCH RECEIVED DATA BYTE
000217 A1            61    174          MOV     @R1,A           SAVE RECEIVED DATA BYTE
000218 19            62    175          INC     R1              INCR. DATA BYTE INDEX
000219 FF            63    176          MOV     A,R7
00021A 93            64    177          RETR
                           178 *
                           179 * SLV; ADDRESSED AS SLAVE SUBROUTINE:
                           180 *
00021B D240          65    181 SLADR    JB6     STADR           JUMP IF ADDRESSED AS SLAVE TRANSMITTER
                           182 *
                           183 * SLV/REC; ADDRESSED AS SLAVE RECEIVER SUBROUTINE:
                           184 *
00021D B93C          66    185 SRADR    MOV     R1,#SRDTR1      SET DATA REGISTER INDEX
00021F 322E          67    186          JB1     GCADR           JUMP IF GENERAL SLAVE ADDRESS
000221 BB04          68    187          MOV     R3,#NRSR        SET SLV/REC DATA BYTE COUNTER
000223 FC            69    188          MOV     A,R4
000224 5318          70    189          ANL     A,#SRDAVF+GCDAVF
000226 C634          71    190          JZ      SRENA           JUMP IF SRDAVF AND GCDAVF ARE NOT SET
                           191 *                                  CONT. WITH RECEPTION OF DATA
000228 9D68          72    192          MOV     S1,#TRX+BB+ESO  SET SLAVE TRANSMITTER MODE
00022A 9CFF          73    193          MOV     S0,#H'FF'       OUTPUT H'FF' AND NEGATIVE ACKN.
00022C FF            74    194          MOV     A,R7
00022D 93            75    195          RETR
                           196 *
                           197 * SLV/REC; ADDRESSED WITH GENERAL CALL ADDRESS SUBROUTINE:
                           198 *
00022E BB04          76    199 GCADR    MOV     R3,#NRGC        SET GENERAL CALL ADDRESS DATA BYTE
                           200 *                                  COUNTER
000230 FC            77    201          MOV     A,R4
000231 53E7          78    202          ANL     A,#.NOT.(SRDAVF+GCDAVF) RESET SRDAVF AND GCDAVF FLAGS
000233 AC            79    203          MOV     R4,A
                           204 *
                           205 * SLV/REC; ENABLE SLC SUBROUTINE:
                           206 *
000234 0C            80    207 SRENA    MOV     A,S0            RELEASE SCL
000235 FF            81    208          MOV     A,R7
000236 93            82    209          RETR
                           210 *
                           211 * SLAVE TRANSMITTER SUBROUTINE:
                           212 *
000237 37            83    213 SLVTR    CPL     A               COMPL. STATUS BITS
000238 123E          84    214          JB0     SLVTR1          JUMP IF ACKN. RECEIVED
                           215 *
                           216 * SLV/TRX; NEGATIVE ACKNOWLEDGE RECEIVED SUBROUTINE:
                           217 *
00023A 9D38          85    218 STNAC    MOV     S1,#BB+PIN+ESO  SET SLV/REC MODE
00023C FF            86    219          MOV     A,R7
00023D 93            87    220          RETR
                           221 *
                           222 * SLV/TRX; ACKNOWLEDGE RECEIVED SUBROUTINE:
                           223 *
00023E 5242          88    224 SLVTR1   JB2     STDAT           JUMP IF AAS = 0
                           225 *
                           226 * SLV/TRX; ADDRESSED AS SLAVE TRANSMITTER SUBROUTINE:
                           227 *
000240 B938          89    228 STADR    MOV     R1,#STDTR1      SET DATA BYTE REG. INDEX
                           229 *
000242 F1            90    230 STDAT    MOV     A,@R1           FETCH NEW DATA BYTE
000243 3C            91    231          MOV     S0,A            OUTPUT DATA BYTE
000244 19            92    232          INC     R1              INCR. DATA BYTE INDEX
000245 FF            93    233          MOV     A,R7
000246 93            94    234          RETR
                           235 *
000247                     236          END
```

3.3 Multi - master routines

3.3.1 Master transmitter/receiver routine with repeated start (MAB84X1-PCD8571)

These routines can be used in a multi-master system.

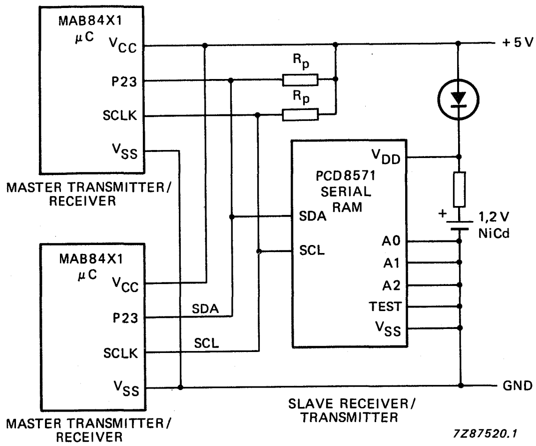


Fig. 3.26 Block diagram multi-master MAB84X1 - PCD8571 configuration.

Initialization:

In a multi master system a microcomputer can be addressed with the general call address and with its own address. In this example the computer can not function as slave so no own slave address is defined. After power-on reset the contents of slave address register is 00H which means that its own slave address is equal to the general call address.

```
000000                   ORG    H'000'
                 *
000000 2400    1  RESET  JMP    INIT              JUMP TO INITIALIZE ROUTINE
                 *
                 *
                 * SERIAL I/O INTERRUPT SERVICE SUBROUTINE ENTRY
                 *
000002                   ORG    H'005'
                 *
000005 24C2    2  SIOINT JMP    SIOIN1            JUMP TO SIO INTERRUPT SERVICE SUBROUTINE
                 *
000007                   ORG    H'100'
                 *
000100 9E44    3  INIT   MOV    S2,#IICFR+ACK     LOAD SCL FREQUENCY WORD
                 *                                SET WITH ACKNOWLEDGE MODE
000102 9D38    4         MOV    S1,#PIN+ESO+BB    ENABLE SIO
                 *                                SET SLAVE RECEIVER MODE
000104 23FF    5         MOV    A,#H'FF'
000106 AD      6         MOV    R5,A              SET POINTER VALUE
000107 A9      7         MOV    R1,A              SET DATA BYTE 1 TO BE WRITTEN
000108 AA      8         MOV    R2,A                  ,,        2      ,,
000109 85      9         EN     SI                ENABLE SIO INTERRUPTS
```

Main program:

The main program calls the two following I^2C-bus transmission subroutines:

- subroutine WMEM which writes the pointer value and two data bytes to the PCD8571 CMOS memory.
  The format of this transmission is shown in Fig. 3.27:

S	MEM. ADR.	W	A	POINT	A	DATA 1	A	DATA 2	A	P

Fig. 3.27 Format of pointer value and 2 data bytes write to PCD8571.

- subroutine RMEM which writes the pointer value to the PCD8571 memory and reads two data bytes out of it.
  The format of this transmission is shown in Fig. 3.28:

S	MEM. ADR.	W	A	POINT	A	S	MEM. ADR.	R	A	DATA 1	A	DATA 2	NA	P

Fig. 3.28 Format of pointer value write 2 data bytes read with PCD8571.

S	is the START condition
MEM ADR	is the slave address of the CMOS memory PCD8571
POINT	is the pointer address (internal RAM location address)
W	is the read/write bit in write state (= 0)
R	is the read/write bit in read state (= 1)
A	is the acknowledge bit; this has to be 0
NA	is the not acknowledge bit; this has to be 1
P	is the STOP condition

The subroutines given in these examples are the same as given in Section 3.1.4 but are extended to a multi-master system.

If one of the transmissions does not succeed (A ≠ 0), this transmission is repeated.

```
                          82 * The slave address of the CMOS memory is 1010 XXX.
                          83 MEMAD  EQU     H'A0'
                          84 *
00010A 1D           10    85 MAIN   INC     R5              INCR. POINTER VALUE
000108 2301         11    86        MOV     A,#1            INCR. DATA TO BE WRITTEN
00010D 6A           12    87        ADD     A,R2             R1,R2 + 1 --> R1,R2
00010E AA           13    88        MOV     R2,A
00010F 27           14    89        CLR     A
000110 79           15    90        ADDC    A,R1
000111 A9           16    91        MOV     R1,A
                          92 *
000112 3420         17    93 MAIN1  CALL    WMEM            WRITE DATA TO CMOS MEMORY
000114 9612         18    94        JNZ     MAIN1           REPEAT IF TRANSMISSION IS NOT SUCCESSFUL
000116 EE16         19    95        DJNZ    R6,$            DELAY
                          96 *
000118 346A         20    97 MAIN2  CALL    RMEM            READ DATA OUT OF CMOS MEMORY
00011A 9618         21    98        JNZ     MAIN2           REPEAT IF TRANSMISSION IS NOT SUCCESSFUL
00011C EE1C         22    99        DJNZ    R6,$            DELAY
00011E 240A         23   100        JMP     MAIN            CONTINUE
                         101 *
```

Master writes two data bytes to CMOS memory (PCD8571) subroutine:

Before starting the transmission this subroutine tests for a maximum time of 7 x 256 x 30 x txtal to see if the bus has become free. If not, it resets the bus-busy bit "BB" with the MOV S1, #PIN + ESO instruction and initializes the other status bits with a second MOV S1, #PIN + ESO instruction. To release an addresed slave transmitter the subroutine BBTST starts a dummy master transmission with slave address (and R/W bit) H'FF'. This dummy transmission is repeated until the bus is free again.

During the initialization, the serial I/O interrupt is enabled to release the bus if the computer receives the general call address or its data.

As the subroutines WMEM and RMEM are based on polling the PIN bit, the serial I/O interrupt is disabled during the execution of these subroutines.

If during the transmission, a not-acknowledge is received the transfer is terminated with a STOP condition and with Accu ≠ 0.

If an arbitration lost situation is detected, the subroutine terminates with Accu ≠ 0. It also enables the serial I/O interrupt again. Now the interrupt service subroutine will release the bus and reset the PIN bit.

entry: R5 contains the pointer value to be transmitted
R1 contains the first data byte which has to be written in the memory
R2 contains the second data byte which has to be written in the memory
exit: A = 0 if transmission is successful
A ≠ 0 if transmission is not successful.
The contents of register R0 are modified.

```
000120 3445          24    129 WMEM   CALL    BBTST           TEST BUS BUSY
000122 95            25    130        DIS     SI              DISABLE INTERRUPT
000123 98FB          26    131        ANL     P0,#.NOT.TRIG1  TRIGGER !!!!!!
000125 8804          27    132        ORL     P0,#TRIG1       TRIGGER !!!!!!
000127 9CA0          28    133        MOV     S0,#MEMAD       LOAD MEMORY SLAVE ADDRESS AND WRITE BIT
000129 9DF8          29    134        MOV     S1,#STRTC       OUTPUT START COND, SLAVE ADDRESS AND WRITE BIT
00012B 3461          30    135        CALL    MTTBS           TEST BUS STATUS
00012D 96B3          31    136        JNZ     ERMEM2          JUMP IF NO ACKN. RECEIVED OR ERROR
00012F FD            32    137        MOV     A,R5            FETCH POINTER VALUE
000130 3C            33    138        MOV     S0,A            TRANSMIT POINTER VALUE
000131 3461          34    139        CALL    MTTBS           TEST BUS STATUS
000133 96B3          35    140        JNZ     ERMEM2          JUMP IF NO ACKN. RECEIVED OR ERROR
000135 F9            36    141        MOV     A,R1            FETCH FIRST DATA BYTE
000136 3C            37    142        MOV     S0,A            TRANSMIT DATA BYTE 1
000137 3461          38    143        CALL    MTTBS           TEST BUS STATUS
000139 96B3          39    144        JNZ     ERMEM2          JUMP IF NO ACKN. RECEIVED OR ERROR
00013B FA            40    145        MOV     A,R2            FETCH SECOND DATA BYTE
00013C 3C            41    146        MOV     S0,A            TRANSMIT DATA BYTE 2
00013D 3461          42    147        CALL    MTTBS           TEST BUS STATUS
00013F 96B3          43    148        JNZ     ERMEM2          JUMP IF NO ACKN. RECEIVED OR ERROR
                           149 *
000141 9DD8          44    150 STPC   MOV     S1,#STOPC       OUTPUT STOP CONDITION
000143 85            45    151        EN      SI              ENABLE SIO INTERRUPTS
000144 83            46    152        RET
```

Bus busy test subroutine:

```
000145 B880          47    156 BBTST   MOV     R0,#H'80'       LOAD BUS BUSY COUNTER
000147 85            48    157         EN      SI
                           158 *
000148 0D            49    159 BBTST1  MOV     A,S1            WAIT UNTILL BUS IS FREE FOR MAXIMUM
000149 37            50    160         CPL     A                7 X 256 X 30 X Txtal
00014A B269          51    161         JB5     BBTST2          JUMP IF BUS IS FREE
00014C E848          52    162         DJNZ    R0,BBTST1       DECR AND TEST BUS BUSY COUNTER
00014E 9D18          53    163         MOV     S1,#PIN+ESO     RESET BB BIT; BUS IS FREE NOW
000150 9D18          54    164         MOV     S1,#PIN+ESO     INITIALIZE OTHER STATUS BITS
000152 9E44          55    165         MOV     S2,#IICFR+ACK   INITIALIZE IIC-FREQ. AND ACKN. MODE BIT
000154 95            56    166         DIS     SI              DISABLE SERIAL I/O INTERRUPTS
000155 9CFF          57    167         MOV     S0,#H'FF'       LOAD SIO DATA REGISTER
000157 9DF8          58    168         MOV     S1,#STRTC       OUTPUT START CONDITION AND H'FF' AS SLAVE
                           169 *                                ADDRESS
000159 34B9          59    170         CALL    MRTBS           TEST BUS STATUS
00015B D301          60    171         XRL     A,#LRB          INVERT ACKN. BIT
00015D 9645          61    172         JNZ     BBTST           JUMP IF ERROR
00015F 2441          62    173         JMP     STPC            OUTPUT STOP CONDITION
```

Master transmitter test bus status subroutine:

```
000161 0D            63    177 MTTBS   MOV     A,S1            FETCH BUS STATUS
000162 F265          64    178         JB7     MTTBS1          JUMP IF MST = 1
000164 83            65    179         RET
                           180 *
000165 9261          66    181 MTTBS1  JB4     MTTBS           JUMP IF PIN = 1
000167 D3E0          67    182         XRL     A,#MTTST        TEST MST/TRX BUS STATUS
                           183 *
000169 83            68    184 BBTST2  RET
```

Master reads two data bytes out of CMOS memory (PCD8571) subroutine:

entry: R5 contains the pointer value to be written to the memory.
exit: A = 0 if transmission is successful and the received data is stored in registers R3 and R4.
A ≠ 0 if the transmission is not successful.
The contents of register R0, R3 and R4 are modified.

```
00016A 3445          69    194 RMEM    CALL    BBTST           TEST BUS BUSY
00016C 95            70    195         DIS     SI              DISABLE INTERRUPT
00016D 98F7          71    196         ANL     P0,#.NOT.TRIG2  TRIGGER !!!!!!
00016F 8808          72    197         ORL     P0,#TRIG2       TRIGGER !!!!!!
000171 9CA0          73    198         MOV     S0,#MEMAD       LOAD SLAVE ADDRESS AND WRITE BIT
000173 9DF8          74    199         MOV     S1,#STRTC       OUTPUT START COND, SLAVE ADDRESS AND WRITE BIT
000175 3461          75    200         CALL    MTTBS           TEST BUS STATUS
000177 96B3          76    201         JNZ     ERMEM2          JUMP IF NO ACKN. RECEIVED OR ERROR
000179 FD            77    202         MOV     A,R5            FETCH POINTER VALUE
00017A 3C            78    203         MOV     S0,A            TRANSMIT POINTER VALUE
00017B 3461          79    204         CALL    MTTBS           TEST BUS STATUS
00017D 96B3          80    205         JNZ     ERMEM2          JUMP IF NO ACKN. RECEIVED OR ERROR
00017F 9D18          81    206         MOV     S1,#PIN+ESO     RELEASE SDA AND SCL LINE
                           207 *                                NO STOP CONDITION
000181 B80A          82    208         MOV     R0,#10          LOAD REPEATED START COUNTER
                           209 *
000183 0D            83    210 RPSTR1  MOV     A,S1            FETCH BUS STATUS
000184 B288          84    211         JB5     RPSTR2          JUMP IF BB = 1
000186 128C          85    212         JB0     RPSTR3          JUMP IF BB = 0 AND LRB = 1
                           213 *                                SCL HAS BECOME HIGH
000188 E883          86    214 RPSTR2  DJNZ    R0,RPSTR1       DECR. AND TEST REP. START COUNTER
00018A 24B3          87    215         JMP     ERMEM2          JUMP IF ERROR
                           216 *
00018C 9CA1          88    217 RPSTR3  MOV     S0,#MEMAD+RW    LOAD SLAVE ADDRESS AND READ BIT
00018E 9DF8          89    218         MOV     S1,#STRTC       OUTPUT START COND, SLAVE ADDRESS AND READ BIT
000190 34B9          90    219         CALL    MRTBS           TEST BUS STATUS
000192 96B3          91    220         JNZ     ERMEM2          JUMP IF NO ACKN. RECEIVED OR ERROR
000194 0C            92    221         MOV     A,S0            START RECEPTION OF DATA
000195 34B9          93    222         CALL    MRTBS           TEST BUS STATUS
000197 96B3          94    223         JNZ     ERMEM2          JUMP IF ERROR
000199 9E04          95    224         MOV     S2,#IICFR       SET WITHOUT ACKNOWLEDGE MODE
00019B 0C            96    225         MOV     A,S0            FETCH FIRST RECEIVED DATA BYTE
00019C AB            97    226         MOV     R3,A            SAVE
00019D 34B9          98    227         CALL    MRTBS           TEST BUS STATUS
00019F 53FE          99    228         ANL     A,#.NOT.LRB     CLEAR LRB
```

```
0001A1 96B1          100   229          JNZ     ERMEM1          JUMP IF ERROR
0001A3 9DA9          101   230          MOV     S1,#MST+BB+ESO+BCO SET BIT COUNTER TO ONE
0001A5 0C            102   231          MOV     A,S0            FETCH SECOND DATA BYTE
0001A6 AC            103   232          MOV     R4,A            SAVE DATA BYTE
0001A7 34B9          104   233          CALL    MRTBS           TEST BUS STATUS
                           234 *                                GENERATE NOT ACKNOWLEDGE
0001A9 D301          105   235          XRL     A,#LRB          INVERT NOT ACKNOWLEDGE BIT
0001AB 9681          106   236          JNZ     ERMEM1          JUMP IF ERROR
0001AD 9E44          107   237          MOV     S2,#IICFR+ACK   SET WITH ACKNOWLEDGE MODE
0001AF 2441          108   238          JMP     STPC            OUTPUT STOP CONDITION
                           239 *
0001B1 9E44          109   240 ERMEM1   MOV     S2,#IICFR+ACK   SET WITH ACKNOWLEDGE MODE
                           241 *
0001B3 0D            110   242 ERMEM2   MOV     A,S1            FETCH SIO STATUS
0001B4 F241          111   243          JB7     STPC            JUMP IF MASTER; NO ACKN. RECEIVED
0001B6 85            112   244          EN      SI              ENABLE SIO INTERRUPTS
0001B7 37            113   245          CPL     A               A # 0; TRANSMISSION NOT SUCCESSFUL
0001B8 83            114   246          RET
```

Master receiver test bus status subroutine:

```
0001B9 0D            115   250 MRTBS    MOV     A,S1            FETCH BUS STATUS
0001BA F2BD          116   251          JB7     MRTBS1          JUMP IF MST = 1
0001BC 83            117   252          RET
                           253 *
0001BD 92B9          118   254 MRTBS1   JB4     MRTBS           JUMP IF PIN = 1
0001BF D3A0          119   255          XRL     A,#MRTST        TEST MST/REC BUS STATUS
0001C1 83            120   256          RET
```

Serial I/O interrupt service subroutine:

In a multi-master system a general call message could be transmitted; every MAB84X1 microcontroller has to respond to this. In this example it is done by enabling the SIO interrupt and setting the PIN bit with the instruction MOV S0,A in the SIO interrupt service subroutine.

In the case of an arbitration lost, this instruction also resets the bits AL and AAS.

```
0001C2 3C            121   266 SIOIN1   MOV     S0,A            SET PIN, RESET AL AND AAS BITS;
                           267 *                                ENABLE SCL IF GENERAL CALL ADDRESS OR ITS
                           268 *                                DATA RECEIVED OR IF ARBITRATION LOST IN THE
                           269 *                                MASTER TRANSMITTER MODE
0001C3 93            122   270          RETR
                           271 *
0001C4                     272          END
```

3.3.2 Master transmitter/receiver routine; slave transmitter/receiver routine (MAB84X1 - MAB84X1)

These routines can be used in a multi-master system.

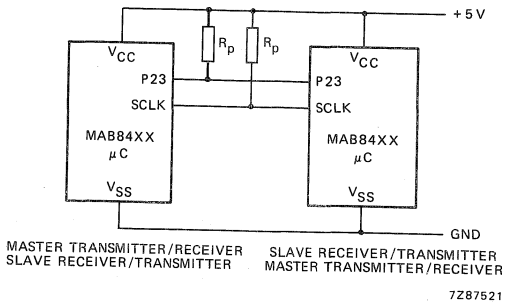


Fig. 3.29 Block diagram multi-master MAB84X1 - MAB84X1 configuration.

Symbol definition:

```
IICFR   EQU  H'02'    fscl = 98,4 kHz @ fcrystal = 4,43 MHz.
NRSR    EQU  4        Number of data bytes which have to be received in
                      SLV/REC mode.
NRGC    EQU  4        Number of data bytes to be transmitted in SLV/TRM
                      mode.
NRST    EQU  4        Number of data bytes which have to be received in
                      GENERAL CALL mode.
```

Data register definition:

```
Register Bank 1

R11     EQU  H'19'    SIO interrupt data index register
R13     EQU  H'1B'    SIO interrupt data byte counter register
R14     EQU  H'1C'    SIO interrupt I2C-bus status register
```

Status flag definition of status register R14:

```
SRDAVF  EQU  H'10'    SLV/REC data valid flag
```

The flag SRDAVF is set in the serial I/O interrupt service subroutine if the microcomputer has received its own address and "NRSR" data bytes. It is cleared in the main program when the received data has been handled.

```
GCDAVF  EQU  H'08'    GENERAL CALL data valid flag
```

The flag GCDAVF is set in the serial I/O interrupt service subroutine if the microcomputer has received the general call address and "NRGC" data bytes. It is cleared in the main program when the received data has been handled.

```
R17     EQU  H'1F'     Accumulator-save register during interrupt service
                       subroutines.
```

master transmitter data registers:

```
MTDTR1  EQU  H'30'     Master transmitter data byte 1 register
MTDTR2  EQU  MTDTR1+1          "                    2    "
MTDTR3  EQU  MTDTR2+1          "                    3    "
MTDTR4  EQU  MTDTR3+1          "                    4    "
```

master receiver data registers:

```
MRDTR1  EQU  H'34'     Master receiver data byte 1 register
MRDTR2  EQU  MRDTR1+1          "                    2    "
MRDTR3  EQU  MRDTR2+1          "                    3    "
MRDTR4  EQU  SRDTR3+1          "                    4    "
```

slave transmitter data registers:

```
STDTR1  EQU  H'38'     Slave transmitter data byte 1 register
STDTR2  EQU  STDTR1+1          "                    2    "
STDTR3  EQU  STDTR2+1          "                    3    "
STDTR4  EQU  STDTR3+1          "                    4    "
```

slave receiver data registers:

```
SRDTR1  EQU  H'3C'     Slave receiver data byte 1 register
SRDTR2  EQU  SRDTR1+1          "                    2    "
SRDTR3  EQU  SRDTR2+1          "                    3    "
SRDTR4  EQU  SRDTR3+1          "                    4    "
```

Initialization:

```
000000                          74          ORG     H'000'
                                75  *
000000 2400              1      76  RESET   JMP     INIT            JUMP TO INITIALIZE ROUTINE
                                77  *
                                78  *
                                79  * SERIAL I/O INTERRUPT SERVICE SUBROUTINE ENTRY
                                80  *
000002                          81          ORG     H'005'
                                82  *
000005 4400              2      83  SIOINT  JMP     SIOIN1          JUMP TO SIO INTERRUPT SERVICE SUBROUTINE
                                84  *
                                85  * POWER-ON RESET INITIALISATION SUBROUTINE:
                                86  *
000007                          87          ORG     H'100'
                                88  *
000100 B83F              3      89  INIT    MOV     R0,#H'3F'
000102 27                4      90          CLR     A
                                91  *
000103 A0                5      92  INIT1   MOV     @R0,A           CLEAR ALL DATA REGISTERS
000104 E803              6      93          DJNZ    R0,INIT1
                                94  *
                                95  * INITIALIZE SIO ROUTINE:
                                96  *
000106 9E42              7      97          MOV     S2,#IICFR+ACK   INITIALISE IIC-BUS FREQUENCY;
                                98  *                                SET WITH ACKNOWLEDGE MODE
000108 9C50              8      99          MOV     S0,#OWNAD       SET OWN SLAVE ADDRESS
00010A 9D18              9     100          MOV     S1,#PIN+ESO     INITALIZE SIO STATUS
00010C 85               10     101          EN      SI
```

Main program:

The software given in this example is meant for a MAB84X1 microcontroller which has to function in a multi-master system and which has to receive and transmit data according the following formats:

- Master transmitter

S	uC ADR.	W	A	DATA 1	A	DATA 2	A	DATA 3	A	DATA 4	A	P

Fig. 3.30 Master transmitter format.

- Master receiver

S	uC ADR.	R	A	DATA 1	A	DATA 2	A	DATA 3	A	DATA 4	NA	P

Fig. 3.31 Master receiver format.

- Slave receiver general call address

S	GC ADR.	W	A	DATA 1	A	DATA 2	A	DATA 3	A	DATA 4	A	P

Fig. 3.32 Slave receiver general call format.

- Slave receiver

S	OWN ADR.	W	A	DATA 1	A	DATA 2	A	DATA 3	A	DATA 4	A	P

Fig. 3.33 Slave receiver format.

- Slave transmitter

S	OWN ADR.	R	A	DATA 1	A	DATA 2	A	DATA 3	A	DATA 4	NA	P

Fig. 3.34 Slave transmitter format.

```
          S         is the START condition
          uC ADR    is the slave address of the other microcomputer
          OWN ADR   is the own slave address
          GC ADR    is the general call address
          W         is the read/write bit in write state (= 0)
          R         is the read/write bit in read state (= 1)
          A         is the acknowledge bit; this has to be 0
          NA        is the not acknowledge bit; this has to be 1
          P         is the STOP condition

UCAD      EQU       H'60'
GCAD      EQU       H'00'
OWNAD     EQU       H'50'
```

The main program increments the four data bytes, stored in register MTDTR1-4, and writes these data bytes to another microcomputer. It also reads four data bytes out of this microcomputer and stores these data bytes in register MRDTR1-4.

In slave mode the program functions as described in Section 3.3.2.

```
00010D B933          11   141 MAIN   MOV    R1,#MTDTR4
00010F BA04          12   142        MOV    R2,#4
000111 97            13   143        CLR    C
000112 A7            14   144        CPL    C
                          145 *
000113 F1            15   146 MAIN1  MOV    A,@R1          INCR. CONTENTS OF REGISTERS MTDTR1-4
000114 1300          16   147        ADDC   A,#0
000116 A1            17   148        MOV    @R1,A
000117 C9            18   149        DEC    R1
000118 EA13          19   150        DJNZ   R2,MAIN1
                          151 *
00011A 3428          20   152 MAIN2  CALL   SUBR           TEST SRDAVF AND GCDAVF FLAGS
00011C 3445          21   153        CALL   WUC            WRITE MTDTR1-4 TO OTHER UC
00011E 961A          22   154        JNZ    MAIN2          REPEAT IF ERROR
                          155 *
000120 3428          23   156 MAIN3  CALL   SUBR           TEST SRDAVF AND GCDAVF FLAGS
000122 348A          24   157        CALL   RUC            READ DATA OUT OF OTHER UC
000124 9620          25   158        JNZ    MAIN3          REPEAT IF ERROR
000126 2400          26   159        JMP    MAIN
                          160 *
                          161 *
000128 B81C          27   162 SUBR   MOV    R0,#R14        SET SYSTEM STATUS REGISTER INDEX
00012A F0            28   163        MOV    A,@R0          FETCH SYSTEM STATUS
00012B 9230          29   164        JB4    SUBR1          JUMP IF SRDAVF IS SET
00012D 7230          30   165        JB3    SUBR1          JUMP IF GCDAVF IS SET
00012F 83            31   166        RET
                          167 *
000130 B83C          32   168 SUBR1  MOV    R0,#SRDTR1
000132 B938          33   169        MOV    R1,#STDTR1     TRANSFER DATA FROM SRDTR1-4 TO STDTR1-4
000134 BB04          34   170        MOV    R3,#4
                          171 *
000136 F0            35   172 SUBR2  MOV    A,@R0          TRANSFER DATA
000137 A1            36   173        MOV    @R1,A
000138 19            37   174        INC    R1
000139 18            38   175        INC    R0
00013A EB36          39   176        DJNZ   R3,SUBR2
00013C B81C          40   177        MOV    R0,#R14
00013E 23E7          41   178        MOV    A,#.NOT.(SRDAVF+GCDAVF)
000140 95            42   179        DIS    SI
000141 50            43   180        ANL    A,@R0          RESET SRDAVF OR GCDAVF
000142 A0            44   181        MOV    @R0,A
000143 85            45   182        EN     SI
000144 83            46   183        RET
```

Master writes four bytes of data to other microcomputer subroutine:

entry: MTDTR1-4 contains data to be written
exit: A = 0 if the transmission is successful
A ≠ 0 if the transmission is not successful

The contents of register R0 are modified.

```
000145 3465          47   191 WUC     CALL    BBTST           BUS BUSY TEST ROUTINE
000147 95            48   192         DIS     SI              DISABLE INTERRUPT
000148 98FB          49   193         ANL     P0,#.NOT.TRIG1  TRIGGER !!!!!!
00014A 8804          50   194         ORL     P0,#TRIG1       TRIGGER !!!!!!
00014C 9C60          51   195         MOV     S0,#UCAD        LOAD UC SLAVE ADDRESS AND WRITE BIT
00014E 9DF8          52   196         MOV     S1,#STRTC       OUTPUT START COND, SLAVE ADDRESS AND WRITE BIT
000150 3481          53   197         CALL    MTTBS           TEST BUS STATUS
000152 96C6          54   198         JNZ     ERUC2           JUMP IF NO ACKN. RECEIVED OR ERROR
000154 B830          55   199         MOV     R0,#MTDTR1      SET DATA REGISTER INDEX
000156 BA04          56   200         MOV     R2,#4
                          201 *
000158 F0            57   202 WUC1    MOV     A,@R0           FETCH DATA BYTE
000159 3C            58   203         MOV     S0,A            TRANSMIT DATA BYTE
00015A 18            59   204         INC     R0              INCR INDEX
00015B 3481          60   205         CALL    MTTBS           TEST BUS STATUS
00015D 96C6          61   206         JNZ     ERUC2           JUMP IF NO ACKN. RECEIVED OR ERROR
00015F EA58          62   207         DJNZ    R2,WUC1         DECR DATA BYTE COUNTER
                          208 *
000161 9DD8          63   209 STPC    MOV     S1,#STOPC       OUTPUT STOP CONDITION
000163 85            64   210         EN      SI              ENABLE SIO INTERRUPTS
000164 83            65   211         RET
                          212 *
                          213 * BUS BUSY TEST SUBROUTINE
                          214 *
000165 B880          66   215 BBTST   MOV     R0,#H'80'       LOAD BUS BUSY COUNTER
000167 85            67   216         EN      SI
                          217 *
000168 0D            68   218 BBTST1  MOV     A,S1            WAIT UNTILL BUS IS FREE FOR MAXIMUM
000169 37            69   219         CPL     A                7 X 256 X 30 X Txtal
00016A B289          70   220         JB5     BBTST2          JUMP IF BUS IS FREE
00016C E868          71   221         DJNZ    R0,BBTST1       DECR AND TEST BUS BUSY COUNTER
00016E 9D18          72   222         MOV     S1,#PIN+ESO     RESET BB BIT; BUS IS FREE NOW
000170 9D18          73   223         MOV     S1,#PIN+ESO     INITIALIZE OTHER STATUS BITS
000172 9E42          74   224         MOV     S2,#IICFR+ACK   INITIALIZE IIC-FREQ. AND ACKN. MODE BIT
000174 95            75   225         DIS     SI              DISABLE SERIAL I/O INTERRUPTS
000175 9CFF          76   226         MOV     S0,#H'FF'       LOAD SIO DATA REGISTER
000177 9DF8          77   227         MOV     S1,#STRTC       OUTPUT START CONDITION AND H'FF' AS SLAVE
                          228 *                                ADDRESS
000179 3481          78   229         CALL    MTTBS           TEST BUS STATUS
00017B D301          79   230         XRL     A,#LRB          INVERT ACKN. BIT
00017D 9665          80   231         JNZ     BBTST           JUMP IF ERROR
00017F 2461          81   232         JMP     STPC            OUTPUT STOP CONDITION
```

Master transmitter test bus status subroutine:

```
000181 0D            82   236 MTTBS   MOV     A,S1            FETCH BUS STATUS
000182 F285          83   237         JB7     MTTBS1          JUMP IF MST = 1
000184 83            84   238         RET
                          239 *
000185 9281          85   240 MTTBS1  JB4     MTTBS           JUMP IF PIN = 1
000187 D3E0          86   241         XRL     A,#MTTST        TEST MST/TRX BUS STATUS
                          242 *
000189 83            87   243 BBTST2  RET
```

Master reads four data bytes out of other microcomputer subroutine:

exit: A = 0 if the transmission is successful
Received data bytes are sorted in register MRDTR1-4
A ≠ 0 if the transmission is not successful

The contents of register R0 are modified.

```
00018A 3465          88   251 RUC     CALL    BBTST           TEST BUS BUSY
00018C 95            89   252         DIS     SI              DISABLE INTERRUPT
00018D 98F7          90   253         ANL     P0,#.NOT.TRIG2  TRIGGER !!!!!!
00018F 8808          91   254         ORL     P0,#TRIG2       TRIGGER !!!!!!
000191 9C61          92   255         MOV     S0,#UCAD+RW     LOAD SLAVE ADDRESS AND READ BIT
000193 9DF8          93   256         MOV     S1,#STRTC       OUTPUT START COND, SLAVE ADDRESS AND READ BIT
000195 34CC          94   257         CALL    MRTBS           TEST BUS STATUS
000197 96C6          95   258         JNZ     ERUC2           JUMP IF NO ACKN. RECEIVED OR ERROR
000199 0C            96   259         MOV     A,S0            DUMMY READ; START RECEPTION OF DATA
00019A BA02          97   260         MOV     R2,#4-2         SET DATA BYTE COUNTER
00019C B834          98   261         MOV     R0,#MRDTR1      SET DATA BYTE INDEX
                          262 *
00019E 34CC          99   263 RUC1    CALL    MRTBS           TEST BUS STATUS
0001A0 96C6         100   264         JNZ     ERUC2           JUMP IF NO ACKN. RECEIVED OR ERROR
0001A2 0C           101   265         MOV     A,S0            FETCH RECEIVED DATA BYTE
0001A3 A0           102   266         MOV     @R0,A           STORE RECEIVED DATA BYTE
0001A4 18           103   267         INC     R0              INCR DATA BYTE INDEX
0001A5 EA9E         104   268         DJNZ    R2,RUC1         DECR. DATA BYTE COUNTER
0001A7 34CC         105   269         CALL    MRTBS           TEST BUS STATUS
0001A9 96C6         106   270         JNZ     ERUC2           JUMP IF ERROR
0001AB 9E02         107   271         MOV     S2,#IICFR       SET WITHOUT ACKNOWLEDGE MODE
0001AD 0C           108   272         MOV     A,S0            FETCH LAST BUT ONE TO BE RECEIVED DATA BYTE
0001AE A0           109   273         MOV     @R0,A           SAVE  ''
0001AF 18           110   274         INC     R0              INCR. DATA BYTE INDEX
0001B0 34CC         111   275         CALL    MRTBS           TEST BUS STATUS
0001B2 53FE         112   276         ANL     A,#.NOT.LRB     CLEAR LRB
0001B4 96C4         113   277         JNZ     ERUC1           JUMP IF ERROR
0001B6 9DA9         114   278         MOV     S1,#MST+BB+ESO+BCO SET BIT COUNTER TO ONE
0001B8 0C           115   279         MOV     A,S0            FETCH LAST TO BE RECEIVED DATA BYTE
0001B9 A0           116   280         MOV     @R0,A           SAVE  ''
0001BA 34CC         117   281         CALL    MRTBS           TEST BUS STATUS
                          282 *                               GENERATE NOT ACKNOWLEDGE
0001BC D301         118   283         XRL     A,#LRB          INVERT NOT ACKNOWLEDGE BIT
0001BE 96C4         119   284         JNZ     ERUC1           JUMP IF ERROR
0001C0 9E42         120   285         MOV     S2,#IICFR+ACK   SET WITH ACKNOWLEDGE MODE
0001C2 2461         121   286         JMP     STPC            OUTPUT STOP CONDITION
                          287 *
0001C4 9E42         122   288 ERUC1   MOV     S2,#IICFR+ACK   SET WITH ACKNOWLEDGE MODE
                          289 *
0001C6 0D           123   290 ERUC2   MOV     A,S1            FETCH SIO STATUS
0001C7 F261         124   291         JB7     STPC            JUMP IF MASTER; NO ACKN. RECEIVED
0001C9 85           125   292         EN      SI              ENABLE SIO INTERRUPTS
0001CA 37           126   293         CPL     A               A # 0; TRANSMISSION NOT SUCCESSFUL
0001CB 83           127   294         RET
                          295 *
                          296 * MASTER RECEIVER TEST BUS STATUS SUBROUTINE:
                          297 *
0001CC 0D           128   298 MRTBS   MOV     A,S1            FETCH BUS STATUS
0001CD F2D0         129   299         JB7     MRTBS1          JUMP IF MST = 1
0001CF 83           130   300         RET
                          301 *
0001D0 92CC         131   302 MRTBS1  JB4     MRTBS           JUMP IF PIN = 1
0001D2 D3A0         132   303         XRL     A,#MRTST        TEST MST/REC BUS STATUS
0001D4 83           133   304         RET
                          305 *
                          306 *
                          307 *
0001D5                    308         ORG     H'200'
                          309 *
                          310 * SERIAL I/O INTERRUPT SERVICE SUBROUTINE:
                          311 *
000200 D5           134   312 SIOIN1  SEL     RB1             SELECT REGISTER BANK 1
000201 AF           135   313         MOV     R7,A            SAVE ACCU
000202 0D           136   314         MOV     A,S1            FETCH SIO STATUS
000203 721D         137   315         JB3     MSTAL           JUMP IF ARB LOST
                          316 *
000205 D23D         138   317 SIOIN2  JB6     SLVTR           JUMP IF TRX = 1
000207 5223         139   318         JB2     SRADR           JUMP IF ADDRESSED AS SLAVE RECEIVER
                          319 *
                          320 * SLV/REC; INPUT DATA SUBROUTINE:
                          321 *
000209 67           140   322 SRDAT   RRC     A
```

```
00020A 67          141    323          RRC     A                ADD IN CARRY
00020B FB          142    324          MOV     A,R3
00020C C63A        143    325          JZ      SRENA            JUMP IF MORE THAN NRSR OR NRGC DATA
                          326 *                                    BYTES RECEIVED
00020E E818        144    327          DJNZ    R3,SRDAT2        DECR. AND TEST DATA BYTE COUNTER
000210 2310        145    328          MOV     A,#SRDAVF        SET SRDAVF
000212 E616        146    329          JNC     SRDAT1           JUMP IF ADD = 0
000214 2308        147    330          MOV     A,#GCDAVF        SET GCDAVF
                          331 *
000216 4C          148    332 SRDAT1   ORL     A,R4             SET SRDAVF OR GCDAVF FLAG
000217 AC          149    333          MOV     R4,A
                          334 *
000218 0C          150    335 SRDAT2   MOV     A,S0             FETCH RECEIVED DATA BYTE
000219 A1          151    336          MOV     @R1,A            SAVE RECEIVED DATA BYTE
00021A 19          152    337          INC     R1               INCR. DATA BYTE INDEX
00021B FF          153    338          MOV     A,R7
00021C 93          154    339          RETR
                          340 *
                          341 * MASTER HAS LOST ARBITRATION SUBROUTINE:
                          342 *
00021D 5205        155    343 MSTAL    JB2     SIOIN2           JUMP IF ADDRESSED AS SLAVE
00021F 443A        156    344          JMP     SRENA            RELEASE BUS
                          345 *
                          346 * SLV; ADDRESSED AS SLAVE SUBROUTINE:
                          347 *
000221 D246        157    348 SLADR    JB6     STADR            JUMP IF ADDRESSED AS SLAVE TRANSMITTER
                          349 *
                          350 * SLV/REC; ADDRESSED AS SLAVE RECEIVER SUBROUTINE:
                          351 *
000223 B93C        158    352 SRADR    MOV     R1,#SRDTR1       SET DATA REGISTER INDEX
000225 3234        159    353          JB1     GCADR            JUMP IF GENERAL SLAVE ADDRESS
000227 BB04        160    354          MOV     R3,#NRSR         SET SLV/REC DATA BYTE COUNTER
000229 FC          161    355          MOV     A,R4
00022A 5318        162    356          ANL     A,#SRDAVF+GCDAVF
00022C C63A        163    357          JZ      SRENA            JUMP IF SRDAVF AND GCDAVF ARE NOT SET
                          358 *                                    CONT. WITH RECEPTION OF DATA
00022E 9D68        164    359          MOV     S1,#TRX+BB+ESO   SET SLAVE TRANSMITTER MODE
000230 9CFF        165    360          MOV     S0,#H'FF'        OUTPUT H'FF' AND NEGATIVE ACKN.
000232 FF          166    361          MOV     A,R7
000233 93          167    362          RETR
                          363 *
                          364 * SLV/REC; ADDRESSED WITH GENERAL CALL ADDRESS SUBROUTINE:
                          365 *
000234 BB04        168    366 GCADR    MOV     R3,#NRGC         SET GENERAL CALL ADDRESS DATA BYTE
                          367 *                                    COUNTER
000236 FC          169    368          MOV     A,R4
000237 53E7        170    369          ANL     A,#.NOT.(SRDAVF+GCDAVF) RESET SRDAVF AND GCDAVF FLAGS
000239 AC          171    370          MOV     R4,A
                          371 *
                          372 * SLV/REC; ENABLE SLC SUBROUTINE:
                          373 *
00023A 0C          172    374 SRENA    MOV     A,S0             RELEASE SCL
00023B FF          173    375          MOV     A,R7
00023C 93          174    376          RETR
                          377 *
                          378 * SLAVE TRANSMITTER SUBROUTINE:
                          379 *
00023D 37          175    380 SLVTR    CPL     A                COMPL. STATUS BITS
00023E 1244        176    381          JB0     SLVTR1           JUMP IF ACKN. RECEIVED
                          382 *
                          383 * SLV/TRX; NEGATIVE ACKNOWLEDGE RECEIVED SUBROUTINE:
                          384 *
000240 9D38        177    385 STNAC    MOV     S1,#BB+PIN+ESO   SET SLV/REC MODE
000242 FF          178    386          MOV     A,R7
000243 93          179    387          RETR
                          388 *
                          389 * SLV/TRX; ACKNOWLEDGE RECEIVED SUBROUTINE:
                          390 *
000244 5248        180    391 SLVTR1   JB2     STDAT            JUMP IF AAS = 0
                          392 *
                          393 * SLV/TRX; ADDRESSED AS SLAVE TRANSMITTER SUBROUTINE:
                          394 *
000246 B938        181    395 STADR    MOV     R1,#STDTR1       SET DATA BYTE REG. INDEX
                          396 *
000248 F1          182    397 STDAT    MOV     A,@R1            FETCH NEW DATA BYTE
000249 3C          183    398          MOV     S0,A             OUTPUT DATA BYTE
00024A 19          184    399          INC     R1               INCR. DATA BYTE INDEX
00024B FF          185    400          MOV     A,R7
00024C 93          186    401          RETR
                          402 *
00024D                    403          END
```

4.0 SERIAL I/O SOFTWARE EXAMPLES: APPLICABLE TO MAB8048

4.1 Single-master routines

4.1.1 Master transmitter routine (MAB8048 - SAA1300)

This software is intended for the MAB8048 microcomputer family which is the only master in an I^2C-bus system. In this system none of the slaves are allowed to stretch the clock low time. This software example only meets the I^2C-bus protocol functionally, electrically there are some differences.

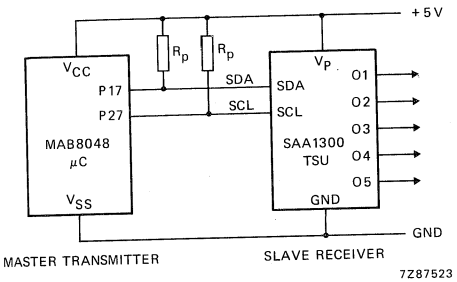


Fig. 4.1 Block diagram of MAB8048 - SAA1300 configuration.

Initialization:

```
000000                    61          ORG     H'000'
                          62 *
000000 2400          1    63 RESET    JMP     INIT          JUMP TO INITIALIZE ROUTINE
                          64 *
                          65 * INITIALISATION:
                          66 *
000002                    67          ORG     H'100'
                          68 *
000100 B8FF          2    69 INIT     MOV     R0,#H'FF'     LOAD DATA TO BE WRITTEN TO TSU
                          70 *
```

Main program:

The main program transmits data to TSU by calling the subroutine WTSU. If the transmission is not successful it repeats the data transfer, otherwise it increments the data and starts again.

The format of the data transfer to TSU is shown in Fig. 4.2:

S	TSU	ADR	W	A	DATA BYTE	A	P

Fig. 4.2 Data format of transfer to TSU

S is the START condition
TSU ADR is the slave address of TSU
W is the read/write bit in write state (=0)
A is the acknowledge bit; this has to be 0
P is the STOP condition

The slave address of TSU is 0100 0XX

```
TSUAD    EQU  H'40'
```

I^2C-bus signals of MAB8048 are:

```
SDA EQU  H'80'     Pin P17 Pin nr. 34
SCL EQU  H'80'     Pin P27 Pin nr. 38
```

Note: The bus outputs SCL and SDA of the MAB8048 do not have an open drain output as specified in the I^2C-bus protocol.

The logic levels of the MAB8048 bus signals do not match the I^2C-bus protocol:

	I^2C-bus protocol	MAB8048
VILmax	1,5 V	0,8 V
VIHmin	3,5 V	2,0 V
VOLmax	0,4 V @ 3 mA	0,45 V @ 1,6 mA

Because the MAB8048 has an internal large and small pull-up, two pins of different I/O ports (Ports 1 and 2) are taken for the I^2C-bus signals.

If the MAB8048 inputs the acknowledge bit, it has to generate a clock pulse at output SCL. This is done with the ORL Px,#SCL and the ANL Px, #.NOT.SCL instructions. While changing the output latch of the SCL pin the MAB8048 also updates the data of the other I/O pins of the same port.

Because the output latch of the SDA output is switched high, the large pull-up of this output latch of the SDA output will be activated. If the slave pulls the SDA line low to generate an acknowledge, a short circuit situation occurs. To overcome this problem two I/O pins of different ports are defined as the I^2C-bus signals SDA and SCL.

The maximum MAB8048 crystal frequency for this software example is 6 MHz to reach the minimum SCL clock low and high times as given in the I^2C-bus protocol. If a higher crystal frequency is chosen some delays (e.g. NOP instructions) have to be inserted in the master transmitter data (MTDAT) subroutine.

```
                          71 * MAIN PROGRAM:
                          72 *
000102 18            3    73 MAIN    INC     R0          INCR. DATA TO BE TRANSMITTED
                          74 *
000103 5400          4    75 MAIN1   CALL    WTSU        TRANSMIT DATA BYTE IN R0 TO TSU
000105 9603          5    76         JNZ     MAIN1       REPEAT IF TRANSMISSION IS NOT SUCCESSFUL
000107 BA00          6    77         MOV     R2,#0       SET DELAY COUNTER
000109 EA09          7    78         DJNZ    R2,$        DELAY
00010B 2402          8    79         JMP     MAIN        CONTINUE
                          80 *
00010D                    81         ORG     H'200'
```

Master writes one data byte to TSU (SAA1300) subroutine:

entry: R0 contents to be written to TSU
exit: A = 0 if transmission is successful
contents of register R2 are modified.

```
                          87 *
000200 2340          9    88 WTSU   MOV     A,#TSUAD        LOAD TSU SLAVE ADDRESS AND WRITE BIT
000202 5410         10    89        CALL    START           OUTPUT START CONDITION, SLAVE ADDRESS AND
                          90 *                               WRITE BIT; INPUT ACKNOWLEDGE BIT
000204 9609         11    91        JNZ     STOP            JUMP IF NO ACKNOWLEDGE RECEIVED
000206 F8           12    92        MOV     A,R0            FETCH DATA BYTE TO BE TRANSMITTED
000207 5414         13    93        CALL    MTDAT           OUTPUT DATA BYTE
```

Output stop condition subroutine:

```
                          96 *
000209 997F         14    97 STOP   ANL     P1,#.NOT.SDA    OUTPUT STOP CONDITION
00020B 8A80         15    98        ORL     P2,#SCL
00020D 8980         16    99        ORL     P1,#SDA
00020F 83           17   100        RET
```

Master outputs start condition, slave address, R/W bit and inputs acknowledge bit subroutine:

entry: A contains slave address and R/W bit
exit: A = 0 if acknowledge is received
contents of register R2 are modified.

```
                         107 *
000210 997F         18   108 START  ANL     P1,#.NOT.SDA    OUTPUT START CONDITION
000212 9A7F         19   109        ANL     P2,#.NOT.SCL
```

Master outputs data byte and inputs acknowledge bit subroutine:

entry: A contains data byte to be transmitted
exit: A = 0 if acknowledge is received
contents of register R2 are modified.

```
000214 BA08         20   116 MTDAT  MOV     R2,#8           SET BIT COUNTER
                         117 *
000216 E7           21   118 MTDAT1 RL      A               SHIFT DATA BYTE
000217 121D         22   119        JB0     MTDAT2          JUMP IF TO BE TRANSMITTED BIT IS HIGH
000219 997F         23   120        ANL     P1,#.NOT.SDA    RESET SDA OUTPUT
00021B 4421         24   121        JMP     MTDAT3          CONTINUE
                         122 *
00021D 8980         25   123 MTDAT2 ORL     P1,#SDA         SET SDA OUTPUT
00021F 00           26   124        NOP                     DELAY  ### CAN BE DELETED ###
000220 00           27   125        NOP                     DELAY  ### CAN BE DELETED ###
                         126 *
000221 8A80         28   127 MTDAT3 ORL     P2,#SCL         SET SCL OUTPUT
000223 9A7F         29   128        ANL     P2,#.NOT.SCL    RESET SCL OUTPUT
000225 EA16         30   129        DJNZ    R2,MTDAT1       DECR. AND TEST BIT COUNTER
000227 8980         31   130        ORL     P1,#SDA         SET SDA OUTPUT TO INPUT MODE
000229 8A80         32   131        ORL     P2,#SCL         SET SCL OUTPUT
00022B 09           33   132        IN      A,P1            INPUT ACKNOWLEDGE BIT
00022C 9A7F         34   133        ANL     P2,#.NOT.SCL    RESET SCL OUTPUT
00022E 5380         35   134        ANL     A,#SDA          MASK ACKN. BIT
000230 83           36   135        RET
                         136 *
000231                   137        END

  NO ERRORS DETECTED
```

4.1.2 Master receiver routine (MAB8048 - SAB3028)

This software is intended for members of the MAB8048 microcomputer family where there is only one master in an I^2C-bus system. In this system none of the slaves are allowed to stretch the clock low time. This software example only meets the I^2C-bus protocol functionally, electrically there are some differences. (See section 4.1.1).

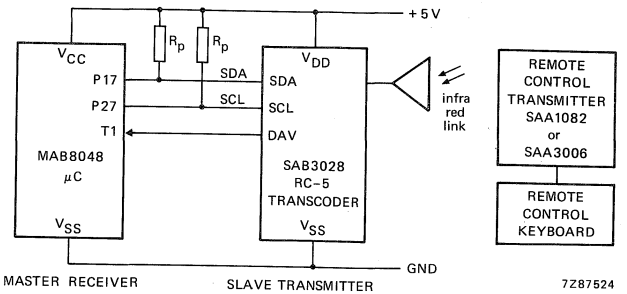


Fig. 4.3 Block diagram of MAB8048 - SAB3028 configuration.

Initialization:

```
000000                    52          ORG     H'000'
                          53 *
000000 2400          1    54 RESET    JMP     INIT        JUMP TO INITIALIZE ROUTINE
000002                    58          ORG     H'100'
                          59 *
000100 00            2    60 INIT     NOP                 NO INITIALISATION
                          61 *
```

Main program:

The main program polls the input T1, which is connected to the data valid output (DAV) of the RC-5 transcoder SAB3028. This output goes low if the RC-5 transcoder has received a valid remote control message. If low the main program calls subroutine RRCT, which reads the data out of the RC-5 transcoder via the I^2C-bus.

The subroutine RRCT returns with Accu = 0 if the data transfer is successful.

The received data is stored in the registers MRDTR1-4. If the transfer is successful the signal DAV is set to 1 again. If the data transfer is not successful ($A \neq 0$), the transmission is repeated.

The received data is stored in the registers MRDTR.

```
MRDTR1   EQU  H'04'     MASTER RECEIVER DATA BYTE 1 REGISTER
MRDTR2   EQU  MRDTR1+1            "               2     "
MRDTR3   EQU  MRDTR2+1            "               3     "
MRDTR4   EQU  MRDTR3+1            "               4     "
```

The format of the data transfer from the RC-5 transcoder SAB3028 is shown in Fig. 4.4:

S	RCT ADR	R	A	DATA 1	A	DATA 2	A	DATA 3	A	DATA 4	NA	P

Fig. 4.4 RC-5 transcoder data transfer.

S	is the START condition
RCT ADR	is the slave address of RC-5 transcoder
R	is the read/write bit in read state (=1)
A	is the acknowledge bit; this has to be 0
DATA	is data byte
NA	is the not acknowledge bit; this has to be 1
P	is the STOP condition

The slave address of RC-5 transcoder is 0100 110.

```
RCTAD    EQU  H'4C'
```

The read bit position in the slave address is:

```
RW       EQU  H'01'
```

I^2C-bus signals of MAB8048 are:

```
SDA      EQU  H'80'      Pin P17  Pin nr. 34
SCL      EQU  H'80'      Pin P27  Pin nr. 38
```

```
000101 5601          3     64 MAIN   JT1     MAIN          WAIT FOR T1 = DAV = 0
                           65 *
000103 5400          4     66 MAIN1  CALL    RRCT          READ DATA OUT OF RC-5 TRANSCODER
000105 9603          5     67        JNZ     MAIN1         REPEAT IF TRANSMISSION IS NOT SUCCESSFUL
000107 2401          6     68        JMP     MAIN          CONTINUE
```

Master reads data out of RC-5 transcoder (SAB3028) subroutine:

exit: A = 0 if transmission is successful
contents of registers R0 and R2 are modified

```
                           75 *
000200 234D          7     76 RRCT   MOV     A,#RCTAD+RW   LOAD RCT SLAVE ADDRESS AND READ BIT
000202 541E          8     77        CALL    START         OUTPUT START CONDITION, SLAVE ADDRESS AND
                           78 *                            READ BIT; INPUT ACKNOWLEDGE BIT
000204 9617          9     79        JNZ     STOP          JUMP IF NO ACKNOWLEDGE RECEIVED
000206 B804         10     80        MOV     R0,#MRDTR1    SET DATA REGISTER INDEX
000208 5445         11     81        CALL    MRDAT         RECEIVE AND STORE FIRST DATA BYTE
00020A 543F         12     82        CALL    MRACK         OUTPUT ACKNOWLEDGE
                           83 *                            RECEIVE AND STORE SECOND DATA BYTE
00020C 543F         13     84        CALL    MRACK         OUTPUT ACKNOWLEDGE
                           85 *                            RECEIVE AND STORE THIRD DATA BYTE
00020E 543F         14     86        CALL    MRACK         OUTPUT ACKNOWLEDGE
                           87 *                            RECEIVE AND STORE FOURTH DATA BYTE
000210 27           15     88        CLR     A             TRANSMISSION SUCCESSFUL
```

Output not acknowledge bit and stop condition subroutine:

```
000211 8980          16      92 MRACKN ORL     P1,#SDA          SET SDA OUTPUT
000213 8A80          17      93        ORL     P2,#SCL          SET SCL OUTPUT
000215 9A7F          18      94        ANL     P2,#.NOT.SCL     RESET SCL OUTPUT
```

Output stop condition subroutine:

```
000217 997F          19      98 STOP   ANL     P1,#.NOT.SDA     OUTPUT STOP CONDITION
000219 8A80          20      99        ORL     P2,#SCL
00021B 8980          21     100        ORL     P1,#SDA
00021D 83            22     101        RET
```

Master outputs start condition, slave address, R/W bit and inputs acknowledge bit subroutine:

entry: A contains slave address and R/$\overline{W}$ bit
exit: A = 0 if acknowledge is received
contents of register R2 are modified

```
                             108 *
00021E 997F          23     109 START  ANL     P1,#.NOT.SDA     OUTPUT START CONDITION
000220 9A7F          24     110        ANL     P2,#.NOT.SCL
```

Master outputs data byte and inputs acknowledge bit subroutine:

entry: A contains data byte to be transmitted
exit: A = 0 if acknowledge is received
contents of register R2 are modified

```
                             116 *
000222 BA08          25     117 MTDAT  MOV     R2,#8            SET BIT COUNTER
                            118 *
000224 E7            26     119 MTDAT1 RL      A                SHIFT DATA BYTE
000225 1228          27     120        JB0     MTDAT2           JUMP IF TO BE TRANSMITTED BIT IS HIGH
000227 997F          28     121        ANL     P1,#.NOT.SDA     RESET SDA OUTPUT
000229 442F          29     122        JMP     MTDAT3           CONTINUE
                            123 *
00022B 8980          30     124 MTDAT2 ORL     P1,#SDA          SET SDA OUTPUT
00022D 00            31     125        NOP                      DELAY  ### CAN BE DELETED ###
00022E 00            32     126        NOP                      DELAY  ### CAN BE DELETED ###
                            127 *
00022F 8A80          33     128 MTDAT3 ORL     P2,#SCL          SET SCL OUTPUT
000231 9A7F          34     129        ANL     P2,#.NOT.SCL     RESET SCL OUTPUT
000233 EA24          35     130        DJNZ    R2,MTDAT1        DECR. AND TEST BIT COUNTER
000235 8980          36     131        ORL     P1,#SDA          SET SDA OUTPUT TO INPUT MODE
000237 8A80          37     132        ORL     P2,#SCL          SET SCL OUTPUT
000239 09            38     133        IN      A,P1             INPUT ACKNOWLEDGE BIT
00023A 9A7F          39     134        ANL     P2,#.NOT.SCL     RESET SCL OUTPUT
00023C 5380          40     135        ANL     A,#SDA           MASK ACKN. BIT
00023E 83            41     136        RET
```

Master outputs acknowledge bit, reads data byte and stores data byte in RAM subroutine:

entry: R0 contains index of data register in which received data byte has to be stored
exit: A contains received data byte
contents of register R0 and R2 are modified

```
00023F 997F          42     146 MRACK  ANL     P1,#.NOT.SDA     RESET SDA OUTPUT
000241 8A80          43     147        ORL     P2,#SCL          SET SCL OUTPUT
000243 9A7F          44     148        ANL     P2,#.NOT.SCL     RESET SCL OUTPUT
```

Master reads data byte and stores data byte in RAM subroutine:

entry: R0 contains index of data register in which received data byte has to be stored
exit: A contains received data byte
contents of registers R0 and R2 are modified

```
000245 8980          45    157 MRDAT   ORL     P1,#SDA          SET SDA OUTPUT TO INPUT
000247 BA01          46    158         MOV     R2,#H'01'        INITIALIZE DATA WORD
                           159 *
000249 8A80          47    160 MRDAT1  ORL     P2,#SCL          SET SCL OUTPUT
00024B 09            48    161         IN      A,P1             INPUT BIT
00024C 9A7F          49    162         ANL     P2,#.NOT.SCL     RESET SCL OUTPUT
00024E F7            50    163         RLC     A                SHIFT DATA BIT INTO C
00024F FA            51    164         MOV     A,R2             FETCH DATA REGISTER
000250 F7            52    165         RLC     A                SHIFT DATA BIT INTO DATA WORD
000251 AA            53    166         MOV     R2,A             SAVE DATA WORD
000252 E649          54    167         JNC     MRDAT1           JUMP IF NOT YET 8 BITS RECEIVED
000254 A0            55    168         MOV     @R0,A            SAVE DATA WORD IN MRDTR REGISTER

000255 18            56    169         INC     R0               INCR. DATA REGISTER INDEX
000256 83            57    170         RET
                           171 *
000257                     172         END
```

4.1.3 Master transmitter/receiver routine with repeated start (MAB8048 - PCD8578)

This software is intended for members of the MAB8048 microcomputer family where there is only one master in an I^2C-bus system. In this system none of the slaves are allowed to stretch the clock low time. This software example only meets the I^2C-bus protocol functionally, electrically there are some differences. (See 4.1.1).

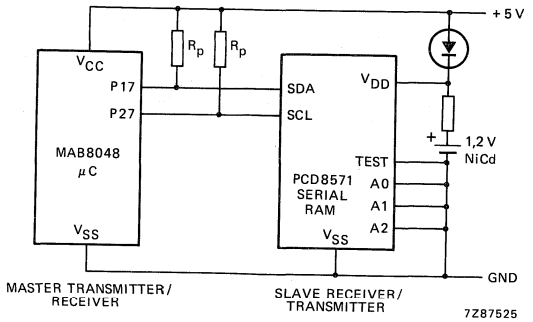


Fig. 4.5 Block diagram MAB8048 - PCD8571 configuration.

Initialization:

```
000100 23FF      2     58 INIT   MOV     A,#H'FF'
000102 AB        3     59        MOV     R3,A          SET POINTER VALUE
000103 AC        4     60        MOV     R4,A          SET DATA BYTE 1 TO BE WRITTEN
000104 AD        5     61        MOV     R5,A                  "    2      "
000000                 50        ORG     H'000'
                       51 *
000000 2400      1     52 RESET  JMP     INIT          JUMP TO INITIALIZE ROUTINE
                       53 *
                       54 * INITIALISATION:
                       55 *
000002                 56        ORG     H'100'
```

Main program:

The main program calls the following two I^2C-bus transmission subroutines:

- subroutine WMEM which writes the pointer value and two data bytes to the PCD8571 CMOS memory.
  The format of this transmission is shown in Fig. 4.6:

S	MEM ADR	W	A	POINT	A	DATA 1	A	DATA 2	A	P

Fig. 4.6 Format to write pointer value and two data bytes to the PCD8571.

- subroutine RMEM which writes the pointer value to the PCD8571 memory and reads two data bytes out of it.
  The format of this transmission is shown in Fig. 4.7:

S	MEM ADR	W	A	POINT	A	S	MEM ADR	R	A	DATA 1	A	DATA 2	NA	P

Fig. 4.7 Format to write pointer value and read two data bytes.

```
S         is the START condition
MEM ADR   is the slave address of the CMOS memory PCD8571
POINT     is the pointer address (internal RAM location address)
W         is the read/write bit in write state (= 0)
R         is the read/write bit in read state (= 1)
A         is the acknowledge bit; this has to be 0
NA        is the not acknowledge bit; this has to be 1
P         is the STOP condition
```

If one of the transmissions does not succeed (A = 0), this transmission is repeated.

The slave address of the CMOS memory is 1010 XXX.

```
MEMAD    EQU  H'AO'
```

The read bit position in the slave address is:

```
RW       EQU  H'01'
```

I^2C-bus signals of MAB8048 are:

```
SDA      EQU  H'80'     Pin P17  Pin nr. 34
SCL      EQU  H'80'     Pin P27  Pin nr. 38
```

```
000105 1B              6    65 MAIN   INC     R3          INCR. POINTER VALUE
000106 2301            7    66        MOV     A,#1        INCR. DATA TO BE WRITTEN
000108 6D              8    67        ADD     A,R5         R4,R5 + 1 --> R4,R5
000109 AD              9    68        MOV     R5,A
00010A 27             10    69        CLR     A
00010B 7C             11    70        ADDC    A,R4
00010C AC             12    71        MOV     R4,A
                            72 *
00010D 5400           13    73 MAIN1  CALL    WMEM        WRITE DATA TO CMOS MEMORY
00010F 960D           14    74        JNZ     MAIN1       REPEAT IF TRANMISSION IS NOT SUCCESSFUL
000111 BA00           15    75        MOV     R2,#0       SET DELAY COUNTER
000113 EA13           16    76        DJNZ    R2,$         DELAY
                            77 *
000115 5415           17    78 MAIN2  CALL    RMEM        READ DATA OUT OF CMOS MEMORY
000117 9615           18    79        JNZ     MAIN2       REPEAT IF TRANSMISSION IS NOT SUCCESSFUL
000119 BA00           19    80        MOV     R2,#0       SET DELAY COUNTER
00011B EA1B           20    81        DJNZ    R2,$         DELAY
00011D 2405           21    82        JMP     MAIN        CONTINUE
                            83 *
00011F                      84        ORG     H'200'
```

Master writes two data bytes to CMOS memory (PCD8571) subroutine:

To write data to the memory, the memory needs to know at which address (location) this data has to be stored. The CMOS memory PCD8571 is designed in such a way that the first data word after the slave address is always the internal address or pointer.

If more data bytes are written into the memory, within one transmission, this pointer is automatically incremented.

```
entry: R3 contains pointer value to be transmitted
       R4 contains first data byte which has to be written in the memory.
       R5 contains second data byte which has to be written in the memory.
exit:  A = 0 if transmission is successful
       A ≠ 0 if transmission is not successful
       contents of register R2 are modified
```

```
                        100 *
000200 23A0       22    101 WMEM   MOV     A,#MEMAD        SET MEMORY ADDRESS AND WRITE BIT
000202 543C       23    102        CALL    START           OUTPUT START CONDITION, SLAVE ADDRESS AND
                        103 *                               WRITE BIT; INPUT ACKNOWLEDGE BIT
000204 9635       24    104        JNZ     STOP            JUMP IF NO ACKNOWLEDGE RECEIVED
000206 FB         25    105        MOV     A,R3            FETCH POINTER VALUE
000207 5440       26    106        CALL    MTDAT           OUTPUT POINTER VALUE
000209 9635       27    107        JNZ     STOP            JUMP IF NO ACKNOWLEDGE
000208 FC         28    108        MOV     A,R4            FETCH FIRST DATA BYTE
00020C 5440       29    109        CALL    MTDAT           OUTPUT DATA BYTE
00020E 9635       30    110        JNZ     STOP            JUMP IF NO ACKNOWLEDGE
000210 FD         31    111        MOV     A,R5            FETCH SECOND DATA BYTE
000211 5440       32    112        CALL    MTDAT           OUTPUT DATA BYTE
000213 4435       33    113        JMP     STOP            OUTPUT STOP CONDITION
                        114 *
```

Master reads two data bytes out of CMOS memory (PCD8571) subroutine:

When the master wants to read data out of the memory, it first has to write the pointer into the memory. Because changing of the data direction is only possible after the R/W bit of the slave address, the master has to repeat the START condition, the slave address and the R/W bit (in read state) to change the direction within one transmission.

entry: R3 contains pointer value to be written in the memory.
exit: A = 0 if transmission is successful and received data is stored in registers R6 and R7.
A ≠ 0 if transmission is not successful
contents of registers R2, R6 and R7 are modified

```
000215 23A0       34    128 RMEM   MOV     A,#MEMAD        LOAD MEMORY SLAVE ADDRESS AND WRITE BIT
000217 543C       35    129        CALL    START           OUTPUT START CONDITION, SLAVE ADDRESS AND
                        130 *                               WRITE BIT; INPUT ACKNOWLEDGE BIT
000219 9635       36    131        JNZ     STOP            JUMP IF NO ACKNOWLEDGE RECEIVED
00021B FB         37    132        MOV     A,R3            FETCH POINTER VALUE
00021C 5440       38    133        CALL    MTDAT           OUTPUT POINTER
00021E 9635       39    134        JNZ     STOP            JUMP IF NO ACKNOWLEDGE RECEIVED
000220 8A80       40    135        ORL     P2,#SCL         SET SCL OUTPUT; SCL AND SDA ARE HIGH NOW
000222 23A1       41    136        MOV     A,#MEMAD+RW     LOAD MEMORY SLAVE ADDRESS AND READ BIT
000224 543C       42    137        CALL    START           OUTPUT START CONDITION, SLAVE ADDRESS AND
                        138 *                               READ BIT; INPUT ACKNOWLEDGE BIT
000226 9635       43    139        JNZ     STOP            JUMP IF NO ACKNOWLEDGE RECEIVED
000228 5463       44    140        CALL    MRDAT           READ FIRST DATA BYTE
00022A AE         45    141        MOV     R6,A            SAVE FIRST DATA BYTE
00022B 545D       46    142        CALL    MRACK           OUTPUT ACKNOWLEDGE
                        143 *                               READ SECOND DATA BYTE
00022D AF         47    144        MOV     R7,A            SAVE SECOND DATA BYTE
00022E 27         48    145        CLR     A               TRANSMISSION SUCCESSFUL
```

Output not acknowledge bit and stop condition subroutine:

```
00022F 8980       49    149 MRACKN ORL     P1,#SDA         SET SDA OUTPUT
000231 8A80       50    150        ORL     P2,#SCL         SET SCL OUTPUT
000233 9A7F       51    151        ANL     P2,#.NOT.SCL    RESET SCL OUTPUT
                        152 *
```

Output stop condition subroutine:

```
000235 997F       52    155 STOP   ANL     P1,#.NOT.SDA    OUTPUT STOP CONDITION
000237 8A80       53    156        ORL     P2,#SCL
000239 8980       54    157        ORL     P1,#SDA
00023B 83         55    158        RET
```

Master outputs start condition, slave address, R/W bit and inputs acknowledge bit subroutine:

entry: A contains slave address and R/$\overline{W}$ bit
exit: A = 0 if acknowledge is received
contents of register R2 are modified

```
00023C 997F       56    166 START  ANL     P1,#.NOT.SDA    OUTPUT START CONDITION
00023E 9A7F       57    167        ANL     P2,#.NOT.SCL
                        168 *
```

Master outputs data byte and inputs acknowledge bit subroutine

```
entry: A contains data byte to be transmitted
exit:  A = 0 if acknowledge is received
       contents of register 2 are modified
```

```
                        173 *
000240 BA08       58    174 MTDAT   MOV     R2,#8           SET BIT COUNTER
                        175 *
000242 E7         59    176 MTDAT1  RL      A               SHIFT DATA BYTE
000243 1249       60    177         JB0     MTDAT2          JUMP IF TO BE TRANSMITTED BIT IS HIGH
000245 997F       61    178         ANL     P1,#.NOT.SDA    RESET SDA OUTPUT
000247 444D       62    179         JMP     MTDAT3          CONTINUE
                        180 *
000249 8980       63    181 MTDAT2  ORL     P1,#SDA         SET SDA OUTPUT
00024B 00         64    182         NOP                     DELAY  ### CAN BE DELETED ###
00024C 00         65    183         NOP                     DELAY  ### CAN BE DELETED ###
                        184 *
00024D 8A80       66    185 MTDAT3  ORL     P2,#SCL         SET SCL OUTPUT
00024F 9A7F       67    186         ANL     P2,#.NOT.SCL    RESET SCL OUTPUT
000251 EA42       68    187         DJNZ    R2,MTDAT1       DECR. AND TEST BIT COUNTER
000253 8980       69    188         ORL     P1,#SDA         SET SDA OUTPUT TO INPUT MODE
000255 8A80       70    189         ORL     P2,#SCL         SET SCL OUTPUT
000257 09         71    190         IN      A,P1            INPUT ACKNOWLEDGE BIT
000258 9A7F       72    191         ANL     P2,#.NOT.SCL    RESET SCL OUTPUT
00025A 5380       73    192         ANL     A,#SDA          MASK ACKN. BIT
00025C 83         74    193         RET
```

Master outputs acknowledge bit and reads data byte subroutinte:

```
exit:  A contains recieved data byte
       contents of register 2 are modified
```

```
00025F 8A80       76    200         ORL     P2,#SCL         SET SCL OUTPUT
000261 9A7F       77    201         ANL     P2,#.NOT.SCL    RESET SCL OUTPUT
                        202 *
```

Master reads data byte subroutine:

```
exit:  A contains received data byte
       contents of register 2 are modified
```

```
                        206 *
000263 8980       78    207 MRDAT   ORL     P1,#SDA         SET SDA OUTPUT TO INPUT
000265 BA01       79    208         MOV     R2,#H'01'       INITIALIZE DATA WORD
                        209 *
000267 8A80       80    210 MRDAT1  ORL     P2,#SCL         SET SCL OUTPUT
000269 09         81    211         IN      A,P1            INPUT BIT
00026A 9A7F       82    212         ANL     P2,#.NOT.SCL    RESET SCL OUTPUT
00026C F7         83    213         RLC     A               SHIFT DATA BIT INTO C
00026D FA         84    214         MOV     A,R2            FETCH DATA REGISTER
00026E F7         85    215         RLC     A               SHIFT DATA BIT INTO DATA WORD
00026F AA         86    216         MOV     R2,A            SAVE DATA WORD
000270 E667       87    217         JNC     MRDAT1          JUMP IF NOT YET 8 BITS RECEIVED
000272 83         88    218         RET
                        219 *
000273                  220         END
```

Section 8 - Development Support

DEVELOPMENT SUPPORT

CONTENTS

1. Development Support

Philips provide a wide variety of development support tools which help to bring your system design faster into the marketplace. For each category of product a choice of tools is available depending on the particular application, these include:

- In-circuit emulation development systems
- Cross-assemblers and symbolic debugging software
- Piggybacks
- Evaluation boards

1.1 Development support for 8-bit microcontrollers with instruction set based on the 8048

One of the development environments is formed by the real-time emulator. Two alternative emulators are offered by Philips: the Stand Alone Debug System, (SDS8400) or the single-board LCDS84. The latter is specially suited in a low cost development environment. In most cases the user will connect his emulator to a PC, in which the cross software resides. This software as well as the product dependant emulator probes are available via the Philips sales organisations, a list containing all these items is given in (***Table* 1**). The rest of this chapter gives summaries of the features found in several development tools for the 84(C)XXX and 33XX families.

Development tools available from Philips for 84CXXX and 33XX single-chip microcontrollers

Quick reference Table 1:

Category	Type number (for ordering)	Description
real-time emulator	OM1087A	SDS8400: Real-time, fully transparent emulator, Stand alone Debug Station with emulator breakpoints, Trace, resident debug software, etc. For different family members only the product specific probe has to be exchanged. (see below)
probes for SDS8400	OM1070	probe for 84C85
	OM1071	,, ,, 3344/47/49
	OM1072	,, ,, 84C430/230
	OM1073	,, ,, 84C121
	OM1074	,, ,, 84C440/640/840
	OM1076	,, ,, 3346
	OM1077	,, ,, 84C270/470
	OM1080	,, ,, 84C00T
	OM1081	,, ,, 84C853
	OM1083	,, ,, 84C12/21/22/41/42/81/ 3315/43/48
	OM1084	,, ,, 8411/21/41/61
	OM1086	,, ,, 84C633

Quick reference Table 1: (continued)

Category	Type number (for ordering)	Description
single-board emulator	OM1025	LCDS84: Real-time, fully transparent single-board emulator, with breakpoints, trace, debug software etc. For different family members, only the product specific daughter board has to be exchanged.
daughter-boards for LCDS84	OM1026	Daughter board for the 84C85
	OM1027	Daughter board for the 84C121/C12/C22/C42
	OM1028	Daughter board for the 3344/47/49/51
software	OM1088	MAC84 cross-assembler
	OM1089	SDS_Windows; creates window oriented development environment on the SDS8400 or SDS8051 (runs on IBM-PC or compatible)
demo/ evaluation	OM1016	I^2C demo board with microcontroller, LCD, LED, Par.I/O, SRAM, EEPROM, DTMF generator, Clock, AD/DA conversion,infrared link.
	OM1023	8400 cross-assembler for MS-DOS (2500 AD)
	OM1017	I^2C demo board with only microcontroller stuffed
	OM1018	manual for OM1016/7
	OM1020	LCD and Driver demo-board
I^2C bus analyzer	OM1022	Hard- and software (running on IBM-PC) analyzer to experiment with and analyze the behaviour of the I^2C bus (incl. documentation)

1.2 The SDS8400:

Features:

- Real-time, fully transparent emulation
- Emulation functions based on the bond-out chip for full emulation at exact specs of the target processor
- Supports: all 33XX, 84XX and 84CXXX microprocessors
- Works stand-alone from a VDU terminal and/or can be connected to a wide variety of computers like PC
- Optional trace memory board
- File format: Intelhex
- Two RS-232 links for serial communication; Recognize Xon/Xoff
- Hardware breakpoints on any combination of address, data, register values or branch instructions
- Resident monitor featuring a.o. in-line assembly and disassembly, memory and register display and alteration

1.2.1 SDS8400 System Architecture

The Stand alone Debug Station (SDS) is a system box with on one side the emulation cable and probe and on the other side, two RS-232 connectors serving the users CRT interface and/or a host computer. The system box consists of two boards; the emulation board and a control board.

The firmware required for emulation is carried on the control board which is designed around the 84C00/8400 CPU. This processor accesses external program memory ROM via an 8-bit address bus and is interfaced with two RS-232 serial ports. One serial port may be used for communication with a Microcomputer Development System (MDS) or PC/XT system from which assembled object code programs can be downloaded, the other port is used by the CRT/VDU terminal.

The emulation board is connected to the emulation cable and probe, this also contains an 84xxx bond-out chip, from which address and control data can be extracted when requested.

The power supply is entirely self-contained within the device, Figure 1 shows a block diagram of the SDS architecture.

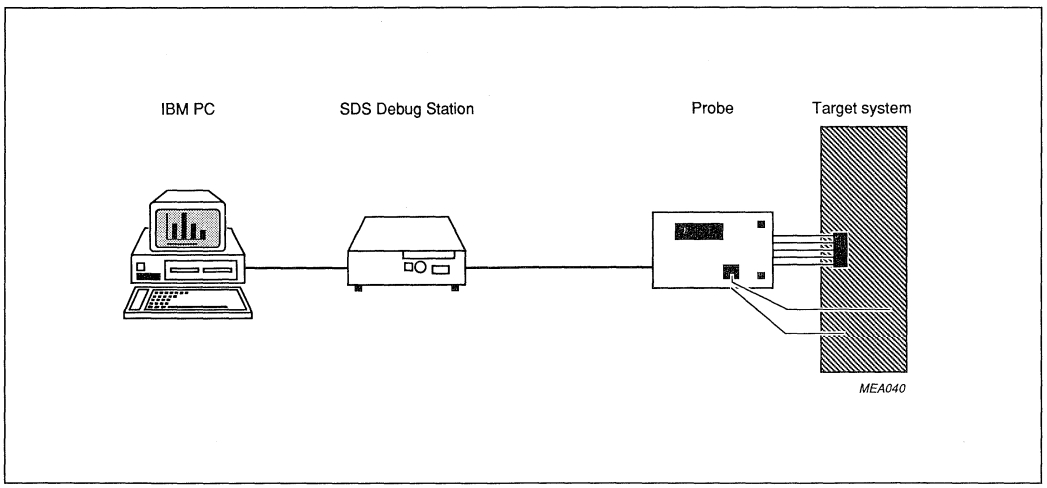


Figure 1 SDS8400 Possible System Set-up

Explanation of system elements in Figure 1

IBM PC or compatible XT, AT, 286, 386 Configuration:

- preferable colour monitor EGA, Mono
- minimum 512K RAM
- DOS version 2.1 or higher
- serial interface RS-232 9600 Baud

The PC, the man-machine interface to the debug-station, handles also software editing, assembling and data storage.

SDS Debug Station: This control unit executes the debug tasks. It is a microprocessor controlled to communicate via the RS-232 interface with the PC as well as to handle the different operation modes of the processor in the probe. In addition a trace memory can be incorporated to give the history of the last 2K instructions executed, as well as log. analysis of the signals on the test-clips.

Probe: For each microcontroller type a special probe is available, it contains the microcontroller bondout configuration. The supply voltage to the microcontroller may be switched from internal to external (from the target system). The built-in oscillator crystal can be bypassed by a switch, for an external one. Additional 8 test leads are available for logic analysis. These signals are monitored in the trace memory. To adapt the probe to the SDS at software level the firmware in 2 EPROMs is included and, it is easy to install them in the field.

The probe may also be used in stand-alone configuration. Its 28 pin socket adopts an EPROM with the customers software , so it will operate as a large piggy-back.

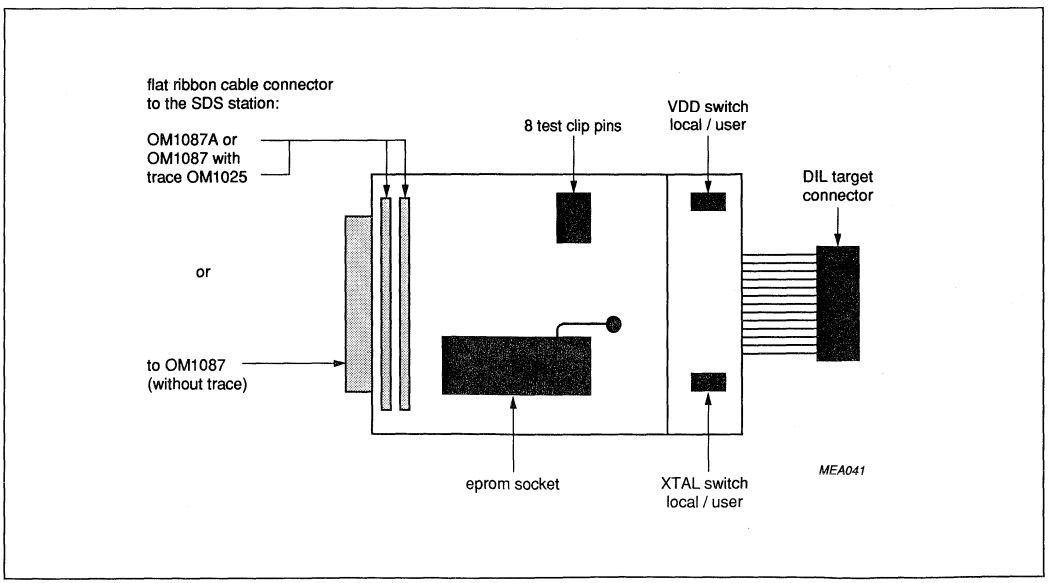


Figure 2 Probe

1.2.2 SDS8400 System software and modes of operation

Monitor software in 12K of EPROM enables the user to perform the following functions:

- To communicate with the SDS using a CRT/VDU terminal or other host computer
- To execute user programs in real-time or single step
- To set breakpoints on program addresses, ranges of program addresses, branch or register values
- To assemble individual 84XXX assembly language instructions into memory, and disassemble memory into assembler mnemonics or equivalents line by line
- To examine and modify locations, registers and bytes, in the on-board program memory of the microcontrollers data memory and/or registers
- To upload or download object code programs from a microcomputer development system or a personal computer system

Modes of Operation

The SDS8400 system has five modes of operation:

- Interrogation (default)
- Assembler mode
- Continuation mode
- Real-time emulation mode
- Single-step execution mode

Interrogation mode is entered following a power-up or system reset. An asterisk prompt is displayed at the left margin indicating that the system is ready to accept a command. All commands are entered through the interrogation mode.

Assembler mode permits the user to enter instructions in 84XXX assembly language. It then assembles the instruction directly into on-board memory. Assembler mode is initiated by the ASM command, and is terminated by pressing Enter/Return. This mode also permits disassembly of the on-board memory into 84XXX mnemonics.

Continuation mode allows the user to enter a list of values for setting memory contents without repeating the type keyword.

Real-time emulation mode permits the user to run code stored in program memory. Emulation starts when the user enters a GO command while in interrogation mode. Real-time emulation is controlled by breakpoints set by the user. If a breakpoint is encountered by the program, the program halts after executing the instruction that contained the breakpoint address. If breakpoints are not used, the program will run until terminated by the user.

Single-step execution mode allows the user to specify the number of steps and run the program one instruction at a time, breaking between steps. Between steps the system also displays instruction mnemonics and register values.

1.2.3 SDS8400 System and cross software

The 84(C)XXX and 33XX microcontrollers are supported by the MAC84 cross-assembler of which the main features are:

- Standard 8048 mnemonics
- Macros
- Memory directive, including files
- Symbol table listing
- Arithmetic and logical operators
- Short assembly time
- Runs on IBM PC or compatible

In addition, an SDS_Windows upgrades your system into a **user friendly**, easy to access, state-of-the-art development environment. The main features of the SDS_WINDOWs are:

- Runs in combination with MAC84 cross-assembler on IBM PC or compatible
- RS-232 driver for upload and/or download
- Menu for selection of editor, assembler, communication port and filenames, stored in a default file
- Windows continuously displaying registers with changes highlighted
- Soft keys for symbolic breakpoints, single step, program step, controller reset, real-time execution, assembly, upload, download and editor window access

1.3 LOW COST Development System LCDS84

The LCDS84 is a low cost, board-level, development system offering full speed (10 MHz) in-circuit emulation of the Philips 84C00 series of microcontrollers. The LCDS84 emulates the 84C21/41/81 directly and other types are supported with the addition of a derivative specific daughter board and footprint adapter.

The LCDS84 provides real time, fully transparent emulation without stretched clock cycles, wait states, stolen I/O, stolen RAM or stolen stack levels. It has an 84C00 bond-out chip for maximum target compatibility, with break points, instruction trace, and test probe trace implemented in additional hardware, guaranteeing full target performance.

The LCDS84 allows the operator control over internal registers and operations of the target microcontroller, thereby allowing development of software in a real time environment. The LCDS84 command language is similar to Intel's ICE* language and is an enhancement of the SDS8400 (OM1087) command set and thus supports SDS windows.

The LCDS84 interfaces via RS-232 to a terminal or the serial port of a personal computer running terminal emulation. The board requires an external 9V 750mA DC power source.

The following LCDS parts are available:

Board	Function	Emulates
LCDS84	Low Cost Development System	PCF84C21P, PCF84C41P, PCF84C81P
LCDS84C85	Derivative and Footprint Adaptor	PCF84C85P
LCDS84C121	Derivative and Footprint Adaptor	PCF84C121P, PCF84C12P, PCF84C22P, PCF84C42P

1.4 Piggybacks

For prototyping and small series of the 84CXXX family, piggyback ICs are available for almost the entire product range.

1.5 Evaluation Boards

I^2C demo-board

The OM1016 is a low cost stand-alone board level product designed to demonstrate Philips I^2C Bus and the 8400 (enhanced 8048) instruction set. It is also used for evaluation of the I^2C Bus peripheral devices and, because of the modular layout of both board and its software, can be used for fast prototyping.

This attractive board is based on the PCF84C00T CMOS microcontroller, and includes documented demonstration software and listings. The 14 sections of the board demonstrate and display eleven I^2C Bus peripheral ICs.

Section	Device
1. Microcontroller	- 84C00T/27C64
2. Clock/Calendar	- PCF8583
3. RAM	- PCF8570
4. EEPROM	- PCF8582
5. DTMF Generator	- PCD3312/TDA7050T
6. IR Receiver Decoder	- SAA3028/TDA3047
7. IR Transmitter	- SAA3006
8. 8-Bit Parallel I/O	- PCF8574A
9. 8-Bit Parallel I/O	- PCF8574
10. A/D - D/A	- PCF8591
11. LED Display	- SAA1064
12. LCD Display	- PCF8577/LTD226F-12
13. Power Supply	
14. 84C21/41/81 Footprint for In-circuit emulation.	

The board is provided with a 3.5mm socket to enable a 9V 150mA plug pack to be connected.

It is suitable as class-sets for teaching institutes.

The manual OM1018 is ordered separately and is a valuable tool in its own right to introduce the I^2C Bus and the 84C00 microcontrollers.

LCD demo-board

The OM1020 is a stand-alone board level product designed to provide a working I^2C and CBus display system to be used to familiarise the users with Philips LCDs and driver IC configurations. Six LCDs and six different driver ICs are used to demonstrate different multiplex rates and various mounting methods.

LCD	DRIVER	RATE	MOUNTING
LTA142F-12	PCF8576T	mux 1:1	Glued-on-pins
LTD321R-12	PCF2110T	mux 1:2	Glued-on-pins
LTD133F-21	PCF1175T	mux 1:2	Heat glued contact strip
LTA232R-11	PCF8566T	mux 1:3	Glued-on-pins
LTD234R-21	PCF8576T	mux 1:4	Elastomer with plastic bezel
LTA341R-11	PCF8578/79T	mux 1:16	Elastomer with metal bezel

The master microcontroller is our PCF84C00T for the I^2C and CBus controlled LCDs. THe OM1020 is primarily intended for demonstration purposes, but because, the software written to show the different features of Philips driver ICs and different techniques in display data handling, it can also be a useful evaluation tool.

The OM1020 comes complete with a full manual and an IBM compatible 5.25" (360K) floppy disk that holds the demonstration source code. There is a 3.5mm power socket ready for your plug pack, which can be 9 to 15V DC (of either polarity) or 7 to 11V AC at 25mA maximum.

I^2C Bus analyzer

The I^2C Bus analyzer consists of software on a diskette, a multimaster bus interface with CMOS piggyback, firmware EPROM and documentation. The main features are:

- Universal I^2C transmitter/receiver menu
- Help functions at any level
- Tracer
- Menu to read or write registers
- System initialization function

Required are an IBM-PC or compatible including a Centronics parallel port.

1.6 Development Tools from third-party vendors

Software

* 2500AD Software, Address: 109 Brookdale Ave.
Box 480, Buena Vista, CA 81211.
Phone: (719) 395-8683
Supplies: cross-assembler for 8400/84CXXX/80C51/80C52

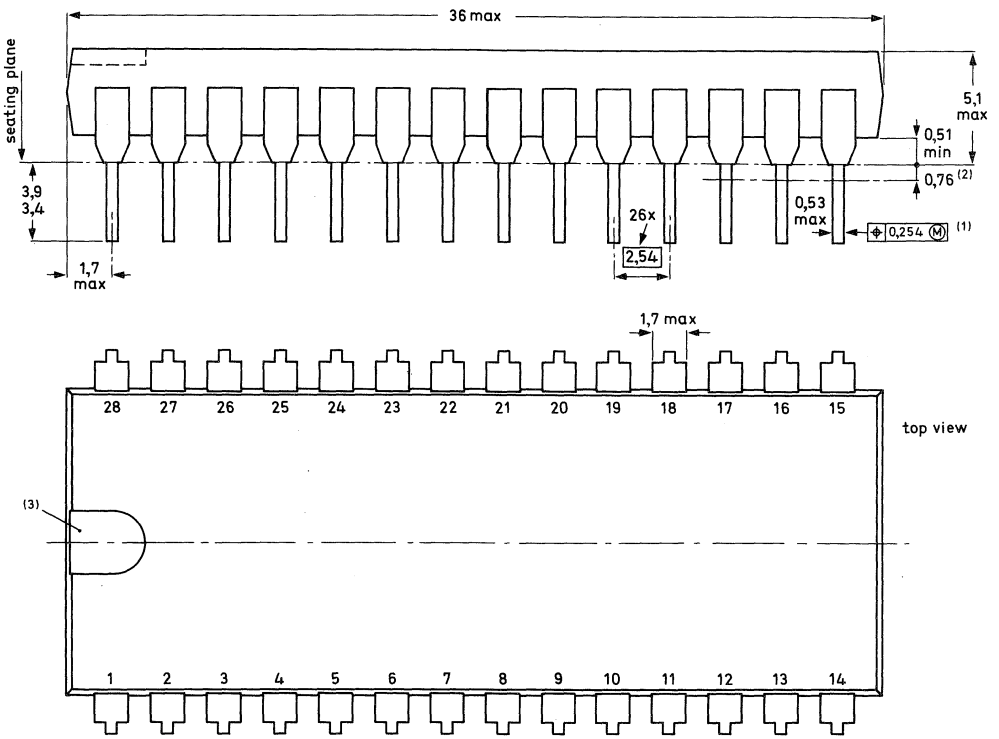
Section 9 - Package Information

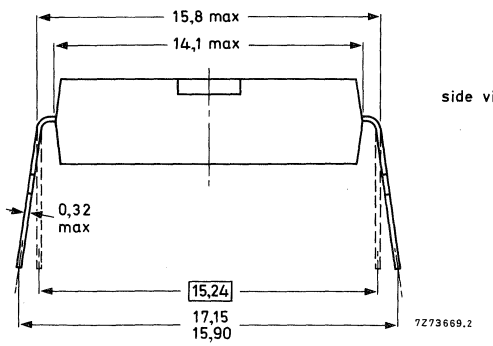
PACKAGE INFORMATION

PACKAGE OUTLINES

For product with prefixes: MAB, MAF, PCD, PCA, PCB, PCF

28-LEAD DUAL IN-LINE; PLASTIC (SOT117)





⊕ Positional accuracy.

Ⓜ Maximum Material Condition.

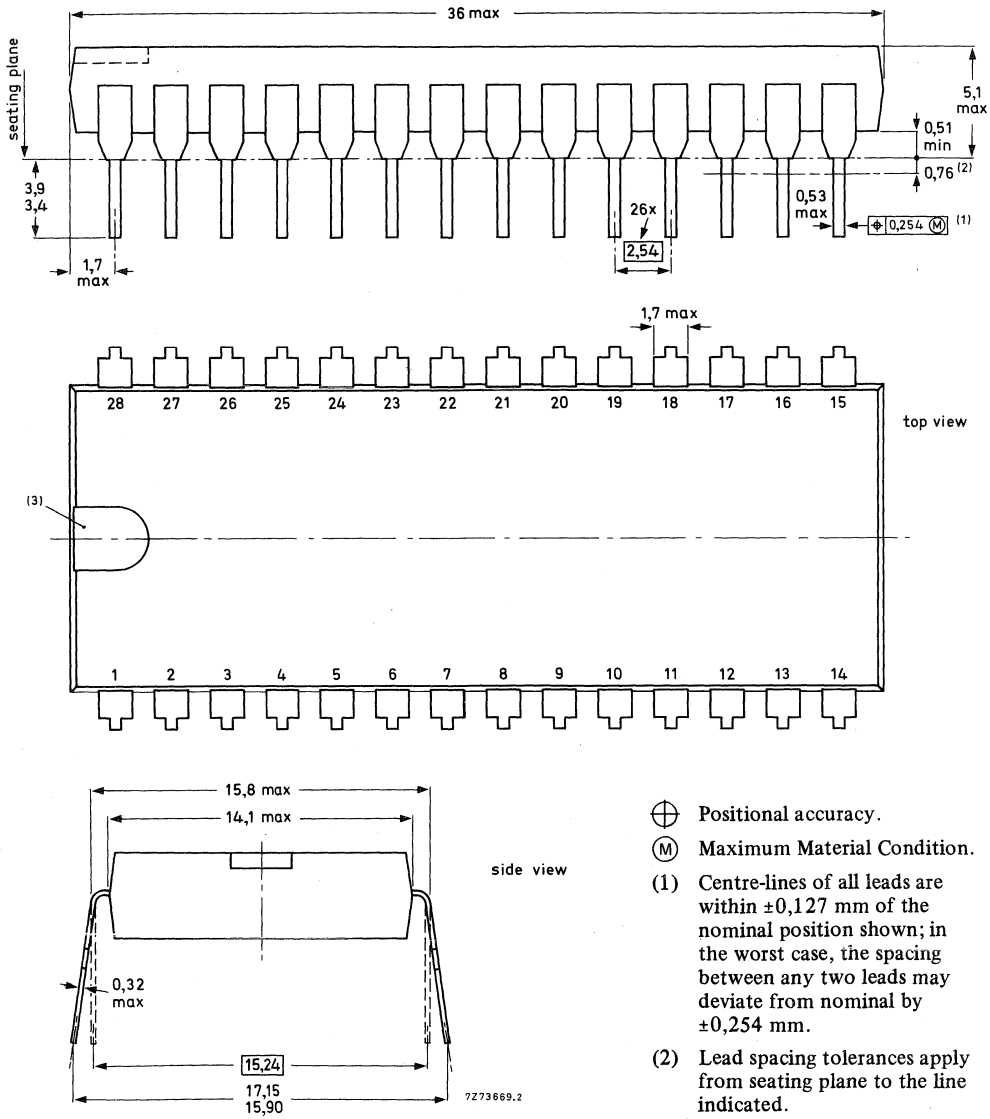
(1) Centre-lines of all leads are within ±0,127 mm of the nominal position shown; in the worst case, the spacing between any two leads may deviate from nominal by ±0,254 mm.

(2) Lead spacing tolerances apply from seating plane to the line indicated.

(3) Index may be horizontal as shown, or vertical.

Dimensions in mm

28-LEAD DUAL IN-LINE; PLASTIC WITH INTERNAL HEAT SPREADER (SOT117)



⊕ Positional accuracy.

Ⓜ Maximum Material Condition.

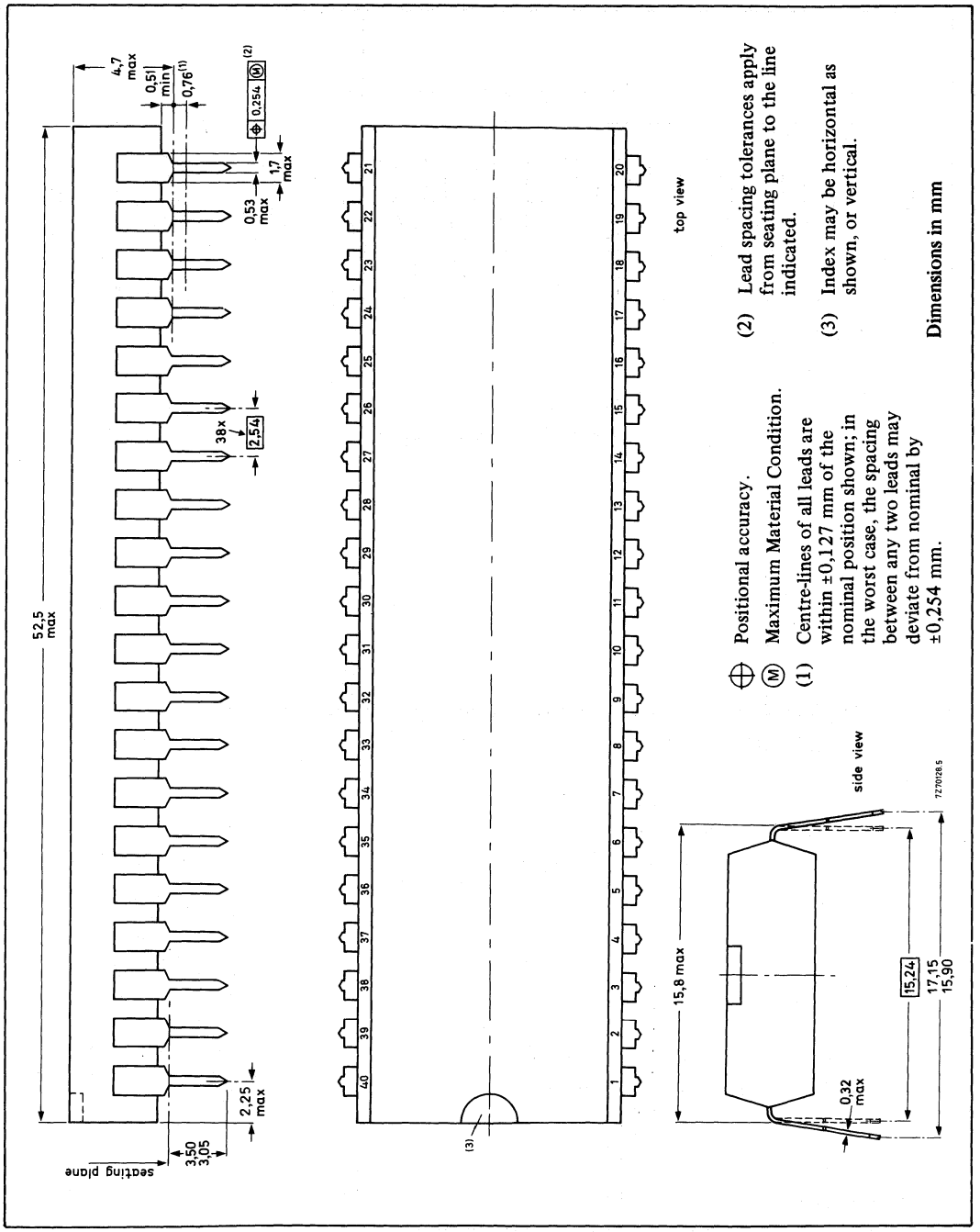
(1) Centre-lines of all leads are within ±0,127 mm of the nominal position shown; in the worst case, the spacing between any two leads may deviate from nominal by ±0,254 mm.

(2) Lead spacing tolerances apply from seating plane to the line indicated.

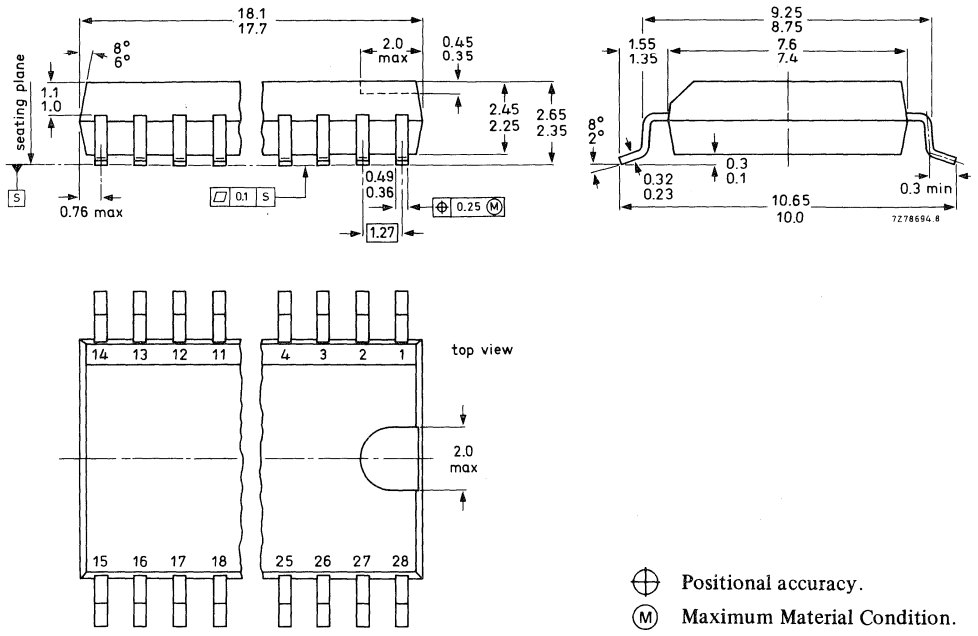
(3) Index may be horizontal as shown, or vertical.

Dimensions in mm

40-LEAD DUAL IN-LINE; PLASTIC (SOT129)

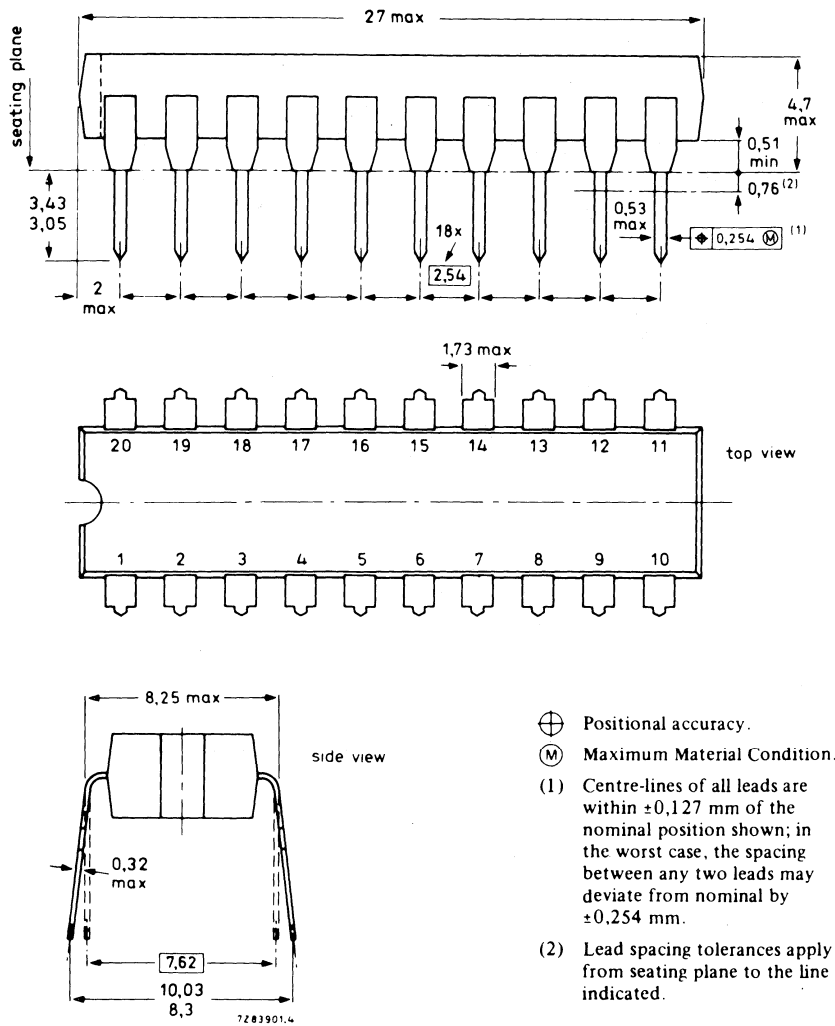


28-LEAD MINI-PACK; PLASTIC (SO28; SOT136A)



Dimensions in mm

20-LEAD DUAL IN-LINE; PLASTIC (SOT146)



⊕ Positional accuracy.

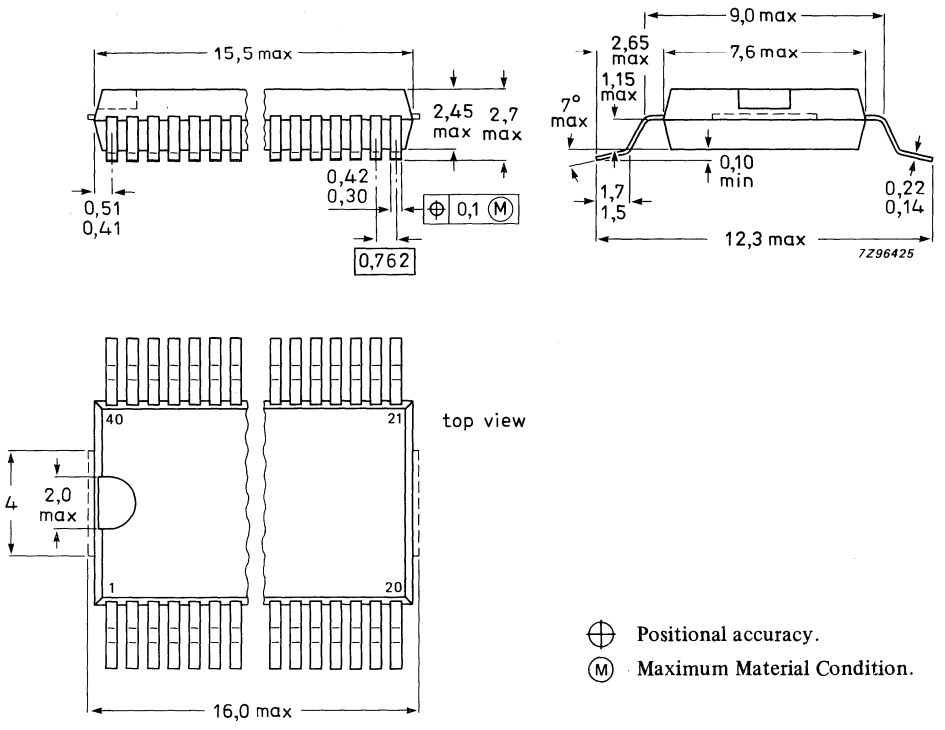
Ⓜ Maximum Material Condition.

(1) Centre-lines of all leads are within ±0,127 mm of the nominal position shown; in the worst case, the spacing between any two leads may deviate from nominal by ±0,254 mm.

(2) Lead spacing tolerances apply from seating plane to the line indicated.

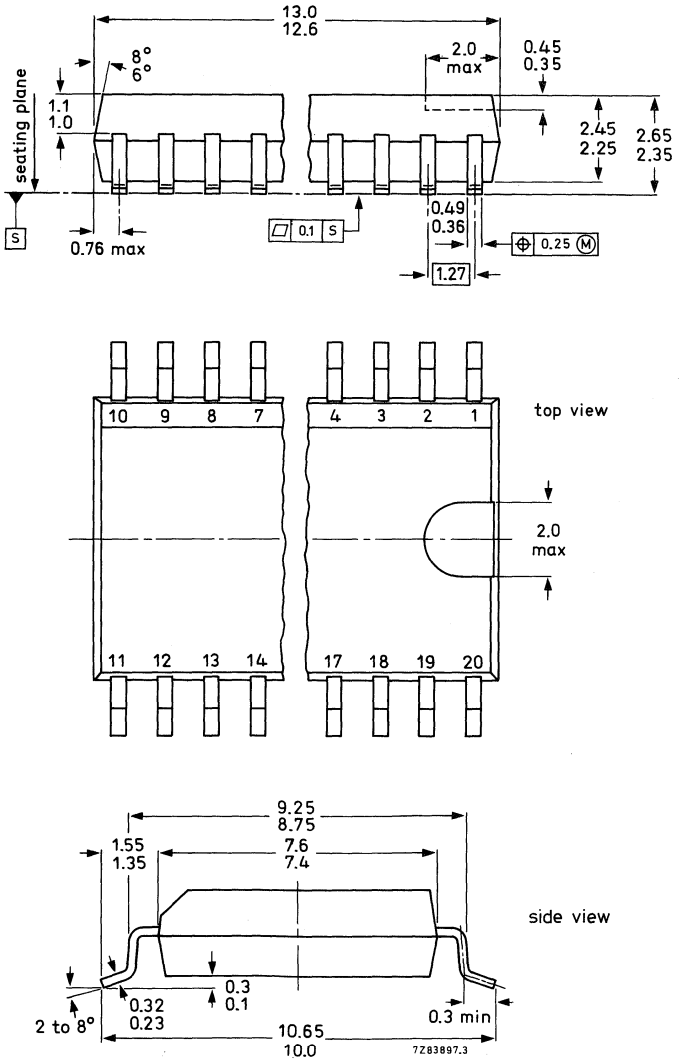
Dimensions in mm

40-LEAD MINI-PACK; PLASTIC (VSO40; SOT158A)



Dimensions in mm

20-LEAD MINI-PACK; PLASTIC (SO20; SOT163A)

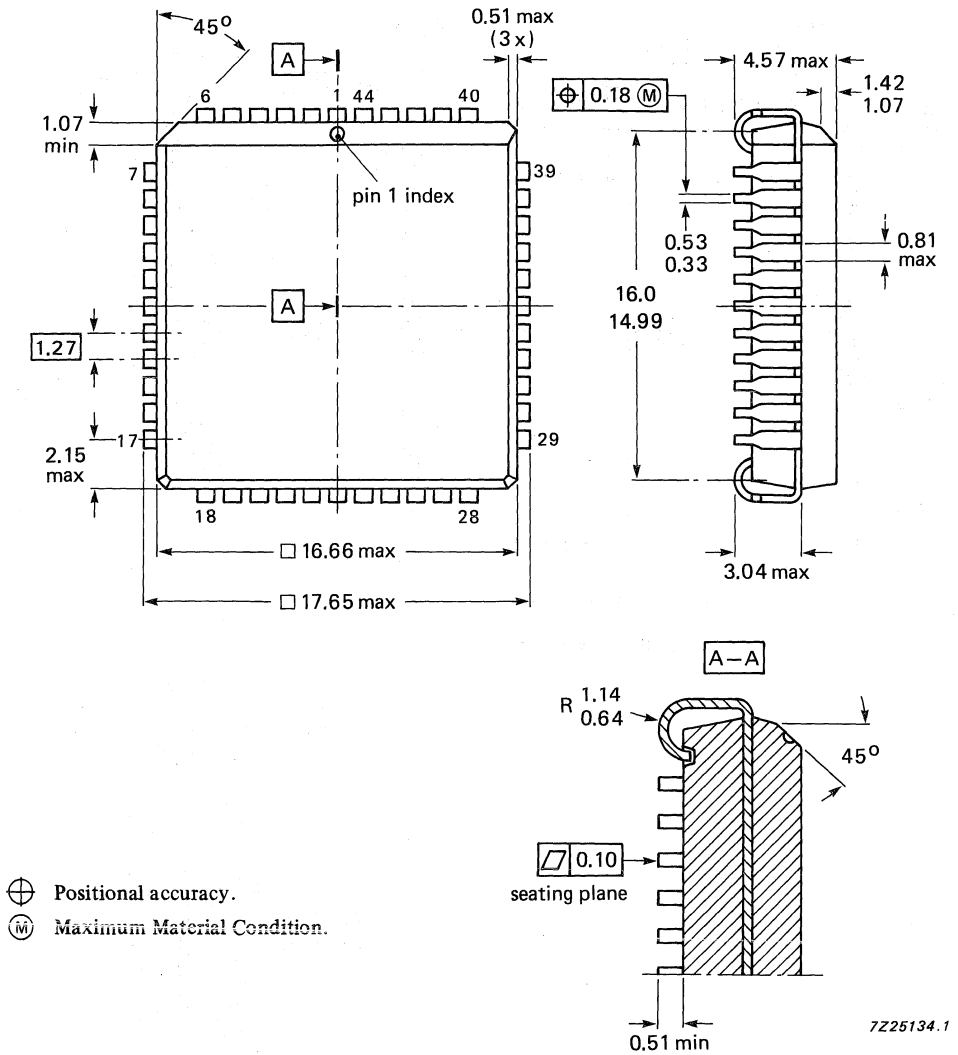


⊕ Positional accuracy.

Ⓜ Maximum Material Condition.

Dimensions in mm

44-LEAD PLASTIC LEADED CHIP CARRIER (PLCC); (SOT187AA)

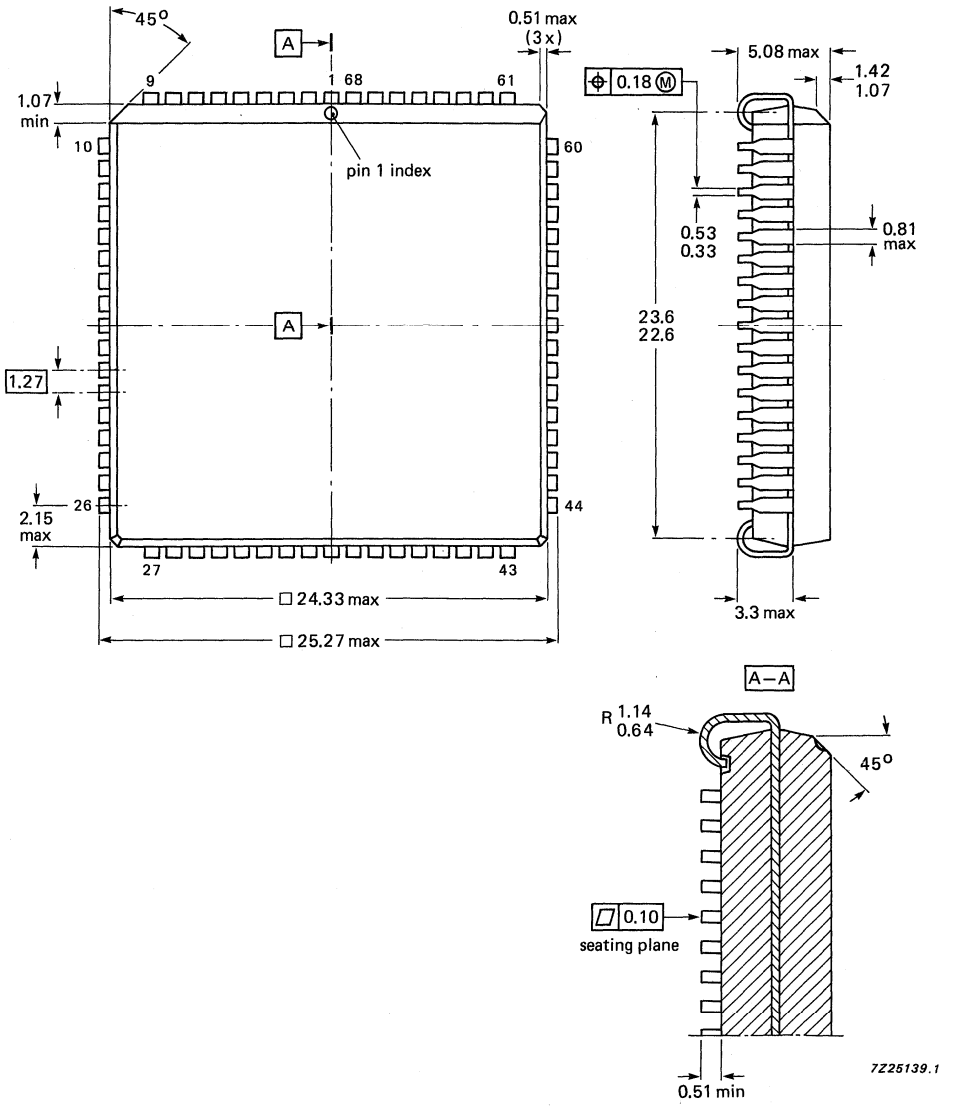


⊕ Positional accuracy.

Ⓜ Maximum Material Condition.

Dimensions in mm

68-LEAD PLASTIC LEADED CHIP CARRIER (PLCC); (SOT188AA, AGA)

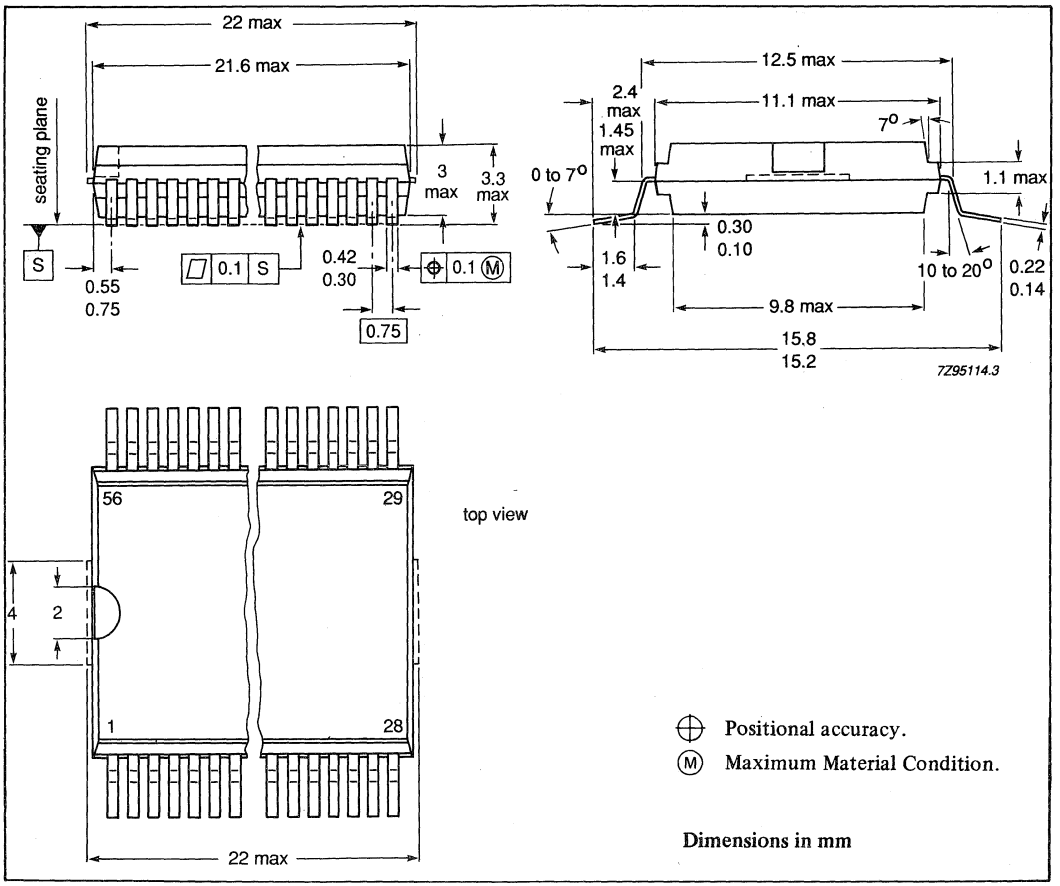


⊕ Positional accuracy.

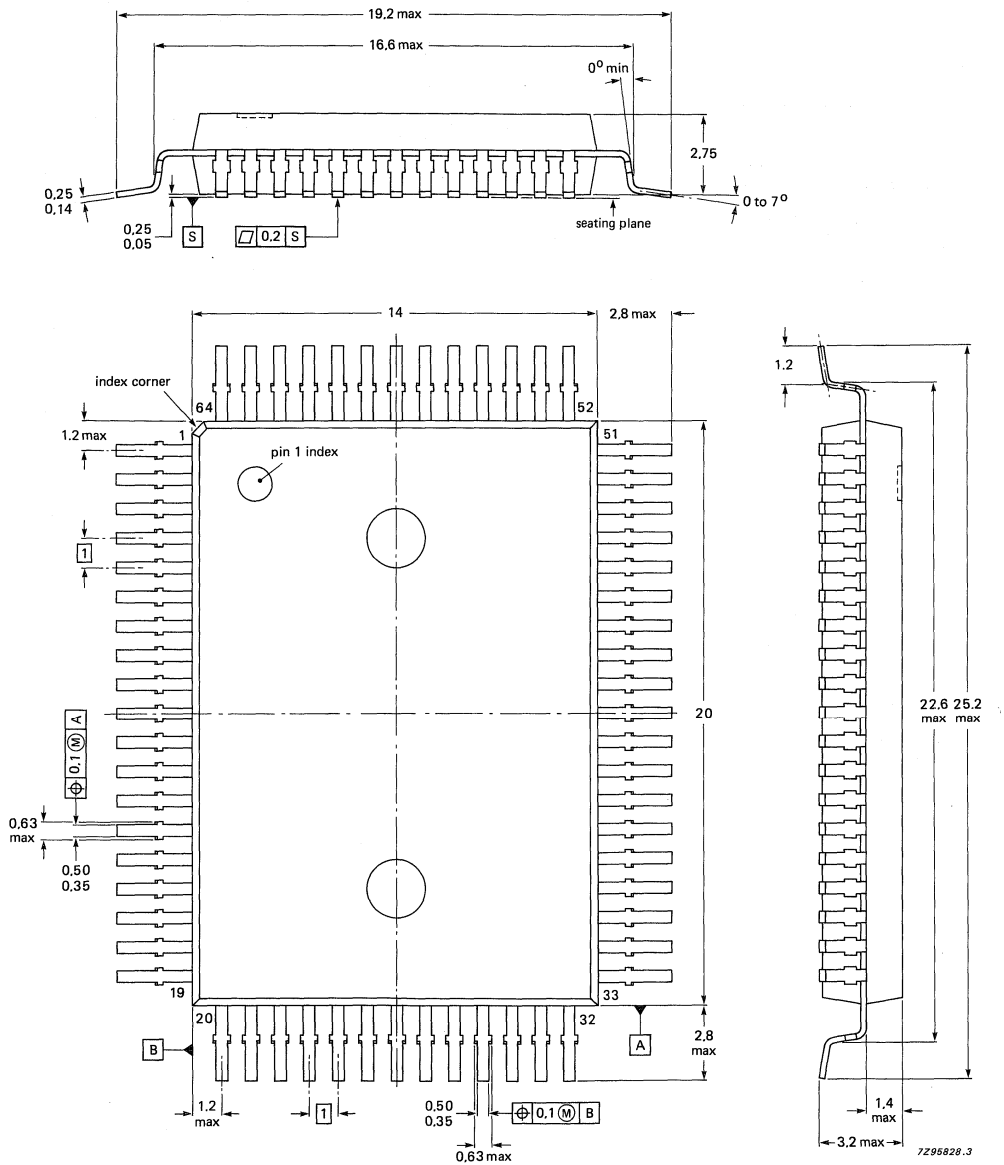
Ⓜ Maximum Material Condition.

Dimensions in mm

56-LEAD MINI-PACK; PLASTIC (VSO56; SOT190)



64-LEAD QUAD FLAT-PACK; PLASTIC (SOT-208)



⊕ Positional accuracy.

Ⓜ Maximum Material Condition.

Dimensions in mm

Soldering

SOLDERING PLASTIC MINI-PACKS

1. By hand-held soldering iron or pulse-heated solder tool

Fix the component by first soldering two, diagonally opposite end leads. Apply the heating tool to the flat part of the lead only. Contact time must be limited to 10 seconds at up to 300 °C. When using proper tools, all other leads can be soldered in one operation within 2 to 5 seconds at between 270 and 320 °C. (Pulse-heated soldering is not recommended for SO packages).

For pulse-heated solder tool (resistance) soldering of VSO packages, solder is applied to substrate by dipping or by an extra thick tin/lead plating before package placement.

2. By wave

During placement and before soldering, the component must be fixed with a droplet of adhesive. After curing the adhesive, the component can be soldered. The adhesive can be applied by screen printing, pin transfer or syringe dispensing.

Maximum permissible solder temperature is 260 °C, and maximum duration of package immersion in solder bath is 10 seconds, if allowed to cool to less than 150 °C within 6 seconds. Typical dwell time is 4 seconds at 250 °C.

3. By solder paste reflow

Reflow soldering requires the solder paste (a suspension of fine solder particles, flux and binding agent) to be applied to the substrate by screen printing, stencilling or pressure-syringe dispensing before device placement.

Several techniques exist for reflowing, for example, thermal conduction by heated belt, infrared, and vapour-phase reflow. Dwell times vary between 50 and 300 seconds according to method. Typical reflow temperatures range from 215 to 250 °C.

Pre-heating is necessary to dry paste and evaporate binding agent.

Pre-heating duration: 45 minutes at 45 °C.

4. Repairing soldered joints

The same precaution and limits apply as in (1) above.

SOLDERING PLASTIC DUAL IN-LINE PACKAGES

1. By hand

Apply the soldering iron below the seating plane (or not more than 2 mm above it). If its temperature is below 300 °C it must not be in contact for more than 10 seconds; if between 300 and 400 °C, for not more than 5 seconds.

2. By dip or wave

The maximum permissible temperature of the solder is 260 °C; this temperature must not be in contact with the joint for more than 5 seconds. The total contact time of successive solder waves must not exceed 5 seconds.

The device may be mounted up to the seating plane, but the temperature of the plastic body must not exceed the specified storage maximum. If the printed-circuit board has been pre-heated, forced cooling may be necessary immediately after soldering to keep the temperature within the permissible limit.

3. Repairing soldered joints

The same precautions and limits apply as in (1) above.

DATA HANDBOOK SYSTEM

DATA HANDBOOK SYSTEM

Our Data Handbook System comprises more than 60 books with specifications on electronic components, subassemblies and materials. It is made up of seven series of handbooks:

INTEGRATED CIRCUITS

DISCRETE SEMICONDUCTORS

DISPLAY COMPONENTS

PASSIVE COMPONENTS*

PROFESSIONAL COMPONENTS**

MAGNETIC PRODUCTS*

LIQUID CRYSTAL DISPLAYS

The contents of each series are listed on pages III to IX.

The data handbooks contain all pertinent data available at the time of publication, and each is revised and reissued periodically.

Where application is given it is advisory and does not form part of the product specification.

Condensed data on the preferred products of Philips Components is given in our Preferred Type Range catalogue (issued annually).

Information on current Data Handbooks and how to obtain a subscription for future issues is available from any of the Organizations listed on the back cover.
Product specialists are at your service and enquiries will be answered promptly.

* Will replace the Components and materials (green) series of handbooks.
** Will replace the Electron tubes (blue) series of handbooks.

INTEGRATED CIRCUITS

This series of handbooks comprises:

code	handbook title
IC01	**Radio, audio and associated systems** Bipolar, MOS
IC02a/b	**Video and associated systems** Bipolar, MOS
IC03	**ICs for Telecom;** Subscriber sets, Cordless Telephones, Mobile/Cellular, Radio Pagers
IC04	**HE4000B logic family** CMOS
IC05	**Advanced Low-power Schottky (ALS) Logic Series**
IC06	**High-speed CMOS; PC74HC/HCT/HCU** Logic family
IC07	**Advanced CMOS logic (ACL)**
IC08	**10/100K ECL Logic/Memory/PLD**
IC09	**TTL logic series**
IC10	**Memories** MOS, TTL, ECL
IC11	**Linear Products**
IC12	**I^2C-bus compatible ICs**
IC13	**Semi-custom** Programmable Logic Devices (PLD)
IC14	**Microcontrollers** NMOS, CMOS
IC15	**FAST TTL logic series**
Supplement to IC15	**FAST TTL logic series**
IC16	**CMOS integrated circuits for clocks and watches**
IC17	**ICs for Telecom;** ISDN
IC18	**Microprocessors and peripherals**
IC19	**Data communication products**
IC20	**8051-based 16-bit microcontrollers**
IC23	**Advanced BiCMOS interface logic**

DISCRETE SEMICONDUCTORS

This series of data handbooks comprises:

current code	new code	handbook title
S1	SC01	Diodes High-voltage tripler units
S2a	SC02	Power diodes
S2b	SC03	Thyristors and triacs
S3	SC04	Small-signal transistors
S4a	SC05	Low-frequency power transistors and hybrid IC power modules
S4b	SC06	High-voltage and switching power transistors
S5	SC07	Small-signal field-effect transistors
S6	SC08a*	RF power bipolar transistors
	SC08b**	RF power MOS transistors
	SC09	RF power modules
S7	SC10	Surface mounted semiconductors
S8b	SC12	Optocouplers
S9	SC13*	Power MOS transistors
S10	SC14	Wideband transistors and wideband hybrid IC modules
S11	SC15	Microwave transistors
S15**	SC16	Laser diodes
S13	SC17	Semiconductor sensors

* Not yet issued with the new code in this series of handbooks.
** New handbook in this series; will be issued shortly.

DISPLAY COMPONENTS

This series of data handbooks comprises:

code	handbook title
DC01	**Colour display components** Colour TV Picture Tubes and Assemblies Colour Monitor Tube Assemblies
DC02	**Monochrome monitor tubes and deflection units**
DC03	**Television tuners, coaxial aerial input assemblies**
DC04	**Loudspeakers**
DC05	**Flyback transformers, mains transformers and general-purpose FXC assemblies**

PASSIVE COMPONENTS

This series of data handbooks comprises:

current code	new code	handbook title
C14	**PA01**	**Electrolytic capacitors; solid and non-solid**
C11	**PA02**	**Varistors, thermistors and sensors**
C12	**PA03**	**Potentiometers and switches**
C7	**PA04**	**Variable capacitors**
C22	**PA05***	**Film capacitors**
C15	**PA06**	**Ceramic capacitors**
C9	**PA07***	**Piezoelectric quartz devices**
C13	**PA08**	**Fixed resistors**

* Not yet issued with the new code in this series of handbooks.

PROFESSIONAL COMPONENTS

This series of data handbooks comprises:

current code	new code	handbook title
T3	**PC01**	**High-power klystrons and accessories**
T5	**PC02***	**Cathode-ray tubes**
T6	**PC03***	**Geiger-Müller tubes**
T9	**PC04**	**Photo multipliers**
T10	**PC05**	**Plumbicon camera tubes and accessories**
T11	**PC06**	**Circulators and Isolators**
T12	**PC07**	**Vidicon and Newvicon camera tubes and deflection units**
T13	**PC08**	**Image intensifiers**
T15	**PC09**	**Dry-reed switches**
	PC11	**Solid state image sensors and peripherals integrated circuits**
T9	**PC12***	**Electron multipliers**

* Not yet issued with the new code in this series of handbooks.

MAGNETIC PRODUCTS

This series of data handbooks comprises:

current code	new code	handbook title
C4, C5	MA01	**Soft Ferrites**
C16	MA02*	**Permanent magnet materials**
C19	MA03*	**Piezoelectric ceramics**

* Not yet issued with the new code in this series of handbooks.

LIQUID CRYSTAL DISPLAYS

current code	new code	handbook title
S14	**LCD01**	**Liquid Crystal Displays and driver ICs for LCDs**

Philips Components – a worldwide company

Argentina: PHILIPS ARGENTINA S.A., Div. Philips Components, Vedia 3892, 1430 BUENOS AIRES, Tel. (01) 541-4261.
Australia: PHILIPS COMPONENTS PTY Ltd, 11 Waltham Street, ARTARMON, N.S.W. 2064, Tel. (02) 439 3322.
Austria: ÖSTERREICHISCHE PHILIPS INDUSTRIE G.m.b.H., UB Bauelemente, Triester Str. 64, 1101 WIEN, Tel. (0222) 60 101-820.
Belgium: N.V PHILIPS PROF. SYSTEMS – Components Div., 80 Rue Des Deux Gares, B-1070 BRUXELLES, Tel. (02) 52 56 111.
Brazil: PHILIPS COMPONENTS (Active Devices & LCD) Av. das Nacoes Unidas, 12495-SAO PAULO-SP, CEP 04578, P.O. Box 7383, Tel. (011) 534-2211.
PHILIPS COMPONENTS (Passive Devices & Materials) Av. Francisco Monteiro 702, RIBEIRAO PIRES-SP, CEP 09400, Tel. (011) 459-8211.
Canada: PHILIPS ELECTRONICS LTD., Philips Components, 601 Milner Ave., SCARBOROUGH, Ontario, M1B 1M8, Tel. (416) 292-5161.
(IC Products) PHILIPS COMPONENTS – Signetics Canada LTD., 1 Eva Road, Suite 411, ETOBICOKE, Ontario, M9C 4Z5, Tel. (416) 626-6676.
Chile: PHILIPS CHILENA S.A., Av. Santa Maria 0760, SANTIAGO, Tel. (02) 77 38 16.
Colombia: IPRELENSO LTDA., Carrera 21 No. 56-17, BOGOTA, D.E., P.O. Box 77621, Tel. (01) 249 76 24.
Denmark: PHILIPS COMPONENTS A/S, Prags Boulevard 80, PB1919, DK-2300 COPENHAGEN S, Tel. 01-54 11 33.
Finland: PHILIPS COMPONENTS, Sinikalliontie 3, SF-2630 ESPOO, Tel. 358-0-50 261.
France: PHILIPS COMPOSANTS, 117 Quai du Président Roosevelt, 92134 ISSY-LES-MOULINEAUX Cedex, Tel. (01) 40 93 80 00.
Germany: PHILIPS COMPONENTS UB der Philips G.m.b.H., Burchardstrasse 19, D-2 HAMBURG 1, Tel. (040) 3296-0.
Greece: PHILIPS HELLENIQUE S.A., Components Division, No. 15, 25th March Street, GR 17778 TAVROS, Tel. (01) 48 94 339/48 94 911.
Hong Kong: PHILIPS HONG KONG LTD., Components Div., 15/F Philips Ind. Bldg., 24-28 Kung Yip St., KWAI CHUNG, Tel. (0)-42 45 121.
India: PEICO ELECTRONICS & ELECTRICALS LTD., Components Dept., Shivsagar Estate 'A' Block, P.O. Box 6598, 254-D Dr. Annie Besant Rd., BOMBAY – 40 0018, Tel. (022) 49 21 500-49 21 515.
Indonesia: P.T. PHILIPS-RALIN ELECTRONICS, Components Div., Setiabudi II Building, 6th Fl., Jalan H.R. Rasuna Said (P.O. Box 223/KBY) Kuningan, JAKARTA 12910, Tel. (021) 51 79 95.
Ireland: PHILIPS ELECTRONICS (IRELAND) LTD., Components Division, Newstead, Clonskeagh, DUBLIN 14, Tel. (01) 69 33 55.
Italy: PHILIPS S.p.A., Philips Components, Piazza IV Novembre 3, I-20124 MILANO, Tel. (02) 6752.1.
Japan: PHILIPS JAPAN LTD., Components Division, Philips Bldg 13-37, Kohnan 2-chome, Minato-ku, TOKYO 108, Tel. (03) 740-5028.
Korea (Republic of): PHILIPS ELECTRONICS (KOREA) LTD. Components Division, Philips House, 260-199 Itaewon-dong, Yongsan-ku, SEOUL, Tel. (02) 794-5011.
Malaysia: PHILIPS MALAYSIA SDN BHD, Components Div., 3 Jalan SS15/2A SUBANG, 47500 PETALING JAYA, Tel. (03) 73 45 511.
Mexico: PHILIPS COMPONENTS, Paseo Triunfo de la Republica, No 215 Local 5, Cd Juarez CHI HUA HUA 32340 MEXICO Tel. (16) 18-67-01/02.
Netherlands: PHILIPS NEDERLAND B.V., Marktgroep Philips Components, Postbus 90050, 5600 PB EINDHOVEN, Tel. (040) 78 37 49.
New Zealand: PHILIPS NEW ZEALAND LTD., Components Division, 110 Mt. Eden Road, C.P.O. Box 1041, AUCKLAND, Tel. (09) 605-914.
Norway: NORSK A/S PHILIPS, Philips Components, Box 1, Manglerud 0612, OSLO, Tel. (02) 74 10 10.
Pakistan: PHILIPS ELECTRICAL CO. OF PAKISTAN LTD., Philips Markaz, M.A. Jinnah Rd., KARACHI-3, Tel. (021) 72 57 72.
Peru: CADESA, Carretera Central 6.500, LIMA 3, Apartado 5612, Tel. 51-14-35 00 59.
Philippines: PHILIPS ELECTRICAL LAMPS INC. Components Div., 106 Valero St. Salcedo Village, P.O. Box 911, MAKATI, Metro MANILA, Tel. (63-2) 810-0161.
Portugal: PHILIPS PORTUGUESA S.A.R.L., Av. Eng. Duarte Pacheco 6, 1009 LISBOA Codex, Tel. (019) 68 31 21.
Singapore: PHILIPS SINGAPORE, PTE LTD., Components Div., Lorong 1, Toa Payoh, SINGAPORE 1231, Tel. 35 02 000.
South Africa: S.A. PHILIPS PTY LTD., Components Division, JOHANNESBURG 2000, P.O. Box 7430.
Spain: PHILIPS COMPONENTS, Balmes 22, 08007 BARCELONA, Tel. (03) 301 63 12.
Sweden: PHILIPS COMPONENTS, A.B., Tegeluddsvägen 1, S-11584 STOCKHOLM, Tel. (0)8-782 1 000.
Switzerland: PHILIPS A.G., Components Dept., Allmendstrasse 140-142, CH-8027 ZÜRICH, Tel. (01) 488 22 11.
Taiwan: PHILIPS TAIWAN LTD., 581 Min Sheng East Road, P.O. Box 22978, TAIPEI 10446, Taiwan, Tel. 886-2-5005899.
Thailand: PHILIPS ELECTRICAL CO. OF THAILAND LTD., 283 Silom Road, P.O. Box 961, BANGKOK, Tel. (02) 233-6330-9.
Turkey: TÜRK PHILIPS TICARET A.S., Philips Components, Talatpasa Cad. No. 5, 80640 LEVENT/ISTANBUL, Tel. (01) 179 27 70.
United Kingdom: PHILIPS COMPONENTS LTD., Mullard House, Torrington Place, LONDON WC1E 7HD, Tel. (071) 580 6633.
United States: (Colour picture tubes – Monochrome & Colour Display Tubes) PHILIPS DISPLAY COMPONENTS COMPANY, 1600 Huron Parkway, P.O. Box 963, ANN ARBOR, Michigan 48106, Tel. 313/996-9400.
(IC Products) PHILIPS COMPONENTS – Signetics, 811 East Arques Avenue, SUNNYVALE, CA 94088-3409, Tel. (408) 991-2000.
(Passive Components, Discrete Semiconductors, Materials and Professional Components & LCD) PHILIPS COMPONENTS, Discrete Products Division, 2001 West Blue Heron Blvd., P.O. Box 10330, RIVIERA BEACH, Florida 33404, Tel. (407) 881-3200.
Uruguay: PHILIPS COMPONENTS, Coronel Mora 433, MONTEVIDEO, Tel. (02) 70-40 44.
Venezuela: MAGNETICA S.A., Calle 6, Ed. Las Tres Jotas, CARACAS 1074A, App. Post. 78117, Tel. (02) 241 75 09.
Zimbabwe: PHILIPS ELECTRICAL (PVT) LTD., 62 Mutare Road, HARARE, P.O. Box 994, Tel. 47211.

For all other countries apply to: Philips Components Division, Strategic Accounts and International Sales, P.O. Box 218, 5600 MD EINDHOVEN, The Netherlands, Telex 35000 phtcnl, Fax. +31-40-723753

AS84

Printed in The Netherlands 9398 177 00011

Philips Components

PHILIPS

PHILIPS